

On Bandwidth Tiered Service

George N. Rouskas, *Senior Member, IEEE*, Nikhil Baradwaj

Abstract—Many network operators offer some type of tiered service, in which users may select only from a small set of service levels (tiers). Such a service has the potential to simplify a wide range of core network functions, allowing the providers to scale their operations efficiently. In this work, we study a number of problem variants related to service tier selection. Our contributions include: (1) a faster algorithm for obtaining optimal service tiers; (2) a new formulation and optimal algorithm to optimize jointly the number and magnitude of each service tier; and (3) the concept of “TDM emulation” in which all service tiers are multiples of the same (software-configurable) bandwidth unit, and a suite of algorithms to select jointly the basic unit and service tiers. Our work provides a systematic framework for reasoning about and tackling algorithmically the general problem of service tier selection, and has applications to a number of networking contexts, including access networks (e.g., determining the tiers for ADSL, cable modem networks or PONs) and core networks (e.g., LSP sizing for MPLS networks).

I. INTRODUCTION

Historically, packet-switched computer networks, including the Internet and legacy networks based on Asynchronous Transfer Mode (ATM) or Frame Relay (FR) technologies, are designed to be *continuous-rate*. In a continuous-rate network, users may request any rate of service (bandwidth), and the network must be able to accommodate arbitrary requests. Theoretically speaking, continuous-rate networks may allocate bandwidth at very fine granularities; for instance, one client may request a rate of 98.99 Megabit per second (Mbps), while another customer may ask for 99.01 Mbps. Taken to the limit, bandwidth in such networks could potentially be allocated at increments of 1 bit per second (bps). Clearly, the option of requesting arbitrary rates offers clients maximum flexibility in utilizing the available network capacity.

On the other hand, supporting bandwidth allocation at such extremely fine granularity may seriously complicate the operation and management of the network. Based on the above example, the network provider faces the problem of designing mechanisms to distinguish between the two rates (i.e., 98.99 Mbps vs. 99.01 Mbps) and enforce them in an accurate and reliable manner. However, the task of differentiating between the two users on the basis of these two rates may be extremely difficult, or even impossible to accomplish for traffic of finite duration, undermining the network’s ability to support important functions such as robust traffic policing or accurate customer billing. Furthermore, given the unpredictability of future bandwidth demands in terms of their size, arrival time,

and duration, link capacity across a continuous-rate network may become fragmented. Such fragmentation poses significant challenges in terms of traffic engineering, and may compromise the ability to achieve an acceptable level of utilization or meet users’ quality of service (QoS) requirements.

A. Tiered-Service Networks

Due to the issues involved in making fine-granular services available to a large, heterogeneous user population cost-effectively, in practice, most network operators have developed a variety of *tiered service* models in which users may select only from a small set of service *tiers* (levels) which offer progressively higher rates (bandwidth). The main motivation for offering such a service is to simplify a wide range of core functions (including network management and equipment configuration, traffic engineering, service level agreements, billing, and customer support), enabling the providers to scale their operations to hundreds of thousands or millions of customers. Returning to the previous example, a tiered-service network might assign both users requesting 98.99 Mbps and 99.01 Mbps to the next higher available rate, say, 100 Mbps. In this case, there is no need to handle the two customers’ traffic differently; furthermore, the network operator only needs to supply policing mechanisms for a small set of rates, independent of the number of users.

For a more formal definition, consider a network that offers a service characterized by a single parameter, e.g., the bandwidth of the user’s access link. A tiered-service network is one that offers p levels (tiers) of service, where typically p is a small integer, much smaller than the number n of (potential) network users (i.e., $p \ll n$). Let

$$S = \{z_1, z_2, \dots, z_p\} \quad (1)$$

denote the set of service tiers offered by the network provider. Without loss of generality, we make the assumption that the service tiers are distinct and are labeled such that

$$z_1 < z_2 < \dots < z_p. \quad (2)$$

Users wishing to receive service are limited to only these p tiers, and may subscribe to any tier depending on their needs and their willingness to pay the corresponding service fee. In particular, z_1 is the minimum and z_p the maximum amount of service that a user may receive. In the case of residential Internet access, for instance, z_1 may correspond to a minimum bandwidth for the service to be considered “broadband,” while z_p may correspond to the capacity of the access link, e.g., as determined by limitations imposed by Asymmetric Digital Subscriber Line (ADSL) technology.

According to this definition, traditional telephone networks and transport networks based on Synchronous Optical Network

This material is based upon work supported by the NSF under grant CNS-0434975.

George N. Rouskas is with the Department of Computer Science, North Carolina State University, Raleigh, NC 27695-8206, USA. Email: rouskas@ncsu.edu .

Nikhil Baradwaj is with MicroStrategy, McLean, VA 22102, USA. Email: nikhilbaradwaj@yahoo.co.in .

or Synchronous Digital Hierarchy (SONET/SDH) technology [18] belong to the class of tiered-service networks. Indeed, such networks allocate bandwidth in discrete tiers that are multiples of a basic unit rate that corresponds to the slot size in the underlying Time Division Multiplexing (TDM) system.

Motivated by the discrete nature of bandwidth allocation in TDM systems, an early study by Lea and Alyatama [30] investigated the benefits of “bandwidth quantization” in packet-switched networks. In their terminology, “bandwidth quantization” refers to sampling the (effectively) continuous range of possible rates to select a small set of discrete bandwidth levels that are made available to users; in essence, these levels correspond to the service tiers we defined earlier. This work was carried out with the goal of reducing the number of states required to analyze broadband ATM networks under the assumption of Poisson traffic, and it presented a heuristic based on simulated annealing to obtain a sub-optimal set of discrete bandwidth levels of service. Because of the significantly reduced state space, it is possible to apply elegant and exact theoretical models to analyze the performance of a tiered-service network efficiently. The main contribution of this study was to demonstrate for the first time that this benefit comes almost for free, as even with a sub-optimal set of tiers the performance degradation (e.g., in terms of call blocking) compared to a continuous-rate network is negligible.

Current tiered service offerings by major ISPs can be broadly classified in two categories based on the tiering structure. The structure of one class of service tiers for Internet access, especially those targeted to business customers, is based on the bandwidth hierarchy of the underlying transport network infrastructure (e.g., DS-1, DS-3, OC-3, etc.). While this is a natural arrangement for the service provider, it is unlikely that hierarchical rates designed decades ago for voice traffic would be a good match for today’s business data applications. The second class employs *exponential tiering* structures in which each tier offers twice the bandwidth of the previous one. The various ADSL tiers (e.g., 384 Kbps, 768 Kbps, 1.5 Mbps, 3 Mbps, 6 Mbps, etc.) available through several ISPs are an example of such an exponential structure. While such simple tier structures may be an appropriate choice for marketing purposes, the relationship between these exponentially increasing levels of service (and their price) and the usage patterns (and willingness or ability to pay) of the population of potential subscribers is open to debate.

So far, we have discussed tiering in the context of broadband Internet access services that are generally characterized by the bandwidth available on the customer’s access link. However, the concept of tiering is equally applicable to other parameters that may characterize the QoS experienced by a customer’s traffic and may be included in the service level agreement (SLA) negotiated between the customer and provider. Consider, for instance, a provider offering a service that guarantees an upper bound on the delay experienced by its customers’ packets. On the one hand, the provider is unlikely to be able to support fine-granularity delay bounds (e.g., at the level of nanoseconds) within a network of realistic size even with the most sophisticated (and expensive) QoS mechanisms. On the other hand, users are unlikely to require (or afford)

delay bounds at such a level of precision. A more reasonable approach would be to offer a small set of delay bound tiers that are tailored to specific applications, e.g., voice, (stored) video on demand, (live) video conferencing, etc. Such delay bounds are likely to be tied to human perception abilities that allow for coarse granularities of tens of milliseconds, making it unnecessary for the network to have to distinguish packet delays at extremely fine precision.

Similar observations apply to other QoS parameters, e.g., level of protection of user traffic. In this case, it may be possible to define and offer a set of discrete grades of service from which customers may select based on the level of quality of protection [17] appropriate for their traffic. Tiered structures may be also be employed when the offered service is not characterized by bandwidth but by amount of traffic generated. As an example, in early 2008, Time-Warner, a major cable ISP in the United States, started a pilot program in Beaumont, Texas, under which it charges customers based on how much data they transfer (i.e., upload *and* download) [22], [23]. For the pilot program, Time Warner put in place an exponential structure with tiers at 5 GB, 10 GB, 20 GB, and 40 GB of monthly traffic.

B. Contributions and Organization

Recently [28], we developed a theoretical framework based on location theory for selecting the service tiers optimally, eliminating the need for guesswork and *ad hoc* approaches. Our work was based on the observation that, by mapping a user to the next higher offered service level, a tiered-service network may use more resources than a continuous-rate one to satisfy the same set of requests for service; alternatively, for a given amount of resources, a tiered-service network will be able to accommodate a smaller fraction of user requests than a continuous-rate one. Therefore, we considered the problem of selecting the service tiers to be offered so as to minimize the additional amount of resources required. We formulated the problem as a *directional p*-median problem, a variant of the well-known *p*-median problem [19], [33] under a new directional distance metric. We presented an optimal dynamic programming algorithm for a single service parameter (e.g., bandwidth), we showed that the problem of obtaining service tiers jointly optimal for multiple service parameters (e.g., bandwidth and delay) is NP-Complete, and we provided effective heuristics in the latter case. Our main finding was that a small set of optimal service levels is sufficient to approximate the resource usage of a continuous-rate network. In other words, the benefits of tiered service in terms of simplified network functions can be achieved with only a small sacrifice in network resources.

With this work, we make several new contributions to the problem of offering bandwidth tiered services. We present a new dynamic programming algorithm for determining the optimal service tiers whose complexity is linear in the number of users, and hence scales to very large numbers of users (a typical scenario); this represents a significant improvement over the complexity of the dynamic programming algorithm of [28] that is quadratic in the number of users. The new

algorithm is based on a novel dynamic programming formulation that exploits intrinsic properties of the problem. We also consider a new version of the service tier selection problem in which the objective is to optimize jointly the number of tiers *and* the magnitude of each tier; in contrast, the number of tiers is provided as an input parameter in the problem we considered in [28]. This problem is of practical significance since typically there is a cost incurred for each additional service tier offered. We then introduce the concept of “TDM emulation” that refers to a tiered-service network in which all service tiers are multiples of a basic bandwidth unit. A network operating with such a set of service levels would resemble a TDM network that allocates bandwidth in multiples of a slot. Consequently, many robust network management functions developed for telecommunication networks, including admission control, routing, traffic grooming, etc., could be easily adapted for the tiered-service packet-switched network. Finally, we develop efficient algorithms to select jointly the basic unit and service tiers in a near-optimal manner. Our work provides a systematic framework for reasoning about and tackling algorithmically the general problem of service tier selection, and has applications to a number of networking contexts, including access networks (e.g., determining the tiers for ADSL, cable modem networks or PONs) and core networks (e.g., LSP sizing for MPLS networks).

