

# On The Impact of Aggregation on The Performance of Traffic Aware Routing

A. Sridharan<sup>†</sup>   S. Bhattacharyya, C. Diot   R. Guérin<sup>†</sup>   J. Jetcheva, N. Taft  
 U. Pennsylvania   Sprint Labs   U. Pennsylvania   Sprint Labs  
 Philadelphia, PA   Burlingame, CA   Philadelphia, PA   Burlingame, CA

**Abstract**—This paper investigates the impact of traffic aggregation on the performance of routing algorithms that incorporate traffic information. The focus is on two main issues. Firstly, we explore the relationship between the *average* performance of the network and the level of granularity at which traffic can be assigned to routes. More specifically, we are interested in how average network performance improves as the ability of the routing protocol to split traffic arbitrarily across multiple paths increases. Secondly, we focus on the impact of traffic aggregation on short-term routing behavior. In particular, we explore the effects of traffic aggregation on traffic variability, which directly affects short-term routing performance. Our analysis is based on traffic traces collected from an operational network. The results of this study provide insights into the cost-performance trade-offs associated with deploying routing protocols that incorporate traffic awareness.

**Keywords**—Routing, Networks, Traffic Engineering, Aggregation.

## I. INTRODUCTION

As IP networks become the life-line of business and commercial applications, the need for better service guarantees and improved performance, are driving the deployment of service differentiation and traffic engineering in IP networks. Both typically involve data path and control path components. The data path relies on mechanisms such packet classification, schedulers, buffer management, etc., and is by now reasonably well understood and studied, e.g., see [16] for a recent survey, even if there is still much debate on issues such as scalability and the level of guarantees that are needed. The control path often involves the use of signalling, e.g., [8], and extensions

<sup>†</sup> The work of these authors was supported in part by a grant from Sprint Labs and by NSF grants ANI99-06855 and ANI99-02943

Authors emails: ashwin@ee.upenn.edu, supratik,cdiot@sprintlabs.com, , guerin@ee.upenn.edu, jor-jeta,nina@sprintlabs.com

to routing protocols, e.g., [3]. In this paper, we focus on routing, and in particular on evaluating the trade-off that exists between the added complexity and cost of the extensions required to accommodate service differentiation and traffic engineering, and the performance benefits it affords. We believe that such an understanding is important to decide whether or not *traffic aware* routing is worth deploying.

Traffic aware routing consists of protocols and algorithms that incorporate in the computation of routes, the knowledge of both available network resources, e.g., available link bandwidth, and traffic requirements. The goal is some optimization of network usage or service guarantees. There have been many studies devoted to the design and evaluation of traffic aware routing algorithms and protocols, and they can be broadly classified in two categories. Those with a *traffic engineering* focus, and those that target an *on-demand* model (see [11], [25], [19], [6] for examples of the first, and [10], [1], [2] and [18] despite its title for examples of the second). In both settings, it is assumed that network topology and link capacities are known, and the two differ primarily in the traffic information available to routing. In the context of traffic engineering, the volume of traffic between different ingress and egress nodes<sup>1</sup>, i.e., a “traffic matrix,” is assumed known. Given this input, the goal of routing is then to *pre-compute* a set of path and associated traffic assignments, so as to optimize some measure of overall network performance, e.g., total network delay. In contrast, in an on-demand model, the input to routing is in the form of traffic *requests*, i.e., requests for a certain amount of bandwidth between given pairs of nodes, that are made in an ongoing manner. Routing is then responsible for the on-line computation and selection of new paths for those requests, and its goal is to “maximize” the traffic carrying ability of the network.

The focus of this paper is on the traffic engineering usage of routing, when a traffic matrix characterizing the

<sup>1</sup>An ingress node is a node at which traffic first enters the routing domain, e.g., a node adjacent to another routing domain or one to which a customer site router is attached. Conversely, an egress node is a node where traffic exits the routing domain, e.g., to enter another domain or a customer site.

bandwidth requirements between pairs of ingress-egress nodes is assumed known<sup>2</sup> and used to compute routes in an attempt to optimize network performance. Our main goal is to gain a better understanding of the cost-benefit trade-off associated with the use of routing for traffic engineering purposes. Two important components to the cost of traffic aware routing are: (1) matching traffic to routes (paths) so as to achieve “optimal” network performance; (2) updating routes to accommodate variations in traffic patterns or intensity.

The first component can be further broken down into a traffic classification cost at the ingress router, and a forwarding state cost in the core network. The cost of ingress traffic classification depends on the granularity at which traffic needs to be split in order to achieve the *route loads* that routing produces. This cost grows with the number of classifier entries required. The forwarding cost in the core network grows with the number of distinct paths needed to achieve “optimal” *link loads*. In particular, because traffic aware routing attempts to optimally distribute traffic over network links, it typically requires a larger number of routes than current IP routing protocols that rely on one or a small number<sup>3</sup> of routes for each possible destination (subnet). This difference is compounded in the presence of route aggregation, i.e., as allowed by CIDR prefixes [13], which further reduces the number of routes that standard IP routing protocols require. In addition to a greater number of forwarding entries in the core network, traffic aware routing also implies a more expensive packet forwarding operation than the standard hop-by-hop IP forwarding. Fortunately, a number of recent developments, e.g., [9], [26] or [23], can help offset some of those cost increases, and make traffic aware routing a more realistic alternative.

These advances notwithstanding, minimizing the number of forwarding entries required by traffic aware routing remains an important criterion to keeping its cost low. This is further motivated by the added cost associated with *installing* forwarding state in routers, e.g., using signalling protocol extensions such as those specified in [5], [17]. This cost grows with the frequency at which routes need to be adjusted, as traffic between pairs of ingress and egress nodes changes. Specifically, routes are computed based on a traffic matrix that specifies the volume of traffic between pairs of ingress and egress nodes. This traffic information is typically (see Section II) obtained by *measuring* at ingress nodes the amount of traffic headed to various destinations. Implicit with any measurement is a time period or set of periods over which the measurement

<sup>2</sup>How to acquire this information is discussed in Section II

<sup>3</sup>Even for protocols such as OSPF [21] that support multiple equal cost paths for load balancing purposes, there is typically a configured limit on this number.

is performed. Because traffic patterns change, traffic matrices can also change from period to period. A different traffic matrix often translates into a different set of “optimal” paths, but frequently adjusting paths to track such changes may not be feasible or desirable. In particular, it temporarily disrupts traffic delivery and requires updating forwarding state in all the affected routers. As a result, the frequency of such changes is best kept as low as possible. Clearly, this must be weighed against the potential performance improvements that closely tracking optimal paths affords, and gaining a better understanding of this trade-off is a goal of this paper. To address the first cost issue regarding classification and forwarding costs, we evaluate the impact of *traffic granularity* (defined below) on the performance of traffic aware routing. To address the second cost issue regarding the frequency of routing updates, we study the influence of *time granularity* (defined below) on routing performance.

Traffic granularity refers to the level of traffic (and route) aggregation that constrains the load balancing ability of traffic aware routing. Coarse granularity aggregation bundles traffic into a small number of “*streams*” that must then be routed individually. This constraint may affect routing performance by limiting its ability to arbitrarily split traffic across paths to achieve *optimal* link loads. In this paper, we use prefix masks of different lengths as the basis for traffic aggregation decisions, with each prefix length defining, therefore, an associated level of traffic granularity. For example, a prefix length of “zero” represents the coarsest level of granularity, and aggregates all the traffic between a pair of ingress and egress routers onto a single stream. Alternatively, a mask length of four corresponds to a finer granularity, where packets between a pair of ingress and egress routers are assigned to different streams based on the first four bits of their destination address.

