Inter-Domain QoS Routing Algorithms

Samphel Norden, Jonathan Turner
Applied Research Lab
Department of Computer Science
Washington University, Saint Louis
{samphel,jst}@arl.wustl.edu

Abstract— **Quality-of-Service routing satisfies performance requirements of applications and maximizes utilization of network resources by selecting paths based on the resource needs of application sessions and link load. QoS routing can significantly increase the number of reserved bandwidth sessions that a network can carry, while meeting application QoS requirements. Most research on QoS routing to date, has focussed on routing within a single domain. We argue that since the peering links joining different network domains are often congestion points for network traffic, it is even more important to apply QoS routing concepts to inter-domain routing. BGP, the *de facto* standard for inter-domain routing provides no support for QoS routing, and indeed it facilitates the use of localized routing polices that can lead to poor end-to-end performance. This paper proposes a new approach to inter-domain routing for sessions requiring reserved resources. We introduce two specific routing algorithms based on this approach and evaluate their performance using simulation.**

Keywords: Quality of service, Reservations, Inter-domain, Routing

I. Introduction

The need for timely delivery of real-time information over local and wide area networks is becoming more common due to the rapid expansion of the internet user population in recent years, and the growing interest in using the internet for telephony, video conferencing and other multimedia applications. Choosing a route that meets the resource needs of such applications is essential to the provision of the high quality services that users are coming to expect.

In this context, it is important to distinguish *datagram* and *flow* routing. In datagram routing, packets of a session may follow different paths to the destination. In *flow routing,* all packets belonging to an application session follow the same path, allowing bandwidth to be reserved along that path, in order to ensure high quality of service. Because many thousands or even millions of packets are typically sent during a single application session, flow routing occurs far less often than datagram routing, making it practical to apply more complex decision procedures than can be used in datagram routing. The current internet follows the *datagram* routing model and relies on adaptive congestion control to cope with overloads. Internet traffic is forwarded on a best-effort basis with no guarantees of performance. This can result in wide variations in performance, resulting in poor service quality for applications such as voice and video. Furthermore, internet routing is typically topology-driven instead of being load-driven. This approach does not allow traffic to be routed along alternative paths, when the primary route to a destination becomes overloaded. While the application of load-sensitive routing to datagram traffic can cause hard-to-control traffic fluctuations, it can be successfully applied to flow routing, since reserved bandwidths sessions typically have holding times of minutes, effectively damping any rapid fluctuations in routes.

This work is supported in part by NSF grant ANI-9714698.

Most research in QoS routing has focussed on routing with a single domain. While the intra-domain problem is important, it is arguably even more important to address the QoS routing problem at the inter-domain level. The reason for this is that the peering links that connect distinct routing domains are often congestion points for network traffic. Managing the resource use at such points of congestion is clearly critical to providing end-to-end quality of service. Inter-doman QoS routing also raises new challenges that are not present in intra-domain routing. Since network operators consider their internal network configurations to be proprietary information, inter-domain routing must be done without detailed knowledge of the overall network structure. The large scale of the global internet also makes it impractical to distribute any highly detailed picture of the topology and resource availability in the overall internet.

The most prominent inter-domain routing protocol in the current internet is the Border Gateway Protocol (BGP). BGP is a path vector based protocol, where a path refers to a sequence of intermediate domains between source and destination routers. BGP suffers from a number of well-documented problems, including long convergence times [1] following link failures. BGP adopts a policy based routing mechanism whereby each domain applies local policies to select the best route and to decide whether or not to propagate this route to neighbouring domains without divulging their policies and topology to others. The immediate effect of the policy based approach is to potentially limit the possible paths between each pair of internet hosts. BGP does not ensure that every pair of hosts can communicate even though there may exist a valid path between the hosts. Also, since every domain is allowed to use its own policy to determine routes, the final outcome may be a path that is locally optimal at some domains but globally sub-optimal due to the lack of a uniform policy or metric used to find an *end-to-end* route. This point is highlighted by [2], [3], where a majority of paths that are picked by BGP do not represent the optimal end-to-end paths. Most domains eventually default to *hot potato* routing, in which each network in the end-to-end path, tries to shunt packets as quickly as possible to the next network in the path, rather than selecting routes that will produce the best end-to-end performance for users. This characteristic is clearly undesirable, even for datagram traffic, and is particularly problematic for sessions that require high quality of service.

Before discussing specific approaches to address these problems, we review the critical issues that need to be considered in the design of new QoS routing protocols.

Routing State: *Local* or *global* state can be maintained by routers. Local state refers to the status of the links connecting a router to all its neighbours. Global state refers to the state of all routers and links in the network. Global state is accummulated gradually via router updates. In a large network, the state information that is available at a router may be *stale* due to changes in network traffic. This can adversely affect routing decisions. In large networks, it may be infeasible to maintain complete global state, making it necessary for routing algorithms to operate with only a partial view of some parts of the network.

Routing Updates: In order to maintain state information, routers must exchange state information from time to time. Updates may be periodic or may be triggered by changes in network traffic. Sending updates too infrequently has been shown to adversely affect the performance of the routing due to the accumulation of stale information [4], [5]. At the same time, too frequent updates can result in excessive routing overhead. Triggering updates following significant changes can be an effective alternative, but care is required to ensure that rapid changes in traffic don't cause excessively high update rates.

Multi-path routing: Typically, QoS routing algorithms find a single "shortest" path using an appropriate QoS metric, and all data packets are routed on that path. However, there are schemes that choose multiple paths [6], [7] and reserve resources on all the paths, resulting in packets being transmitted on multiple paths. Other multi-path routing schemes maintain a set of paths for each source-destination pair and select the best path from this set on demand. [7] shows that multi-path routing can provide significantly better performance than single path routing.

User-centric path selection: A routing protocol that seeks to provide the best performance for users is preferred over one that encourages locally optimal routing policies that can produce poor end-to-end routes.

Privacy and scalability: For reasons of both privacy and scalability, information exchanged between domains must be limited.

Fast reaction to traffic changes: A routing protocol should be able to track changes in network configuration and traffic quickly enough to ensure selection of the best available route for a flow.

New Inter-domain QoS Routing Algorithms: Our strategy for inter-domain routing has two parts. In the *inter-domain part*, a loose source route is selected by the router at origination point of the session. This source route specifies the domains through which the route is to pass and the *peering links* used to pass from one domain to the next. Within each domain, paths are selected between the ingress and egress points, using domain-specific routing policies. This is referred to as the *intra-domain part*. This decomposition of the end-to-end routing problem respects each domain's right to maintain the privacy of its internal network configuration and appropriately limits the amount of information that must be taken into account when selecting routes. At the same time, it allows the large-scale characteristics of the route to be selected with appropriate consideration of the status of the peering links. It should be noted that BGP based approaches specify only the domain in the path vector.

In this paper, we study two inter-domain QoS routing algorithms that follow this overall strategy. The first, uses a fairly conventional shortest-path framework, using a cost metric that accounts for both the intrinsic cost of each link and the amount of bandwidth that the link has available for use. The second dynamically probes several paths in parallel, in order to find a path capable of handling the flow. This approach eliminates the need for regular routing updates, since routing information is obtained on-demand.

There is a significant amount of prior research in the area of *intra-domain* QoS routing. However, the design criteria for *inter-domain* QoS routing is different from *intra-domain* routing, with a special emphasis on scalability. There is an inherent tradeoff between performance and scalability. We believe that this is one of the first papers that extensively evaluates the performance of QoS routing protocols in both the intra and inter-domain context and quantitatively evaluates the tradeoff, in addition to describing new scalable, high performance algorithms for inter-domain QoS routing.

In Section II, we introduce intra-domain versions of our two routing algorithms, and present simulation results characterizing their performance on a realistic network configuration. In Section III, we extend both algorithms to the inter-domain setting, and in section IV, we study their performance in depth.

II. Algorithms for Intra-domain QoS Routing

In this section, we describe and evaluate two QoS routing algorithms in the intra-domain setting before proceeding to the more general inter-domain setting.

A. Least Combined Cost Routing Algorithm (LCC)

The Least Combined Cost Routing (LCC) algorithm selects a route for a flow reservation by selecting a least cost path at the source router where the reservation request first enters the network, then forwarding the reservation request along this path, reserving resources at each hop. If at some point on the path, the selected link does not have sufficient capacity for the reservation, then the reservation is rejected. The cost metric used in the path computation takes into account both the intrinsic cost of the links (which we characterize here by geographic distance) and the amount of available bandwidth relative to the reservation bandwidth. It requires an underlying routing information distribution algorithm, to periodically update the necessary link state information. We assume that the link state update includes the available link capacity in addition to reachability information. Each router periodically computes shortest path trees to the other routers in the network, based on the received routing updates. These precomputed shortest path trees are used at flow setup time to determine the best path to the destination.

