On-line Routing for Permanent Virtual Circuits

Rainer Gawlick
MIT
Laboratory for Computer Science
Cambridge, MA 02139

Charles Kalmanek
AT&T Bell Laboratories
Murray Hill, NJ 07974

K. G. Ramakrishnan
AT&T Bell Laboratories
Murray Hill, NJ 07974

Abstract

This paper considers the problem of routing a set of permanent virtual circuit requests over a backbone network. Several factors make this routing problem complicated. Routing decisions must be made on-line *without any knowledge of future request sets. Furthermore, frequent rerouting to correct inefficiencies that can result from the on-line routing decisions is not possible since rerouting creates a service disruption for the customer. Finally, the forward and reverse bandwidth of a virtual circuit must be routed over the same single path.*

Using an extensive set of simulations, this paper evaluates several different strategies for on-line permanent virtual circuit routing. We find that a strategy based on recent results in competitive analysis and ideas from combinatorial optimization consistently provides the best performance.

The problem of admission control is closely related to the problem of routing. This paper also provides a theoretical lower bound that suggests that non-greedy admission control is a fundamental component of an efficient on-line permanent virtual circuit routing algorithm.

1 Introduction

This paper considers the problem of a service provider who, given a set of customer virtual circuit requests each associated with a bandwidth requirement for the forward and reverse direction, must determine what paths through the backbone network the virtual circuits should follow: we refer to this as the primary routing problem for permanent virtual circuits.

This paper evaluates several strategies for primary routing of permanent virtual circuits using an extensive set of simulations. The network is modeled as having a centralized controller that makes all routing decisions. Customers submit requests to set up virtual circuits in sets. Each virtual circuit request consists of a source, a destination, a forward bandwidth and a reverse bandwidth. The duration of the circuit is assumed to be infinite. The service provider must route the virtual circuits or, if there is insufficient capacity, indicate that some of the circuits cannot be routed. We are interested in the relative performance, in terms of the amount of bandwidth routed, of different routing strategies for primary routing. If one routing strategy is able to route substantially more bandwidth on the backbone network than another, the customers ultimately benefit since the costs of expensive facilities are shared with a larger base of users.

Several factors make primary routing for permanent virtual circuits complicated. Routing decisions must be made on-line. In other words, when a set of requests arrives, the routing decisions must be made without knowledge of future requests. The fact that routing decisions must be made on-line can lead to poor performance when a decision to route a virtual circuit over a particular path has, as a result of subsequent requests, caused the network to be used inefficiently. Unfortunately, frequent rerouting cannot be used to correct these inefficiencies since rerouting in virtual circuit networks, such as ATM and Frame Relay, causes service disruptions to the customer. Finally, a virtual circuit must be routed along a single path in both directions. The requirement can introduce large inefficiencies especially when requests have differing forward and reverse bandwidth requirements.

In addition to primary routing there are two related problems that are relevant to permanent virtual circuits. If a link or switching node fails, the service provider may be able to restore service by rerouting the affected virtual circuits: this is the restoration problem. While this paper focuses on primary routing, our results are relevant to the restoration problem. In particular, an efficient primary routing strategy will generally provide greater flexibility to any restoration effort. Furthermore, since our primary routing strategies route *sets* of virtual circuits, they may be used

0743-166X/95 $04.00

directly for the restoration problem. The second problem related to primary routing is that of admission control. Routing algorithms for permanent virtual circuits are required to pursue a greedy admission control strategy. Specifically, a virtual circuit must be routed as long as sufficient capacity is available. An alternative admission control strategy called *trunk reservation* rejects a virtual circuit if all available routes for that virtual circuit could result in inefficient use of the remaining capacity. In this paper we present a new theoretical result that shows, using competitive analysis, that the use of trunk reservation is fundamental to efficient routing. Previous work analyzing routing under statistical assumptions supports our result [Kru82, Aki84].

Our evaluation considers several different routing strategies. A popular routing strategy simply picks the path with the least number of links since that path uses the fewest amount of resources. We call this *minimum-hop* routing. The restoration problem suggests that one might prefer the path that maximizes over all links the minimum remaining capacity, so as to allow room for circuits to be rerouted. We call this *max-min* routing. Finally, the theory of competitive analysis suggests selecting the shortest path in a cost metric where the cost of a link is exponential in the bandwidth currently routed on a link [AAF+93, AAP93]. We call this *exponential* routing. Each of these strategies can be represented as a shortest path or minimum cost routing algorithm. In particular, the strategies differ only in their cost metric.

We evaluate three ways in which the above routing strategies can be adapted to route a *set* of virtual circuit requests rather than just a single virtual circuit request. The first approach takes a set of requests and processes them sequentially, in decreasing order of bandwidth, in an on-line fashion. We call this the *ordered on-line* approach. The second approach adds a greedy local search in order to improve the route selection. The use of such combinatorial optimization techniques as part of an on-line algorithm is new and may be applicable beyond the area of routing. We call this the *local search* approach. Finally, the third approach uses mathematical programming to directly find the best set of paths for the set of virtual circuit requests. We call this the *convex optimization* approach.

The simulation results show that exponential routing consistently provides the best performance in terms of the amount of bandwidth routed. Depending on the simulation scenario, exponential routing outperforms minimum-hop routing by 0% to 15% and outperforms max-min routing by 3% to 25%.