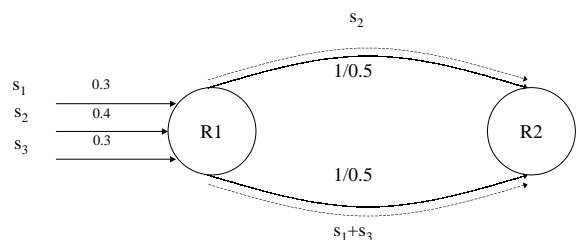


Fig. 1. Impact of traffic granularity on link loads.

We provide a simple example that illustrates the relation between traffic granularity and routing performance. Consider the configuration shown in Figure 1 with two routers, R1 and R2, connected by two unit capacity paths. The total traffic between the two routers is also one unit. From symmetry arguments, it is easy to see that optimal

performance involves assigning half of the traffic to each path. However, because the traffic arriving at router R1 is in the form of three individual streams,  $s_1$ ,  $s_2$ , and  $s_3$ , with respective traffic intensities of 0.3, 0.4, and 0.3, it is not possible to split the traffic so as to optimally load each path with a load of 0.5. Instead, the best stream assignment ends up loading one path with 0.6 units of traffic and the other with 0.4 units. Thus another one our goals is to quantify how “far” from optimal we end-up because of the constraints that traffic granularity imposes.

Time granularity, on the other hand, refers to the variations of traffic intensity as a function of the duration of measurement intervals. The magnitude of those fluctuations depends on both the length of the measurement period and the granularity of the traffic stream that is being monitored. Our goal is to understand this relationship as traffic granularity changes, as well as investigate its impact on routing performance, and in particular short term performance, i.e., over shorter time intervals than the ones used by routing to compute optimal routes.

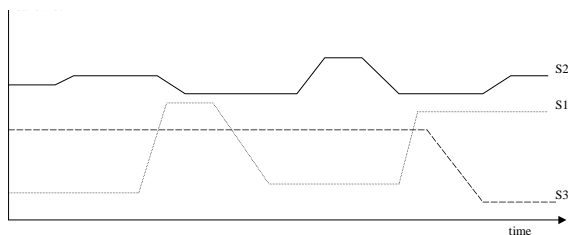


Fig. 2. Interactions Between Time and Traffic Granularity.

In Figure 2 we illustrate for three streams S1, S2, and S3, the influence of time granularity as well as its interaction with traffic granularity. If new routes are assigned to each stream each time it experiences a major change in bandwidth requirement, then it would be possible to avoid overloading network links. However, as mentioned earlier, performing frequent changes may not be feasible, and it is common that routes be assigned initially on the basis of long term averages and kept unchanged for some extended period of time. As a result, the short term bandwidth variations of individual streams can translate into periods of link overloads, especially when multiple streams sharing the same path experience a bandwidth increase.

The interaction between traffic and time granularity in the context of traffic aware routing is as follows. Routes are typically computed on the basis of long term traffic averages, e.g., daily averages, that correspond to the frequency at which routes can be adjusted. Routing specifies a set of “optimal” paths and associated loads for traffic between each source and destination. Achieving those optimal loads often calls for fine grain splitting of traf-

fic, in order to be able to properly distribute traffic across the different paths generated by routing. However, requiring such fine traffic granularity, besides incurring a high classification and possibly<sup>4</sup> forwarding cost, can translate into greater short term traffic variability. This may in turn increase the likelihood of transient link overloads (or underloads), and therefore poorer short term performance. Note that this will obviously depend on the extent to which fine granularity streams are assigned to different paths, and this is one of the aspects we investigate. On the other hand, although using coarser traffic granularity, e.g., using supernets, may not allow us to optimally distribute traffic over links, it forces traffic to remain aggregated. This may then result in smaller short term traffic fluctuations and, therefore, fewer periods of transient overload.

Understanding the extent to which these different parameters affect the trade-off between performance and cost in the context of traffic aware routing is the main goal of this paper. Our approach is based on evaluating the performance of two heuristic<sup>5</sup> routing algorithms for “optimally” routing traffic given certain granularity constraints. We evaluate both short term and long term performance as we vary traffic granularity. In addition, we also investigate the relation that exists between traffic granularity, i.e., the ingress classifying cost of traffic aware routing, and the number of distinct paths actually required in the network, i.e., the forwarding cost of traffic aware routing.

There have been a number of previous studies devoted to the design and evaluation of traffic engineering protocols and algorithms [11], [25], [19], [6] in the context of IP and MPLS networks that we assume here. However, none of these has focused on the interactions between time and traffic granularity and their relation to routing performance, as this paper does. Similarly, there have been a number of papers that have proposed computing routes on the basis of traffic matrix estimates. However, most of these proposals, many of which date back to circuit-switched networks, e.g., see [15], [4] or [20] for a more recent work, have been formulated in the context of on-demand routing in loss networks, which represent a different setting from the traffic engineering application we consider. In the context of traffic engineering, the most relevant recent work regarding the computation of optimal paths for a given traffic matrix and under granularity constraints is that of [22], which applies randomized algorithm to this problem. As we shall see, this method is, however, different from the heuristics we develop in Section III.

<sup>4</sup>Multiple streams may be forwarded onto the same path, and this is an aspect which we investigate further in the paper.

<sup>5</sup>We rely on heuristics as the problem of optimal routing under splitting constraints is well known to be NP-hard, e.g., [14].

The rest of this paper is structured as follows. In Section II, we review the traffic measurement procedures we rely on to estimate the traffic matrices used in the rest of the paper. Section III focuses on the impact of traffic granularity on routing performance. It evaluates routing performance for different levels of traffic granularity, and compares it to the performance of an optimal algorithm that operates without granularity constraints. Section IV investigates how routing performance varies over time as a function of traffic granularity. In particular, it explores how traffic granularity affects the difference that exists between short term and long term performance. Finally, Section V summarizes the findings of the paper and points out potential extensions.

## II. TRAFFIC MEASUREMENT AND TRAFFIC MATRIX GENERATION

The generation of traffic matrices for our study is based on several measurements taken within Sprint’s IP backbone network. SNMP databases provided us with information on POP activity and the amount of data exchanged between POPs. SNMP data is available as part of classic network management tools. The Sprint IP backbone monitoring project [12] provided us with exhaustive traffic data from a single POP (that we will call the “monitored POP”). This data consists of traces, gathered over the duration of a given measurement interval, that contain the first 40 bytes of each IP packet captured on the access links of the monitored POP. Note that this provides us information on the size of the packet. In addition, each IP packet in the trace is time-stamped using a globally synchronized clock (GPS based). From this information, we can determine the number of bytes headed to other destinations across the Sprint network during any time interval. This data forms the basis for determining (i) traffic intensities between the monitored POP and the other  $20^6$  POPs in the Sprint backbone (see [12] for a general description of the overall topology and the internal architecture of a POP), and (ii) the variation of this traffic on different time scales and at different levels of granularity.

A traffic matrix contains, for each pair of POPs in the backbone, the amount of traffic between these two POPs (in each direction) at different levels of granularity. For proprietary reasons, we cannot give the exact topology of the backbone. Figure 3 represents the topology of a classic 1st tier ISP backbone. A typical POP consists of few core routers and of 10 to 20 access routers (see [12] for details). Figure 4). A POP is fully redundant (each link terminates to at least two core routers or to two access routers). Access routers usually connect to core

<sup>6</sup>some POPS are represented by a single router in the topology. We have 16 such routers in the topology. POPS behind a router were aggregated as one entity

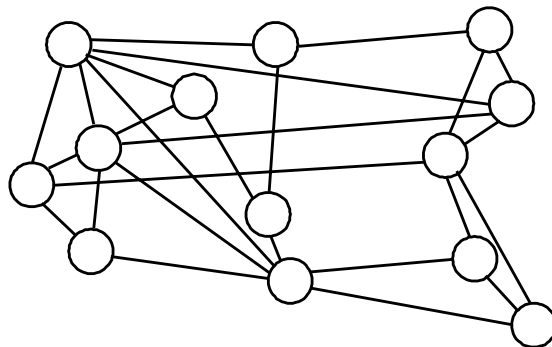


Fig. 3. Topology of a 1st tier ISP backbone.