The cost metric is motivated by the observation that whenever the network is lightly loaded, paths should be selected to minimize the sum of the intrinsic link costs, since this minimizes the cost of the network resources used. When some links are heavily loaded, we want to steer traffic away from those links, even if our most recent link state information indicates that they have enough capacity to handle the flow reservation being setup. The reason for avoiding such links is that in the time since the last link state update, the link may have become too busy to handle the reservation. Rather than risk setting up the reservation on a path with a high likelihood of failure, we would prefer a longer path with a smaller chance of failure.

Term	Explanation
B	Available bandwidth on a link
R	Reservation bandwidth
$L[u,v]$	Length of link joining routers u and v
M	Bandwidth ”margin” where $M = B - R$

TABLE I
NOTATION FOR COST METRIC

Previous studies suggest several variants for a path cost metric including the sum of link utilization, bottleneck bandwidth, etc. Defining the path cost as the sum of link utilization reduces the blocking probability and results in less route oscillation by adapting slowly to changes in network load [8]. Other studies show that assigning each link a cost that is exponential in the current utilization results in optimal blocking probability [9]. The LCC metric has similar elements but is not directly based on these results.

Table I lists several key pieces of notation used in the link cost expression shown below which is referred to as the *Combined Cost Metric (CCM)*.

$$C[u,v,R] = \{L[u,v] + \alpha(\max(0, g - M))^{\beta}\} \times R \qquad (1)$$

The three parameters, α, β and g determine how the cost of a heavily loaded link increases. Specifically, if the link's *margin* (the amount of available bandwidth remaining after subtracting the bandwidth required by the reservation) is greater than g, then the cost of the link is equal to its intrinsic cost, which we characterize here by its length. If its margin is less than g, then its cost increases as the margin shrinks (note the margin may be less than zero). g should be chosen to reflect the likelihood that in the time between the last link state update and the arrival of a reservation, that the link has become too busy to handle the reservation. Specifically for margins of g or greater, the probability of making a bad route selection based on stale link state information should be small, say 1-5%. If the average reservation bandwidth is a small fraction of the link bandwidth, then a reasonable choice for g would be several times the average reservation bandwidth. The parameter β determines how rapidly the cost grows as the margin drops. In the simulation results reported in the next section, β is set to 2, giving quadratic growth.

To determine a reasonable choice for the scaling parameter α, consider the appropriate cost increment in a situation where the margin is equal to zero. Note, that in the time since the last link state update, the “true” margin may have either increased or decreased. If we assume that both possibilities are equally likely, then the added cost when the margin is zero should balance the cost of the two different “incorrect” routing decisions that are possible. A decision to use a path with a zero margin link is incorrect, if that link no longer has enough bandwidth to accommodate the reservation. A decision to not use a path with a zero margin link is incorrect if the link actually does have sufficient capacity for the reservation. The cost of the first type of incorrect decision is that the reservation is rejected. The cost of the second type of incorrect decision is that a longer, higher cost path is used, wasting network resources. This added cost is $\alpha g^{\beta} R$. We equate the cost of rejecting a reservation request to the cost of the resources that the reservation would use if it were accepted and used a minimum length route. If this minimum route length is D, then the cost of rejecting the reservation is DR. Setting this equal to $\alpha g^{\beta} R$ and solving for α gives $\alpha = D/g^{\beta}$. To avoid the implied requirement to calculate D and α for each reservation, we simply specify α based on a typical value of D.

B. Parallel Probe Algorithm (PP)

The LCC algorithm, like most conventional routing algorithm, relies on the regular distribution of routing information throughout the network. One drawback of this approach is that routers must maintain a great deal of information, much of which is never used. Indeed, if no reservation consults a particular piece of routing information before the next update replaces it, then that piece of routing information served no purpose, and the effort spent to create it was wasted.

The Parallel Probe (PP) algorithm takes a different approach. Rather than maintain a lot of dynamic routing information, it sends *probe packets* through the network to collect routing information as it is needed . This means that no extraneous routing information must be maintained. Only that information that is relevant to the selection of paths for actual flow reservations is required.

The PP algorithm uses a precomputed set of paths for each source-destination pair. Probe packets are sent in parallel on all of these paths to the destination, and are intercepted by the last hop router. As the probe packets pass through the network, each router on the path inserts a field specifying the available bandwidth on its outgoing link. This operation is simple enough to be implemented in hardware, allowing probes to be forwarded at wire speed.

When the probe packets reach the last hop router, it selects the best path for the flow, based on the information received. Each probe packet includes a field indicating how many probes were sent, allowing the last hop router to easily determine when it has received all the probes, in the normal case where all probes are received. If one or more probes is lost, the last hop router will proceed following a timeout. The last hop router selects the shortest path for which the bottleneck bandwidth is at least equal to the reservation bandwidth, if there is one or more such path. If there is no such path, the reservation is dropped.

If the last hop router selects a path with a large enough bottleneck bandwidth to handle the reservation, it sends a reservation message back along the selected path to the origination point, reserving resources as it goes. If in the short time since the probe packet was forwarded, a link has become too busy for the reservation, the reservation attempt fails and all reserved resources are released. It should also be noted that: 1) Precomputation is done rarely since the network topology is relatively static and changes at long time intervals; 2) Routes are computed using static information (path lengths) about the network topology, which makes the routes relatively robust to network fluctuations.

We precompute the alternate paths using a simple algorithm outlined as follows. Initially, we find the shortest path (using link length or hop count) between a given pair of routers in the

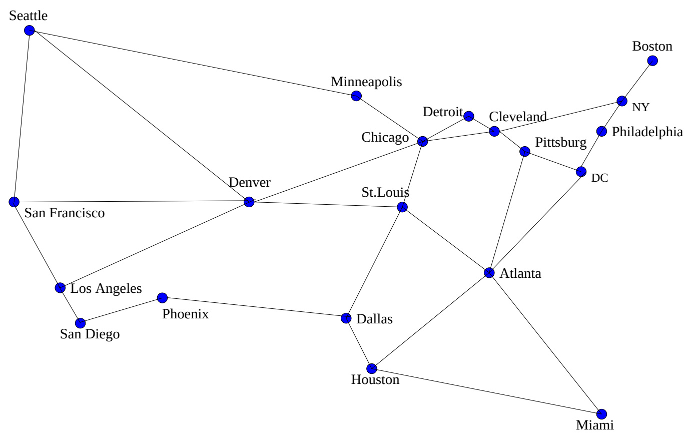

Fig. 1. National Network Topology

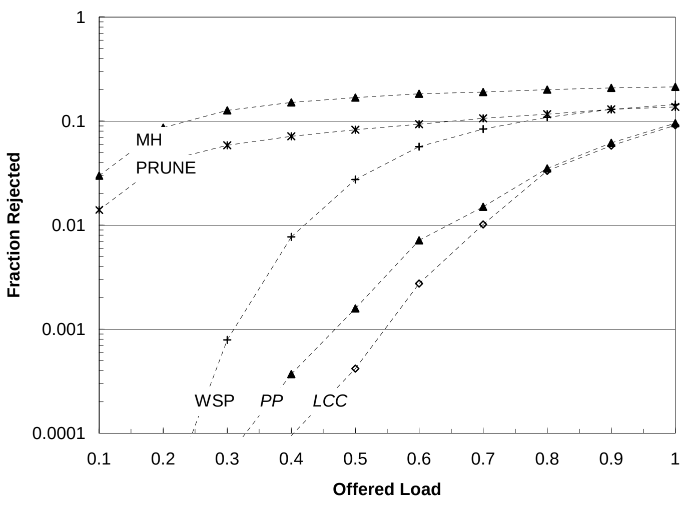

Fig. 2. Rejection Fraction of the QoS Routing Protocols

network and use this as a baseline metric. We then take every intermediate node and verify if the path length via the intermediate node is within some bound of the baseline path and is distinct from the baseline path. If so, we add this path to the set of alternate paths. We also restrict the number of alternate paths so as to minimize the number of probes that are sent. It should be noted that there may be other mechanisms to precompute the paths such as ensuring that paths do not share bottleneck links. However, we have adopted an approach that is simple and is not subject to fluctuations in network state, facilitating fast deployment.