The paper is organized as follows. Section 2 defines the routing strategies evaluated in this paper. The simulation results for each of the routing strategies are presented in Section 3. Section 4 summarizes our results. Finally, the proofs for our results on the importance of trunk reservation appear in Appendix A.

2 The Algorithms

This section describes the algorithms we evaluate for primary routing of permanent virtual circuits. Section 2.1 describes each of the cost metrics we consider. The two methods used to enhance the algorithms to deal with a set of virtual circuit requests appear in Section 2.2.

2.1 Cost metrics

Consider the following notation. For link e let c_e represent the capacity of link e. The bandwidth used by the virtual circuits currently routed over link e normalized by the capacity of the link is denoted by f_e^f in the forward direction and f_e^r in the reverse direction. Finally, for virtual circuit request i, $\Delta f_{i,e}^f$ ($\Delta f_{i,e}^r$) represents the forward(reverse) bandwidth of request i normalized by the capacity of link e.

Minimum-hop routing. A simple cost metric that minimizes the amount of resources used by each virtual circuit assigns a constant state-independent cost to each link. Thus this cost metric chooses the path that consists of the least number of links. In this cost metric there will often be many paths with the same cost, therefore a tie break is needed. An *arbitrary* tie-break picks the first minimum cost path that is found whereas a *random* tie-break randomly chooses from all minimum cost paths. Finally, a *state-dependent* tie break uses the values of f_e^f, f_e^r, Δf_e^f, and Δf_e^r for all the links on the paths in order to break the tie.

Exponential routing. Recent results in the theory of competitive analysis suggest the use of a cost metric based on an exponential function. Competitive analysis is an important tool used in the theoretical community to analyze the performance of an algorithm when no assumptions are made about the inputs to the algorithm. In particular, competitive analysis compares the performance of an algorithm on each

network topology and request sequence to the optimal algorithm for that network topology and request sequence. Let $A(G,\alpha)$ be the amount of bandwidth routed by algorithm A on topology G and request sequence α. (Note, this analysis can obviously be used for performance measures other than amount of bandwidth routed.) Let $\text{OPT}(G,\alpha)$ be the amount of bandwidth routed by the optimal algorithm for topology G and request sequence α. The performance of algorithm A is the characterized by

$$\text{MAX}_{G,\alpha}\left\{\frac{\text{OPT}(G,\alpha)}{A(G,\alpha)}\right\},$$

which is the *competitive ratio* of algorithm A. Since the competitive ratio is based on a worst case analysis, it is clearly a robust, albeit conservative, measure of performance.

Several recent papers show that a cost metric based on an exponential function leads to routing algorithms that have an optimal competitive ratio. Aspnes et al. [AAF+93] consider primary routing of permanent virtual circuits where the measure of performance for the algorithm is not the amount of bandwidth routed, but the amount of capacity used. Specifically, [AAF+93] shows that an algorithm with a cost metric based on an exponential function leads to a competitive ratio of $O(1/\log n)$ where n is the number of nodes in the network. In other words, such an algorithm would need at most $O(\log n)$ more capacity on each link to route the same amount of bandwidth routed by an optimal algorithm. No better competitive ratio is possible [AAF+93]. The function used for the cost of link e in [AAF+93] is $a^{f_e^f+\Delta f_{i,e}^f}-a^{f_e^f}$ where a is a constant[1]. Unfortunately, the analysis in [AAF+93] provides no information about the performance of an exponential function based cost metric when $O(\log n)$ additional capacity is not available on each link.

[1] The function does not use any information about reverse bandwidth since [AAF+93] does not consider virtual circuit where the forward and reverse bandwidth differ.

In [AAF+93] the constant a is set to $1+\epsilon$ where $\epsilon < 1$.

Up to a constant, $a^{x+\Delta x}-a^x$ is just the integral of a^y from x to $x+\Delta x$. Thus a^x can be seen as the marginal cost of the link.

Since the amount of bandwidth required to route all virtual circuit requests in not known in advance, [AAF+93] uses a *scaling* technique. This scaling technique first restricts the algorithm to use at most $1/\log n$ fraction of the available capacity. Once no more virtual circuits can be routed with that amount of capacity, the amount of capacity the algorithm may use is doubled. This continues until the algorithm uses all of the available capacity. With the use of scaling, the performance of an exponential function based cost metric should be similar to the performance of the cost metric that seeks to maximize the minimum residual bandwidth (max-min routing).

In [AAP93] the performance of the exponential function based cost metric is considered in terms of the total amount of bandwidth routed without the use of additional capacity. However, the model in [AAP93] differs from the permanent virtual circuit model since trunk reservation is permitted. In other words, a virtual circuit can be rejected even if sufficient capacity exists to route the virtual circuit. A simplified version of the proof in [AAP93] can be used to show that a cost metric based on the function $a^{f_e^f+\Delta f_{i,e}^f}-a^{f_e^f}$ leads to a competitive ratio of $O(\log n)$ in terms of the amount of bandwidth routed if trunk reservation is permitted[2].

