Network-Wide Load Balancing Routing With Performance Guarantees

Kartik Gopalan, Florida State University, kartik@cs.fsu.edu
Tzi-cker Chiueh, Stony Brook University, chiueh@cs.sunysb.edu
Yow-Jian Lin, Telcordia Research, yjlin@research.telcordia.com

***Abstract*— As wide-area network connectivity becomes commoditized, network service providers are offering premium services that generate higher revenues by supporting performance sensitive traffic (such as voice, multimedia, and online trading). An emerging example is a virtual private network path with quality of service (QoS) guarantees, or QVPN. The main technical challenge in offering the QVPN service is how to allocate a physical route for each QVPN so as to maximize the total number of QVPNs that a given physical network infrastructure can support simultaneously. We make the case that the key to addressing this challenge is to maintain network-wide load balance when selecting QVPN routes. By ensuring that different parts of the network are evenly loaded, no single critical link will tend to become a bottleneck resource. This paper describes a Link Criticality Based Routing (LCBR) algorithm, which achieves high network resource utilization efficiency while supporting QVPNs with end-to-end delay and bandwidth guarantees. In addition, LCBR can select primary and backup routes for each QVPN simultaneously to support fast recovery from node or link failures. Using a simple yet effective metric that accurately quantifies network-wide load balance, LCBR significantly improves the total number of supported QVPNs when compared to state-of-the-art traffic engineering approaches.**

I. Introduction

As Internet connectivity becomes a commodity, large enterprises increasingly want high levels of performance and reliability guarantees for their performance sensitive traffic (such as voice, multimedia, online financial trading or electronic commerce). Carriers and network service providers are responding to such demands with QoS-guaranteed VPN (or QVPN) service. Technologies such as Multi-Protocol Label Switching (MPLS) can meet the QoS requirements of QVPNs by mapping each QVPN into a Label Switched Path (LSP). However, a major challenge that carriers face today is how to map each QVPN into a physical network path such that as many QVPNs as possible can be supported on its network infrastructure, thus maximizing the revenue base for the carrier.

We make the case that key to this problem is to maintain network-wide load balance during the route selection process. Without network-wide load balance, it is possible that critical links become saturated much earlier than others, rendering large number of routes unusable. While a number of QoS routing [1]–[8] and traffic engineering approaches [9]–[14] exist, none of them have attempted to explicitly quantify the notion of network-wide load balance under QoS constraints. Without a quantitative metric, it becomes difficult, if not impossible, to evaluate various load balancing routing algorithms.

In this paper, we propose the *Link Criticality Based Routing* (LCBR) algorithm that explicitly maintains network-wide load balance in the route selection process. LCBR is the first algorithm to explicitly quantify the notion of network-wide load balance and to use it systematically in the selection of primary and backup routes for QVPNs with bandwidth, end-to-end delay, and fault-tolerance guarantees. LCBR incorporates a simple and yet effective measure of *link criticality* that helps to capture the importance of a link with respect to its current residual capacity and expected future load. Link criticality is then used as the basis to derive a network-wide load balancing metric that characterizes the degree of load balance across the network and encourages the selection of less critical links in the routing process. Finally, LCBR applies a *backup resource aggregation* technique to select disjoint primary and backup routes such that the backup reservations for different QVPNs can be shared to reduce the cost of supporting failure recovery. LCBR routes incoming QVPNs in an *online* fashion. In contrast to offline schemes, online algorithms do not possess apriori knowledge of future QVPN arrivals and employ intelligent heuristics to adapt to future resource demands. We develop upon an earlier version of LCBR algorithm that was first briefly introduced in [15]. This paper's significant additional contributions include the detailed development and performance evaluation of a comprehensive set of LCBR algorithms that provide bandwidth guarantees alone, bandwidth-delay guarantees, and fault-tolerance guarantees along with backup resource aggregation. We consider the wide-area physical network managed by a carrier that has complete administrative control of resources in the network. Each customer's QVPN spans from one point of presence of the carrier's network to another and, once activated, lasts for several days, weeks, or months. A network management system monitors the traffic matrix and per-link traffic load in real-time and uses this feedback to route QVPNs.