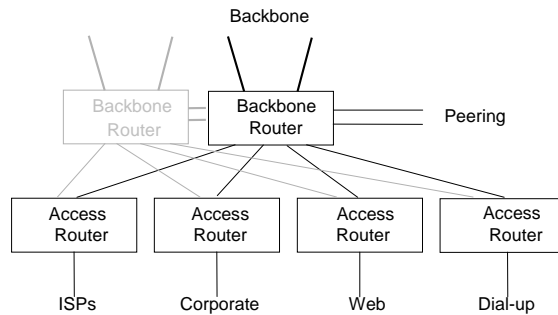


Fig. 4. Architecture of a POP.

routers via OC-12 links. Core routers are connected to other POPs with OC-48 links. Sprint customers connect to the IP network via the access routers. Peering points are connected to both access and core routers.

We are interested in computing the intensity of a typical traffic “stream” between two POPs at a given level of traffic granularity, where as mentioned earlier, granularity refers to the level of aggregation used to determine which packets are mapped onto a given stream. Packets between two POPs can be aggregated into streams according to different criteria. For example, packets can be mapped to streams based on their source and destination addresses, port numbers and protocol numbers. This corresponds to relatively fine granularity streams. Alternatively, coarser granularity streams can be obtained by aggregating packets on the basis of a common destination address prefix. Furthermore, by using prefix masks of different lengths, it is possible to vary the level of aggregation and, therefore, stream granularity over a pre-determined range. We use this approach in the paper to evaluate the impact of stream granularity on routing performance.

### A. Measurement Methodology

Constructing a traffic matrix requires generating estimates for the total traffic between each pair of POPs. We

used both SNMP data and measurements from our monitored POP, and we first describe how we construct the row in our traffic matrix that corresponds to the monitored POP. The first step consists of identifying the set of destinations addresses associated with the other POPs in the network. For this purpose, we downloaded IBGP tables from the monitored POP for the measurement interval during which the traffic traces were gathered. For every packet in the trace, the egress POP was then identified using information in the IBGP table in conjunction with detailed knowledge about the network topology.

For the results reported in the paper, the duration of the measurement interval at the monitored POP was 800 minutes. As a result, a 800 minutes average represents the coarsest time granularity for traffic between the monitored POP and other POPs in the network. Similarly, the coarsest stream or traffic granularity is achieved by aggregating all the traffic between two POPs, the monitored POP and one of the 20 other POPs, into a single stream. As mentioned earlier, finer levels of granularity are generated by aggregating packets into streams based on destination address prefixes of variable lengths. We do this for prefix lengths of 4, 6, and 8, in addition to the previously mentioned prefix length of “zero” that corresponds to aggregating all the traffic between two POPs onto a single stream. As the length of the address prefix used to aggregate packets increases, so does the number of streams, and conversely, traffic granularity decreases.

We will explore in Section III how this decrease in granularity, which gives routing more flexibility in splitting traffic by routing streams individually, affects routing performance. In order to carry out this investigation, we measure the traffic intensity of individual streams for each level of granularity. In addition, we also measure traffic intensities for different time granularities, i.e., the average bandwidth of individual streams is measured over time intervals of different durations, namely 10, 30, and 60 and 800 minutes. Measuring the average bandwidth of streams on different time scales enables us to identify short-term fluctuations around longer-term averages. For example, the eighty ten-minute estimates obtained for each stream, show how the traffic associated with a given destination prefix varies around its 800 minutes average during those eighty consecutive ten-minute measurement intervals. Table 1 summarizes the result of this process, and gives the typical number of streams and associated traffic intensities for each granularity level. The first column indicates the granularity level in terms of the prefix length used to aggregate packets onto streams. The second column gives, for each granularity level, the range of the number of streams from the monitored POP to the other POPs. Finally, the third column gives the corresponding ranges of 800-minute stream bandwidth aver-

granularity level	number of streams	bandwidth ranges (Mbps)
p0	1	[1-14]
p4	[5-10]	[0-8]
p6	[10-25]	[0-4]
p8	[25-64]	[0-4]

Table 1. Traffic characteristics versus granularity

ages.

More specifically, the traffic matrix “row” obtained as a result of this process provides us with a set of bandwidth estimates of the form  $B_{i,j}^k[n, m]$ , where  $i = 1, \dots, 16$  identifies the egress node;  $j \in \{0, 4, 6, 8\}$  indicates the prefix length used to separate traffic into finer granularity streams; and  $k \in \{10, 30, 60, 800\}$  identifies the time granularity at which traffic is being measured. In particular,  $B_{10,8}^{10}[\cdot, \cdot]$  is itself a “matrix” of bandwidth estimates for traffic from the monitored POP to egress POP number 10. Each row in this matrix corresponds to a single stream associated with all the packets heading towards POP number 10 with the same 8-bit destination address prefix. Each column of this matrix corresponds to one of the eighty ten-minute bandwidth estimates. As a result,  $B_{10,8}^{10}[5, 22]$  gives the average traffic intensity in the 22nd ten-minute monitoring interval for stream number 5 associated with an 8-bit destination address prefix for packets from the monitored POP to POP number 10.

This completes the process used to generate one row of our  $16 \times 16$  traffic matrix that corresponds to the monitored POP. It still remains to fill out the other 15 rows, and we describe next how this was done. The basic approach used was to combine coarse traffic information obtained for other POPs using SNMP, with the detailed structural information provided by the measurements done at the monitoring POP. As a simple check, we verified that this approach and the SNMP data available at the monitored POP were consistent with each other.

Additional details on this procedure are given in the next section.

### B. Populating the traffic matrix at the highest granularity level

The first step to populate the rest of the traffic matrix consisted of characterizing the ability of different POPs to act as traffic sources and sinks. In general, most of the traffic traversing the backbone is destined to Sprint customers who are attached to the network at one of the 16 POPs. As a result, a POPs ability to act as a traffic sink is approximately proportional to the number of customers attached to it. Customers are of two types. Dialup cus-

tomers subscribe to dialup networks which connects to the POPs. Corporate customers usually have dedicated connections (through access routers) at various POPs. Hence, the “customer weight” of a POP can be measured based on the number of each type of customers attached to it. Alternatively, the ability of a POP to source traffic is dominated by its external connections, i.e., the number of peering links it has, as the bulk of the traffic destined for Sprint customers originates outside of Sprint’s network. As a result, one can assume that the volume of traffic sourced by a POP is proportional to the number of peering points it has.

Based on the above reasoning, we characterize each POP with a three-tuple  $(PP, CC, DC)$ , where  $PP$  corresponds to the number of peering points,  $CC$  corresponds to the number of corporate customers and  $DC$  corresponds to the number of dialup customers. Each of these attributes can take one of three possible values HIGH, MEDIUM and LOW. For example, a POP that receives a large volume of traffic from peer networks is assigned the value HIGH for its  $PP$  attribute. Assignment of values to attributes is done in an approximate manner using information about the network topology and SNMP data. For a given pair of POPs  $i$  and  $j$ , the *LEVEL* of traffic from  $i$  to  $j$  is approximated as

$$LEVEL = MIN(PP_i, MAX(CC_j, DC_j)). \quad (1)$$

The above equation states that the level of the volume of traffic between POPs  $i$  and  $j$  is based on the ability of POP  $i$  to source traffic, as expressed by  $PP_i$ , and by the ability of POP  $j$  to sink traffic, as expressed by  $MAX(CC_j, DC_j)$ . From this relation, we determine the *LEVEL* of traffic between all pairs of POP to be one of the three possible values HIGH, MEDIUM and LOW. As mentioned earlier, this coarse-grained approximation of traffic levels between POP pairs was found to be reasonably consistent with SNMP data and the data obtained from measurements at the monitored POP. Once the *LEVEL* of a pair of POPs has been determined, it is used to compute its 40-minute average traffic they exchange. The scaling is done based on the data collected at the monitored POP. As a result, a value of LOW is mapped to a random value in the range of 1.0 – 5.0 Mbps. The values MEDIUM and HIGH are similarly mapped to traffic intensity values in the ranges 7.5 – 15 Mbps and 20 – 35 Mbps respectively.