In Section II we provide some background on the directional p -median problem as a model for service tier selection, and discuss several applications. In Section III we develop a new algorithm of linear complexity for the problem, a significant improvement over our previous algorithm. In Section IV we study a new variant of the problem in which the number of service tiers is also a parameter to be optimized. In Section V we consider the problem of TDM emulation, which includes the additional constraint that all service levels be a multiple of the same bandwidth unit; we then develop a set of algorithms to select both the service levels and the bandwidth unit that are jointly optimal. We present numerical results in Section VI, and we conclude the paper in Section VII.

II. THE DIRECTIONAL p -MEDIAN PROBLEM (DPM1) AND ITS APPLICATIONS

For completeness, we now provide some background on the directional p -median problem and present the dynamic programming formulation we introduced in [28]. Our discussion follows closely that of [28].

We consider a packet-switched network with n users. Let x_i be the amount of bandwidth requested by user i . The network offers $p \geq 1$ levels (tiers) of service; typically, $p \ll n$. The j -th level of service corresponds to bandwidth z_j , $z_1 < z_2 < \dots < z_p$. In such a tiered-service network, each user i is mapped to service level z_j such that $z_{j-1} < x_i \leq z_j$. The additional bandwidth $z_j - x_i$ represents the performance penalty associated with the tiered service. Given the set $X = \{x_i\}$ of user requests and the number of service levels p , we are interested in finding the set of service levels $S = \{z_1, \dots, z_p\}$ which minimizes the performance penalty over all n users; we refer to such set as “optimal.” In the

following, we use terminology standard in discrete location literature, and refer to bandwidth requests as “demand points” and to service tiers as “supply points”.

The traditional p -median problem asks us to find, for a given set of n demand points on the real line, the set of p supply points that minimizes the total distance of each demand point to its nearest supply point. Let $d(x_i, z_j)$ be the distance from point x_i to point z_j on the line, according to some distance metric. The decision version of the p -median problem on the line may be formally stated as:

Problem 2.1 (PM1): Given a set $X = \{x_1, x_2, \dots, x_n\}$ of demand points, an integer p , and a bound B , does there exist a set $S = \{z_1, z_2, \dots, z_p\}$ of p supply points such that $\sum_{i=1}^n \min_{1 \leq j \leq p} \{d(x_i, z_j)\} \leq B$?

The choice of distance measure impacts the complexity of the problem as well as the approach needed to find a solution. An $O(np)$ algorithm for PM1 is given in [19], and it is known that the p -median problem is NP-complete in two or more dimensions under either the Euclidean or the rectilinear distance measure [33]. For a comprehensive treatment of location problems, the reader is referred to [11].

We now introduce the *directional* rectilinear distance measure. In general, a l -directional, k -dimensional rectilinear metric (with $1 \leq l \leq k$) defines distance from point (r_1, \dots, r_k) to (q_1, \dots, q_k) to be ∞ if $r_i > q_i$ for some $i \in \{1, \dots, l\}$ and $\sum_{1 \leq i \leq k} |q_i - r_i|$ otherwise. Thus, in a directional p -median problem, a supply point must achieve or exceed the values of the first l coordinates of all the demand points assigned to it. On the real line, this restriction requires that the nearest supply point for a given demand point be located to the right of it, hence, the 1-directional rectilinear distance is:

$$d_{dr}(x_i, z_j) = \begin{cases} z_j - x_i, & z_j \geq x_i \\ \infty, & \text{otherwise} \end{cases} \quad (3)$$

Let $X = \{x_1, \dots, x_n\}$ be a set of n demand points on the real line, such that $x_1 \leq x_2 \leq \dots \leq x_n$, and define the *density* of X to be $\rho_X = \sum_{i=1}^n x_i$. A set of supply points $S = \{z_1, \dots, z_p\}$, $z_1 < z_2 < \dots < z_p$, $1 \leq p \leq n$, is a *feasible solution* for X if and only if $x_n \leq z_p$. Associated with a feasible solution is an *implied mapping* $X \rightarrow S$, where $x_i \rightarrow z_j$ if and only if

$$z_{j-1} < x_i \leq z_j. \quad (4)$$

Figure 1 shows a sample mapping from a set of 13 demand points onto a set of 6 supply points. Let n_j be the number of demand points mapped to supply point z_j . The directional p -median problem on the real line (which we will refer to as problem DPM1) is:

Problem 2.2 (DPM1): Given a set X of n demand points, $x_1 \leq x_2 \leq \dots \leq x_n$, find a feasible set S of p supply points, $z_1 < z_2 < \dots < z_p$, $1 \leq p \leq n$, which minimizes the following objective function:

$$Obj(S) = \sum_{j=1}^p (n_j z_j) - \rho_X = \Psi(X, p) - \rho_X \quad (5)$$

The density ρ_X is the amount of load requested by the original set of demand points, while the term $\Psi(X, p)$ is the load assigned to the users under the tiered service. Hence, $Obj(S)$

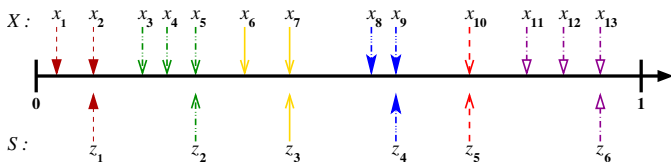


Fig. 1. Sample mapping of demand points x_i to supply points z_j

is the amount of excess bandwidth needed by the tiered-service network to accommodate the demand set X after mapping it to the supply set S .

We note that algorithms for the classical p -median problem cannot be applied directly to the directional variant DPM1, since such algorithms do not obey the directionality constraints (4) and hence may assign a demand point x_i to a supply point $z_{j-1} < x_i$ if their distance is smaller than that between x_i and $z_j > x_i$.

It is straightforward to show [28] that, given a set X of n demand points, there exists an optimal supply set $S = \{z_1, \dots, z_p\}$ such that $z_j \in X$, for each $j = 1, \dots, p$. Based on this observation and the fact that for a given demand set X the density ρ_X is constant, it is possible to solve DPM1 by using the following dynamic programming algorithm to compute $\Psi(X, p)$ recursively, where $X_k = \bigcup_{i=1}^k \{x_i\}$, $k = 1, 2, \dots, n$, is the set with the k smallest demand points in X :

$$\Psi(X_1, l) = x_1, \quad l = 1, \dots, p \quad (6)$$

$$\Psi(X_k, 1) = kx_k, \quad k = 1, \dots, n \quad (7)$$

$$\Psi(X_k, l+1) = \min_{q=l, \dots, k-1} \{\Psi(X_q, l) + (k-q)x_k\} \\ l = 1, \dots, p-1, k = 2, \dots, n \quad (8)$$

Expression (6) states that if there is only one demand point, it is the optimal supply point. Expression (7) is due to the fact that when $p = 1$, the optimal supply point is equal to the largest demand point. The recursive expression (8) can be explained by noting that the $(l+1)$ -th supply point must be equal to the demand point x_k . If the l -th supply point is equal to x_q , $q = l, \dots, k-1$, the tiered-service load is given by the expression in brackets in the right-hand side of (8), since $k-q$ demand points are mapped to supply point x_q . Taking the minimum over all values of q provides the optimal value.

The running time complexity of the above dynamic programming algorithm is $O(pn^2)$. Experimental results with this algorithm that can be found in [28] demonstrate that by using 10-20 optimal service levels (supply points), a tiered-service network will use no more than 5-8% bandwidth resources beyond the amount requested. More importantly, our experiments indicate that this result is valid across a wide range of distributions of user requests.

A. Applications

The p -median problem arises in domains where the objective is to (1) group elements described by a vector of characteristics, and (2) assign a representative to each group. It was originally conceived in the context of facility location (where the representative element is a facility that serves all the elements of the corresponding group), but it has also

been applied to statistical cluster analysis (where elements in the same group are somehow more similar than elements in different groups) and data or vector compression (where the representative is used to describe the whole group).

The directional p -median problem similarly arises in clustering applications in which the representative of each group must be such that for certain characteristics, its values exceed those of the other members of the group (i.e., by imposing directionality constraints similar to (4)). Such applications emerge in modern networking and computing environments where catering to very large sets of heterogeneous users/demands poses significant scalability challenges. To overcome these challenges, it is often desirable to arrange users with similar service requirements in a small number of groups, and treat all users within a given group identically by providing each with the same level of service. To ensure that all users receive at least as good a service as they have requested, the service level of a group is determined by the requirements of the most demanding user in that group; hence the directionality constraints in the form of (4) arise naturally in such a setting. A practical and effective approach to clustering users in such groups is for the provider to offer a tiered service, i.e., make available only a small set of service levels (tiers) to which customers may subscribe, thus avoiding the complexities associated with supporting the potentially infinite number of service levels that users may request.

Tiered service, and hence the directional p -median problem in its several variants, is useful in several contexts, including, but not limited to:

- *Broadband Internet Access.*

As we discussed earlier, most ISPs already offer some form of tiered Internet access service, along with a corresponding multi-tiered price structure that is either capacity-based or usage-sensitive. In most cases, the tiering is either *ad-hoc* or based on simple exponential structures. In the sections that follow, we develop a formal framework for optimizing the service tier structure.

- *Dimensioning of MPLS Tunnels for Virtual Private Networks (VPNs).*

A virtual private network (VPN) is a network that uses a public telecommunication infrastructure, such as a carrier's network, to connect various remote sites of an organization to each other in a secure and efficient manner. Traffic among the various sites is *tunneled* through the public network using various tunneling technologies. Multi-protocol label switching (MPLS) [5], [12], [16], [37] is a widely-used tunneling technology that forwards traffic over what are referred to as label switched paths (LSP) [4].