II. Quantifying Network-Wide Load Balance

Consider the simple example in Figure 1. We need to select a route for a QVPN F_1 between nodes S_1 and D_1. There are two candidate routes: (S_1, E, B, C, D, D_1) and (S_1, E, F, G, D, D_1). Which of these two routes is better from perspective of long-term network resource usage efficiency? Suppose it is likely that future QVPN requests may arrive between (S_2, D_2) and (S_3, D_3) as well, but we do not know the exact QoS requirements of these QVPNs. Then the better route to select for F_1 would be (S_1, E, F, G, D, D_1) because it leaves the resources along the links (B, C) and (C, D) free for future QVPN requests between (S_2, D_2) and (S_3, D_3). Hence, the routing algorithm should, as far as possible, avoid overloading the links which are critical to a large number

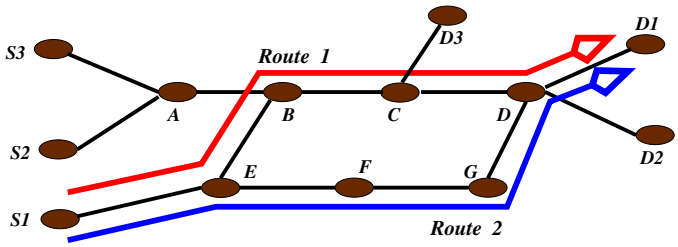

Fig. 1. An example of load balancing route selection problem.

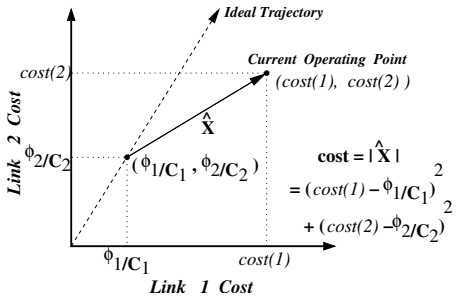

Fig. 2. Cost of a simple 2-link network. An n-dimensional plot would represent a network with n links.

of source-destination pairs, so that no single link becomes a bottleneck for the whole network.

The fundamental challenge here is the following: *Without precise knowledge of future QVPN request pattern, how exactly can we determine the importance of a network link and its impact on network-wide load balance?* In this section, we answer this question by quantifying the notions of link criticality, expected load and network-wide load balance.

A. Link Criticality and Expected Load

The importance of a link in LCBR is measured by the future *expected load* on each link, i.e. the expected amount of traffic between different source-destination pairs to be carried over each link. A link that carries higher amount of traffic between different source-destination nodes is considered more critical than one that carries less. More formally, assume that a total of x network routes are possible between a source-destination pair (s, d) and y of these routes pass through a link l. Then the *criticality* $\phi_l(s, d)$ of the link l with respect to source-destination pair (s, d) is defined as the fraction y/x.

The total *expected load* ϕ_l on link l is defined as the fractional sum of *expected* bandwidth demands on the link from all possible source-destination pairs in the network. Since ϕ_l represents a future expected load on the link, we need to begin with an initial estimate of ϕ_l and then refine it incrementally at run-time as successive QVPNs are admitted. The initial estimate of ϕ_l is computed using a matrix of expected bandwidth demands $B(s, d)$ between each source-destination pair (s, d) in the network. $B(s, d)$ can be obtained from measured daily traffic profiles and/or service-level agreements. If $B(s, d)$ were to distribute equally over each possible route between s and d, then the initial estimate of ϕ_l can be calculated as $\phi_l = \sum_{(s,d)} \phi_l(s, d)B(s, d)$.

Of course, the initial estimate of ϕ_l may not be accurate because (1) actual bandwidth demands may deviate from the $B(s, d)$ values and (2) equal-load distribution assumption may not hold at run-time. Section III describes how the initial ϕ_l values are incrementally corrected at run-time with each new QVPN. Another important consideration in computing ϕ_l is that the total number of routes between any source-destination pair grows quickly with network size and connectivity. In practice, we can tackle this issue as follows. (1) Since only a small subset of all the n nodes in the network are typically possible sources or destinations for QVPN traffic, computing ϕ_l does not involve an exhaustive computation of all the n^2 possible values of $\phi_l(s, d)$. (2) Between a given source-destination pair, we can restrict the choice of routes to k-shortest candidate routes (or disjoint route pairs), where k is typically small (around 5 in our evaluations). (3) Finally, the link criticality $\phi_l(s, d)$ itself is largely a static metric that changes only when the topology itself changes. Thus the values of ϕ_l can be periodically precomputed (such as on a daily basis) and kept ready for use.