In the next section, we will describe a simulation environment that represents a typical ISP network. We will initially show results for the aforementioned routing protocols on this simple network. We then extend the network design mechanism used to construct this ISP network to build a hierarchical network that represents multiple autonomous systems. We will subsequently show simulation results on this inter-domain topology.

C. *Results for Intra-domain QoS Routing*

We now present simulation results for the LCC and PP routing protocols for the *intra-domain context*. The network configuration for this simulation study was chosen to be representative of a real wide area network. This network has nodes in each of the 20 largest metropolitan areas in the United States (see Fig. 1). The traffic originating and terminating at each node was chosen to be proportional to the population of the metro area served by the node, and the traffic between nodes was also chosen, based on the populations of the two nodes. This leads to the sort of uneven traffic distribution that is typical of real networks. The links in the network are also dimensioned to enable them to carry the expected traffic. Dimensioning links to have appropriate capacity is important for a realistic study of routing, since a badly engineered network can easily distort the results, leading to inappropriate conclusions about the relative merits of different routing algorithms. The link dimensioning was carried out using the constraint-based design method developed by Fingerhut in [10]. The network was modelled after a similar design described in [11]. The link dimensioning process results in a wide range of link capacities, with the largest capacity link being 32 times as large as the smallest.

In the simulations, reservation requests arrive at each node, at rates that are proportional to the population of the area served by the node. The interarrival times and the reservation holding times are exponentially distributed. Uniform reservation bandwidths are used. The destination of each reservation is chosen randomly, but with the choice weighted by the relative population size of the possible destinations. Several numerical parameters are varied in the simulations. Each parameter has a *default value*. Whenever one parameter is varied in a given chart, the other parameters are assigned their default values. One key parameter is the bandwidth of the reservations, relative to the bandwidth of the smallest capacity network link. This *link fraction* is assigned a default value of .05. The default update period (time between state updates) is $MHT/2$ where *MHT* is the mean holding time for the flows. The default number of alternate paths on which PP sends probes is 6.

Blocking Probabilty: Figure 2 compares the blocking probability observed for several different routing algorithms, including the LCC and PP algorithms. The others include a minimum hop algorithm (MH), a variant of MH that first removes links that lack the bandwidth to carry the reservation (PRUNE) and the widest shortest path (WSP) algorithm. The widest shortest path first (WSP) algorithm [12], [13] maintains a set of alternate paths between every source and destination arranged in increasing order of hop count. When a request arrives, the path with the largest bottleneck bandwidth is picked from the set of paths with the shortest hop count.

Fig. 2 shows that the LCC and PP algorithms significantly outperform the other algorithms, with LCC performing slightly better than PP. The MH and PRUNE algorithms perform much worse than the other algorithms, showing that QoS routing can provide substantially better performance than conventional methods. traditional routing. Additional results for these algorithms can be found in reference [14].

D. *Comparison of Intra-domain QoS Routing Algorithms*

The LCC and PP protocols presented above represent two extreme approaches. The *LCC* protocol represents link state protocols such as OSPF. The *PP* approach represents a hybrid multi-

path routing scheme. We examine these protocols with respect to essential routing metrics such as the *call setup overhead*, *message overheads*, *Router processor complexity*, and *Robustness*.
Call Setup Overhead: The *PP* algorithm also has low setup time since probes simply query hardware port processors using precomputed paths. While there is additional processing at the last hop, the procedure effectively takes a round trip time. The *LCC* protocol has the longest call setup time since it computes the shortest path on-demand.
Message Overhead: From the *message overhead* perspective, LCC sends a single reservation request on the chosen path. The *PP* algorithm sends probes on k paths incurring a slightly higher overhead. However, since no resources are reserved in the forward pass, this does not lead to any wastage.
Router processor complexity: refers to the complexity in processing information from other routers as well as processing reservation requests. *LCC* has relatively low complexity requirements since residual link bandwidth is the only state information that needs to be exchanged. *PP* does not require any overhead for distributing and processing link state, since the probes obtain the required information on demand. However, *PP* does require that routers be capable or processing probe packets in the data path, preferably in hardware.
Robustness: From a robustness perspective, the PP algorithm is more resilient to link failures due to the intrinsic alternate path mechanism that allows a router to choose an alternative either statically from the set of paths, or by dynamically probing the alternate paths from the point of failure. LCC on the other hand would require recomputation of the shortest path from the point of failure.

III. Inter-domain QoS Routing

In this section, we show how the LCC and PP algorithms can be generalized to the inter-domain routing context. As mentioned earlier, we adopt a two part strategy. In the inter-domain part, a loose source route is selected. This route comprises a list of domains and the peering links used to pass between domains. Each domain routes the flow within its own boundaries using whatever intra-domain algorithm it chooses to use, but the ingress and egress points remain fixed.

A. Inter-domain Version of the LCC Algorithm

The objective of an inter-domain routing algorithm is to select a loose source route, joining a flow's endpoints. The route is selected with the objective of ensuring a high probability of successful completion, while minimizing the use of network resources. The status of the peering links is a key element of the route selection. Since peering links are often congestion points for network traffic, careful selection of peering links can have a significant impact on the probability of success. Because the inter-domain route selection must be done without the benefit of detailed knowledge of each domain's internal configuration, it's necessary to estimate the amount of resources that a flow will consume within a domain.

The inter-domain LCC algorithm includes two parts. One part distributes information about the connectivity among the various domains and the peering links that join domains. The peering link information includes the intrinsic costs of the peering links (characterized here by the links' physical length) and their available capacity. Peering link information can be aggregated to improve scalability, but we do not address the aggregation problem here. The network status information is distributed to all domains, allowing routers to select routes for flows based on their knowledge of the current network status. The second part of the inter-domain LCC algorithm makes per-flow routing decisions. Conceptually this is done by computing a least cost path between the endpoints. The cost of a path is defined to be the cost of the peering links on the path (computed using the combined cost metric, introduced in the previous section), plus the estimated cost of the segments within each domain on the path. We investigate the performance of the LCC algorithm for two estimation methods, which are described below.

- *Geographic Estimation (GEO)* The geographic estimation method uses the geographic length of the path segment within a domain, as the estimated cost. This implies that the distributed state information include the geographic coordinates of the routers at the ends of peering links. Because this information is static, it need not be updated frequently.
- *True Value Estimation (TRU)* True value estimation is not so much a practical estimation method, as it is a benchmark for bounding the performance of this class of algorithms. In this method, we assume that the cost of the path within each domain is calculated using knowledge of the underlying network structure. The combined cost metric (Equation 1) is used to determine the path costs within each domain.

In most of the simulation results reported in the next section, the inter-domain LCC algorithm is combined with the use of LCC at the intra-domain level, as well. Note however, that the use of LCC at the inter-domain level does not require the use of LCC (or any other specific algorithm) at the intra-domain level

B. Inter-Domain Version of Parallel Probe

In this section, we extend the PP algorithm for interdomain QoS routing. As in the intra-domain context, the PP algorithm involves the transmission of probe packets from the origination point of a flow along several pre-computed inter-domain paths to the destination point for the flow. These inter-domain paths are loose source routes, which specify the domains to be traversed and the peering links used to pass between domains. As the probes pass along the path, they gather status information that is used by the router at the destination end, to determine the best path for the flow to take. The collected status information includes the available bandwidth on the peering links, and an estimate of the available bandwidth on the path segments within the individual domains.

In keeping with our overall framework, different routing methods may be used within the domains. However, here we focus on the case where parallel probe is used at the intra-domain level, as well as at the inter-domain level. To describe this combined algorithm clearly, we distinguish between *macro-probes*, which are used for inter-domain routing and *micro-probes* which are used within domains.

In the combined algorithm, macro-probes are launched by the router at the origination point of the flow, and the passage of macro-probes through the internet triggers the transmission of micro-probes within each domain. More precisely, when

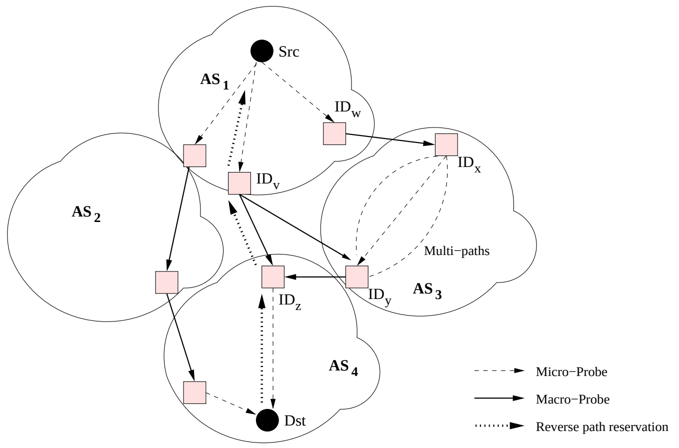

Fig. 3. Multipath routing using PP

a macro-probe arrives at the ingress router for a domain, the ingress router launches several micro-probes which pass through the domain to the egress router specified in the inter-domain path contained in the macro-probe. The egress router collects the information in the micro-probes, selects the best path and then forwards the macro-probe along the inter-domain path, after adding to it, information describing the available bandwidth on the selected intra-domain path.