### C. Populating the traffic matrix at lower granularity levels

In order to populate the remainder of the pop-to-pop traffic matrices at finer granularity levels, we use the 40-minute averages generated as described in the previous sections, and couple that information with the structural

information, i.e., number and intensity of streams at different levels of granularity, obtained from the monitored POP. In other words, we replicate the stream structure of the monitored POP to other POPs. This generalization is supported by the fact that (i) the characteristics of the traffic in term of protocol types, packet size distribution, and application type were found to very similar for each POP reached from the monitored POP and (ii) the characteristics of the traffic appears to be independent of the observed POP (see <http://www.caida.org> for similar observations).

Based on this assumption, we then first determine the number of streams at the finest granularity (mask 8) for all other pairs of POPs in the network. This is done by drawing those numbers from a uniform distribution created using the statistical data from the original trace. The intensity of each stream is then selected in the following fashion:

Do until streams have been assigned:

- Pick a source-destination pair randomly
- Pick a flow from that source-destination pair randomly
- Perturbate the stream by a parameter  $0.75 \leq \alpha \leq 1.25$
- Cyclically shift the perturbed stream by a random number  $P : 0 \leq P \leq TIME\_SLOTS$ , where *TIME\_SLOTS* represents the time scale over which the trace was measured.
- Assign this stream to source-destination pair.
- Repeat till all streams have been assigned.

To maintain consistency across masks, traffic matrices for smaller prefix lengths, e.g., 6, were generated by aggregating streams with longer prefix lengths, e.g., 8. The number of longer prefix length streams that were aggregated together is again determined by data from the original trace. Specifically, the ranges of the number of streams for prefix lengths of 4, 6, and 8 in the original trace, are used to determine how many fine granularity streams need to be aggregated to form a coarser granularity one. For example, when constructing a prefix length 6 stream, we select a number of prefix length 8 streams randomly from a range based on the respective numbers of streams of prefix lengths 6 and 8 in the original trace. This process is repeated for each prefix length.

## III. ON THE IMPACT OF TRAFFIC GRANULARITY

In the context of traffic engineering, the goal of traffic aware routing is to distribute network traffic, so as to optimize network performance, e.g., minimize average network delay or maximum link load. In this work, we use a delay-based “cost” function that relies on a standard M/M/1 queueing delay expression (see [7, Sections 5.4 to 5.7]). Specifically, the cost function  $\mathcal{C}(\gamma)$  on which we

rely, is of the form

$$C(\gamma) = \frac{S}{\gamma} \sum_{l \in E} \frac{B_l}{C_l - B_l}, \quad (2)$$

where  $S$  corresponds to the average packet size,  $\gamma$  is the average total traffic offered to the network,  $E$  is the set of links in the network,  $C_l$  are the link capacities, and  $B_l$  are the average link loads achieved by routing. There are obviously many other cost functions that are possible, but in the context of traffic engineering, minimizing the delay experienced by user packets traversing the network is a reasonable target. In addition, several other “typical” cost functions, e.g., minimizing the maximum link load, are known, e.g., [24] to yield results similar to those of a minimum delay cost function. In the rest of the paper, we limit ourselves to this specific cost function.

Optimal algorithms that can minimize the above cost function are well known (e.g. see [7]). They take as input a traffic matrix that specifies the average traffic requirement for each source-destination pairs, where sources and destinations correspond to the different POPs in the network, and produces and output in the form of a set of paths (routes) for each source-destination pair, together with the specification of the fraction of traffic for that source-destination pair assigned to each path. The resulting set of link loads is such that the average network delay is minimized. However, as alluded to earlier, in practice it is neither possible nor desirable to split traffic arbitrarily across paths in the network. As a result, the traffic assignment produced by the optimal algorithm is usually not feasible. In particular, traffic granularity constraints, e.g., as described in Section II, prevent the splitting of traffic from one stream across multiple paths. Recall that traffic granularity is normally caused by the coupling that exists between the monitoring and measurement process from which traffic intensities are obtained, and the underlying packet classifier which selects packets. This classifier can operate at various levels of granularity, e.g., source address, prefix mask, etc., which then determine the number and intensity of packet *streams* that can be routed individually. Note that it is not a goal of this paper to design classifiers capable of splitting traffic into a specific number of streams. Instead, we want to gain a general understanding of the relation that exists between traffic granularity (number of streams) and routing performance. As a result, we rely on a generic packet classification criteria, i.e., prefix masks of various lengths, as described in Section II. Note that as shown in Table 1, this provides for a rather broad range of traffic streams and associated average bandwidth. In that context, we explore the impact of traffic granularity on the achievable network delay by analyzing the behavior of two heuristics, which are described in detail later in this section. Our goal is to compare the performance of these two heuristics against that of an optimal algorithm that operates without granularity

constraints.

### A. Heuristic Routing Algorithms

Before proceeding with the evaluation of several candidate algorithms, we first describe the general goals of those algorithms more formally and introduce some useful notation. Let a routing domain be described as a directed graph  $\mathcal{G} = (V, E)$ , where  $V$  corresponds to the set of vertices in the graph, i.e., the routers in the routing domain, and  $E$  is the set of edges, i.e., the set of links connecting the routers.  $N = |V|$  denotes the number of vertices in the graph and  $M = |E|$  denotes the number of edges. For each pair of nodes  $(i, j)$ ,  $i, j \in V$ , there is a set  $S_{i,j}$  identifying the traffic streams between nodes  $i$  and node  $j$ , and being monitored at node  $i$ , where  $ts_{i,j}^k$  denotes the average traffic intensity of the  $k$ -th stream between  $i$  and  $j$ ,  $s_{i,j}^k \in S_{i,j}$ . There are  $K$  such source destination pairs. In terms of the methodology described in Section II, the number of streams between two nodes depends on the size of the prefixes used to group packet headers, and the average intensity of a stream corresponds to its bandwidth requirements measured over the maximum measurement interval.

The two heuristics we describe next attempt to approach unconstrained, optimal performances, while accounting for the traffic granularity constraints imposed by the existence of individual streams. We use them to evaluate the impact of traffic granularity on our ability to approach the performance of an optimal routing algorithm. Note that our primary intent is **not** to demonstrate the superior performance of one heuristic over the other, even if one (heuristic 2) performs consistently better but at the cost of a greater complexity. Instead, our main focus is assessing the impact of traffic granularity on routing performance, and hence provide motivations, or lack thereof, for moving towards finer granularity in the context of traffic aware routing. As a result, the two heuristics we use, were chosen as representative samples from a larger set of heuristics which we evaluated. In particular, they are based on rather different approaches to handling the granularity constraints introduced by individual streams, and as a result should provide a reasonable coverage of how traffic granularity affects routing performance.

#### A.1 Heuristic 1

The first heuristic is a simple greedy method that adds streams one at a time, while each time selecting a path that minimizes delay. This heuristic is inspired from methods used in an “on-demand” model of traffic aware routing, i.e., an environment where requests are generated dynamically and need to be routed one at a time. The main variations for this first heuristic are in terms of the *order* in which individual streams are routed, e.g., in ascending,

descending, or random orders in terms of their traffic intensity. Specifically, if  $S = \{s_{i,j}^k\}$  represents the set of streams in the network, with  $s_{i,j}^k$  denoting the  $k$ -th stream between nodes  $i$  and  $j$ , the heuristic proceeds as follows:

**Step 1**  $S \xrightarrow{f} S'$  (applies the function  $f$  to the set  $S$  to generate the *ordered* set  $S'$ ). The function  $f$  will typically be one of three choices:  $f_{\text{ascend}}$ ,  $f_{\text{descend}}$ ,  $f_{\text{random}}$ .