B. Metric for Network-wide Load Balancing

Let C_l be the total bandwidth capacity of a link and R_l be its residual capacity at any time. We begin by defining the dynamic cost of each link, $cost(l) = \phi_l/R_l$, as the *expected load per unit of available link capacity*. Thus, a link with smaller residual capacity R_l or larger expected load ϕ_l will be considered more expensive in routing QVPNs. Network-wide load balancing depends upon the cumulative impact of not only the *magnitude* of individual link costs but also the *variations* among them. Figure 2 geometrically illustrates this relationship for a simple network with two links that have expected loads ϕ_1 and ϕ_2. The term ϕ_l/C_l represents the minimum value of the link cost when the residual capacity is maximum at $R_l = C_l$. For ideal load balance, residual link capacities R_1 and R_2 should ideally evolve towards zero such that $(cost(1), cost(2))$ indeed stays along the ideal load-balance trajectory. At the same time, in order to minimize the *amount* of resources consumed, it may not always be possible to follow the ideal trajectory. The next best alternative is to select routes that minimize the distance between the current operating point of the network $(cost(1), cost(2))$ and idle-state operating point $(\phi_1/C_1, \phi_2/C_2)$. We define the extent of load balance in a network G as the squared magnitude of the distance vector between the actual and the idle-state operating point.

$$cost(G) = \sum_{l \in G} \left(cost(l) - \frac{\phi_l}{C_l} \right)^2 \quad (1)$$

III. Load Balancing Route Selection

We now present the Primary LCBR (P-LCBR) algorithm that selects a single primary route for a QVPN F_N between source s and destination d that minimizes the load balance metric $cost(G)$. F_N requires two forms of QoS guarantees: (a) long-term bandwidth requirement ρ_N of F_N must be satisfied at each link along the route, and (b) end-to-end delay encountered by packets of F_N should be smaller than D_N.

A. With Bandwidth Guarantees

In this section, we consider the QVPNs that require bandwidth guarantees alone. When we select any route for a QVPN F_N, that requires a long-term bandwidth guarantee of ρ_N, the residual capacity R_l of each link l in the selected route would decrease by ρ_N. The contribution of each link to the network-wide metric $cost(G)$ is given by $(\phi_l/R_l - \phi_l/C_l)^2$. Correspondingly, $cost(G)$ increases due to smaller residual capacity R_l along the selected links.

In order to find the route which produces the smallest increment in $cost(G)$, we can first eliminate links with available bandwidth smaller than ρ_N and then apply Djikstra's shortest path algorithm on the reduced network graph, where the weight of each link is defined as follows:

$$w_l = \left(\frac{\phi_l}{R_l - \rho_N} - \frac{\phi_l}{C_l}\right)^2 - \left(\frac{\phi_l}{R_l} - \frac{\phi_l}{C_l}\right)^2 \quad (2)$$

The term w_l represents increase in link l's contribution to $cost(G)$ when F_N is routed through l. It is straightforward to see that the route X_N with the minimum value of $\sum_{l \in X_N} w_l$ is the one which produces the least increase in $cost(G)$.

Every time bandwidth ρ_N is reserved along a route for a new QVPN F_N, the expected load ϕ_l of each link in the network needs to be dynamically updated because (1) the actual traffic profile might deviate from originally expected traffic profile $B(s,d)$ and (2) the original ϕ_l values were computed under the assumption of equal traffic distribution among all candidate routes. This update can be performed *incrementally and with low overhead*, without recomputing each ϕ_l from scratch. For each link l, its ϕ_l value is incrementally updated as follows:

$$\begin{aligned} \phi_l &= \phi_l + (1 - \phi_l(s,d))\rho_N \quad \forall l \in X_N \\ \phi_l &= \phi_l - \phi_l(s,d)\rho_N \quad \forall l \notin X_N \end{aligned} \quad (3)$$

B. With Bandwidth and Delay Guarantees

Consider a QVPN F_N that requires an end-to-end delay guarantee of D_N, in addition to a bandwidth guarantee of ρ_N. Each link l is capable of supporting a range of queuing delay bounds for F_N depending upon the amount of bandwidth reserved for F_N at the link l. This raises a number of possibilities for *partitioning* the end-to-end delay budget of a QVPN among the individual links of a route [8], [16], [17]. However, in order to perform the delay partitioning, we need to know the complete set of links along the selected route. Since the classical shortest path algorithm incrementally builds the route one link at a time, it cannot handle the end-to-end delay partitioning problem mentioned above. At the same time, it is also impractical to examine the delay partitioning along every possible candidate route between a given source and destination because the number of routes between any source-destination pair grows quickly with network size.