Unfortunately the $O(\log n)$ competitive ratio from [AAP93] does not extend to permanent virtual circuit routing where trunk reservation is not permitted. In fact, when trunk reservation is not permitted, there is a lower bound that states that any on-line algorithm has a competitive ratio that is at best $\Omega(n)$ in the amount of bandwidth routed. Appendix A provides the proof of this lower bound. Appendix A also shows that any cost metric will achieve an $O(n)$ competitive ratio. The $\Omega(n)$ lower bound in Appendix A along with $O(\log n)$ competitive ratio from [AAP93] suggest that trunk reservation may be fundamental to efficient usage of network resources. It is noteworthy that statistical analysis of network routing algorithms also points to the fundamental importance of trunk reservation [Kru82, Aki84].

Even though the analyses of [AAF+93, AAP93] do not apply directly to our permanent virtual circuit setting, the success of the exponential function in related settings suggests its use here. The particular function we use for the cost of a link is

$$a^{(f_e^f+\Delta f_{i,e}^f)}-a^{f_e^f}+a^{(f_e^r+\Delta f_{i,e}^r)}-a^{f_e^r} +c(\Delta f_{i,e}^f+\Delta f_{i,e}^r). \qquad (1)$$

The constants a and c are varied in order to determine their best values. Intuitively, a is a measure of how quickly the algorithm retreats from minimum hop routing, while the constant c causes the algorithm to favor minimum-hop routing when the network is lightly loaded.

Max-min routing. The desire to provide a maximum amount of flexibility in the event a virtual circuit must be rerouted due to a link failure, suggests selecting the path that maximizes the residual bandwidth

[2] The function used by [AAP93] has the same form as the function used by [AAF+93]. However, the constants, a, used by [AAP93] and [AAF+93] differ substantially.

for the links in the network. This path selection is achieved with a cost metric that assigns the following cost to path P:

$$-\text{MIN}_{e\in P}\left\{\text{MIN}\left\{1-f_e^f-\Delta f_{i,e}^f, 1-f_e^r-\Delta f_{i,e}^r\right\}\right\}.$$

2.2 Multiple requests

This section describes three approaches to primary routing of a *set* of permanent virtual circuit requests. Each of the approaches can be used in conjunction with any of the cost metrics described in Section 2.1.

Ordered on-line. The ordered on-line approach is based on a simple heuristic. For a given network state and source destination pair a virtual circuit request with a low bandwidth requirement generally has more feasible paths than a virtual circuit request with a high bandwidth requirement. Thus, in order to maximize the chance that every virtual circuit request in a set of requests has a feasible path, the ordered on-line approach gives the requests with high bandwidth requirements the chance to choose a path before the requests with a low bandwidth requirement. In particular, the ordered on-line approach sorts the requests in decreasing order of the maximum of the forward and reverse bandwidth. Once sorted, the virtual circuit requests are routed on-line using one of the methods described in Section 2.1. In other words, after the ith request in the sorted list has been routed, the network state is updated with the bandwidth from ith request and then the $(i+1)$st request is routed based on the new network state.

Unfortunately, this approach does not permit the ith request to take any of the $(i+j)$th requests for $j > 0$ into consideration. The following example illustrates some of the resulting problems.

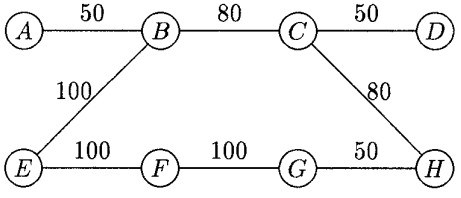

Figure 1: Network for Example 2.1.

Example 2.1 Consider the network in Figure 1 and the set of requests $(E, H, 46)$, $(F, G, 52)$, $(A, D, 45)$ where the elements of each tuple are the source, the destination, and the bandwidth respectively. The forward and reverse bandwidth is the same for each of the requests. The ordered on-line approach first routes the request $(F, G, 52)$ on the direct path F, G and then routes request $(E, H, 46)$ on the path E, B, C, H. As a result, the request $(A, D, 45)$ has no feasible path. However, it is possible to find a feasible path for all three requests. In particular, if request $(E, H, 46)$ is routed on the path E, F, G, H, request $(A, D, 45)$ could use path A, B, C, D.

Local search. The local search approach attempts to address the problems outlined in Example 2.1. Rather than considering each virtual circuit request individually, the local search approach considers all requests in a set together. Let p_i be the path used by request i. Then the local search approach used in conjunction with exponential routing attempts to find the *set* of feasible paths that will minimize the following cost function that is the generalization of (1) to multiple requests.

$$\sum_e a^{(f_e^f+\sum_{i,e\in p_i}\Delta f_{i,e}^f)} - a^{f_e^f} + \sum_e a^{(f_e^r+\sum_{i,e\in p_i}\Delta f_{i,e}^r)} - a^{f_e^r} + \sum_e \sum_{i,e\in p_i} c(\Delta f_{i,e}^f + \Delta f_{i,e}^r) \qquad (2)$$

The problem of finding such a set of paths is clearly NP-complete due to the single path constraint for each virtual circuit. We use a greedy local search to find a set of paths whose cost is close to the minimum cost.