**Step 2** While  $S' \neq \emptyset$  do

– Let  $s_{i,j}^k$  be the *first* element of  $S'$ ;

Compute a minimum delay path  $p_{i,j,k}$  between nodes  $i$  and  $j$  for stream  $s_{i,j}^k$ , given the current network link loads and the traffic intensity  $ts_{i,j}^k$  of stream  $s_{i,j}^k$ ;

$$p_{i,j,k} = \operatorname{argmin}_p \left( \sum_{l \in p} \frac{1}{C_l - (B_l + ts_{i,j}^k)} \right)$$

where  $C_l$  is the capacity of link  $l$  and  $B_l$  denotes the allocated bandwidth on link  $l$ .

–  $\forall l \in p_{i,j,k}, B_l = B_l + ts_{i,j}^k$ .

–  $S' = S' \ominus \{s_{i,j}^k\}$

In the next section, we explore the performance of this heuristic and evaluate its sensitivity to variations in the number and traffic intensity of the individual streams it needs to route. A key feature of this first heuristic is that it is independent of any information regarding the actual traffic offered between different nodes. This clearly makes for greater simplicity, but also points to a limitation of the approach, as it does not exploit key information that is available to the routing algorithm.

## A.2 Heuristic 2

The second heuristic takes a different approach that attempts to remedy the shortcoming we have just identified in the first one. Specifically, this heuristic takes into account the knowledge of the total traffic matrix, and in particular the output (set of paths and associated loads) generated by an optimal routing algorithm, i.e., an algorithm that computes optimal paths and loads without considering the granularity constraints imposed by streams. The motivation for using this information is that it represents the best performance achievable. The goal of the heuristic will then be to attempt to get as close as possible to the performance of this unconstrained solution.

The optimal routing problem can be set up as a straightforward multi-commodity flow problem of the form

$$\min_{X_1, X_2, \dots, X_K} f(X_1, X_2, \dots, X_K)$$

subject to

$$AX_k = R_k \quad k = 1, \dots, K \quad (3)$$

$$\sum_{k=1}^K X_k \leq C \quad (4)$$

where  $X_k \in \mathcal{R}^M$  is the flow vector of each commodity  $k$ ,  $k = 1, \dots, K$ .  $A \in \mathcal{R}^{N \times M}$  is the arc-node incidence matrix,  $f : \mathcal{R}^{K \times N} \rightarrow \mathcal{R}$  is the real valued cost function of the network,  $R \in \mathcal{R}^{N \times K}$  is the source-sink matrix and  $C \in \mathcal{R}^M$  is the capacity vector of the network. The first equation conserves the flow between a source and destination pair, while the second inequality constrains flow to not exceed the physical link capacity. We used PPRN<sup>7</sup> to solve the above multi-commodity flow problem.

The heuristic proceeds by assigning streams to (optimal) paths in a manner that attempts to get as close as possible to the link loads achieved by the optimal algorithm. This relies on a two phase procedure. The first phase involves routing streams one-by-one, as in the first heuristic, but on a network with (fictitious) link capacities initially set equal to the desired optimal loads. As each stream is added, it is routed on the path that yields the minimum “delay” given the assumed link capacities. We noticed, that frequently, because of the granularity of the streams, the ‘optimal’ path for a stream may have slightly less capacity than required by the stream. Not assigning the stream to the ‘intended’ path leads to an avalanche effect, wherein, streams get blocked or routed on paths that were ‘meant’ for other streams and this effect multiplies as we route other streams. Hence, we relax the capacity constraint while routing over this network with fictitious link capacities, by allowing a stream to be routed if it does not exceed the capacity of any link on that path by more than a factor of  $(1 + \Delta)$ . The parameter  $\Delta$  controls the amount by which the link constraint is violated. In practice we found  $0.1 \leq \Delta \leq 0.5$  to work reasonably well. The second phase accounts for the fact that, because of traffic granularity, it is unlikely that the first step will succeed in routing all streams even after relaxing the link constraints by  $\Delta$ . In other words, because optimal loads are almost surely not feasible given the granularity constraints, the fictitious capacities of some links will typically be exceeded before all streams have been routed. Any stream for which no feasible<sup>8</sup> path is found during the first phase, is set aside and routed in the second phase.

<sup>7</sup>the PPRN package (<http://www-eio.upc.es/~jcastro/pprn.html>) was developed at the Statistics and Operations Research Dept. at Universitat Politècnica de Catalunya, Barcelona, Spain, by Jordi Castro and Narcs Nabona for solving multi-commodity network flow problems with linear and non-linear cost functions.

<sup>8</sup>Note that feasibility is only in the context of the fictitious

This second phase simply consists of routing the streams that were left-over from phase one using a standard minimum delay algorithm, but using now the actual link capacities together with the link loads that resulted from the first phase. To summarize, this second heuristic proceeds as follows:

### Phase 1

1. Run optimum routing algorithm, ignoring traffic granularity constraints;

For each pair of nodes  $(i, j)$ ,  $i, j \in V, i \neq j$ , this generates a set  $P_{i,j}$  of optimal paths together with corresponding optimal link loads  $c_l, \forall l \in E$ .

2. As for the first heuristic, we generate next an ordered set of streams  $S'$  from the initial set  $S$ , i.e.,

$S \xrightarrow{f} S'$ , where as before the ordering function  $f$  can be one of three choices, ascending, descending, or random.

3.  $\forall l \in E$ , set  $C'_l = c_l$  and  $B'_l = 0$ , where  $C'_l$  represents the new, fictitious capacity assigned to link  $l$  and  $B'_l = 0$  is a variable used to track the amount of bandwidth allocated on link  $l$ .

4.  $S'' = \emptyset$  ( $S''$  is used to retain streams for which no feasible path is found in Phase 1).

5. While  $S' \neq \emptyset$

– Let  $s_{i,j}^k$  be the *first* element of  $S'$ ;

Identify the minimum delay path  $p_{i,j,k} \in P_{i,j}$  between nodes  $i$  and  $j$ ;

$$p_{i,j,k} = \operatorname{argmin}_{p \in P_{i,j}} \left( \sum_{l \in p} \frac{1}{C'_l - (B'_l + ts_{i,j}^k)} \right)$$

– If a feasible path  $p_{i,j,k}$  is identified, i.e.,  $B'_l + ts_{i,j}^k \leq C'_l(1 + \Delta), \forall l \in p_{i,j,k}$ ,

\*  $B'_l \leftarrow B'_l + ts_{i,j}^k, \forall l \in p_{i,j,k}$

\*  $S' = S' \ominus \{s_{i,j}^k\}$

Note that in order to allow the bandwidth allocation  $B'_l$  to exceed the link capacity  $C'_l$ , we set infinite and negative costs to a “large” positive value.

– Else

\*  $S' = S' \ominus \{s_{i,j}^k\}$

\*  $S'' = S'' \cup \{s_{i,j}^k\}$

**Phase 2** While  $S'' \neq \emptyset$

1. Let  $s_{i,j}^k$  be the *first* element of  $S''$ ;

Identify the minimum delay path  $p_{i,j,k} \in P_{i,j}$  between nodes  $i$  and  $j$  given the *actual* link capacities  $C_l, \forall l \in E$ , and the link loads  $B'_l, \forall l \in E$  produced by Phase 1, and the traffic intensity  $ts_{i,j}^k$  of  $s_{i,j}^k$ ;

$$p_{i,j,k} = \operatorname{argmin}_{p \in P_{i,j}} \left( \sum_{l \in p} \frac{1}{C_l - (B'_l + ts_{i,j}^k)} \right)$$

Note that we still limit our search to the set of paths originally generated by the optimal (unconstrained) routing algorithm.

link capacities, and a feasible path can typically be found when using the *real* link capacities.