In practice, shorter routes typically tend to utilize fewer network resources in comparison to longer routes and hence it is more likely that the route which best minimizes the network-wide cost metric is one among the k-shortest candidate routes. The P-LCBR algorithm for bandwidth-delay guaranteed route selection applies this insight to narrow down the set of candidate routes that can minimize $cost(G)$ to those having fewer links. P-LCBR performs route selection in two phases - *offline* and *online*. In the **offline phase**, performed once for the entire network, P-LCBR pre-computes the set of k-shortest candidate routes between each source and destination and computes the expected load ϕ_l for each link based on the computed candidate routes. A set of k-shortest candidate routes can be pre-computed using well known algorithms such as [18]. Fortunately, as results in Section V-C will demonstrate, a small value of k is sufficient in practice to achieve good network-wide load balance, which also lowers the computation overhead.

The **online phase** of P-LCBR executes upon the arrival of each new route set up request. The algorithm first computes $cost(l) = \phi_l/R_l$ for each link l in network using the pre-computed value ϕ_l and current residual capacity R_l. For each pre-computed candidate route X between s and d, P-LCBR performs the following sequence of three operations. (1) It checks if the QoS requirements (D_N, ρ_N) of F_N can be satisfied by the available resources along route X. (2) If there are sufficient resources, then P-LCBR partitions the end-to-end delay D_N among the links of route X. Specific delay partitioning algorithms are described in detail in [8], [16], [17]. The result of delay partitioning is a bandwidth reservation value $\rho_{Nl} \geq \rho_N$ for each link l along route X that guarantees a per-link delay budget D_{Nl} such that $\sum_{l \in X} D_{Nl} \leq D_N$. (3) Next, P-LCBR recomputes the per-link remaining capacity R_l and the projected value of $cost(G)$ that would result if the route X is assigned to F_N. The route set up request for F_N is rejected if either (a) no route X has sufficient resources to satisfy F_N's QoS requirements or (b) the minimum projected value of $cost(G)$ for any route X is greater than a pre-defined cost threshold α. If these two checks do not reject the QVPN F_N, then P-LCBR assigns F_N to that route X_N which yields the minimum increment in value of $cost(G)$. The ϕ_l values are correspondingly updated as follows.

$$\begin{aligned} \phi_l &= \phi_l - \phi_l(s,d)\rho_N + \rho_{Nl} \quad \forall l \in X_N \\ \phi_l &= \phi_l - \phi_l(s,d)\rho_N \quad \forall l \notin X_N \end{aligned} \quad (4)$$

IV. Primary-Backup Route Selection

Given a new QVPN request F_N, the goal of Primary-Backup LCBR (PB-LCBR) algorithm is to select a disjoint primary-backup route pair (X_N, Y_N) that minimizes $cost(G)$. The disjoint backup route Y_N guarantees that, if at most one network element (a link or a node) fails and the failed element lies on the primary route X_N, then F_N's traffic would be diverted to Y_N with the same QoS guarantees on bandwidth ρ_N and end-to-end delay bound D_N.

A. PB-LCBR Algorithm

A simplistic approach to primary-backup route selection is to first select a primary route X_N that yields the smallest value of $cost(G)$, and then select a disjoint route Y_N from the residual network graph G' that does not have network elements of X_N. However, this sequential approach leads to inefficient and skewed solutions [19]. Hence the PB-LCBR algorithm

simultaneously examines both primary and backup components of candidate *route pairs*. The algorithm is similar in structure to P-LCBR algorithm, although with important variations that deserve mention.

When a QVPN F_N requires only a bandwidth guarantee of ρ_N, then PB-LCBR finds the *shortest path-pair*, rather than the shortest path, that increases $cost(G)$ by the smallest margin. Specifically, PB-LCBR selects the path-pair (X_N, Y_N) with minimum value of $\sum_{l \in (X_N \cup Y_N)} w_l$, where w_l is given by Equation 2. Corresponding to Djikstra's shortest path algorithm, [20] describes a shortest path-pair algorithm.

When F_N requires an end-to-end delay bound of D_N in addition to bandwidth guarantee of ρ_N, using the shortest path-pair algorithm is not feasible due to reasons mentioned earlier in Section III-B. In this case, PB-LCBR follows the framework of P-LCBR in Section III-B with two major variations. The first variation is that the offline phase of PB-LCBR pre-computes a set of $k = k_1 \times k_2$ candidate primary-backup route pairs for every source-destination pair (as opposed to just k_1 candidate primary routes in the case of P-LCBR). The second variation is that the input to the online phase consists of route pairs (X, Y) that were pre-computed during the offline phase. The difference from the online phase of P-LCBR is that for each candidate route pair (X, Y) (1) both primary route X and backup route Y are checked to ensure that sufficient resources are available to satisfy new QVPN F_N's QoS requirements and (2) the end-to-end delay requirement D_N is partitioned along both X and Y. Finally, the candidate route pair (X_N, Y_N) that yields minimum $cost(G)$ is chosen for F_N.