When the macro-probes are processed at the terminating router for the flow, it selects the best path and forwards a reservation packet back along the loose source route. The intra-domain portions of the path are filled in by the individual domains, based on the previously selected micro-probe. This implies of course that the egress router retain this information for a short period of time, so that it is available if and when the reservation packet comes back along the path. Note that this process maintains the private nature of the intra-domain path segments. The only thing communicated outside the domain is the available bottleneck bandwidth on the chosen path.

The loose source routes used for the macro-probes are precomputed, based on overall knowledge of the network configuration, but without reference to current traffic conditions. We use the same precomputation algorithm described for the intra-domain routing. The information stored at routers is location-dependent:

Internal Router: A router within a domain stores alternate paths to every edge router in that domain. If there are a relatively large number of edge routers, the internal router can store a single path to each edge router.

Edge Router (ID): We designate a router that connects to other peering routers in other domains as an ID or *Intermediate Destination router*. An ID stores alternate paths to other ID's within a domain. The ID also stores paths to other ID's in other domains.

We show a sample request and the probes generated by it in Figure 3. We will trace out a sample end-to-end path as it traverses domains or autonomous systems (AS). Initially, *micro-probes* sent out from the source reach all the IDs within the AS (For example, reaching $[\mathrm{ID}_w, \mathrm{ID}_v]$). The ID's store the path included so far locally and only advertise the bottleneck bandwidth or related cost metrics to other domains, repsecting the privacy constraints of the domain. Next, *macro-probes* are spawned to other domains ($\mathrm{ID}_w \rightarrow \mathrm{ID}_x \in \mathrm{AS}_3$). ID_x in turn spawns micro-probes to reach ID_y. ID_y stores the winning probe path locally and then spawns a *macro-probe* to $\mathrm{ID}_z \in \mathrm{AS}_4$. ID_z receives macro-probes from two different domains $[\mathrm{AS}_1, \mathrm{AS}_3]$, and selects the better option which is the probe from ID_v in our example. It spawns micro-probes which finally reach the destination. After a short period, the winning end-to-end probe is selected and reservation is initiated on the reverse path (Dst $\rightarrow \mathrm{ID}_z \rightarrow \mathrm{ID}_v \rightarrow$ Src).

To recapitulate, *macro-probes* on reaching IDs in other domains result in the spawning of *micro-probes* in order to find the optimal intra-domain path to reach an egress ID. Once an egress ID is reached and an intra-domain path chosen from the various micro-probes, a macro-probe is recreated to reach the next domain. This process repeats till the destination is reached. A reverse probe then establishes the reservation by reversing the path information stored by the winning probe. Note that the winning probe carries a loose source route comprised of IDs. The reverse reservation will need to expand on this to select the route within a domain as it traverses back to the source. In summary, the salient features of our scheme are:

Flexible Routing Metrics: Each probe can be ”injected” with a custom routing policy or QoS metric which it uses to find the best possible path as it traverses across heterogeneous ASes.
Privacy across ASes: The probe works across ASs by collapsing and expanding source routes at edge routers. This allows privacy of router topologies between ISP's. As the probe enters an ISP (ingress edge router), it uses the intra domain path information stored at the ingress node. This information is then stored at the egress node when the probe leaves the ISP.
Optimal Route selection: PP uses a larger search space that searches multiple alternatives between ISP's as opposed to the single route maintained by BGP. Also, the link information is collected by the probe on-demand as opposed to using stale link and forwarding state in current routing approaches. More importantly, all probes use the same metric as they span domains allowing for a unified end-to-end path that is consistently optimal. This is the main distinguishing feature from BGP that leads to significant performance gains for PP as confirmed by our simulation results.
Rerouting: A probe when blocked can retrace itself to the nearest ingress router for that domain and inform the router of a failed path as well as a failed link. This allows network management operations to be inherently incorporated using PP allowing for a more robust solution and fast recovery from link failures.
Scalability: The only information exchanged is between the peering nodes as before and this reduces the overhead significantly. Also, paths are precomputed and there is no computation on-demand apart from transmitting probes.

IV. Results for Inter-domain QoS Routing

In this section, we will first describe the design of an inter-domain topology to evaluate the performance of the inter-domain QoS routing algorithms. As before, we dimension links according to traffic constraints between nodes similar to the intra-domain design.

A. Design of an Interdomain Topology

In this section, we describe the design of an inter-domain routing network that can be used to realistically evaluate the performance of the QoS routing protocols. The basis for network design is similar to the design of the intra-domain routing network described in Section II-C. We chose the 30 largest metropolitan areas in the United States. There are two basic kinds of network providers in this topology: *National* and *Regional* ISP's. We use the following heuristics to pick members of either ISP:

National ISP: A city is considered a member of a national ISP with a probability $p = 0.8$.

Regional ISP: Once a region is decided for a regional ISP, we locate the approximate center of the region and find the 10 closest cities in order of distance. We then pick these cities to be a member with probability p. We use a distribution of 2 National ISP's, and 6 Regional ISP's. The national ISP's are complete graphs and cover 80% of the network (25 nodes). Among the regional ISP's, there are 3 best star topologies, 2 delaunay triangulations and 1 complete graph topology . A delaunay triangulation [15] topology allows parallel paths between nodes, while minimizing the number of such parallel paths allowing for a cost-effective topology. This diverse mix of topologies allows the simulation results to be applicable to a general topology as opposed to a particular topology. We use a constraint-based design method similar to the approach for the intra-domain routing topology. Since traffic can now either be sent within a domain or across domains, we separately dimension the links for intra and inter-domain routing, and take the sum of the dimensions for the overall network. In the simulation, we ensure that intradomain traffic is restricted to use the links within the domain, and not use peering links.

As before, reservation requests arrive at each node, at rates that are proportional to the population of the area served by the node. The interarrival times and the reservation holding times are exponentially distributed. Uniform reservation bandwidths are used. The destination of each reservation is chosen randomly, but with the choice weighted by the relative population size of the possible destinations. The traffic in this case is distributed not just among a set of destination nodes, but also among a set of destination domains. The link capacities vary with the smallest being 80 Mbps and the largest being 15 Gbps. With such a large variance, we use a default bandwidth of 20 Mbps rather than a fixed link fraction. The mean holding time (MHT) is also increased to 120 time units. The update period is 30 time units. The default number of alternate paths on which PP sends probes is 6.

Call Blocking Probability: Figures 4-7 show the performance of the QoS routing schemes. The overall rejection fraction results (Figure 4) show that PP is clearly superior to TRU (factor of 2 improvement at 10^{-3}) and the GEO algorithm (factor of 8 improvement at 10^{-3}). It is surprising that the PP algorithm is able to outperform TRU inspite of TRU calculating a shortest path on-demand for every request. The TRU algorithm still relies on periodic link state updates and the use of stale link information can result in non-optimal path selection. It should also be noted that with the requested bandwidth (20 Mbps) which is almost $1/4^{th}$ of the smallest link capacity, it is easier to saturate a link by choosing a non-minimal path. The PP algorithm also is able to pick alternate paths that do not have bottleneck links in common allowing for better load distribution. It is also possible that the TRU algorithm picks longer paths than necessary (no upper bound on path length) increasing the chance of reservation failure as well as placing extra load that could affect other connections. Interestingly, the LCC algorithm was marginally better than the PP algorithm in the smaller ISP topology for intra-domain routing. The lack of distinct paths collected by the precomputation algorithm of PP allows LCC to perform better than PP. However, this is not true for larger networks and inter-domain routing as shown by Figures 5 and 6. Figure 5 shows the various alternate paths with choice 1 being the default shortest path. As expected, most requests choose this option and other alternatives share links with this path eliminating them from being chosen frequently. With a larger topology as in the inter-domain case, the path choices are more distinct with less sharing of links allowing for a more even distribution of requests as seen in Figure 6 leading to a higher performance for PP over TRU.