Typically, the size of (i.e., bandwidth associated with) an LSP (equivalently, VPN tunnel) connecting two remote sites is directly related to the traffic demands between the two sites. In principle, the bandwidth of an LSP may take any value up to the capacity of the physical links over which the LSP is routed. From a traffic engineering point of view, however, supporting LSPs of arbitrary size can be a challenge, and may lead to fragmentation of link capacity, hence low utilization of the underlying

network resources. Alternatively, the VPN provider may offer a tiered service in which clients may select from a small, appropriately selected set of LSP sizes (bandwidth levels) [38], the one that best suits their needs. In addition to the benefits regarding the operation and management of the network, such a tiered service also simplifies what in MPLS parlance is referred to as *LSP resizing*, since a client can easily upgrade to the next higher tier whenever its traffic exceeds the current allocation.

- *Efficient Packet Fair Scheduling.*

In packet-switched networks, the scheduling algorithm is central to the QoS architecture. Timestamp-based schedulers, such as weighted fair queueing (WFQ) [35] and its variants, may be employed to allocate the bandwidth fairly among competing flows, as well as to ensure bounded delay under certain conditions. The main drawback of such schedulers is their high complexity, which makes it difficult to implement in hardware so as to operate at wire speeds. The high complexity is due, to a certain extent, to the fact that these schedulers are designed to accommodate packet flows with *arbitrary* weights (equivalently, bandwidth shares). Consequently, recent schedulers such as stratified round robin [36] reduce the complexity by arranging flows of arbitrary weights into classes using exponential grouping.

In [29], [39] we have developed and implemented a new class of schedulers that capitalize on tiered service (as well as certain properties of Internet traffic), to realize the properties of packet fair scheduling with low complexity that is amenable to hardware implementation. In this context, tiered service implies that flows cannot request arbitrary weights (bandwidth shares), but must select among a small number of pre-determined weights (i.e., tiers). The directionality constraints in this case ensure that each flow subscribes to a tier that offers a share of the bandwidth at least as high as the flow requires.

- *Task Scheduling in Multiprocessor, Cluster, or Grid Systems.*

Multiprocessor facilities, computational clusters, and Grid systems allow multiple users to share a pool of high-performance computational resources. If such a system provides only a small set of service levels, the scheduling, management, and handling of dynamic task requests can be simplified significantly. Consider, for instance the preemptive scheduling of a set of n periodic tasks on m identical processors [27], [31], [32]. A periodic task is made up of subtasks, each of length one, which have to be scheduled at regular intervals. In the discrete-time version of the problem, time is assumed to be slotted, and each task has a density ρ_i , $0 < \rho_i < 1$, that represents the task's demand for processing time, in terms of subtasks per slot.

A feasible schedule for this problem exists if and only if $\sum_{i=1}^n \rho_i \leq m$, i.e., the total demand does not exceed the capacity of the m -processor system. A *proportionally fair* schedule [7] closely mimics the ideal fluid system in which both time and the subtasks are infinitely divisible. The fastest algorithm for constructing a proportionally

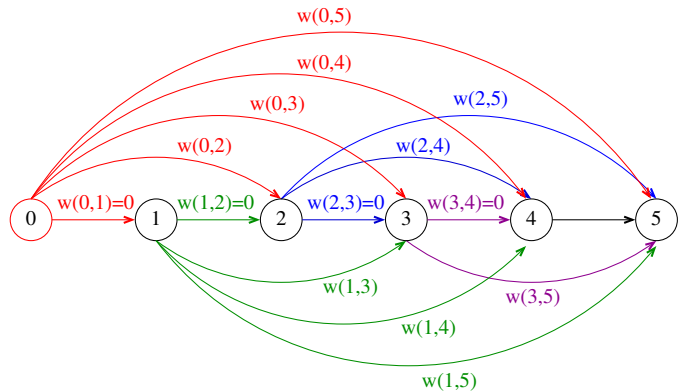


Fig. 2. DAG representation of an instance of the directional p -median problem with $n = 5$

fair schedule whenever one exists takes time $O(m \log n)$ per slot [3], hence it cannot scale to systems with a large number n of tasks (users). On the other hand, quantizing the task densities ρ_i to a small set of service tiers makes it possible to reduce the complexity of the proportionally fair algorithm to $O(m)$ [26], that is *independent* of the user population, and for a given system (i.e., a given number m of processors), it is constant.

III. AN $O(pn)$ ALGORITHM FOR DPM1

We now show how to exploit a property of DPM1 to develop an $O(pn)$ optimal algorithm which scales well to large problem instances with demand sets of size n in the order of hundreds of thousands and beyond. To this end, we restate DPM1 as a constrained shortest path problem. Let $G = (V, E)$ be a weighted, complete, directed acyclic graph (DAG), with vertex set $V = \{0, 1, \dots, n\}$. In the DAG representation of DPM1, demand x_i gives rise to vertex i , and we create another vertex 0. Arc weight $w(i, k)$ represents the cost of mapping demand points x_{i+1}, \dots, x_k , to point x_k :

$$w(i, k) = \begin{cases} 0, & k = i + 1 \\ (k - i - 1)x_k - \sum_{j=i+1}^{k-1} x_j, & k > i + 1 \end{cases} \quad (9)$$

As an example, Figure 2 shows the graph for a DPM1 instance with $n = 5$. It is not difficult to prove the following lemma:

Problem 3.1: Solving an instance of DPM1 with n demand points is equivalent to finding a minimum weight p -link path from vertex 0 to vertex n in the corresponding DAG.

A weighted DAG satisfies the concave Monge condition if

$$w(i, j) + w(i + 1, j + 1) \leq w(i, j + 1) + w(i + 1, j) \quad (10)$$

holds for all $0 < i + 1 < j < n$. By substituting (9) into (10), it can be shown that the DAG representing any instance of DPM1 obeys the concave Monge condition.

Consider a matrix M of real elements, and let $I(t)$ denote the index of the leftmost column containing the maximum value in row t of M . Matrix M is said to be *monotone* if

$$t_1 > t_2 \Rightarrow I(t_1) \geq I(t_2), \quad \forall t_1, t_2. \quad (11)$$

Matrix M is said to be *totally monotone* if all its sub-matrices are monotone [1]. It has been shown [2] that a 2-dimensional Monge array is totally monotone.

The work in [1] presents an algorithm that can find the minimum entry in each column of a totally monotone $n \times m$ matrix, $n \geq m$, in $\Theta(n)$ time. This elegant matrix searching algorithm has many geometric applications, and we show next how it can be applied to obtain a faster optimal algorithm for DPM1. For the details of the matrix searching algorithm, the reader is referred to [1], [6].

The faster algorithm for DPM1 is based on the following dynamic programming formulation to obtain the optimal value of the objective function $Obj(S)$ in (5):¹

$$F(1, l) = 0, \quad l = 1, \dots, p \quad (12)$$

$$F(k, 1) = w(0, k), \quad k = 1, \dots, n \quad (13)$$

$$F(k, l+1) = \min_{q=l, \dots, k-1} \{F(q, l) + w(q+1, k)\} \\ l = 1, \dots, p-1, k = 2, \dots, n \quad (14)$$

where $w(i, k)$ are the arc weights of the DAG corresponding to this instance of DPM1, as defined in (9). The optimal value for the objective function $Obj(S)$ in (5) is obtained as the value of $F(n, p)$. Expressions (12)-(14) correspond to (6)-(8), respectively, and can be explained in a similar manner.

Our objective is to obtain the value of $F(n, p)$ by computing all the elements of each column l of the matrix defined by F in $O(n)$ time. Note that for $l = 1$, the elements of the first column can be computed in $O(n)$ time from expressions (13) and (9). Therefore, we concentrate on computing expression (14) efficiently. To this end, we introduce a new function $\Gamma(q, k)$:

$$\Gamma(q, k) = F(q, l-1) + w(q+1, k), \quad q, k = 1, \dots, n \quad (15)$$

We can now see that filling out the l -th column defined by matrix F , i.e., computing the n elements $F(k, l+1)$, $k = 1, \dots, n$, from expression (14) is equivalent to finding the minimum elements in each column of the $n \times n$ matrix defined by function $\Gamma(q, k)$. Also, $\Gamma(q, k)$ depends on the values of the elements of the $(l-1)$ -th column of the matrix defined by F , which have already been calculated.

We now show that the function $\Gamma(q, k)$ obeys the concave Monge condition. From (10) we know that

$$w(q+1, k) + w(q+2, k+1) \leq w(q+1, k+1) + w(q+2, k) \quad (16)$$

If we add the term $F(q, l-1) + F(q+1, l-1)$ to both sides of (16) and use the definition of $\Gamma(q, k)$ in (15), we get:

$$\Gamma(q, k) + \Gamma(q+1, k+1) \leq \Gamma(q, k+1) + \Gamma(q+1, j) \quad (17)$$

Since $\Gamma(q, k)$ obeys the concave Monge condition, the matrix represented by $\Gamma(q, k)$ is totally monotone [2].

Based on the above observations, in order to solve the dynamic programming algorithm (12)-(14) we proceed by filling the $n \times p$ matrix defined by function $F(k, l)$ one column at a time. The first column ($l = 1$) is filled in $O(n)$ time using expression (13). In order to compute the n elements of the l -th column, $l = 2, \dots, p$, we use expression (15) to form an $n \times n$ totally monotone matrix containing $\Gamma(q, k)$ values that depend on the values of the $(l-1)$ -th column of the matrix F .

¹In contrast, recall that the dynamic programming formulation (6)-(8) was used to compute the optimal value of the term $\Psi(X, p)$ in (5).

The minimum elements in each column of the $n \times n$ matrix Γ are the n elements needed to fill the l -th column of matrix F . These elements can be obtained in $O(n)$ time using the algorithm in [1] we discussed in the previous section. Hence, the time to fill all p columns of matrix F , i.e., the time to find the optimal value for the objective function (5), is $O(np)$.

However, there remains one important issue that we need to address. The $O(n)$ algorithm in [1] assumes that the totally monotone matrix Γ is provided as input, since building this matrix would take time $O(n^2)$. In our case, the matrix Γ is not provided but has to be built anew for computing each column of matrix F . Rather than actually building the matrix in time $O(n^2)$, we now show how to evaluate the value of each element $\Gamma(q, k)$ in constant time. Whenever the algorithm in [1] needs to use the value of some element $\Gamma(q, k)$, rather than accessing the value from memory, we compute its value in constant time. By replacing one constant-time operation (memory access) with another (computing the value), we ensure that the algorithm runs in $O(n)$ time.

From (15) we see that element $\Gamma(q, k)$ depends on the values of $F(q, l-1)$ and $w(q+1, k)$. The value of $F(q, l-1)$ is already computed and hence can be accessed in constant time. In order to evaluate $w(q+1, k)$ in constant time, we perform the following preprocessing operation before starting to solve the dynamic programming algorithm. Define, for $q = 1, \dots, n$, $A(q) = \sum_{i=1}^q x_i$. It takes time $O(n)$ to compute and store the values $A(q)$ for all q . Once this is done, one can compute the value of $w(q, k)$ in constant time using the expression:

$$w(q, k) = -A(k) + A(q-1) + (k-q+1)x_k, \quad q \leq k \quad (18)$$

Hence the value of $\Gamma(q, k)$ can be calculated in constant time for any two values q and k .