B. Backup Resource Aggregation

In order to maximize resource usage efficiency along backup routes, we now introduce the notion of *backup resource aggregation* which attempts to share QVPN reservations along common backup links. Backup resource aggregation was first proposed in the RAFT approach [21] in the context of QVPNs that require bandwidth guarantees alone. Our additional contribution here is the application of the resource aggregation concept to both bandwidth guaranteed as well as delay-bandwidth guaranteed QVPNs within the comprehensive admission control and resource allocation framework of the LCBR algorithm.

Two QVPNs are said to *intersect* with each other at a network element e if both their primary routes pass through element e. Every link l has one *primary set*, $Primary(l)$, that contains the IDs of all QVPNs whose primary routes pass through the link l. In addition, each link has a total of $(m + n)$ *backup sets* of QVPN reservation, where each set corresponds to one network element; m is the number of links and n is the number of nodes in the entire network. The backup sets at any link l are represented by $Backup(l, e)$, $1 \leq e \leq (m+n)$, where each backup set corresponds to one network element e.

Recovery from failure of a single network element occurs as follows. During normal operations, each link scheduler operates with the reservations for QVPNs in its primary set $Primary(l)$. Whenever a network element e fails, its corresponding backup sets, $Backup(l, e)$, are activated at all the links l and the respective link schedulers start operating with reservations in primary set $Primary(l)$ plus those in backup set $Backup(l, e)$. In the meantime, the route selection mechanism would attempt to recover completely by rediscovering new primary and/or backup routes of those QVPNs that are affected by the failure of network element e. The only change in PB-LCBR algorithm with backup resource aggregation is the manner in which residual link capacities are calculated. If we represent the primary reservation for a QVPN F_i at link j by $\rho_{i,j}$ and its backup reservation at link k by $\beta_{i,k}$, then the residual capacity R_l at link l is calculated as follows.

$$R_l = C_l - \sum_{i \in Primary(l)} \rho_{i,l} - \max_{\forall e} \left\{ \sum_{i \in Backup(l,e)} \beta_{i,l} \right\} \quad (5)$$

V. Performance Evaluation

We now compare the performance of LCBR against two earlier traffic engineering based approaches – the Widest Shortest Path (WSP) [2] based algorithm and the Minimum Interference Routing Algorithm (MIRA) [11] – that were originally proposed for bandwidth guaranteed primary route selection. The primary route selection version of WSP (P-WSP) works as follows. For QVPNs that require bandwidth guarantees alone, P-WSP selects that route which has maximum residual bottleneck link capacity from among all the feasible routes having minimum length. The primary-backup WSP (PB-WSP), selects that feasible route pair with minimum-length primary path which has maximum bottleneck link capacity. For bandwidth-delay guaranteed route selection, P-WSP (PB-WSP) examines a set of k candidate routes (route pairs) and select the minimum length candidate with maximum bottleneck capacity. MIRA defines the weight of a link l as the number of source-destination pairs whose mincuts include l. For bandwidth guaranteed route selection, P-MIRA (PB-MIRA) selects the route (route pair) with minimum sum of link weights, as defined above. For bandwidth-delay guaranteed route selection, P-MIRA (PB-MIRA) examines a set of k candidate routes (route pairs) and selects the one with minimum sum of link weights.

We developed a simulation tool to compare the resource usage efficiency of the three algorithms. In this paper, we present simulation results over the AT&T nationwide backbone topology with 41 nodes and 64 links. Due to space constraints, the results for two other topologies – the North American IP backbone topology of Sprint and a 5×5 Grid topology – are presented in [19]. Link capacities are chosen as a mix from 5 Gbps to 20 Gbps and sources-destination pairs are selected uniformly. Bandwidth demand profile $B(s, d)$, is uniformly distributed between 5 Gbps and 20 Gbps. New QVPNs request an average bandwidth guarantee of 10Mbps. Excluding signal propagation delays, which can typically range anywhere around 25–150ms, the end-to-end *queuing* delay budget of up to 20ms can be requested by each QVPN. QVPNs are constantly admitted till the network saturates. Intra-path delay partitioning is performed using LSS algorithm [16] and backup resource aggregation is applied to all the three PB-* algorithms.