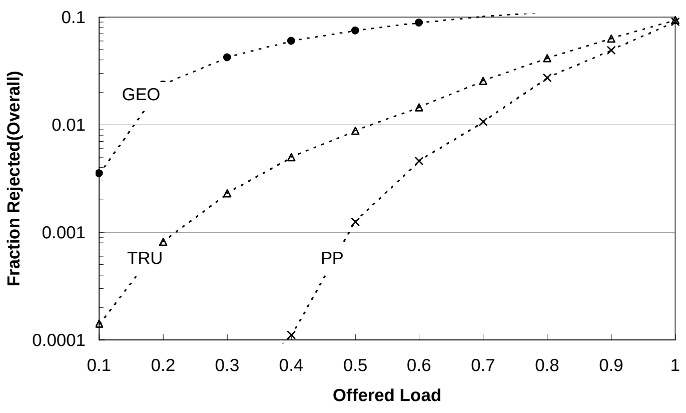

Fig. 4. Rejection Fraction for Interdomain QoS Routing Protocols

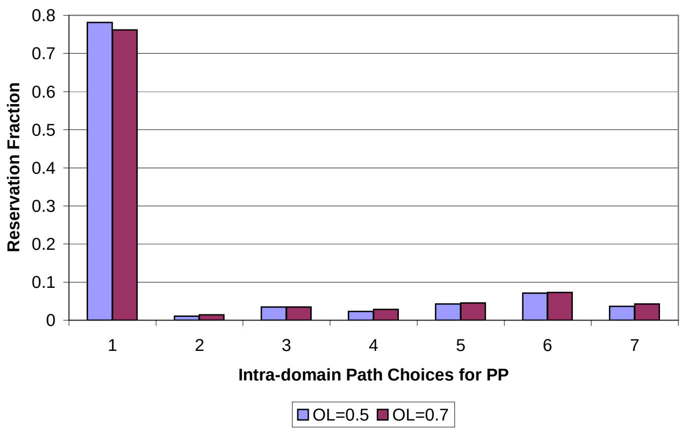

Fig. 5. Request Distribution over individual paths (Intra-Domain)

The GEO algorithm is signficantly worse than both TRU and PP. Recall that the GEO algorithm first uses geographical distances to pick peering nodes, and then uses the LCC algorithm within the domain. The use of geographical distances assumes that physical links follow the virtual geographical links which is not the case.

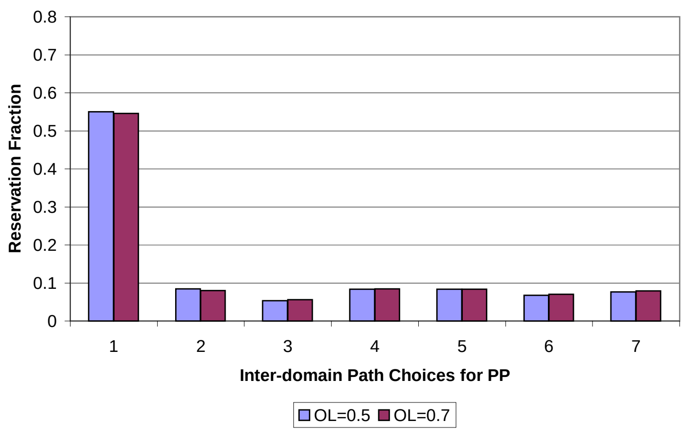

Fig. 6. Request Distribution over individual paths (Inter-Domain)

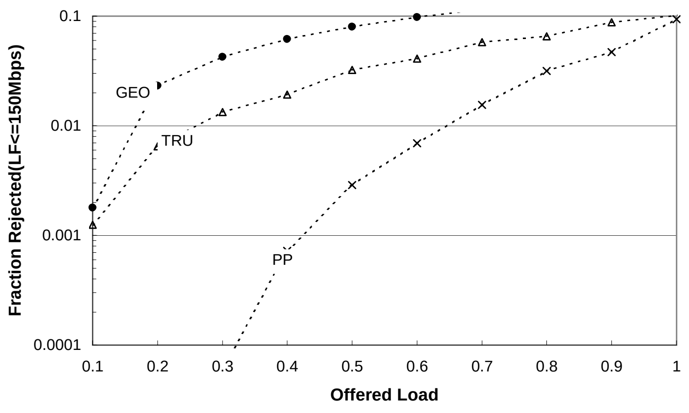

Fig. 7. Rejection Fraction for Interdomain QoS (Low Bottleneck Bandwidth)

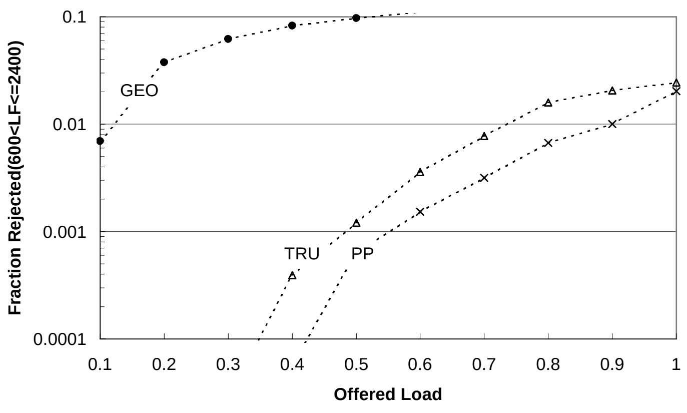

Fig. 8. Rejection Fraction for Interdomain QoS (Large Bottleneck Bandwidth)

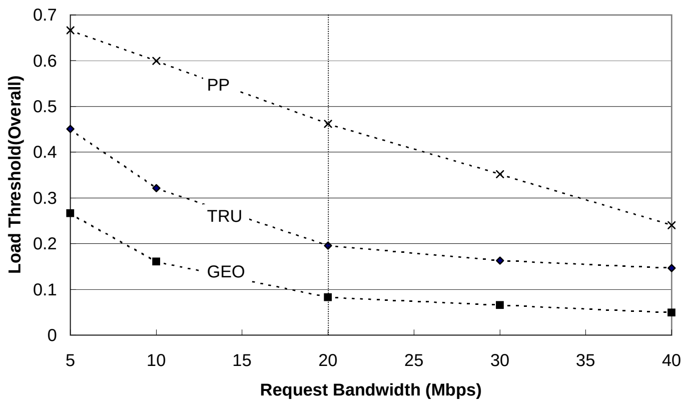

Fig. 9. Variation of Threshold with Request Bandwidth Size

Figure 7 shows that PP provides significant gains for paths with small bottleneck bandwidths (≤ 150 Mbps), by distributing the load more effectively and reducing the probability of saturating these paths. For paths with larger bottleneck bandwidths (between 600 Mbps and 2.4 Gbps), the gains for PP over the other schemes are consequently less as shown in Figure 8, indicating that the combination of stale link state and small links can cause traditional shortest path protocols to perform poorly compared to a dynamic multipath protocol like the PP algorithm.

Effect of Larger Bandwidth Requests: Figure 9 shows the impact of increasing the request bandwidth on the load threshold. The *load threshold* is the maximum load that can be supported at a rejection fraction of 10^{-3}. PP as expected supports a load of 0.48, TRU a load of 0.2 and GEO a load of 0.1 at the default value of 20 Mbps. The load threshold decreases as the request bandwidth size is increased.

Degrading Peering Link Bandwidth: Figure 10 plots the ratio of the rejection fractions of GEO and TRU algorithms at a load of 0.4, when varying the bandwidth of peering links. At a degradation factor of 0.5, the link capacity of all peering links is halved. As a lower degradation factor, the peering links are saturated quickly and the choice of the peering link is not as important. This affects the end-to-end TRU algorithm which uses the combined cost metric (Equation 1) throughout. Since the GEO algorithm uses the cost metric only for the peering links and the peering links are saturated in a short time, the performance of both TRU and GEO will approximately converge at low degradation factors. Thus, the ratio of their rejection fractions is lower. As we increase the peering link bandwidth, the TRU algorithm starts to outperform GEO and the ratio of the two increases.

Impact of Update Period: Figure 11 shows the impact of varying the update period on the load threshold for both TRU and GEO. As expected, the threshold decreases with larger updates due to the use of stale link state information by the route selection mechanism. Both TRU and GEO fall by significant amounts as the update period becomes as large as the MHT.