IV. JOINT OPTIMIZATION OF THE NUMBER AND MAGNITUDE OF SERVICE TIERS

In the above definition of DPM1, the number p of service points is provided as an input parameter and there is no cost associated with a supply point. In a practical system, there may be a cost (e.g., additional hardware and/or software complexity) involved in providing more service tiers. In this case, it is desirable to jointly optimize both the number p of supply points and their placement, so as to strike a balance between the bandwidth penalty due to quantization and the cost of establishing additional supply points.

Let us assume that there is a cost associated with supporting a given number of service tiers, e.g., in terms of the relevant software and/or hardware mechanisms that would need to be implemented at the routers. Specifically, we assume that the *incremental cost* (e.g., due to additional queueing structures, policing mechanisms, control plane support, etc.) of offering one additional service tier is equal to α . Hence, the total cost for p tiers is αp .

The problem of jointly optimizing the number and magnitude of service tiers, a variant of DPM1 which we will refer to as JDPM1, can be expressed as:

Problem 4.1 (JDPM1): Given a set X of n demand points, $x_1 \leq x_2 \leq \dots \leq x_n$, find an integer $p \leq n$ and a feasible set

S of p supply points, $z_1 < z_2 < \dots < z_p$, $1 \leq p \leq n$, which minimizes the following objective function:

$$J\text{-Obj}(S) = \left(\sum_{j=1}^p (n_j z_j) - \rho_X \right) + \alpha p = \text{Obj}(S) + \alpha p \quad (19)$$

We observe that the two terms in the right hand side of (19) behave differently as a function of $p = 1, \dots, n$. Specifically, as p increases, the first term that represents the error due to quantization decreases, but the second term increases linearly with p . Hence, the optimal number of supply points will depend on the cost α . However, given a fixed number p of supply points, the optimal *placement* of these points is independent of the value of α (since for a fixed p the second term in (19) becomes constant).

Based on the above observations, one straightforward approach to tackling this joint optimization problem would be to run the new dynamic programming algorithm for DPM1 in Section III n times, once for each value of $p = 1, \dots, n$. Since the running time of the algorithm for a fixed value of p is $O(pn)$, the running time complexity of this approach is $O(n^3)$. However, we have been able to adapt the dynamic programming algorithm for DPM1 to this problem by making small modifications to the boundary conditions and the cost of a solution to account for the additional term in the objective function. This modified algorithm runs in time $O(n^2)$, since it has to examine all n possible values for the number of supply points. In fact, the algorithm can be extended to handle general non-decreasing cost functions in p , not just the linear cost function αp in (19).

Figure 3 plots the objective function (19) against the number p of supply points for the uniform distribution (refer to Table I) and $n = 1000$ demand points. The results shown are representative of the behavior we have observed for the other distributions in Table I and other values of n . Two curves are plotted, each corresponding to a different value of the cost α ($= 1, 10$, respectively) for establishing an additional supply point. As we can see, the exact shape of the curves depends on the value of α , but both exhibit the same trend. Specifically, for $p = 1$, the objective function is equal to $nd_n + \alpha$, where d_n is the largest demand point; since the demand points are generated from a uniform distribution in $(0, 1)$, this value is approximately $n + \alpha$. For the selected values of α , the first term of the objective function dominates in the region where p is small. As p increases, initially the objective function decreases, reflecting a decrease in the dominant first term, until a minimum is reached; the optimal value of p at which the minimum is reached is different for each curve, as it depends on the value of α . Further increases of p beyond this optimal value result in an increase in the overall objective value, reflecting the fact that the second term αp becomes dominant for large p .

V. TDM EMULATION

Many existing networks operate over an infrastructure that is based on time division multiplexing (TDM); examples include the legacy telephone network, the synchronous optical network/synchronous digital hierarchy (SONET/SDH) transport architecture, and time division multiple access wireless

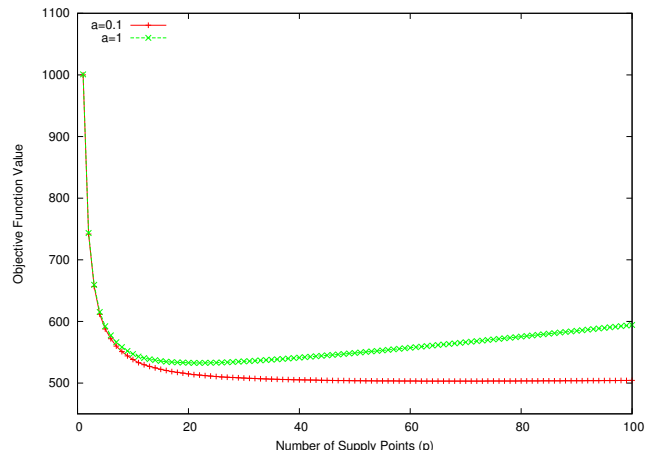


Fig. 3. Joint optimization of the number and placement of supply points, uniform distribution, $n = 1000$, $\alpha = 1, 10$

communications networks such as those employing the global system for mobile communications (GSM) standard. Whereas networks based on TDM technology today offer a multitude of data services in addition to classical voice service, it is often convenient for providers to allocate bandwidth to users in multiples of a unit rate that is typically tied to the slot size of the underlying transmission system. Legacy telephone networks and SONET/SDH networks, in particular, impose a rigid hierarchy of rates that are multiples of a voice channel (i.e., 64 Kbps), hence there is a proliferation of data services with rates ranging from DS-1 (1.5 Mbps) and DS-3 (45 Mbps) to STS-3 (155 Mbps) and STS-48 (2.5 Gbps) and beyond. These services are implemented by mapping a user's payload directly onto specific time slots of the transport system, e.g., using the generic framing procedure (GFP) [8], [21], [24].

In this section we introduce the concept of *TDM emulation* to describe a tiered-service network that allocates bandwidth in *arbitrary* multiples of a basic bandwidth unit (data rate). A packet-switched network operating with such a set of service levels would resemble a TDM network. Consequently, many robust network management functions developed for telecommunications networks, including admission control, routing, traffic engineering and grooming, etc., could be easily adapted for the tiered-service packet-switched network. The key premise of TDM emulation is that both the bandwidth unit and the service tiers be *configurable* (under software control) rather than fixed by the intrinsic structure of the underlying transport architecture as is the case with classical SONET/SDH networks. TDM emulation can be useful in a wide range of networking contexts and applications, including, but not limited to:

- *Bandwidth allocation in native packet-switched networks.* We emphasize that, as defined here, TDM emulation only affects the way that bandwidth is allocated to network users, *not* the data plane operation of packet-switched networks, e.g., those employing 1 or 10 Gigabit Ethernet (GE) links. For example, while bandwidth is allocated in multiples of the basic unit, users (flows) are not limited to using a particular slot. Similarly, unlike TDM networks where an unused slot is wasted, excess bandwidth can be

allocated to active flows by the scheduling algorithm. Furthermore, the bandwidth unit is not fixed or determined by hardware, as in a TDM network, but, it is configurable and can be optimized for the characteristics of the carried traffic. In addition, the routers provide for free the functionality of a time-slot interchange. Overall, with TDM emulation, tiered-service packet-switched networks may enjoy many of the benefits, in terms of control and management, of a TDM network, but without the data plane rigidities of such a network.

- *Flexible bandwidth allocation in next generation SONET/SDH networks.* The emergence of framer technologies such as virtual concatenation (VCAT) [9] coupled with the link capacity-adjustment scheme (LCAS) [25] enable providers to make significantly more efficient use of the existing SONET/SDH infrastructure [10], [13], [20], [25]. Virtual concatenation allows finer granularity for provisioning of bandwidth services, making it possible to configure service tiers in any multiple of 64 Kbps. LCAS, on the other hand, offers a means to reconfigure (i.e., enlarge or shrink) dynamically the size of a SONET/SDH data pipe without impacting the transported data, permitting users to move between service tiers as their requirements evolve.
- *Traffic grooming.* Traffic grooming [14] refers to techniques used to combine low-speed traffic streams for transport over high-speed wavelengths so as to minimize the network cost in terms of line terminating equipment and/or electronic switching [15], [34], [42]. The traffic grooming problem has been investigated extensively in the literature [40], [41], but most studies make the assumption that traffic demands are multiple of a basic unit, e.g., STS-3 for an OC-48 link; this assumption is a reflection of the dominance of SONET/SDH in the transport network infrastructure. Packet networks employing TDM emulation (e.g., MPLS networks with LSP sizes that are multiples of a basic unit) may employ these traffic grooming solutions without any modifications, and thus leverage the vast body of research that already exists.

A. TDM Emulation As A Constrained DPM1 Problem

In a tiered-service network with TDM emulation, all service tiers are multiples of some quantity r that represents the unit bandwidth, i.e., the smallest amount in which bandwidth may be allocated. In packet-switched networks, r is itself a configurable parameter, and is not tied to any underlying, hardware-imposed slot structure. Therefore, the objective in such a network is to select *jointly* the unit rate r and the service tiers in some optimal manner. To this end, in this section we pose the optimization problem arising in TDM emulated networks as a DPM1 problem with an additional constraint on the values of the optimal supply points.

Figure 4 shows a sample mapping from a set X of 13 demand points onto a set S of 6 supply points, under this constraint. The set X is identical to the one in Figure 1 which shows a sample mapping under DPM1. The main difference is that with the new constraint, the supply points are all multiples of the same unit r .

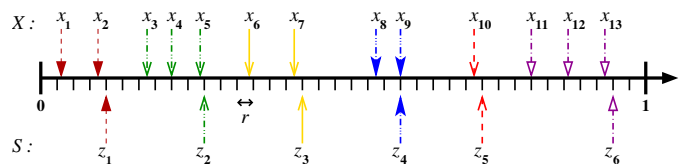


Fig. 4. Sample mapping of demand points x_i to supply points z_j that are multiples of a basic unit r

This constrained directional p -median problem, which we will refer to as TDM-DPM1, can be expressed as follows (recall that n_j is the number of demand points mapped to supply point z_j , $\rho_X = \sum_{i=1}^n x_i$, and $Obj(S)$ is the objective function (5) of the DPM1 problem):

Problem 5.1 (TDM-DPM1): Given a set X of n demand points, $x_1 \leq x_2 \leq \dots \leq x_n$, and a constant C , find a real r and a feasible set S of p supply points, $z_1 < z_2 < \dots < z_p$, $1 \leq p \leq n$, so as to minimize the objective function:

$$TDM-Obj(S) = \sum_{j=1}^p (n_j z_j) - \rho_X + \frac{C}{r} = Obj(S) + \frac{C}{r} \quad (20)$$

under the constraints: $z_j = rk_j$, k_j : integer, $j = 1, \dots, p$.