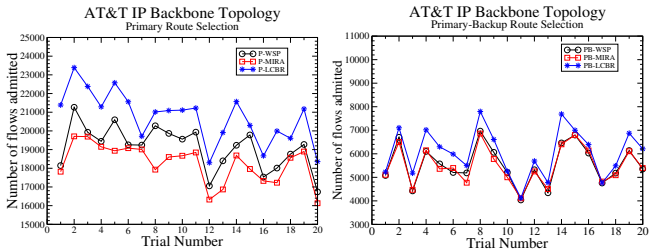

Fig. 3. Number of QVPNs admitted with bandwidth guarantees alone. $\rho_i^{avg} = 10Mbps$, $k = 5$

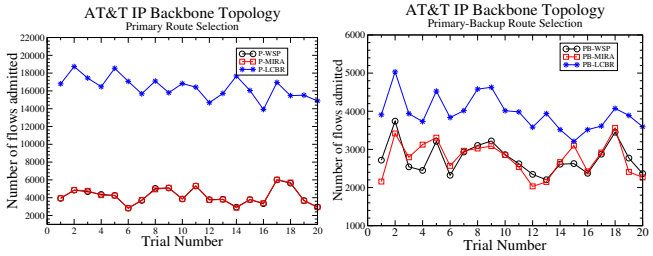

Fig. 4. Number of QVPNs admitted with bandwidth-delay guarantees. $\rho_i^{avg} = 10Mbps$, $D_i = 10ms$, $k = 5$

A. Effectiveness of LCBR Algorithm

We first provide a snapshot comparison of the performance of LCBR against WSP and MIRA. Figure 3 and 4 plot the number of QVPNs admitted with bandwidth guarantees and bandwidth-delay guarantees respectively by different algorithms for 20 simulation runs, where each run uses a different random seed to controls link capacities. The figures demonstrate that in all scenarios, LCBR consistently admits more QVPNs than WSP and MIRA algorithms since it bases routing decisions on a network-wide load balancing criteria. On the other hand, WSP performs only limited load balancing by selecting widest shortest route among all candidates. The performance of MIRA and WSP are in general close to each other.

There are noticeable differences in the relative performance of algorithms with and without delay guarantees. Specifically, we find that performance of LCBR is significantly better compared to WSP and MIRA in the presence of delay requirement than without delay requirement. This is explained by the fact that tight delay requirements of a QVPN F_N would require a bandwidth reservation ρ_{Nl} at each link l that is larger than its average bandwidth requirement ρ_N. Thus with delay-guaranteed QVPNs, there is a tendency towards higher network-wide load imbalance and the benefits from LCBR's load balancing approach become more evident. Another noticeable trend is that the relative difference in performance of LCBR compared to WSP and MIRA is smaller for primary-backup route selection. The disjointed-ness requirement on each primary-backup route pairs reduces the number of good quality candidates available for LCBR to choose from.

B. Network-Wide Load Balance

We now show that LCBR indeed maintains better network-wide load balance. Figure 5 shows that the standard deviation in final percentage loads among all the links in the network is indeed consistently smaller for LCBR compared to MIRA and WSP. The reason for LCBR's smaller load deviation is that it explicitly minimizes the load-balance metric $cost(G)$. While the $cost(G)$ values of WSP and MIRA saturate after admitting fewer QVPNs, LCBR continues to admit more QVPNs.

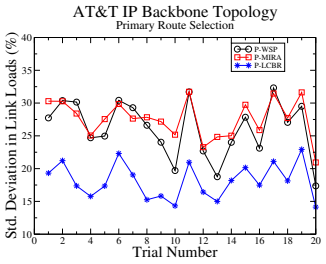

Fig. 5. Comparison of standard deviation in link loads for WSP, MIRA and LCBR. $\rho_i^{avg} = 10Mbps$, $D_i = 10ms$, $k = 5$.

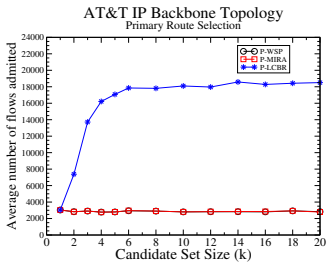

Fig. 6. Number of QVPNs admitted vs. candidate set size k. $\rho_i^{avg} = 10Mbps$, $D_i = 10ms$.