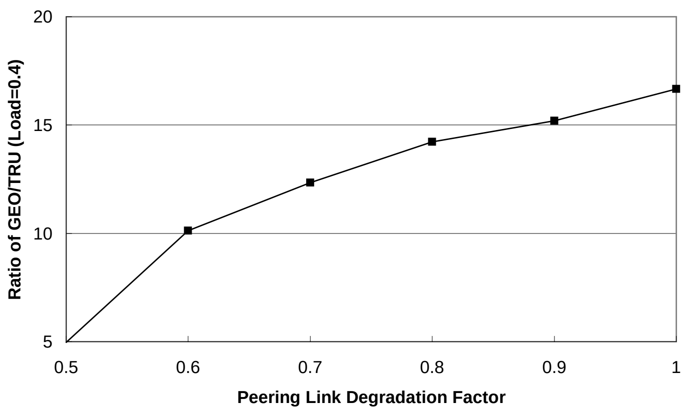

Fig. 10. Impact of Reducing Peering Link Bandwidth (Inter-domain)

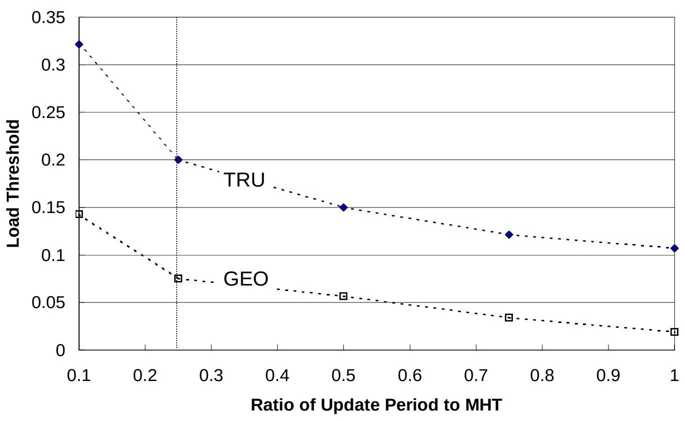

Fig. 11. Variation of Threshold with Update Period (Inter-domain)

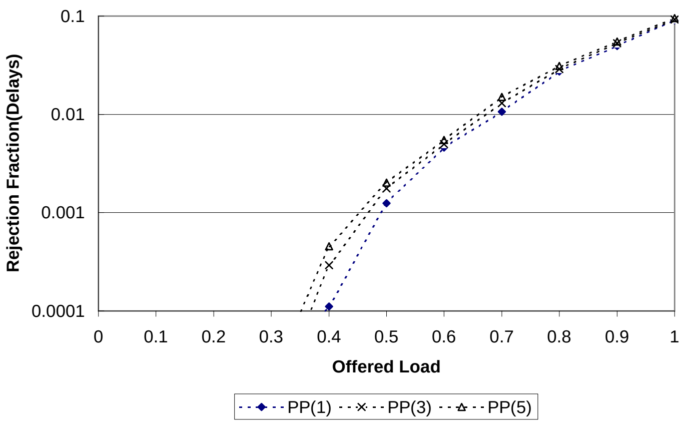

Fig. 12. PP with Propagation Delays

Note that the PP algorithm uses dynamic probing and does not rely on link state updates.

Routing Delays: Figure 12 shows the impact of propagation delay of control messages on the PP algorithm. One drawback of PP is that if two requests originating at different nodes initiate probes (say r_1 and r_2) and share paths such that one possible path for r_2 is a subset of a possible path P for r_1. Let us further assume that probing for r_1 is initiated before r_2, but r2 initiates the reverse reservation before r1 on the subset of path P. Finally, if we assume that the path P is chosen for r_1 and the reverse reservation is subsequently initiated. It may happen that the reverse reservation for r_1 could fail as resources were taken away by r_2. Unfortunately the last hop router on the path P is unaware of the other request r_2. While it is easy to circumvent this problem by installing some sort of soft state in the forward pass of the probe, we show that this problem does not affect the performance of PP. We modify the PP algorithm to include propagation delays for the probes as they traverse the path. Obviously, the propagation delays are significantly smaller than the link state updates which are only sent periodically. Thus, we show the PP algorithm for delays of 3 and 5 units compared to the baseline PP algorithm in Figure 12. It is clear that the delays do not impact the performance of the PP algorithm.

V. Enhancing the Routing Algorithms

We will now describe enhancements to the aforementioned QoS routing algorithms that explore the tradeoff between performance and algorithm complexity/scalability. We have the PP and GEO algorithms at two extremes. The PP algorithm shows the best performance, but also incurs a sizeable overhead due to probes being sent, as well as requiring that all domains use the algorithm consistently for inter and intra domain routing on the end-to-end path. Both of the above do not make the algorithm scalable to large networks with diverse autonomous systems. The GEO algorithm on the other hand uses purely static information that can be obtained without any privacy constraints between domains and is easy to deploy. However, it has significantly worse performance than PP. We seek to improve both algorithms in different ways.

A. Improving Performance of GEO

In this section, we describe enhancements that improve the performance of the GEO algorithm while maintaining its ease of deployment. Our first modification (G-SP) is to use real physical link lengths as opposed to using virtual geographical link lengths for routing within the domain[1]. While this information is domain specific, it is not necessarily proprietary since another domain will not be able to benefit by information about link lengths, without knowing additional information about the link such as link capacity. Thus, the top-level route selection mechanism uses the *combined cost metric* at the peering links and physical link lengths at all other links as the link weights and computes the shortest path. As before, intra domain routing is performed using the LCC algorithm. The next logical step is to combine the two variants above and create a hybrid protocol (G:PP+SP), which uses real physical link lengths to find the inter-domain path of peering nodes (based on G-SP) and uses the parallel probe algorithm for intra domain routing (based on G-PP).

From Fgure 13, we see a 30% improvement of G-SP over GEO. Our next modification (G-PP) is to use the PP algorithm for intra domain routing, while using virtual geographic link lengths for inter-domain routing. This yields a greater improvement of 40% over the GEO algorithm. We see a significant improvement in performance as shown in Figure 13. The original GEO and preliminary variants (G-PP,G-SP) are also shown along with the TRU and PP algorithms. As we can see, the hybrid protocol (G:PP+SP) performs almost as well as the end-to-end TRU algorithms and offers a factor of 5 improvement over the baseline GEO algorithm. This version achieves the proper balance between using static information assuring ease of deployment, and performance.

B. A Scalable Parallel Probe Algorithm

The PP and TRU algorithms reflect two extremes in inter-domain routing. The TRU algorithm is a source routing approach that assumes all information is known at the source and

[1]G-SP represents an idealized version of BGP routing using physical path lengths as the cost metric. Most vendor implementations commonly default to using the path length as the primary criterion for route selection. While hop count is a common metric, physical lengths are more applicable in the interdomain scenario where routers are physically separated by large distances

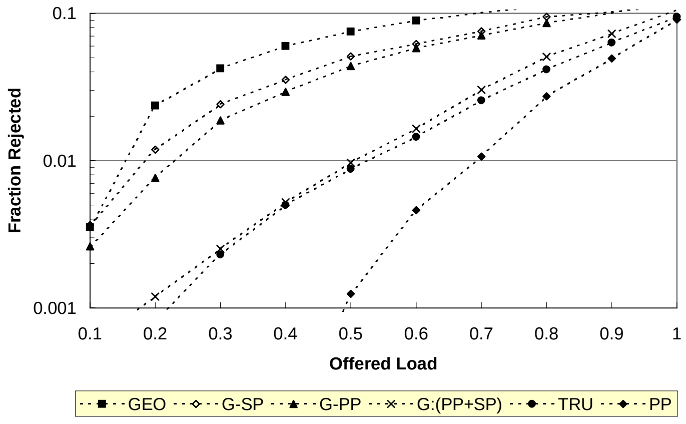

Fig. 13. Performance of a Hybrid GEO Scheme

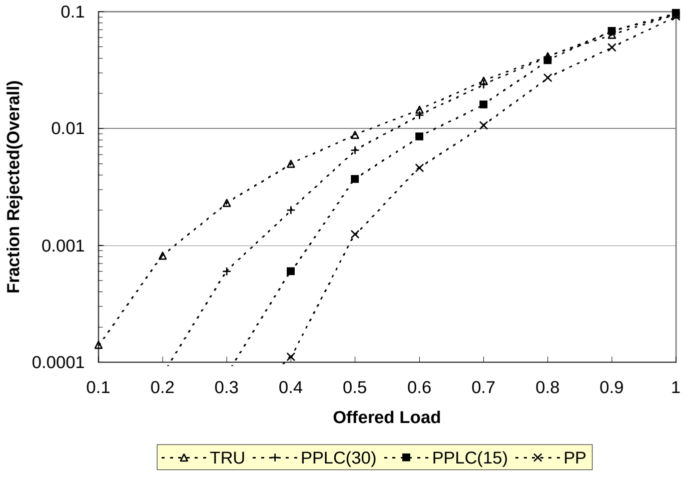

Fig. 14. Performance of PPLC compared to PP and TRU Algorithms

computes an end-to-end route on connection arrival. The PP algorithm sends probes on precomputed paths to find the best possible route. However, the PP algorithm assumes the existence of alternate routes via precomputation. Both protocols are applied in a consistent manner across all domains. BGP allows each domain to use its own route selection mechanism leading to locally optimal segments but a globally sub-optimal paths. We realise that forcing every domain to use the same routing protocol is not practical in real networks. In addition, while PP clearly outperforms the other variants, there is still an overhead associated with transmitting probes both within and across domains. We propose a hybrid version of the PP and LCC algorithms denoted as *PPLC* in the subsequent discussion that is more scalable than the original PP algorithm.