2.  $B'_l \leftarrow B'_l + ts_{i,j}^k, \forall l \in p_{i,j,k}$ ;
3.  $S'' = S'' \ominus \{s_{i,j}^k\}$ ;

As mentioned, the main difference between this second heuristic and the first one, is that it attempts to use the results of the optimal, unconstrained routing as guidelines when assigning streams to paths.

### B. Performance Evaluation

In this section, we investigate the impact of traffic granularity on routing performance, where performance is measured in terms of the total network average delay computed using the long-term average load. First, we compare the performance of the two heuristics against the optimal routing algorithm, while assuming the finest granularity available from our traffic measurements. i.e., the use of an address prefix mask of length 8. This provides some insight on both the general impact of flow granularity, albeit a relatively fine one, and the differences that exist between heuristics that incorporate knowledge of the traffic matrix (Heuristic 2) and those that don't (Heuristic 1). Second, we study more carefully the impact of the granularity of flows on the performance of the network. Unless specified, the stream ordering used for both the heuristics was in decreasing fashion, i.e., larger streams were routed first.

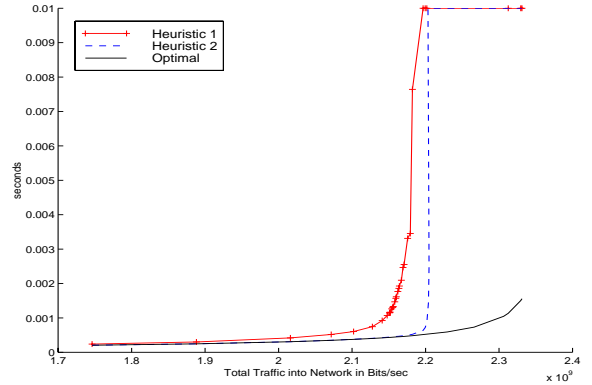


Fig. 5. Optimal Routing vs Routing Under Granularity Constraints.

In order to compare the performance of the heuristics, we used a suitably sized version of the Sprint Topology, and scaled the intensity of a randomly selected set of source-destination pairs in the Traffic Matrix to create hot spots. We show the performance of the two heuristics and of the optimal routing algorithm in Figure 5. Note that while the two heuristics were constrained to routing individually streams generated from length 8 prefixes, the optimal algorithm did not follow any such constraint. As can be seen, Heuristic 2 outperforms Heuristic 1 and closely follows the optimal till the ‘knee’, thereafter the granu-

larity of the streams forces a different sub-optimum allocation. This illustrates the fact that using the information available from the traffic matrix remains important, even when the presence of granularity constraints makes it difficult to take full advantage of this information. In what follows, we explore further the impact that traffic granularity has on routing performance. For that purpose, we limit ourselves to Heuristic 1 as it is simpler than Heuristic 2 and should still capture general trends.

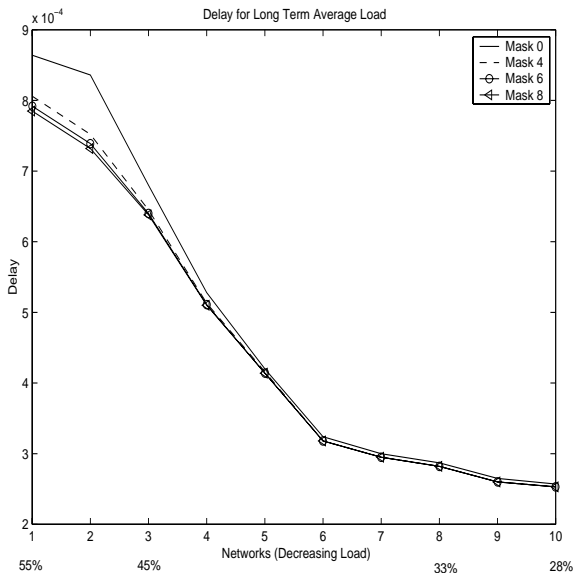


Fig. 6. On the Impact of Traffic Granularity. Delay computed for long term average load

We start this comparison by using the different levels of traffic granularities generated from our traffic measurements. Specifically, traffic can be aggregated onto streams using prefix of lengths 0, 4, 6, or 8, which, as shown in Table 1, translate into average number of streams ranging from 1 (prefix length of 0) to 64 (prefix length of 8). Clearly, one can expect a larger number of streams to result in better performance, as it gives routing more flexibility in assigning traffic to paths. The aspect we want to explore is the *evolution* of routing performance as the number of streams varies, i.e., what is the magnitude of performance improvements as the number of streams varies. We proceeded to do so, by computing the average delay averaged over 30 traffic matrices for each granularity level, each of which was routed on 10 networks with the same topology but increasing capacities (decreasing loads). The results are plotted in Figure 6, which shows the average network wide delay as a function of network sizing. We immediately see that a large fraction of the performance gain is achieved when going from a prefix length of 0 to one of 4, and subsequent improvements are much more modest. This stands to reason, since the num-

Granularity	Average No. of Distinct Paths	Max. No. of Distinct Paths
Mask 0 ( $p_0$ )	1	1
Mask 4 ( $p_4$ )	2	7
Mask 6 ( $p_6$ )	2.3	12
Mask 8 ( $p_8$ )	2.5	16

Table 2. Statistics for the number of distinct paths used at each granularity level for topology 1 (Most Heavily Loaded)

ber of paths through the network is limited, so that beyond a certain level, additional streams are still routed over the same set of paths. We justify this claim by providing, in Table 2, statistics regarding the actual number of distinct paths used for each granularity level. We notice that the average number of paths used doubles from Mask 0 to Mask 4, but increases slowly thereafter indicating the limited availability of paths for all the source-destination pairs. We also note that the improvement in performance decreases rapidly as we increase the capacity of the network, which is not unexpected.

#### IV. ON THE IMPACT OF TIME GRANULARITY

The purpose of this section is to explore another dimension of how performance is affected by the granularity at which routing is performed. Specifically, we saw in Section III that finer granularity leads to a lower *average* load on the links, which translates into lower *average* delay, that is, better long term performance. However, performance measured over finer time scales can be significantly different. The traffic, and hence link loads measured at finer time scales fluctuate around the long term average values. This combined with the non-linear nature of the delay function can result in significantly different performance as compared to that obtained using long term average load.

To see this consider a simple example of a single link with capacity  $C$ . Let the long term average load on the link be  $\bar{\eta}$ . Let the deviation of load from the average value be  $\Delta_k$  in time slot  $k$ . The long-term average performance, which is the delay resulting from the long term average load is simply

$$\begin{aligned}
 D &= \frac{1}{C - \bar{\eta}} \\
 &\approx \frac{1}{C} \left( 1 - \frac{\bar{\eta}}{C} + \frac{\bar{\eta}^2}{C^2} \right) \quad (5)
 \end{aligned}$$

where we have used a second order approximation for purposes of simplicity. Now, if we look at the average of the delay measured over finer time scales, we obtain

$$\tilde{D} = E_k \left( \frac{1}{C - (\bar{\eta} + \Delta_k)} \right)$$

$$\approx \frac{1}{C} \left( 1 - \frac{\bar{\eta}}{C} + \frac{\overline{\eta^2}}{C^2} + \frac{\overline{\Delta^2}}{C^2} \right)$$

where we have again used a second order approximation for simplicity. If we now compare the average of the short term delay with the delay of long term average load (5), we note that the second moment of the deviation from the mean load plays a key role in determining the delay. If the traffic has high variability, it may result in a much higher delay over smaller time scales as compared to the long term delay. Hence, both the mean load *and* the variability of the traffic determine how routing performance, measured over short time intervals (10 minutes), differs from its expected value based on the long term average load (800 minutes). The goal of this section is to shed some light on how these different factors interact and ultimately affect routing performance. Especially since, as we show in the next sub-section, splitting traffic into finer streams *increases* their variability. This can potentially offset the advantage of a lower operating link load, and result in a higher delay over smaller time scales as compared to that obtained with coarser granularity streams.