The search starts with the set of paths determined using the ordered on-line approach. If the ordered on-line approach cannot find a feasible path for a particular request, it uses the cheapest infeasible path. Denote the set of paths found using the ordered on-line approach by $\mathcal{P}_0$. The set $\mathcal{P}_0$ is said to be the *current* set. Let $C(\mathcal{P}_0)$ be the cost of the paths in $\mathcal{P}_0$ based on (2). Define $N(\mathcal{P}_0)$ to be the set of path sets that differ from $\mathcal{P}_0$ in at most one path, i.e., at most one virtual circuit has a different path. The local search now finds the lowest cost path set in $N(\mathcal{P}_0)$. Call this path set $\mathcal{P}_1$. If $C(\mathcal{P}_1) < C(\mathcal{P}_0)$ then $\mathcal{P}_1$ becomes the current set. This procedure is repeated to find the lowest cost path set in $N(\mathcal{P}_1)$, etc. The process stops after the $i+1$st iteration where i is the lowest index such that $C(\mathcal{P}_{i+1}) \geq C(\mathcal{P}_i)$. The final set $\mathcal{P}_i$ is called a local minimum of (2). However, $\mathcal{P}_i$ need not be a global minimum. Thus, $\mathcal{P}_i$ may still have infeasible paths even if the global minimum has no infeasible paths. New feasible paths are found for the virtual circuits without a feasible path in $\mathcal{P}_i$ using the ordered

on-line approach. Finally, the virtual circuit requests are routed based on the resulting path set. Simple calculations will show that the local search approach successfully routes all three requests in Example 2.1.

In order to ensure a reasonable running time we limit the number of iteration for the local search to several hundred. The local search is clearly not guaranteed to find the optimal solution since the solution space may have exponential number of local minima and the local procedure terminates after finding the first local minimum. One could explore different regions of the solution space by starting the local search with different (perhaps random) starting points.

Convex optimization. The convex optimization approach tries to find a global minimum for the cost function (2). To overcome the fact that this problem is NP-complete, this approach relaxes the single path constraint. In particular, let $P_i = \{p_i^1 \ldots p_i^{j_i}\}$ be the set paths over which virtual circuit request i could be routed. Let $b_f(p_i^j)$ to be the fraction of the forward bandwidth of request i that is routed over path p_i^j. Similarly, $b_r(p_i^j)$ is the fraction of the reverse bandwidth routed over path p_i^j. Then the total cost for the path selection defined by the functions b_f and b_r is given by the following generalization of (2) to virtual circuit that are split over multiple paths:

$$\sum_e a^{(f_e^f + \sum_{i,p_i^j \in P_i, e \in p_i^j} b_f(p_i^j)\Delta f_{i,e}^f)} - a^{f_e^f} + \sum_e a^{(f_e^r + \sum_{i,p_i^j \in P_i, e \in p_i^j} b_r(p_i^j)\Delta f_{i,e}^r)} - a^{f_e^r} + \sum_e \sum_{i,p_i^j \in P_i, e \in p_i^j} (b_f(p_i^j)\Delta f_{i,e}^f + b_r(p_i^j)\Delta f_{i,e}^r) \quad (3)$$

The initial goal of the convex optimization approach is to determine $b_f(p_i^j)$ and $b_r(p_i^j)$ for all i, j such that the cost function (3) is minimized, $\sum_{p_i^j \in P_i} b_f(p_i^j) = 1$ for all i and $\sum_{p_i^j \in P_i} b_r(p_i^j) = 1$ for all i. This optimization goal is easy to express as a continuous optimization with a convex objective function and linear constraints. Thus the global optimum solution of this problem can be found in polynomial time. The optimal objective function value of this problem is a lower bound to the real objective of minimizing the value of (2) subject to the single path constraint. We solve the optimization problem using MINOS 5.3, a software package for solving linear and nonlinear programming problems available from Stanford University [MS87].

The next step in the convex optimization approach is to change the $b_f(p_i^j)$ and $b_r(p_i^j)$ for all i, j found the optimization of cost function (3) to satisfy the single path constraint. In other words we must set $b_f(p_i^j), b_r(p_i^j) \in \{0, 1\}$ for all i, j such that $\sum_{p_i^j \in P_i} b_f(p_i^j) = 1$ for all i and $\sum_{p_i^j \in P_i} b_r(p_i^j) = 1$ for all i. We investigated several randomized strategies for this step but found the best results when using the strategy of Raghavan and Thompson [RT85].

3 Simulations

3.1 Simulation scenario

In order to explore the performance of the different routing strategies, we conducted an extensive set of simulations, varying both the network topology and the characteristics of the request sequences. Where possible, we used data describing existing network topologies and traffic patterns to drive the simulations.

Figure 2 is the topology of an existing network for permanent virtual circuits. The simulation results we present focus on this topology. We also performed simulations using the two topologies pictured in Figure 3. The figure describes the initial topologies of planned networks. In all of the topologies, the capacities of the links are all chosen to be $155Mbps$, which corresponds to SONET OC-3 service.

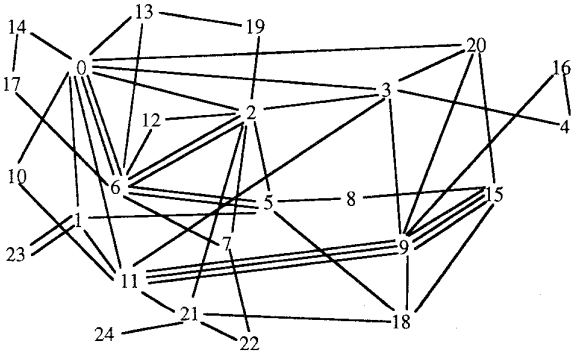

Figure 2: An existing network.