The term $Obj(S)$ in (20) represents the excess bandwidth penalty, as before. However, the objective function for TDM-DPM1 includes the additional term $\frac{C}{r}$, where C is some constant related to the operation of the system, as we explain shortly. The presence of a term which is a monotonically decreasing function of r in (20) is necessary, since without it TDM-DPM1 reduces to DPM1: if nothing prevents r from being very small, then the optimal is obtained for $r = 1$ bps as the solution to DPM1 which minimizes the excess bandwidth penalty.

More importantly, the term $\frac{C}{r}$ is of practical importance as it captures the overhead associated with making the unit r of bandwidth allocation small. To illustrate, let us make the simplifying assumption that all users request and receive the basic rate of r bits/sec. After serving a user, the system incurs some overhead due to the bookkeeping operations, memory lookups, etc., required before it can switch to serving another user. Let β denote the amount of time required to switch between users, expressed as the number of bits that could be transmitted during this time at the given service rate. Therefore, the quantity $\frac{\beta}{r}$ represents the amount of overhead operations relative to the bandwidth unit. This relative overhead, which increases as the unit of bandwidth decreases, is similar in principle to the ‘‘cell tax’’ incurred in carrying IP traffic over ATM networks due to the relatively large fraction of header (i.e., overhead) bits to data bits. In the objective function (20) we use the term $\frac{C}{r}$ where $C = c\beta$ and c is a constant which ensures that the two terms in the rightmost side of (20) are expressed in the same units.

We note that the term $Obj(S)$ in (20) requires that the unit r be small so as to minimize the excess bandwidth. However, making r small would increase the term $\frac{C}{r}$ which represents the bandwidth wasted due to overhead operations. Therefore, the objective of TDM-DPM1 is to determine the value of r so as to strike a balance between these two conflicting objectives.

B. Optimal Solution to TDM-DPM1 for Fixed r

As defined, the objective of TDM-DPM1 is to find jointly optimal values for the basic bandwidth unit r (a real number) and the p supply points. However, let us consider for a moment the special case where the value of r is fixed and not subject to optimization; as we shall see shortly, the algorithm for this problem is useful in tackling the general one. In this case, the term $\frac{C}{r}$ in (20) is constant and does not affect the minimization. Hence, the objective function is identical to that of the DPM1 problem.

Consider an instance of TDM-DPM1 in which the value of the basic bandwidth unit is fixed at $r = r_0$; that is, the p supply points can only take the values $kr_0, k = 1, \dots, K$. Let $U = \{u_1, \dots, u_K\}$ be the set of candidate values for the p supply points, $u_k = kr_0$; in Figure 4, these candidate values are represented by the ticks below the horizontal line. Integer K corresponds to the largest possible multiple of r_0 , i.e., $K = \lceil \frac{x_n}{r_0} \rceil$, where x_n is the largest demand point.

This version of TDM-DPM1 can be represented by a DAG similar to the one in Figure 2. In this case, the DAG has $K + 1$, rather than $n + 1$, vertices: vertex 0 and the K vertices corresponding to the K candidate values for the supply points (recall that in DPM1, the candidate supply points are the n demand points). Similarly, the arc weight $w(i, k)$ in this DAG represents the cost of mapping the demand points with values between candidate supply points u_i and u_k to u_k :

$$w(i, k) = \sum_{u_i < x_j \leq u_k} (u_k - x_j) \quad (21)$$

It is also not difficult to verify that these weights satisfy the concave Monge condition (10) for all $0 < i + 1 < j < K$.

Since this version of TDM-DPM1 has the same objective function as DPM1 and can be represented by a DAG whose weights satisfy the concave Monge condition, we can solve it optimally using the dynamic programming algorithm in Section III. Note that the algorithm will run in $O(pK)$, not $O(pn)$, time, as it has to consider K candidates for the p supply points.

C. The Behavior Of The TDM-DPM1 Objective Function

To obtain insight into how the additional parameter r affects the optimization, let us investigate the behavior of the objective function $CObj$ in (20) as we vary r . In Figure 5 we plot the objective function against the value of r for an instance of TDM-DPM1 with $n = 1000$ demand points, $p = 10$ supply points, and $C = 0.01$, with the set of n demand points generated from a uniform distribution in $(0, 1)$. We varied the value of the basic unit r in increments of $\delta_r = 10^{-5}$ across the range shown in the figure. For each (fixed) value of r we obtained the optimal supply points in the manner we described in the previous subsection, from which we evaluated the objective function (20), including the term $\frac{C}{r}$.

The behavior exhibited in Figure 5 is representative of the TDM-DPM1 instances we have studied. At low values of r , the term $\frac{C}{r}$ representing the overhead cost dominates, resulting in large overall values. As r increases, there is an initial period of rapid decrease in the objective function as

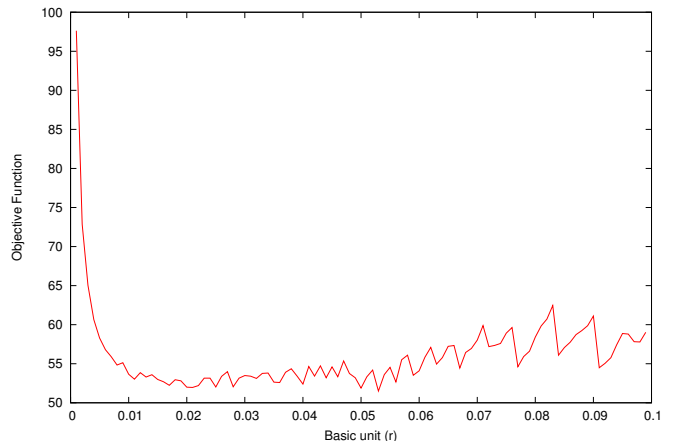


Fig. 5. Objective function value against r , $n = 1000$, $p = 10$, $C = 0.01$, demand points generated from a uniform distribution in $(0, 1)$

the term representing the excess bandwidth penalty starts to become important. Following this initial decrease, the curve settles into a seesaw pattern. The high and low points along this pattern depend on the values of the multiples of r relative to the demand points: when multiples of r are aligned close to demand points, there is little bandwidth penalty for mapping these demand points to supply points that are multiples of r , hence the objective function has a lower value; the opposite is true when there is a mismatch between multiples of r and demand points. We also note that as the value of r increases further, the curve trends upwards. This behavior is due to two factors that come into play when r becomes large: the excess bandwidth term in (20) starts to dominate, and at the same time this term increases in value as large values of r are too coarse to minimize the excess bandwidth.

It is clear from Figure 5 that the objective function is non-convex and includes several troughs at irregular intervals. This non-convex nature makes standard optimization techniques (e.g., steepest descent methods) impractical, as they are likely to get trapped in a local minima. We now describe an exhaustive search approach for identifying the value of r and the supply points that minimize the objective function, and in the next section we develop a suite of heuristics that trade solution quality for running time.

We first observe that for a TDM-DPM1 instance with p supply points and x_n the largest demand point, the largest value that r may take under the constraint that all p supply points be an integer multiple of r is $r_{max} = \frac{x_n}{p}$. Hence, the optimal value of r lies in the interval $(0, r_{max}]$. Let δ_r be a small increment value, and consider the set $R = \{r_m = m\delta_r \leq r_{max}, m = 1, 2, \dots\}$. For each (fixed) $r_m \in R$, we use the approach in Section V-B to obtain the optimal supply points, and evaluate the objective function (20). The optimal solution to the TDM-DPM1 problem is obtained as the value of r_m and the corresponding supply points which produce the smallest value for the objective function.

In order to determine the running time complexity of this exhaustive search algorithm, let $L = \lfloor \frac{r_{max}}{\delta_r} \rfloor$ be the size of set R (i.e., the number of candidate values of r to be considered),

and $K_m = \frac{x_n}{r_m}$ be the number of candidate supply points when the value of $r = r_m$. The dynamic programming algorithm will be run L times, and during the m -th iteration, i.e., when $r = r_m$, the algorithm will take $O(pK_m)$ time. Since

$$\sum_{m=1}^L pK_m = \frac{px_n}{\delta_r} \sum_{m=1}^L \frac{1}{m} = \frac{px_n}{\delta_r} (\ln L + \gamma) \quad (22)$$

where $\gamma = 0.577\dots$, is Euler's constant, the complexity of the algorithm is $O(\frac{px_n}{\delta_r} \ln(\frac{x_n}{p\delta_r}))$.

As we can see, the complexity of the exhaustive search depends critically on the value of the increment δ_r which determines the granularity of the search. With finer granularity (i.e., smaller δ_r), the accuracy of the algorithm increases, but its complexity also increases dramatically; the opposite is true when δ_r becomes larger and the granularity coarser. We also note that the time complexity is independent of the number n of demand points, and depends only on the largest demand x_n . In the applications we consider, the input value x_n is bounded above by the bandwidth available on the highest capacity link in the network. To get a sense of the values involved in expression (22), consider a network with 10 Gbps links. A reasonable value for the bandwidth increment is $\delta_r = 64$ Kbps. Assuming that the largest demand can be equal to the capacity of a link, we have that $\frac{x_n}{\delta_r} \approx 10^6$, which demonstrates that the exhaustive search is taxing in terms of both computational and memory requirements.

D. Optimization Heuristics

We now present a set of heuristics for the TDM-DPM1 problem. Each heuristic trades solution quality for speed by using its own approach to reduce the size of the space of candidate values for r and/or the supply points that it considers.

1) *Demand Driven Heuristic (DDH)*: Recall that, for each candidate value r_m of r , the exhaustive search algorithm considers all the $K_m = \frac{x_n}{r_m}$ multiples of r_m as the set of potential supply points, where K_m can be much larger than the number n of demand points. The intuition behind this heuristic is that the optimal supply points are more likely to be located just above a demand point, since otherwise there would be a larger penalty in terms of excess bandwidth. Therefore, the heuristic only considers the n multiples of r_m that are located immediately to the right of (or coincide with) the n demand points. In other words, the set U of candidate values for the p supply points is $U = \{u_i = r_m \times \lceil \frac{x_i}{r_m} \rceil, i = 1, \dots, n\}$. Since there have to be n different candidate supply points, the range of values for r is in the interval $(0, \frac{x_n}{n} = r_{max}]$. Using n instead of K_m and the new value for r_{max} in expression (22), we find that the running time complexity of the DDH heuristic is $O(\frac{px_n}{\delta_r})$, which represents an improvement over the exhaustive algorithm, especially for small values of δ_r which allow for a finer granularity search.