This algorithm uses the parallel probe approach in order to find the top-level path of peering nodes as before. While the baseline PP algorithm used *micro* probes for routing within an domain, we remove that requirement and allow the domain to use any shortest path mechanism. As our results for intra-domain routing have shown, the LCC algorithm performs the best for routing within a domain. Hence, we use LCC for intra-domain routing and use PP for routing between domains. This variant allows an domain to use a simpler intra-domain routing mechanism and is more scalable. The emphasis in the subsequent charts is to see the performance gap narrowing between the PPLC variant and the original PP algorithm. The link state update period is one additional parameter of PPLC that is carried over from the LCC component. The periodicity of routing updates decide the accuracy of the LCC algorithm for intra-domain routing.

Figure 14 shows the performance of the PPLC variant at different update periods along with the performance of the individual PP and TRU algorithms, where PPLC(x) is PPLC with an update period of x. There is a significant performance boost of PPLC(30) over TRU with nearly 65% improvement at a rejection fraction of 10^{-3}. PPLC(15) obtains a factor of 2 improvement over TRU and the baseline PP algorithm only has a 17% improvement over it. Thus, we see that this hybrid variant not only has significantly reduced overhead compared to both PP and TRU but also provides a balance between the PP and TRU algorithms in performance as well as processing complexity.

C. *Summary of Inter-domain QoS Routing Algorithms*

The three schemes (PP,TRU,GEO) represent protocols which primarily differ in the nature of routing information exchanged. The GEO algorithm uses static information about the geographical distance between routers assuming that the physical location of routers in domains is known *a priori*. It additionally assumes that information about peering links are distributed to peering nodes in all domains periodically. This is not a significant overhead since other mechanisms like threshold based triggers can minimize the routing update messages. This is the easiest protocol to deploy of the three schemes requiring minor modifications to OSPF. The TRU algorithm represents an upper bound on a shortest path first scheme based on its superior performance for intra-domain routing. As a result, we use the LCC algorithm for intra-domain routing for the GEO scheme. The PP algorithm uses a completely different semantic compared to the other two schemes. This scheme uses precomputed paths using static information about the distance between links. We have already shown the efficiency of PP algorithm in intra-domain routing.

The PP algorithm emerges as the best interdomain routing algorithm compared to the other schemes. This algorithm uses up-to-date information in the path selection mechanism using information collected by probes. It has also been adapted to suite the isolated nature of each domain whereby probes only carry peering information between domains, while other domain information remains strictly private to the domain. While there may be concerns about the scalability of the PP algorithm due to the number of probes, it should be noted that precomputation of the paths is a rare event, and the probes restrict themselves to a small set of paths.

In order to address the issue that not all domains may wish to use the PP algorithm, the PPLC variant is presented. This hybrid approach uses a combination of the PP algorithm for finding the top-level path of peering nodes, and the LCC algorithm for intra domain routing and is found to provide a balance in both performance as well as processing overhead between the two extremes. The baseline GEO algorithm performed significantly worse than the PP and TRU schemes. We presented two variants of GEO (G-SP and G-PP) that achieved some improvement over the baseline GEO approach. G-PP used PP for intra-domain routing, while G-SP used real physical link lengths for inter-domain routing. However, the combination of both approaches yielded a new hybrid scheme G:(PP+SP) that achieved a significant performance gain over the baseline GEO algorithms. The primary advantage of this variant is that it is based on static information for inter-domain routing allowing it to be easily deployed in current networks.

VI. Implementation of Routing Algorithms

In this section, we describe implementation details about the PP algorithm and its variants. This section also describes the details for a practical implementation of LCC which obviates the need to do a per-session computation of a shortest path on demand. We assume the use of OSPF-TE [16] for link state updates. OSPF-TE (OSPF with Traffic Engineering extensions) uses opaque link state advertisements (LSAs) to disseminate traffic engineering information. Such an LSA carries a special type of TLV (Type-length-value structure) called a link TLV that encodes information about a specific link. This includes the maximum bandwidth on the link, the maximum reservable bandwidth on the link and the unreserved bandwidth.

A. Implementing Parallel Probe

We will first focus on details for intra-domain routing. Adapting the PP algorithm for inter-domain routing requires an incremental effort and will be described subsequently. We use the following mechanisms:

Probe packets: The PP algorithm requires routers sending probe packets to the destination. We assume the use of IP packets, in particular the IP options field to designate the probes. The IP options occupy 40 bytes following the standard IP header. IPv4 options are organized into single and multi-byte options as shown in Figure 15. The *type* field is decimal or binary if a single byte option is used. Otherwise, it is comprised of the *copied, class and number* fields. The *copied* flag is examined when a packet with options is fragmented and indicates whether an option should be copied into the IP header of the fragments. We envisage a probe packet to fit within a single frame and fragmentation is not an issue. The *class* field indicates the whether the option is a control option or is used for debugging and measurement. A multi-byte option also has a *len* field and a data field. Some of the currently defined options in the current verion of IP (RFC 791) are *record route, timestamp, stream identifier, strict source and record route, loose source and record route*.

The *strict source and record route* option allows the route taken by a packet to be recorded within the packet, while forcing a source route to be followed by the routers routing this particular packet. This is suited for recording the path in the probe packet. The last hop router reverses the source route contained in the *winning* probe to allow the reservation of resources in the reverse direction. The source and destination addresses in the IP header and the offset and address list in the option specify the route and the packet's current location within the route. It is easy for a router to verify if it is the last hop router in this scheme since if the offset points to an address within the option, it indicates that the router is an intermediate node. Otherwise, if it is directly connected to the destination, it would have an en-

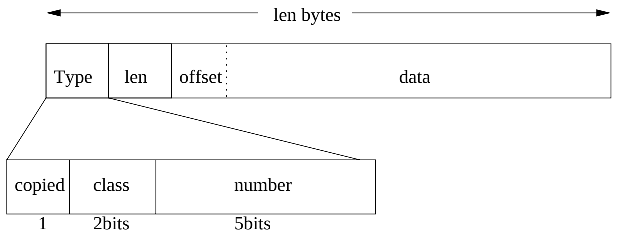

Fig. 15. Organization of IP Options

try in the routing table indicating the destination is the next hop. In the event that the destination node is on a LAN, then the address list would contain a network broadcast address (resolved using ARP). In summary, the probe packets will have a the following options active: 1) *strict source record route* option; 2) *no operation* which is a 1-byte sequence number generated by the source for identifying the probe with a particular connection. The option is not processed and is purely used for the purpose of identification.

Interactions with Router Hardware: The probe collects information regarding the bottleneck bandwidth capacity on outgoing links in addition to other relevant link load statistics. This information can be collected by the line cards at the routers and copied onto the probe packet in hardware without any interaction with the router control processor or router management software. The router management software is a user-level, centralised entity, which could become a bottleneck, and any interaction should be minimized in order to facilitate fast processing. Reconfigurable technologies such as field programmable gate arrays (FPGA's) can be used to maintain statistics at each interface. The probe packet collects this information directly from the interface allowing for fast processing.

Inter-domain issues: For larger networks, the 40-byte option space may not be sufficient to record longer routes. Approximately, 9 addresses can be stored in the address list while recording the source route. While the PP protocol can adapt by *expanding* and *collapsing* routes using peering nodes (routes longer than typical intra-domain paths need not be stored), an end-to-end version of LCC could record a path longer than can be stored in the IP options. IPv6 allows for a larger options header and more flexibility. With paths exceeding 9 hops, we use the *loose source record route* option which omits certain intermediate routers and concentrates on the significant hops. The following discussion assumes the use of PP in an end-to-end manner.