The source and destination of the simulated virtual circuit requests are generated using a traffic matrix that describes the relative amount of traffic between pairs of nodes in the network of Figure 2. In particular, if S denotes the sum of all the entries in the traffic matrix and $s_{i,j}$ denotes the entry in the i^{th} row and j^{th} column, then $s_{i,j}/S$ denotes the fraction of the total traffic that goes between node i and node j. The bandwidth of a circuit request is chosen independently

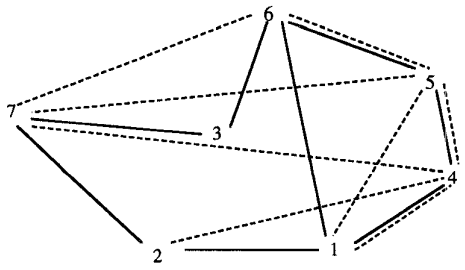

Figure 3: Planned networks. One consists of only the solid links. The other consists of the solid and the dashed links.

of its source and destination. Initially, the bandwidth is drawn from a hyper-exponential distribution with a mean of $1Mbps$. This distribution weights requests with smaller bandwidth more heavily, however it has a long tail, meaning that higher bandwidth requests do occur. The forward and reverse bandwidths are the same. Finally, the average number of virtual circuit requests in a set of requests is chosen from a binomial distribution with a mean of 10. We term this simulation scenario the *base case*. The base case uses the topology of Figure 2.

In Section 3.3 we describe the relative performance of the routing algorithms in the base case and in simulation scenarios that are variations of this case. All simulation results have a 95% confidence interval that is within 1% of the sample mean.

3.2 Performance measure

The performance measure we use in our simulations is the total amount of bandwidth routed by an algorithm when, for the first time, more than 50% of a set of virtual circuit requests is rejected. By terminating the simulation when 50% of a set of virtual circuit requests is rejected, we avoid operating the algorithms in the unrealistic scenario where most circuit requests are being rejected. In practice, the network capacity will always be increased to avoid very high rejection rates. (This performance measure preserves the lower bound of Appendix A. In fact, much stronger lower bounds are possible with this performance measure.)

3.3 Simulation results

Our simulation results compare exponential routing with constants a and c set to 1000 and 10 respectively, minimum-hop routing with random tie break, minimum-hop routing with state dependent tie break[3], and min-max routing. The type of state dependent tie break used for minimum-hop routing had little effect on the simulation results. Unless otherwise stated, we use the ordered on-line approach to deal with the fact that circuit requests arrive in sets.

Bandwidth. Figure 5 provides simulation results for the base case and for simulations that differ from the base case in the mean circuit bandwidth. The y-axis gives amount of bandwidth accepted (in 10,000,000 ATM cells per second) and the y-axis gives the mean circuit bandwidth. In the base case (mean bandwidth is $1Mbps$) minimum-hop routing with a state dependent tie break and minimum-hop routing with a random tie break are respectively 6.5% and 10% worse than exponential routing. Max-min routing is 25% worse than exponential routing. (Although the results are not presented in the figure, we note that minimum-hop routing with an arbitrary tie break is up to 15% worse than exponential routing while pure random routing accommodates up to 60% less bandwidth than exponential routing.)

Figure 5 shows further that exponential routing maintains its performance advantage over a broad spectrum of mean bandwidths. The simulations were also repeated for circuit requests whose bandwidths are chosen from several discrete distributions rather than a continuous distribution. The discrete distributions appear in Figure 4. The use of discrete bandwidth distributions has little effect of the relative performance of the routing algorithms.

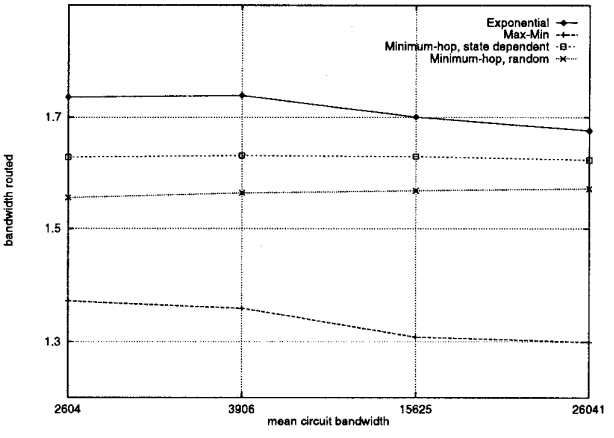

Figure 5: Relative performance as mean bandwidth is varied.

[3] We use the max-min measure for the state dependent tie break.

256K/256K	512K/512K	1.5M/1.5M	1.5M/64K	6M/6M	6M/64K	10M/10M	10M/64K
20%	20%	50%	0%	5%	0%	5%	0%
10%	10%	50%	0%	15%	0%	15%	0%
20%	20%	25%	25%	2.5%	2.5%	2.5%	2.5%
20%	20%	5%	45%	.5%	4.5%	.5%	4.5%
10%	10%	25%	25%	7.5%	7.5%	7.5%	7.5%
10%	10%	5%	45%	1.5%	13.5%	1.5%	13.5%

Figure 4: Discreet bandwidth distributions. The top row indicates a forward/reverse bandwidth. For example, 1.5M/64K means 1.5 Mbps in the forward direction and 64 Kbps in the reverse direction. Each other row represents a bandwidth distribution by indicating the percentage of requests that have a specific forward/reverse bandwidth.