2) *Supply Driven Heuristics*: Both the DDH and the exhaustive search algorithms apply the dynamic programming algorithm in Section III for each candidate value for parameter r . The two heuristics we present in this section are based

on the assumption that the optimal supply points for TDM-DPM1 are likely to be close to the optimal supply points for the corresponding unconstrained DPM1 problem with the same demand set. Therefore, each heuristic initially runs the dynamic programming algorithm for the corresponding DPM1 problem, and computes the optimal set $S^{DPM1} = \{z_1^{DPM1}, \dots, z_p^{DPM1}\}$ of supply points for that problem. This step takes time $O(pn)$, and this dynamic programming algorithm is not used again by the heuristics.

The first algorithm, which we call the *unidirectional supply driven heuristic (USDH)*, sets the i -th supply point for a given candidate value r_m of r to the smallest multiple of r_m that is greater than or equal to supply point z_i^{DPM1} . In other words, the set S'_m of supply points for candidate r_m is defined as $S'_m = \{\lceil \frac{z_i^{DPM1}}{r_m} \rceil r_m, i = 1, \dots, p\}$. The heuristic returns the value r_m and corresponding set S'_m which result in the minimum value for the objective function (20).

The second algorithm is called the *bidirectional supply driven heuristic (BSDH)*, and computes a set of $2p$ possible values for the supply points for each candidate value r_m . The first set of p values is identical to the set S'_m used by the USDH algorithm above. In addition, this heuristic considers the set S''_m consisting of the p largest multiples of r_m that are less than the corresponding supply points z_i^{DPM1} , i.e., $S''_m = \{\lfloor \frac{z_i^{DPM1}}{r_m} \rfloor r_m, i = 1, \dots, p\}$. The $2p$ elements of these two sets collectively become the candidates for being one of the p supply points when the value of $r = r_m$. We use the dynamic programming algorithm in Section V-B to select the optimal set of p supply points from the set $S'_m \cup S''_m$; it is easy to see that the dynamic programming algorithm works even when the set of the candidate supply points is a proper subset of the set of all integer multiples of r_m . As with USDH, the heuristic returns the value r_m and corresponding p supply points that minimize (20).

We expect the BSDH heuristic to perform better than USDH since it considers a larger number of candidate supply points; this improved performance, however, is at the expense of having to run the dynamic programming algorithm on a set of $2p$ points, which takes time $O(p^2)$.

3) *The Power of Two Heuristic (PTH)*: This heuristic simply selects the set of p supply points as the set of the p consecutive powers of two such that the largest element in the set is the smallest power of two that is larger than or equal to the largest demand point x_n ; in other words, $S = \{2^{q+1}, 2^{q+2}, \dots, 2^{q+p} \mid 2^{q+p-1} < x_n \leq 2^{q+p}\}$. This solution is consistent with TDM-DPM1 in that it consists of supply points all of which are a multiple of a basic unit, in this case 2^{q+1} . However, as we shall see shortly, the excess bandwidth penalty for this solution can be quite high compared to the other algorithms. We consider this solution here as a baseline case as it is similar in spirit to approaches that assign packet flows in classes (e.g., as in [36]) whose boundaries are defined by powers of two.

VI. NUMERICAL RESULTS

We now present simulation results to investigate the relative performance of the various algorithms we presented and to

TABLE I
FORMULAE FOR THE PDF AND CDF OF THE INPUT DISTRIBUTIONS

Distribution	pdf	cdf	Domain
Uniform	1	x	$0 \leq x \leq 1$
Increasing	$2x$	x^2	$0 \leq x \leq 1$
Decreasing	$-2x + 2$	$-x^2 + 2x$	$0 \leq x \leq 1$
Triangle	$4x$	$2x^2$	$0 \leq x < 0.5$
	$-4x + 4$	$-2x^2 + 4x - 1$	$0.5 \leq x \leq 1$
Unimodal	$4/9$	$4x/9$	$0 \leq x < 0.25$
	6	$6x - 25/18$	$0.25 \leq x < 0.35$
	$4/9$	$4x/9 + 5/9$	$0.35 \leq x \leq 1$
Bimodal	$1/4$	$x/4$	$0 \leq x < 0.25$
	4	$4x - 15/16$	$0.25 \leq x < 0.35$
	$1/4$	$x/4 + 3/8$	$0.35 \leq x < 0.65$
	1	$4x - 33/16$	$0.65 \leq x < 0.75$
	$1/4$	$x/4 + 3/4$	$0.75 \leq x \leq 1$

determine their effect on the operation of a network. The demand sets X of the problem instances we consider throughout this section were generated from one of six distributions whose pdf and cdf are listed in Table I. Note that in general, bandwidth demands will be in the range $(0, B]$, where B is the link capacity. However, in order to obtain results that are independent of the link capacity, we assume that all demands are normalized with respect to B ; thus, the domain of the pdf and cdf of all distributions in Table I is $[0,1]$. Also, we used an increment value $\delta_r = 10^{-5}$ whenever applicable. Given that we assume unit link capacity, this value of δ_r roughly corresponds to a SONET OC-192 link that allocates bandwidth in increments of 64 Kbps. Finally, we used $C = 0.5$ for the constant in the second term of the objective function (20) for all results presented in this section. This value of C ensures that the two terms of the objective function are dominant at different regions of the unit rate r (refer also to Figure 5), avoiding the special cases where only one term dominates over all values of r . A comprehensive set of results across a wide range of values for the various parameters are available in [6]. In this section we only present a small subset of results that are representative of the behavior we have observed, and which allows us to draw general conclusions regarding the relative performance of the various algorithms and the impact of tiered service on resource usage and network performance.

Algorithm comparison. Let us first investigate the relative performance of the various algorithms for TDM-DPM1 with respect to the objective function (20). Figure 6 plots the value of the objective function against the value of r for the stated problem instance, and for four TDM-DPM1 algorithms: DDH, BSDH, USDH, and PTH. We also show the optimal value for the corresponding DPM1 problem; this value does not include the overhead term $\frac{C}{r}$ of (20), hence it serves as a lower bound for the TDM-DPM1 algorithms. Note that the DPM1 and PTH solutions do not take parameter r into account, hence they are shown as horizontal lines in the figure. We see that PTH performs much worse than all other algorithms, confirming our earlier observation that using powers of two to define classes of traffic is not an efficient approach; this result is representative of the behavior of PTH, hence, we do not consider this heuristic in the remainder of this section. We also observe that the three algorithms DDH, BSDH, and

USDH perform close to the lower bound.

For the results shown in Figures 7 and 8, we have considered thirty problem instances with $n = 100$, $p = 5$, and $C = 0.05$, generated from the increasing and triangle distributions, respectively. The figures plot the objective function value returned by each of four algorithms, DDH, BSDH, USDH, and DPM1, for each problem instance; again, the DPM1 solution provides a lower bound for the other three algorithms. The graphs show that, except for a few instances, all three TDM-DPM1 algorithms are close to the lower bound. Of the three algorithms, DDH produces the lowest objective function values, followed closely by BSDH. The objective function values returned by USDH are generally higher than those of the DDH and BSDH heuristics, but USDH has a much faster running time. Hence, these results indicate that there is a tradeoff between quality of solution and running time complexity of the algorithms.

Bandwidth penalty due to tiered service. Let us now turn our attention to determining the penalty in terms of excess resources needed due to tiered service. Given a demand set X , a continuous-rate link will use an amount of bandwidth equal to $\rho_X = \sum_i x_i$ to satisfy all the demands in X . A link of a tiered-service network, on the other hand, will in general use more bandwidth, as each demand x_i will be mapped to the next offered level of service (i.e., supply point). For a network with service levels obtained through the DPM1 (respectively, TDM-DPM1) algorithm, the amount of bandwidth used is given by the objective function (5) (respectively, the objective function (20) after subtracting the term $\frac{C}{r}$). In our study, we use the *normalized bandwidth requirement* metric, defined as the ratio of the amount of bandwidth used by a tiered-service network to the amount of bandwidth ρ_X used by a continuous network, to characterize the bandwidth penalty incurred by a tiered-service network.

Figures 9 and 10 plot this metric against the number p of service levels offered by the network. Each point in these curves is the average over 30 different problem instances generated by a uniform distribution; similar results were obtained for all other distributions shown in Table I and can be found in [6].

Figure 9 presents results for two tiered service scenarios: one in which the service levels are obtained from the DPM1 algorithm, and one in which they are obtained from the DDH algorithm; DDH is selected as a representative algorithm for the TDM-DPM1 problem in which the service levels are all multiples of a basic bandwidth unit r . As we can see, the curve for DDH is above the one for DPM1. This result is expected, since the (optimal) DPM1 algorithm is only concerned with minimizing the excess bandwidth due to tiered service, while the DDH algorithm also has to take into account the constraint that all service levels be multiples of a basic unit. However, the additional penalty due to the constraint imposed by the TDM-DPM1 problem is relatively small; we have observed similar behavior for all distributions. Also, the normalized bandwidth requirement decreases rapidly with the number of service levels; this result can be explained by noting that as the number of levels becomes very large, the tiered-service network reduces to a continuous-rate network.