#### A. Evaluating the Impact of Time Variability

In order to investigate this potential trade-off, i.e., between improved average performance and greater traffic variability, we first proceed to evaluate how traffic granularity affects its variability. In order to measure this, we compute the coefficient of variation of the traffic intensity of individual streams for different levels of granularity. The coefficient of variation per stream is computed over the 80 10-minute-interval measurements. This provides an indication of the traffic variability that exists, when considering 10-minute intervals and splitting traffic at different levels of granularity. The results are summarized in Figure 7, which clearly indicates that as the number of streams increases, i.e., from Mask 0 ( $p0$ ) to Mask 8 ( $p8$ ), so does the variability of individual streams.

This implies that the potential benefits of improved average performance may be offset by the impact of this greater stream variability, if it also translates into greater variability at the *link* level, i.e., after streams have been assigned to paths. In other words, although average link loads and, therefore, cost will be lower, short term link load variations need not be, and could even be larger, which may adversely affect network performance. In order to assess the relative effect of these two competing factors, we evaluated the network delay averaged over all the 80 10-minute measurement intervals, for 30 traffic matrices routed over the 10 networks used in Figure 6. The results are shown in Figure 8.

From comparing Figure 8 with Figure 6, we immediately observe that the benefits of lower average link loads and, therefore, delay, do not always translate into visi-

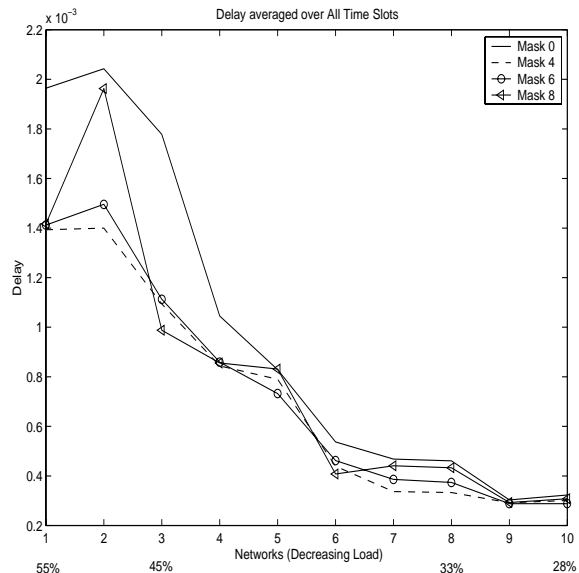


Fig. 8. Delay averaged over time slots .

bly better performance when time variability is taken into account. The figure shows the delay, averaged over the 80 10-minute time slots, for the different levels of traffic granularity under consideration, i.e., masks 0, 4, 6, and 8. The figure clearly shows that when routing is limited to a single stream (mask 0), performance is poor even on the shorter 10-minute time scale. This is because although link traffic may exhibit smaller short term load variations, the higher average link loads amplify those variations when it comes to delay because of the non-linear nature of the curve. As traffic granularity decreases, i.e., to masks 4 or 6, we observe that the resulting lower average link loads manage to also improve short term performance. This is because, despite the potentially larger short term traffic fluctuations, the lower overall average link loads sufficiently dampen the impact of those variations on short term delay. However, this improvement does not readily extend as traffic granularity decreases further. Specifically, we see that routing using mask 8 streams often results in worse average short term delays than when masks 4 or 6 are used. This is because, as seen from Figure 6, the improvement in average link loads that mask 8 streams afford, is marginal compared to what is achievable with mask 4 or 6. On the other hand, mask 8 streams exhibit higher short term traffic fluctuations that result in degraded average short term delays. Hence the higher short term variability more than offsets any advantage of the lower average link loads. Note that this behavior is observed even though, as shown in Table 2, the number of paths used with mask 8 is similar to what is used with masks 4 and 6. This indicates that although the number of paths is similar, the assignment of traffic (streams) to them is different.

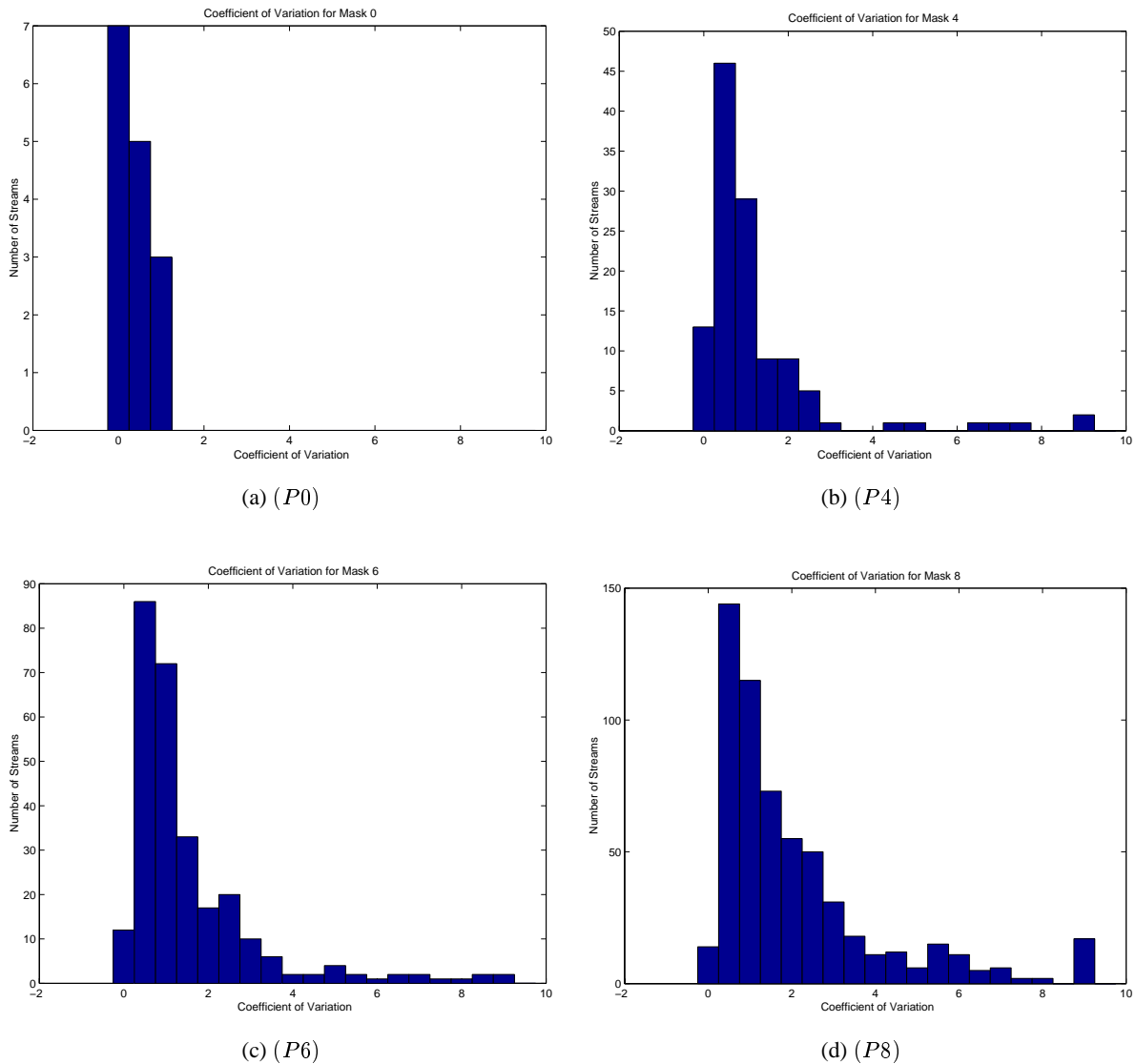


Fig. 7. Relation between Traffic Variability and Stream Granularity.