Traffic matrix. Next we consider the effect of changing the degree to which the traffic matrix and the topology are matched. In particular, the simulations use the topology of Figure 2 while the traffic matrix is varied from the base case. At the extreme ends of the set of traffic matrices considered are a random traffic matrix and one that matches the topology perfectly. For a perfectly matched traffic matrix, the arrival rate of the circuit requests for any pair of nodes depends on the number of direct links between the nodes. If a pair of nodes has no direct links, there are no circuit requests that have those nodes as the source and destination. In Figure 6 the traffic matrix is varied from a completely random traffic matrix on the left to base case traffic matrix in the middle and the perfectly matched traffic matrix on the right. The figure shows that the performance advantage of exponential routing evaporates for both a perfectly matched traffic matrix and a random traffic matrix. (The exception is max-min routing, which continues to perform poorly even when the topology and the traffic matrix match perfectly.) The results from the perfectly matched traffic matrix are not surprising. In this case, each of the algorithms, except max-min routing, will generally choose the single link path. Thus, each algorithm essentially routes optimally. The results for the random traffic matrix are more surprising. Inspection of the link utilizations suggests that the results are due to "hot spots". In particular, the random traffic matrix creates regions of the network with traffic demands that completely overwhelm the available capacity. These regions are so dramatically capacity limited that all of the routing algorithms quickly reject most of the requests in that region. Thus, each of the algorithms reaches the termination condition of the simulations approximately simultaneously. (Recall that the termination condition is the first set of circuit request where over half of the requests is rejected. Our performance measure is the amount of bandwidth routed when the termination condition is reached.)

We conclude that exponential routing provides the greatest benefits approximately in the most common mode of network operations. In particular, most networks will have a traffic matrix that lies somewhere between being perfectly matched to the topology and being random.

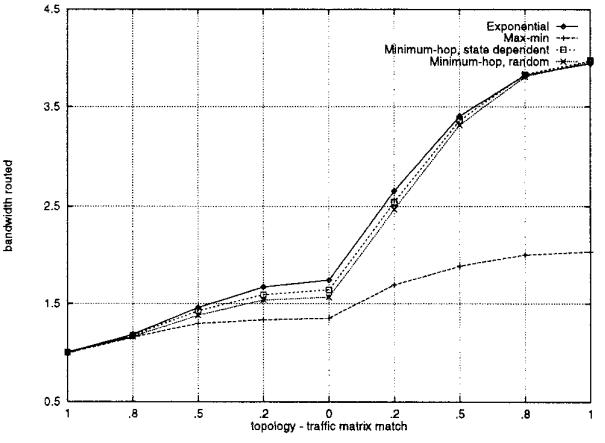

Figure 6: Relative performance as the degree to which the traffic matrix and the topology are matched is varied.

Net flows. For the simulations discussed so far, all circuits have the same forward and reverse bandwidth. Thus, there are no net traffic flows between any pair of nodes. However, there are many situations, such as video servers, where one might expect net traffic flows. To measure the relative performance of the algorithms with net flows, we modify the base case simulations as follows. For some percentage of the circuit requests, the reverse bandwidth is reduced to 10% of the forward bandwidth. Our traffic matrix only specifies node pairs without specifying which node is the source and which the destination. Since the forward and reverse bandwidth now differ, it now mat-

ters which node is picked as the source and which is picked as the destination. Whenever the node pair (i, j) is picked by the traffic matrix, we now choose i to be the source 80% of the time.

Figure 7 provides the results of simulations where the percentage of circuits with reverse bandwidth that is 10% of the forward bandwidth is varied from 30% to 90%. Exponential routing continues to outperform the other routing algorithms. However the relative performance advantage of exponential routing decreases with the magnitude of the net flow. Inspection of the link utilizations suggests that hot spots (c.f. results for random traffic matrix) are responsible for the decline the performance advantage of exponential routing.

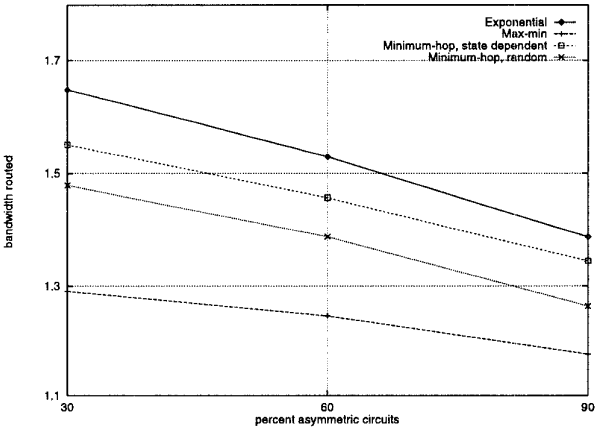

Figure 7: Relative performance as magnitude of net flows is varied.

Topology. We also ran the simulations using the topologies in Figure 3. We omit the results due to space considerations, but note that they are qualitatively similar to the results presented for the network of Figure 2.