C. Candidate Set Size

Figure 6 shows that, with increasing candidate set size k, the number of QVPNs admitted with bandwidth-delay guarantees increases rapidly for LCBR and then quickly saturates. Increasing k implies more choices for selecting a route leading to more admitted QVPNs. However, for larger k values, the candidate set now includes longer routes which can consume more resources and hence are rarely selected. WSP and MIRA are hardly affected by varying k. *This shows that LCBR improves performance without deviating too much from the shortest route – it tends to choose among 4 or 5 shortest candidates.* WSP does not examine the entire candidate set; increasing k helps WSP only if it results in inclusion of more feasible routes with shortest path length. Similarly, increasing k helps MIRA only if it results in inclusion of routes with fewer links that impact the maxflow between various source-destination pairs.

D. End-to-end Delay

Figure 7 shows that, with increasing end-to-end delay, number of QVPNs admitted increases initially and then tends to saturate for all the algorithms. Initially, each QVPN's delay-derived bandwidth requirement is tighter than the average rate requirement and hence delay is the tighter requirement. For larger delay requirement, the delay-derived bandwidth becomes smaller than the average rate requirement. In the saturation region, QVPNs are essentially being serviced at their average rates. Also, LCBR admits more QVPNs than WSP and MIRA across all delay ranges.

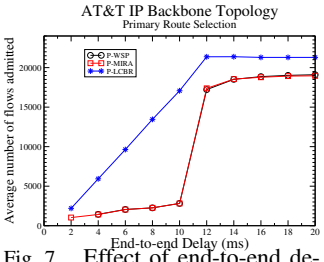

Fig. 7. Effect of end-to-end delay on average number of admitted QVPNs.

E. Computation Cost

The computation cost for LCBR on a typical operational network is quite small. For P-LCBR algorithm, offline computation

time, on a 1.8GHz Pentium 4 machine, varies from 3ms to 18ms as k increases from 2 to 10. For PB-LCBR, the corresponding variation is from 10ms to 50ms. The online computation cost is less than 2ms for P-LCBR and less than 4.5ms for PB-LCBR. The computation cost is more for PB-LCBR because route selection and QoS partitioning are performed for both primary and backup routes.

VI. Related Work

Traditional *hop-by-hop shortest path* (SP) algorithms [22] for best-effort traffic ignore the fact that links along a selected route might already be congested while non-congested alternatives may never be considered. Widest Shortest Path (WSP) [2] and several variations [3], [22], [23] perform limited load balancing and congestion management but finally suffer the same fundamental problems as SP.

QoS routing algorithms attempt to find a feasible short term route that satisfies either single [1]–[3] or multiple [4]–[8], [18], [24] QoS requirements, but not to optimize for long term traffic engineering considerations. On the other hand, *traffic engineering* (TE) solutions attempt to achieve long term network-wide resource management objectives. A class of TE schemes [13], [25], [26] manipulate link weights to maintain load balance and minimize congestion with best-effort traffic, but do not support explicit QoS guarantees.

Minimum Interference Routing Algorithm (MIRA) [11] is one of the first *explicit routing* TE schemes that selects the entire route for a QVPN in advance. MIRA uses a notion of mincut-based link criticality (described in Section V) which is useful in avoiding links that impact the carrying capacity between large number of ingress-egress pairs, but does not identify important links that may not be part of any mincuts. Profile-Based Routing (PBR) [14] uses measured traffic profiles to solve a multi-commodity network flow problem and provide bandwidth guarantees. The key features that distinguish LCBR from the earlier explicit TE approaches are (1) LCBR's network cost metric quantifies the impact of *all links* on the network-wide load balance, whether or not they are part of any mincuts, and (2) LCBR supports QVPNs with any combination of bandwidth, end-to-end delay, and fault-tolerance guarantees.

VII. Conclusions

In this paper, we made the case that the key to maximize network resource usage efficiency under performance constraints is to achieve network-wide load balance every step of the way in the route selection process. We have proposed the Link Criticality Based Routing (LCBR) algorithm, which can select primary and backup routes for wide-area QVPNs with bandwidth, end-to-end delay, and fault-tolerance guarantees. To the best of our knowledge, our work is the first to explicitly quantify the notion of network-wide load balance and use it systematically in developing algorithms for traffic engineering route selection with QoS guarantees. LCBR defines a simple and yet effective network-wide load balancing metric that encourages the selection of less critical links in the routing process. In our evaluations, LCBR consistently supports more QVPNs under the same input workloads, when compared against state-of-the-art traffic engineering approaches.

References

[1] R. Guerin and A. Orda, "QoS-based routing in networks with inaccurate information," in *Proc. of IEEE INFOCOM'97, Japan*, April 1997.