Note that the first step is to generate an inter-domain path comprising of peering nodes that connect the source and destination. Inter-domain paths are stored at every router containing a set of alternate top-level peering nodes. Thus, at a source router s_i, probes are sent on the set of alternate paths using local information. Once a probe enters an AS, it spawns micro-probes to find the best intra-domain path. We assume the edge routers of an AS$_i$ to maintain two separate tables. The first table is the normal routing table reflecting routes within the domain and represents information private to the AS$_i$. The second table is a *mapping table* with each entry being a 4-tuple

$$< f_{id}, \text{PEER}_{\text{in}}, \text{PEER}_{\text{out}}, \text{path} >$$

where PEER$_{\text{in}}$, PEER$_{\text{out}}$ specify the ingress and egress peering nodes belonging to AS$_i$ that are part of the inter-domain path for the flow identified by f_{id}. The path field is the chosen path within the domain.

In the forward pass of the parallel probe, the edge router PEER$_{\text{in}}$ sends *micro-probes* on alternate paths to reach PEER$_{\text{out}}$. Each probe collects relevant intra-domain routing information. The egress router collects information about each probe including the intra-domain route and stores it locally. After waiting for a short time period, information about the best probe is stored at the egress router along with the *path* field which is filled using the path recorded by the best probe. This state is transient and will be flushed if a reverse path reservation is not made along this segment of the path. The last hop router selects the best probe, reverses the source routed path, and initiates a reservation in the reverse mode. The reserve_probe uses the top-level information maintained in the reversed source route to reach the appropriate peering nodes. A table lookup is made to the mapping table to expand on the intra-domain path allowing the probe to reserve resources within the domain. The intra-domain path is not recorded in the probe during the reverse reservation process to prevent AS-sensitive information from being revealed.

We will now describe mechanisms to optimize the LCC algorithm in the next section, which will reduce the overhead of LCC as well as the PPLC variant.

B. Implementing LCC

In this section, we will describe mechanisms to improve the LCC algorithm as well as any derivative algorithm that uses LCC for routing including the PPLC[Section V-B] and variants of the GEO algorithm[Section V-A]. Recall that the end-to-end LCC computed a shortest path on a per-session basis which results in a significant computational overhead. More importantly, computation on a per-session basis is essentially wasteful since the link state updates occur at a larger interval than request arrivals. Without an update, recomputing a path between a given source and destination is unnecessary and does not result in a new path. In this regard, we propose the following enhancements:

Bandwidth Quantization: Our first enhancement is to quantize connection bandwidth into a dynamic set of ranges or bins. A similar approach using a fixed set of bins was proposed in [17]. Whenever a request arrives and no path exists between the source and destination nodes, the shortest path algorithm is executed using the complete set of nodes (For intra-domain routing, we use all the nodes in the local AS). Now, this not only provides a path from the source to the destination, but also provides shortest paths from the source to a set of intermediate nodes. For every source node, we can store paths from the given source to destinations based on the shortest path tree that results from a single execution of the shortest path algorithm. The key idea here is to assign these shortest paths to the bins such that given a bin x that corresponds to bandwidth b_x, all the paths in that bin from the given source to the relevant destinations have bottleneck bandwidth of atleast b_x units. The bandwidth range spanned by a particular bin can increase exponentially to minimize the number of bins.

Using Precomputed Shortest Path Trees: The second enhancement is to cache the shortest path trees corresponding to the bins and reuse the paths in them on a per-session basis whenever possible. Any recomputation of paths is done only when a link state update occurs, reducing the computational overhead significantly. Given a request with bandwidth b_i, we first map the request to a quantized bin B_i. We then examine the shortest path tree from the source to the destination stored corresponding to B_i and select the relevant route. From a storage perspective, the peering or edge routers will store a new table called the *Bin Mapper* table shown in Figure 16. The table uses a 2-level hashed data structure similar to a hashed bucket where each entry or bucket points to a linked list of nodes. The first step is to hash the bandwidth to the appropriate bin. The next step is to extract the shortest path tree(s) for that source. Considering the overhead involved in computing the path on request arrival, there are significant savings by using an efficient structure like a hashtable. Since path discovery proceeds transparently in parallel to connection arrivals, we can reduce the storage requirements further by merging different shortest path trees of the same source into a single tree.

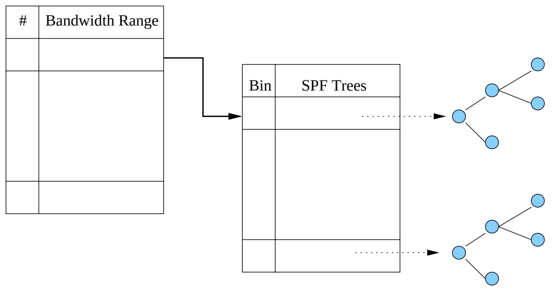

Fig. 16. Bin Mapper Table

Recomputing Paths without the overhead: We propose a third enhancement to LCC that allows routers to use the best possible path on a hop-by-hop basis for optimal route selection. Essentially, we allow every intermediate router in the path originally picked at the source to examine its local cache and use the best possible path in its table. This improves the selection mechanism since each router uses up-to-date information about its local links in the route selection decision.

VII. Conclusions

In this paper, we studied the performance of various QoS routing schemes that cover a wide spectrum from hop-by-hop to source routing and hybrid multi-path routing schemes. We evaluated these schemes on realistic ISP and inter-domain routing topologies that have links appropriately dimensioned using traffic constraints. In particular, the parallel probe algorithm appears to be the most effective of all the routing algorithms for both intra-domain and inter-domain routing. We also outlined mechanisms based on the shortest path algorithm and described variants that can be easily deployed in networks. Implementation details that outline the manner of deployment were also presented. We showed through comprehensive simulation studies that these protocols achieve consistently high performance over existing algorithms.

References

[1] Garcia-Luna-Aceves J. Loop-free routing using diffusing computations. *IEEE/ACM Trans. on Networking*, February 1993.

[2] Tangmunarunkit H., Govindan R., Shenker S., and Estrin D. The impact of routing policy on internet paths. In *Proc. of IEEE INFOCOM*, April 2001.

[3] Savage S., Collins A., Hoffman E., Snell J., and Anderson T. The end-to-end effects of internet path selection. In *Proc. of ACM SIGCOMM*, April 2001.

[4] Apostolopoulos G., Guerin R., Kamat S., and Tripathi S. K. Quality of service based routing: A performance perspective. In *Proc. of ACM SIGCOMM*, August 1998.

[5] Shaikh A. *Efficient dynamic routing in wide-area networks*. PhD thesis, University of Michigan, May 1999.

[6] Chen S. and Nahrstedt K. Distributed quality-of-service routing for next generation high speed networks based on selective probing. In *Proc. of IEEE Local Computer Networks*, pages 80–89, August 1998.

[7] Cidon I., Rom R., and Shavitt Y. Multi-path routing combined with resource reservation. In *Proc. of IEEE INFOCOM*, April 2000.

[8] Matta I. and Shankar A. U. Dynamic routing of real-time virtual circuits. In *Proc. IEEE Int. Conf. Network Protocols*, pages 132–139, 1996.

[9] Plotkin S. Competitive routing of virtual circuits in atm networks. *IEEE Journal on Selected Areas in Communication*, 13:1128–1136, August 1995.

[10] Fingerhut J. A. *Approximation algorithms for configuring nonblocking communication networks*. PhD thesis, Washington University, May 1994.

[11] Ma H. Singh I. and Turner J. S. Constraint based design of atm networks, an experimental study. Technical Report WU-CS-97-15, Department of Computer Science, Washington University, April 1997.

[12] Guerin R., Orda A., and Williams D. Qos routing mechanisms and ospf extensions. In *Proc. of IEEE INFOCOM*, March 1997.

[13] Apostolopoulos G., Guerin R., Kamat S., Orda A., Przygienda T., and Williams D. *QoS Routing mechanismsn and OSPF Extensions*. Internet Engineering Task Force, December 1998. Internet Draft.

[14] ———. Performance of deferred reservations in data networks. Technical report, Washington University, June 2001. http://www.arl.wustl.edu/$\sim$samphel/results.ps.

[15] O'Rourke J. *Computational Geometry in C*. Cambridge University Press, 1994.

[16] Katz D. and Yeung D. *Traffic engineering extensions to OSPF*. Internet Engineering Task Force, July 1997. Internet Draft.

[17] Norden S., Buddhikhot M., Waldvogel M., and Suri S. Routing bandwidth guaranteed paths with restoration in label switched networks. In *Proc. of ICNP*, November 2001.