Requests set size. The next set of simulations evaluates the effectiveness of the various approaches for dealing with the fact that virtual circuit requests arrive in sets. The simulations shown in Figure 8 vary the number of requests in a set of requests from 20 to 60. (The simulations use the denser of the two topologies in Figure 3. Furthermore, the simulations use exponential routing.). The greedy local search provides some small improvement over the simple ordered on-line strategy while the convex optimization approach provides no improvement. Since it is easy to implement, we recommend the use of the local search approach. The good performance of the ordered on-line approach relative to the other two approaches, both of which attempt to find globally optimal routes for a set of requests, is encouraging since it suggests that on-line exponential routing may perform close to the optimal routing strategy.

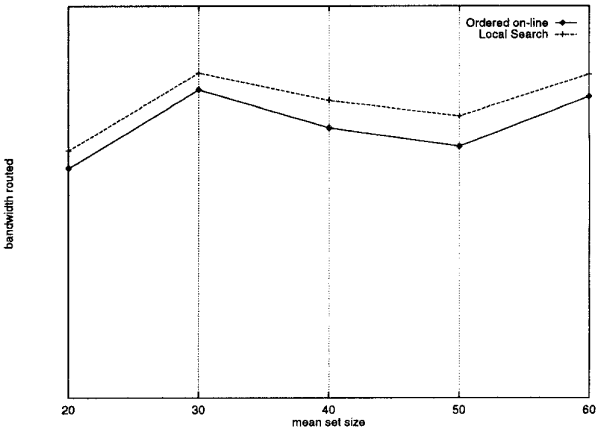

Figure 8: Relative performance as size of request set is varied.

Exponential function constants. In the simulations presented so far, the constants a and c for exponential routing are set at 1000 and 10 respectively. These values were arrived at using simulations. The results of the simulations suggest that the constants a and c are fairly forgiving in the sense that exponential routing performs well over a wide range of a and c values. In particular a can take any value from 10^3 to 10^7 while c can take any value between $a/50$ to 0[4]. The size of the constant a may be somewhat surprising initially. It arises from the fact that the exponent in the cost function is normalized bandwidth and therefore always between 0 and 1. Consider the following example.

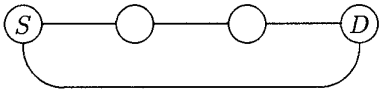

Figure 9: Network for Example 3.2.

Example 3.1 Assume that we wish to route a virtual circuit from S to D in the network of Figure 9.

[4] Values for a that are above 10^7 may also lead to good performance, however we did not test them.

The current bandwidth on the top edges is f_1 in both the forward and the reverse direction, and the current bandwidth on the bottom edge is f_2 in both directions. Assume that the bandwidth of the virtual circuit that we wish to route is Δf in both directions. We now determine the minimum value for a as a function of f_1 and f_2 so that the three hop path using the top edges is chosen over the one hop path using the lower edge. For the purpose of the calculation, we assume that $c = 0$ and that Δf is small enough so that $a^{f_i+\Delta f} - a^{f_i}$ can be approximated by $\Delta f a^{f_i}$ for $i = 1, 2$. If the three hop path is chosen, it must be the case that $2\Delta f 3a^{f_1} \leq 2\Delta f a^{f_2}$, which leads to the condition that $a \geq 3^{1/(f_2-f_1)}$. Figure 10 plots a as a function of $f_2 - f_1$. Notice the steep rise of a for small $f_2 - f_1$ despite the logarithmic scale. The graph suggest that a must be fairly large for the behavior of exponential routing to differ significantly from the behavior of minimum-hop routing. Furthermore, for exponential routing to behave like max-min routing it must be the case that $a \geq h^{1/x}$ where h is the maximum hop count for a virtual circuit (5 in our simulations) and x is the faction of link bandwidth used by the lowest bandwidth connections (approximately 1/2000 in our simulations).

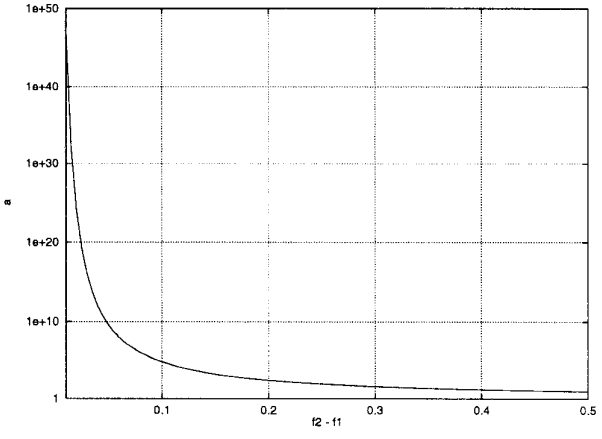

Figure 10: The minimum value of a as a function of $f_2 - f_1$. See Example 3.2.

Discussion. Since minimum-hop routing results in paths that use the fewest amounts of resources, the results showing that exponential routing outperforms minimum-hop routing may seem counter intuitive. The principle behind the exponential function is that it behaves similarly to minimum-hop routing under light loading conditions. However, when links become heavily loaded exponential routing starts choosing long lightly loaded paths in favor of shorter heavily loaded paths. The potential advantages of this approach are illustrated in the following example.