Figure 10 shows the effect of the number n of demands on

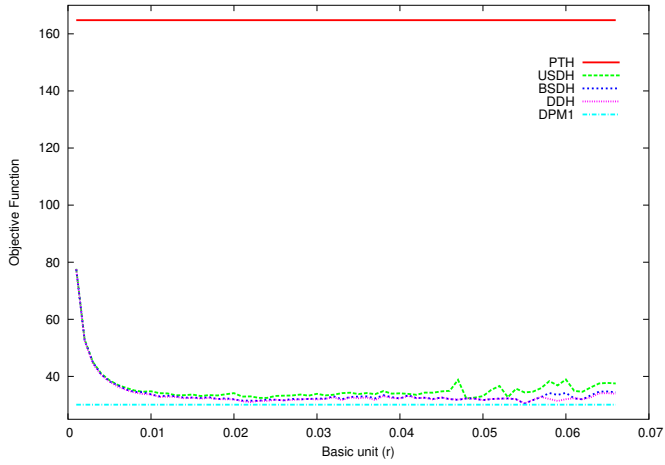


Fig. 6. Objective function value against r , $n = 1000$, $p = 15$, $C = 0.05$, triangle distribution

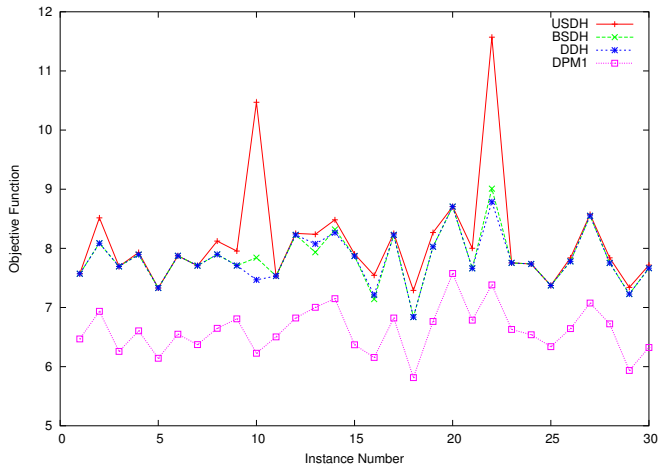


Fig. 7. Objective function value returned by the algorithms, $n = 100$, $p = 5$, $C = 0.05$, increasing distribution

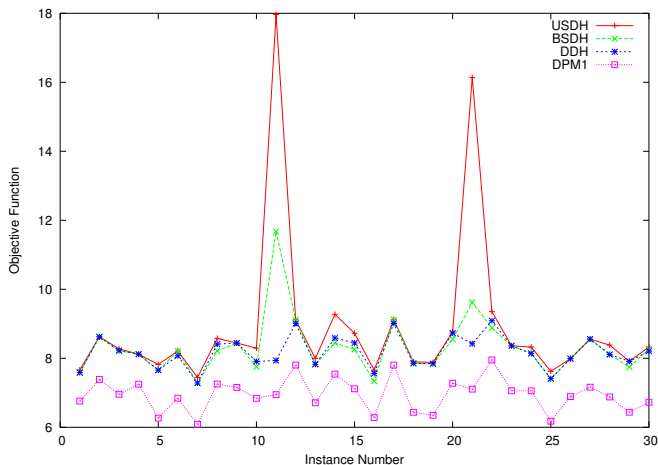


Fig. 8. Objective function value returned by the algorithms, $n = 100$, $p = 5$, $C = 0.05$, triangle distribution

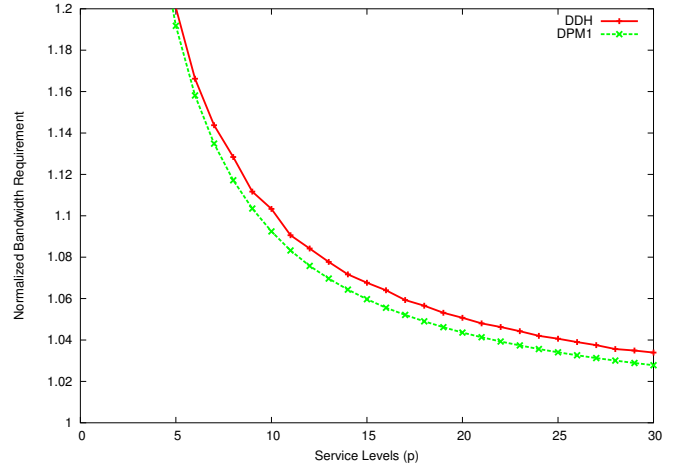


Fig. 9. Normalized bandwidth requirement against p , uniform distribution

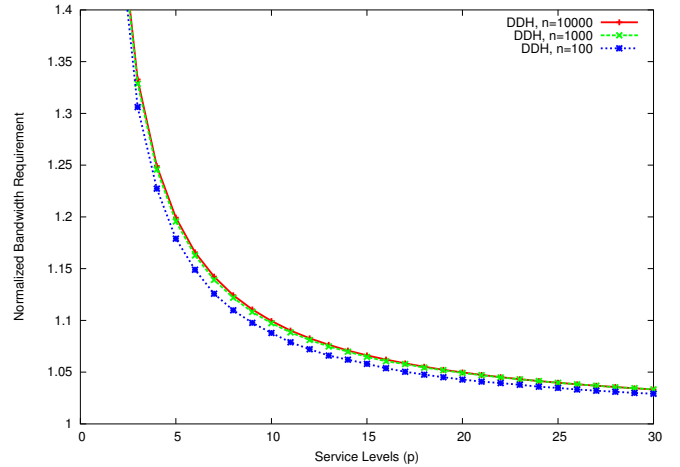


Fig. 10. Normalized bandwidth requirement against p , uniform distribution

the normalized bandwidth requirement for the DDH algorithm; the effect on the DPM1 algorithm is similar. We observe that as the number n of demands increases, the normalized bandwidth requirement does increase slightly, but the effect diminishes quickly; in fact, the curve for $n = 10,000$ almost coincides with the curve for $n = 1,000$ in the figure. The conclusions we can draw from these figures, and similar ones which can be found in [6], is that (1) with $p = 10 - 15$ levels, the bandwidth required by a tiered-service network is only about 5-10% higher than that of a continuous-rate network; (2) the additional constraint that all service levels be a multiple of a basic unit only slightly adds to the bandwidth penalty; and (3) increasing the number n of demands imposes only an incremental penalty on bandwidth.

Impact on network performance. Finally, let us examine the practical impact of tiered service on overall network performance. To this end, we consider an MPLS network scenario in which LSPs arrive and depart dynamically. An LSP between source-destination pair (s, d) requires a certain amount of bandwidth; if a path between s and d with sufficient resources can be found, the LSP is established, otherwise, it is rejected (blocked). The performance measure of interest in this

context is the LSP blocking probability. We use simulation to compare the blocking probability of a continuous network to that of a tiered-service network. In a continuous network, an LSP requiring bandwidth x_i is accepted if a path with at least that much bandwidth can be found. In a tiered-service network, the bandwidth demand x_i is first mapped to the next highest service level offered, say, z_j , and the LSP is accepted if a path with bandwidth at least equal to z_j is found. The service levels for the tiered-service network are computed in advance for the given demand distribution, using the appropriate algorithm (DPM1 or a TDM-DPM1 heuristic).

In our simulation model, LSP requests arrive as a Poisson process with rate λ , and the mean LSP holding time is an exponentially distributed random variable with rate $\mu = 1$. Each simulation run lasts until 100,000 LSP requests have been served. Each point in the blocking probability curves shown here is the average of thirty simulation runs; we also plot 95% confidence intervals which we estimated using the method of batch means. All the results are for the NSFNet network topology, which can be found in [6]. The capacity of all links is set to two units of bandwidth; since the demand distributions in Table I are defined in the interval $[0, 1]$, this assumption implies that the bandwidth requested by any LSP is at most one half the link capacity.

Figure 11 plots the blocking probability against the LSP arrival rate for a continuous network and two tiered-service networks, one using DPM1 to obtain the service levels and one using DDH, a representative algorithm for the TDM-DPM1 problem. As expected, the blocking probability of the continuous-rate network is lowest, that of the tiered-service network allocating bandwidth in multiples of a basic unit is highest (DDH algorithm), and that of a network (DPM1 algorithm) which minimizes the excess bandwidth is in between the other two. The higher blocking probability experienced by a tiered-service network is a direct result of the additional resources that such a network uses for each traffic demand. However, the increase in blocking probability is rather small and it may be more than compensated by the advantages of tiered service.

Figure 12 shows the behavior of the blocking probability for the DDH algorithm as we vary the number of service levels p . The curves confirm the intuition that as p increases, the blocking probability of the tiered-service network decreases and tends towards that of a continuous-rate network. This figure suggests that the network designer/engineer may select the number p of the service levels to be offered so as to combine the advantages of tiered service with the performance of a continuous-rate one.

VII. SUMMARY AND OUTLOOK

Tiered service has many potential applications in networking, especially in contexts where catering to very large sets of heterogeneous users/demands poses significant scalability problems. We have developed a theoretical framework for reasoning about service level selection. Our ongoing research aims to extend this work by (1) including pricing considerations in service level selection, and (2) building a tiered-service

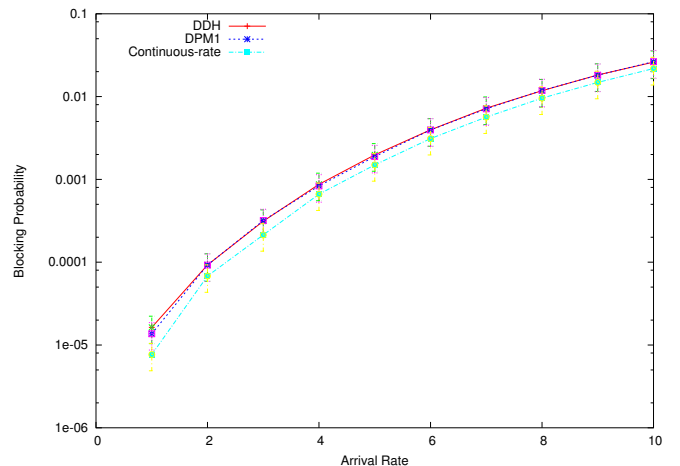


Fig. 11. Blocking probability against the arrival rate, $n = 100,000$, $p = 30$, $C = 0.05$, uniform distribution

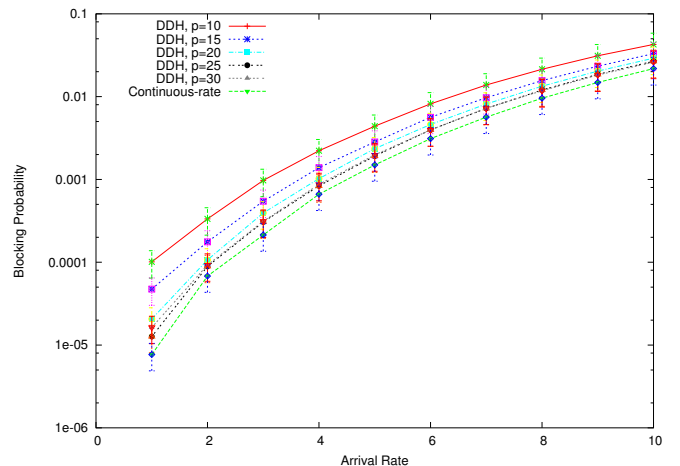


Fig. 12. Blocking probability against the arrival rate, $n = 100,000$, $C = 0.05$, uniform distribution

network testbed for quantifying experimentally the benefits of tiered service.