The findings of Figure 8 confirm the existence of a trade-off between short-term and long-term performance, and the use of fine traffic granularity to achieve lower average link loads. The figure suggests using the coarsest possible traffic granularity that achieves a “significant” decrease in average link loads, e.g., a mask of 4 in the current environment. Further reductions in traffic granularity result in only minor improvements in average loads, and the greater short term traffic variability they induce often becomes the dominant effect, worsening short term performance.

## V. CONCLUSION

We have investigated a new aspect of traffic aware routing in IP networks. Specifically, we explored the impact of traffic and time granularity on routing performance. Our performance measure was based on a traditional delay based cost function, but we expect comparable findings with other cost functions. The investigation was carried out using actual traffic and flow data collected on the Sprint operational Internet backbone.

One of our goals was to assess the relation that exists between routing performance and routing’s ability to split traffic across paths at different levels of granularity. The method used in the paper to generate different levels of traffic granularity was to apply masks of different sizes to

the destination IP address of packets. This was meant for illustration purposes as it provided a systematic way of varying traffic granularity; however, it is also meaningful to operators as it gives a general sense of blocks of customers that belong to the same subnet. We plan to extend this to other fields and combination of fields in the packet header, to study how different classification methods affect the relation between traffic granularity and variability. Ideally, we would like a classification scheme that lets us achieve fine granularity without significantly increasing variability. In particular, we investigated the trade-off that exists between improving long term performance by finer splitting of traffic across paths, and the potential degradation of short term performance that greater traffic variability can induce.

The main contributions of the paper consist of: (1) identifying the impact of traffic granularity on routing performance, and in particular establishing that the bulk of the improvement occurs with a relatively small number of streams; (2) designing and evaluating a routing heuristic (Heuristic 2) that approximates the performance of “optimal” routing reasonably well, while taking the constraints of traffic granularity into account; (3) observing that while finer granularity routing improved average performance, this did not always carry over to smaller time scales, as the greater variability of finer grain traffic often offset most of the resulting performance improvements.

We believe that the findings of the paper provide initial guidelines for deploying traffic aware routing, i.e., most of the benefits can be achieved using a small number of paths and relatively coarse traffic splitting. Furthermore, finer grain optimization may turn out to be detrimental to short term performance. This being said, we believe that finer grain routing has the potential to offer additional performance improvements, provided that the splitting of traffic can be done without significantly increasing traffic variability. We intend to design and evaluate new aggregation heuristics that directly incorporate both traffic and time granularities, in order to achieve those performance improvements. We are collecting more data from different POPs and will use them in this investigation.

#### ACKNOWLEDGEMENTS

The authors would like to thank Jordi Castro for his help in using the PPRN software package and applying it to the relatively large optimization problems of this paper.

#### REFERENCES

- [1] G. Apostolopoulos, R. Guérin, S. Kamat, and S. Tripathi. Quality of service based routing: A performance perspective. In *Proceedings of SIGCOMM*, pages 17–28, Vancouver, Ontario, CANADA, September 1998.
- [2] G. Apostolopoulos, R. Guérin, S. Kamat, and S. Tripathi. Improving QoS routing performance under inaccurate link state information. In *Proceedings of ITC'16*, June 1999.
- [3] G. Apostolopoulos, R. Guérin, S. Kamat, A. Orda, T. Przygienda, and D. Williams. QoS routing mechanisms and OSPF extensions. Request For Comments (Experimental) RFC 2676, Internet Engineering Task Force, August 1999.
- [4] G. R. Ash. *Dynamic Routing in Telecommunications Networks*. McGraw-Hill Telecommunications, 1997.
- [5] D. Awduche, L. Berger, D.-H. Gan, T. Li, G. Swallow, and V. Srinivasan. RSVP-TE: Extensions to RSVP for LSP tunnels. INTERNET-DRAFT, draft-ietf-mpls-rsvp-lps-tunnel-05.txt, February 2000. (work in progress).
- [6] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus. Requirements for traffic engineering over MPLS. Request For Comments (Informational) RFC 2702, Internet Engineering Task Force, September 1999.
- [7] D. Bertsekas and R. G. Gallager. *Data Networks*. Prentice Hall, Englewood Cliffs, NJ, 2nd edition, 1992.
- [8] R. Braden (Ed.), L. Zhang, S. Berson, S. Herzog, and S. Jamin. Resource reSerVation Protocol (RSVP) version 1, functional specification. Request For Comments (Proposed Standard) RFC 2205, Internet Engineering Task Force, September 1997.
- [9] A. Brodnik, S. Carlsson, M. Degermark, and S. Pink. Small forwarding tables for fast routing lookups. In *Proceedings of SIGCOMM*, Nice, France, September 1997.
- [10] S. Chen and K. Nahrstedt. An overview of quality-of-service routing for next generation high-speed networks: Problems and solutions. *IEEE Network Magazine, Special Issue on Transmission and Distribution of Digital Video*, 12(6):64–79, November/December 1998.
- [11] B. Fortz and M. Thorup. Internet traffic engineering by optimizing OSPF weights. In *Proceedings of INFOCOM'2000*, Tel Aviv, Israel, March 2000.
- [12] C. Fraleigh, S. Moon, B. Lyles, F. Tobagi, and C. Diot. High performance IP monitoring system with GPS synchronization. Technical report, Sprint Labs, July 2000. Available at ftp://www.sprintlabs.com/PUB/sbmoon/smop-007.ps.
- [13] V. Fuller, T. Li, J. Yu, and K. Varadhan. Inter-domain routing (CIDR): An address assignment and aggregation strategy. Request For Comments (Standard) RFC 1519, Internet Engineering Task Force, September 1993.
- [14] B. Gavish and S. Hantler. An algorithm for optimal route selection in SNA networks. *IEEE Trans. Commun.*, COM-31(10), October 1983.
- [15] A. Girard. *Routing and Dimensioning in Circuit-Switched Networks*. Addison-Wesley, 1990.
- [16] R. Guérin and V. Peris. Quality-of-service in packet networks: Basic mechanisms and directions. *Computer Networks*, 31(3):169–179, February 1999.
- [17] B. Jamoussi (Ed.). Constraint-based LSP setup using LDP. INTERNET-DRAFT, draft-ietf-mpls-cr-ldp-03.txt, September 1999. (work in progress).
- [18] M. Kodialam and T.V. Lakshman. Minimum interference routing with applications to MPLS traffic engineering. In *Proceedings of INFOCOM'2000*, Tel Aviv, Israel, March 2000.

- [19] T. Li and Y. Rekhter. A provider architecture for differentiated services and traffic engineering (PASTE). Request For Comments (Informational) RFC 2430, Internet Engineering Task Force, October 1998.
- [20] D. Mitra, J. A. Morrison, and K. G. Ramakrishnan. Virtual private networks: Joint resource allocation and routing design. In *Proceedings of INFOCOM'99*, New York, NY, March 1999.
- [21] J. Moy. OSPF Version 2. Request For Comments (Standard) RFC 2178, Internet Engineering Task Force, July 1997.
- [22] P. Raghavan and C. D. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.
- [23] C. Rosen, A. Viswanathan, and R. Callon. Multiprotocol label switching architecture. INTERNET-DRAFT, `draft-ietf-mpls-arch-06.txt`, August 1999. (work in progress).
- [24] K. S. Vastola. A numerical study of two measures of delay for network routing. M.S. Thesis, U. Illinois, Dept. Elec. Eng., Urbana, IL, 1979.
- [25] C. Villamizar. MPLS optimized multipath MPLS-OMP. INTERNET-DRAFT, `draft-villamizar-mpls-omp-01.txt`, February 1999. (work in progress).
- [26] M. Waldvogel, G. Varghese, J. Turner, and B. Plattner. Scalable high speed IP routing lookups. In *Proceedings of SIGCOMM*, Nice, France, September 1997.