[2] R. Guerin, A. Orda, and D. Williams, "QoS routing mechanisms and OSPF extensions," in *Proc. of IEEE GLOBECOM'97, Phoenix, AZ*, Nov. 1997, vol. 3, pp. 1903–1908.

[3] S. Plotkin, "Competitive routing of virtual circuits in ATM networks," *IEEE J. Selected Areas in Comm.*, vol. 13(6), pp. 1128–1136, 1995.

[4] S. Chen and K. Nahrstedt, "On finding multi-constrained paths," in *Proc. IEEE ICC'98*, June 1998.

[5] Z. Wang and J. Crowcroft, "Bandwidth-delay based routing algorithms," in *Proc. of IEEE GLOBECOM'95, Singapore*, Nov. 1995, pp. 2129–2133.

[6] D. Raz and Y. Shavitt, "Optimal partition of QoS requirements with discrete cost functions," in *Proc. of INFOCOM, Israel*, March 2000.

[7] F. Ergun, R. Sinha, and L. Zhang, "QoS routing with performance dependent costs," in *Proc. of INFOCOM, Israel*, March 2000.

[8] D.H. Lorenz, A. Orda, D. Raz, and Y. Shavitt, "Efficient QoS partition and routing of unicast and multicast," in *Proc. of IWQoS 2000, Pittsburgh, PA*, June 2000, pp. 75–83.

[9] J. Boyle, V. Gill, A. Hannan, D. Cooper, D.Awduche, B. Christian, and W.S. Lai, "Applicability statement for traffic engineering with MPLS," RFC 3346, August 2002.

[10] B. Fortz, J. Rexford, and M. Thorup, "Traffic engineering with traditional IP routing protocols," *IEEE Communications Magazine*, Oct. 2002.

[11] M.S. Kodialam and T.V. Lakshman, "Minimum interference routing with applications to MPLS traffic engineering," in *Proc. of INFOCOM, Israel*, March 2000, pp. 884–893.

[12] H. Smit and T. Li, "IS-IS extensions for traffic engineering," Internel Draft, August 2003.

[13] A. Sridharan, R. Guerin, and C. Diot, "Achieving near optimal traffic engineering solutions in current OSPF/ISIS networks," in *Proc. of IEEE Infocom 2003, San Fransisco, CA*, March 2003.

[14] S. Suri, M. Waldvogel, D. Bauer, and P.R. Warkhede, "Profile-based routing and traffic engineering," *Computer Communications*, vol. 26(4), pp. 351–365, 2003.

[15] K. Gopalan, T. Chiueh, and Y.J. Lin, "Load balancing routing with bandwidth-delay guarantees," *IEEE Communications Magazine*, June 2004.

[16] K. Gopalan and T. Chiueh, "Delay budget partitioning to maximize network resource usage efficiency," in *Proc. IEEE INFOCOM'04, Hong Kong, China*, March 2004.

[17] R. Nagarajan, J. Kurose, and D. Towsley, "Local allocation of end-to-end quality-of-service in high-speed networks," in *Proc. of Workshop on Perf. analysis of ATM Systems*, Jan. 1993, pp. 99–118.

[18] D. Eppstein, "Finding the k shortest paths," in *Proc. of Symposium on Foundations of Computer Science*, Nov. 1994, pp. 154–155.

[19] K. Gopalan, "Efficient provisioning algorithms for network resource virtualization with QoS guarantees," Ph.D. Thesis, Computer Science Dept., Stony Brook University, Aug. 2003.

[20] J.W. Suurballe and R.E. Tarjan, "A quick method for finding shortest pairs of disjoint paths," *Networks*, vol. 14, pp. 325–336, 1984.

[21] K. Dovrolis and P. Ramanathan, "Resource aggregation for fault tolerance in integrated services packet networks," *ACM Computer Communications Review*, vol. 28(2), pp. 39–53, April 1998.

[22] J. Moy, "OSPF Version 2," RFC 2328, April 1998.

[23] S. Blake, D. Black, D. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services," RFC 2475, Dec. 1998.

[24] R. Hassin, "Approximation schemes for restricted shortest path problem," *Mathematics of Operations Research*, vol. 17(1), pp. 36–42, Feb. 1992.

[25] B. Fortz and M. Thorup, "Optimizing OSPF/IS-IS weights in a changing world," *IEEE Journal on Selected Areas in Communications*, vol. 20, pp. 756–767, May 2002.

[26] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao, "Overview and principles of internet traffic engineering," RFC 3272, May 2002.