REFERENCES

- [1] A. Aggarwal, M. Klawe, S. Moran, P. Shor, and R. Wilber. Geometric applications of a matrix searching algorithm. *Algorithmica*, 2(2):195–208, 1987.
- [2] A. Aggarwal and J. Park. Notes on searching in multidimensional monotone arrays. In *Proceedings of the 29th Annual IEEE Symposium on Foundations of Computer Science*, pages 497–512, 1988.
- [3] J. H. Anderson and A. Srinivasan. Mixed Pfair/ERfair scheduling of asynchronous periodic tasks. *Journal of Computer and System Sciences*, 68(1):157–204, February 2004.
- [4] D. Awduche, L. Berger, D.-H. Gan, T. Li, V. Srinivasan, and G. Swallow. RSVP-TE: Extensions to RSVP for LSP tunnels. IETF Draft <draft-ietf-mpls-rsvp-lsp-tunnel-08.txt>, February 2001. Work in progress.
- [5] D. O. Awduche. MPLS and traffic engineering in IP networks. *IEEE Communications*, 37(12):42–47, December 1999.
- [6] Nikhil Baradwaj. Traffic quantization and its application to QoS routing. Master's thesis, North Carolina State University, Raleigh, NC, August 2005. (2006 Graduate School Nancy G. Pollock MS Thesis Award).
- [7] S. K. Baruah, N. K. Cohen, C. G. Plaxton, and D. A. Varvel. Proportionate fairness: a notion of fairness in resource allocation. *Algorithmica*, 15(6):600–625, 1996.
- [8] P. Bonenfant and A. Rodriguez-Moral. Generic framing procedure (GFP): The catalyst for efficient data over transport. *IEEE Communications Magazine*, 40(5):72–79, May 2002.

- [9] D. Cavendish, K. Murakami, S-H. Yun, O. Matsuda, and M. Nishihara. New transport services for next-generation SONET/SDH systems. *IEEE Communications Magazine*, 40(5):80–87, May 2002.
- [10] D. Cavendish, K. Murakami, S-H. Yun, O. Matsuda, and M. Nishihara. New transport services for next-generation SONET/SDH systems. *IEEE Communications Magazine*, 40(5):80–87, May 2002.
- [11] M. Daskin. *Network and Discrete Location: Models, Algorithms, and Applications*. John Wiley and Sons, New York, 1995.
- [12] B. Davie and Y. Rekhter. *MPLS Technology and Applications*. Morgan Kaufmann Publishers, San Diego, California, 2000.
- [13] R. Dutta, A. E. Kamal, and G. N. Rouskas. Grooming mechanisms in SONET/SDH and next-generation SONET/SDH. In R. Dutta and A. E. Kamal and G. N. Rouskas (Editors), *Traffic Grooming for Optical Networks: Foundations, Techniques, and Frontiers*, pages 39–55. Springer, 2008.
- [14] R. Dutta, A. E. Kamal, and G. N. Rouskas, editors. *Traffic Grooming in Optical Networks: Foundations, Techniques, and Frontiers*. Springer, 2008.
- [15] R. Dutta and G. N. Rouskas. Traffic grooming in WDM networks: Past and future. *IEEE Network*, 16(6):46–56, November/December 2002.
- [16] G-S. Kuo (Ed.). Special issue on multiprotocol label switching. *IEEE Communications Magazine*, 37(12), December 1999.
- [17] O. Gerstel and G. Sasaki. Quality of protection: A quantitative unifying paradigm to protection service grades. In *Proceedings of SPIE Opticomm 2001*, pages 12–23, 2001.
- [18] W. Goralski. *SONET*. McGraw-Hill, 2000.
- [19] R. Hassin and A. Tamir. Improved complexity bounds for location problems on the real line. *Operations Research Letters*, 10:395–402, 1991.
- [20] E. Hernandez-Valencia. Hybrid transport solutions for tdm/data networking services. *IEEE Communications Magazine*, 40(5):104–112, May 2002.
- [21] E. Hernandez-Valencia, M. Scholten, and Z. Zhu. The generic framing procedure (gfp): An overview. *IEEE Communications Magazine*, 40(5):63–71, May 2002.
- [22] C. Holahan. Time warner’s net metering precedent. *Business Week*, June 4 2008.
- [23] C. Holahan. Time warner’s pricing paradox. *Business Week*, January 18 2008.
- [24] International Telecommunication Union (ITU). Generic framing procedure (GFP). In *ITU-T G.7041*, 2001.
- [25] International Telecommunication Union (ITU). Link capacity adjustment scheme (LCAS) for virtually concatenated signals. In *ITU-T G.7042*, 2004.
- [26] L. Jackson and G. N. Rouskas. Optimal quantization of periodic task requests on multiple identical processors. *IEEE Transactions on Parallel and Distributed Systems*, 14(7):795–806, July 2003.
- [27] L. E. Jackson and G. N. Rouskas. Deterministic preemptive scheduling of real time tasks. *IEEE Computer*, 35(5):72–79, May 2002.
- [28] L. E. Jackson, G. N. Rouskas, and M. F. M. Stallmann. The directional p -median problem: Definition, complexity, and algorithms. *European Journal of Operations Research*, 179(3):1097–1108, June 2007.
- [29] Shrikrishna Khare. Testbed implementation and performance evaluation of the tiered service fair queueing (TSFQ) packet scheduling discipline. Master’s thesis, North Carolina State University, Raleigh, NC, August 2008.
- [30] C-T. Lea and A. Alyatama. Bandwidth quantization and states reduction in the broadband ISDN. *IEEE/ACM Transactions on Networking*, 3(3):352–360, June 1995.
- [31] J. Y-T. Leung. A new algorithm for scheduling periodic, real time tasks. *Algorithmica*, 4:209–219, 1989.
- [32] J. Y-T. Leung and M. L. Merrill. A note on preemptive scheduling of periodic, real time tasks. *Information Processing Letters*, 11(3):115–118, Nov 1980.
- [33] N. Megiddo and K. J. Supowit. On the complexity of some common geometric location problems. *SIAM Journal on Computing*, 13(1):182–196, February 1984.
- [34] E. Modiano and P. J. Lin. Traffic grooming in WDM networks. *IEEE Communications*, 39(7):124–129, Jul 2001.
- [35] A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The single-node case. *IEEE/ACM Transactions on Networking*, 1(3):344–357, June 1993.
- [36] S. Ramabhadran and J. Pasquale. Stratified round robin: A low complexity packet scheduler with bandwidth fairness and bounded delay. In *Proceedings of ACM SIGCOMM ’03*, pages 239–249, August 2003.
- [37] E. Rosen, A. Viswanathan, and R. Callon. Multiprotocol label switching architecture. RFC 3031, January 2001.
- [38] G. N. Rouskas and N. Baradwaj. A framework for tiered service in MPLS networks. In *Proceedings of IEEE INFOCOM 2007*, pages 1577–1585, May 2007.
- [39] G. N. Rouskas and Z. Dwekat. A practical and efficient implementation of WF²Q+. In *Proceedings of IEEE ICC*, pages 172–176, June 2007.
- [40] P-J. Wan, G. Calinescu, L. Liu, and O. Frieder. Grooming of arbitrary traffic in SONET/WDM BLSRs. *IEEE Journal on Selected Areas in Communications*, 18(10):1995–2003, 2000.
- [41] K. Zhu and B. Mukherjee. Traffic grooming in an optical WDM mesh network. *IEEE Journal on Selected Areas in Communications*, 20(1):122–133, Jan 2002.
- [42] K. Zhu and B. Mukherjee. A review of traffic grooming in WDM optical networks: Architectures and challenges. *Optical Networks Magazine*, 4(2):55–64, March/April 2003.



George N. Rouskas (S '92, M '95, SM '01) is a Professor of Computer Science at North Carolina State University. He received the Diploma in Computer Engineering from the National Technical University of Athens (NTUA), Athens, Greece, in 1989, and the M.S. and Ph.D. degrees in Computer Science from the College of Computing, Georgia Institute of Technology, Atlanta, GA, in 1991 and 1994, respectively. During the 2000-2001 academic year he spent a sabbatical term at Vitesse Semiconductor, Morrisville, NC, and in May 2000, December 2002,

and July 2006 he was an Invited Professor at the University of Evry, France. His research interests include network architectures and protocols, optical networks, multicast communication, and performance evaluation. He is co-editor of the book *Traffic Grooming for Optical Networks: Foundations, Techniques and Frontiers* (Springer 2008).

Dr. Rouskas received the 2004 ALCOA Foundation Engineering Research Achievement Award, and the 2003 NCSU Alumni Outstanding Research Award. He is a recipient of a 1997 NSF Faculty Early Career Development (CAREER) Award, and the recipient of the 1994 Graduate Research Assistant Award from the College of Computing, Georgia Tech. He is a co-author of two papers that received Best Paper Awards at the 1998 SPIE conference on All-Optical Networking and at the 2006 CSNDSP conference. Dr. Rouskas is especially proud of his teaching awards, including his induction in the NCSU Academy of Outstanding Teachers in 2004, and the Outstanding New Teacher Award he received from the Department of Computer Science in 1995.

Dr. Rouskas is founding co-editor-in-chief of Optical Switching and Networking (OSN), an Elsevier journal, he has served on the editorial boards of the *IEEE/ACM Transactions on Networking*, *Computer Networks*, and *Optical Networks*, and he was a co-guest editor for the *IEEE Journal on Selected Areas in Communications*, Special Issue on Protocols and Architectures for Next Generation Optical WDM Networks, published in October, 2000. He was technical program co-chair of the Networking 2004 conference, general co-chair of the IEEE LANMAN 2005 workshop, program chair of the IEEE LANMAN 2004 workshop, and program co-chair of the Traffic Grooming workshop 2004. He is serving as general co-chair for BROADNETS 2007. He is a member of the ACM and of the Technical Chamber of Greece.



Nikhil Baradwaj is a software design engineer with MicroStrategy, McLean Virginia. He received his M.S. degree in Computer Science from North Carolina State University, Raleigh, North Carolina, in 2005, and his B.E. in Computer Science and Engineering from Nitte Mahalinga Adhyanta Memorial Institute of Technology, Nitte, India, in 2002. He was the recipient of the 2006 Nancy G. Pollock M.S. Thesis Award from the Graduate School at North Carolina State University for his thesis “Traffic Quantization and its Application to QoS Routing.”