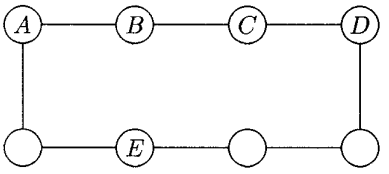

Figure 11: Network for Example 3.1.

Example 3.2 Consider the network in Figure 11. Let the capacity of each link be 10. Furthermore, the links from A to B, from B to C and from C to D already carry 9 units of bandwidth in both directions while the remaining links currently carry no bandwidth. Now consider a virtual circuit request with source A, destination D and bandwidth 1 in both directions. Minimum-hop path routing would pick the path passing through the edge from B to C, thus blocking any future request from B to C. In contrast, exponential routing would pick the path through node E thus preserving the ability to accepts requests from B to C.

4 Summary

The simulations presented in this paper show that exponential routing outperforms several other routing algorithms over a wide range of simulation scenarios. Furthermore, the performance advantages of exponential routing are most pronounced when the traffic matrix lies between being perfectly matched to the topology and being random. If circuit requests arrive in sets, our simulations suggest the use of the local search approach. Finally, we note that the constants used in exponential routing are very forgiving.

5 Acknowledgments

We thank Bala Rajagopalan, Vikram Saksena, Bill Aghinii, and Bob Wentworth for invaluable discussions about the system model and the performance measures for primary permanent virtual circuit routing. Finally, discussions with Yossi Azar and Baruch Awerbuch contributed to the lower bound in Appendix A.

References

[AAF+93] J. Aspnes, Y. Azar, A. Fiat, S. Plotkin, and Orli Waarts. On-line load balancing with applications to machine scheduling and virtual circuit routing. In *Proceedings of the 23^{rd} Annual ACM Symposium on Theory of Computing, San Diego, California*, May 1993.

[AAP93] B. Awerbuch, Y. Azar, and S. Plotkin. Throughput competitive on-line routing. In 34^{th} *Annual Symposium on Foundations of Computer Science,* Palo Alto, California, November 1993.

[Aki84] J. Akinpelu. The overload performance of engineered networks with nonhierarchical and hierarchical routing. *AT&T Bell Laboratories Technical Journal*, 63(7):1261–1281, September 1984.

[BG92] D. Bertsekas and R. Gallager. *Data Networks.* Prentice-Hall, NJ, second edition, 1992.

[Kru82] R. Krupp. Stabilization of alternate routing networks. In *Proceedings of IEEE Internation Conference on Communications*, page 3I.2.1, 1982.

[MS87] B.A. Murtagh and M.A. Saunders. MINOS 5.1 User's Guide. Technical report, Systems Optimization Laboratory, Dept. of Operations Research, Stanford University, Stanford, CA, 1983 revised 1987. Tech report no SOL83-20R.

[RT85] P. Raghavan and C.D. Thompson. Provably good routing in graphs: Regular arrays. In *Proceedings of the 17^{th} Annual ACM Symposium on Theory of Computing, Providence, Rhode Island*, May 1985.

A Appendix

Theorem A.1 (Lower bound) *Let B be an algorithm for on-line routing of permanent virtual circuits with greedy admission control. Assume that B is deterministic or randomized with an adaptive adversary. Then the competitive ratio of B in terms of the amount of bandwidth routed is $\Omega(n)$, where n is the number of nodes in the network.*

Proof: Consider the network G in Figure 12 consisting of $2n+2$ nodes and links of capacity 1. Call the links between nodes v_i and v_{i+1} for $3 \leq i \leq n-1$ the *v-links* and the links between nodes u_i and u_{i+1} for $3 \leq i \leq n-1$ the *u-links*. To prove the theorem we construct a sequence α of virtual circuit requests such that the amount of bandwidth routed by B is $O(1)$ while the amount o bandwidth routed by an optimal off-line algorithm is $\Omega(n)$. The request sequence α is a concatenation of four request sequences α_1, α_2, α_3, and α_4. All of the requests in α have a forward and backward bandwidth of $1/r$ for some $r > 0$.

Let α_1 be a sequence of r virtual circuit requests from S to D. After B routes the requests in α_1, either the *v-links* or the *u-links* are at least 50% utilized. Without loss of generality assume that the *v-links* are at least 50% utilized. Now, α_2 consists of requests from u_2 to u_0 such that after B routes the requests in α_1 followed by the requests in α_2 the link from u_1 to u_2 and the link from u_1 to u_0 are 100% utilized. The sequence α_3 of consists of $r/2$ requests from S to D. Since the *v-links* are at least 50% utilized after the requests in α_1 and the links from u_1 to u_2 and u_1 to u_0 are 100% utilized as a result of the requests in α_2, the *v-links* are 100% utilized after the requests in α_3. Finally, α_4 consists of $(n-3)r/2$ requests such that there are $r/2$ virtual circuit requests between v_i and v_{i+1} for each $3 \leq i \leq n-1$.

Since *v-links* are 100% utilized as a result of the requests in $\alpha_1.\alpha_2.\alpha_3$, B is forced to reject all of the requests in α_4. Thus, the total bandwidth routed by B after routing the requests in α is at most 3. An off-line algorithm, however, can route a total bandwidth of $n/2$. Consider the following strategy. Route all requests in α_1 using the *u-links*. As a result, all of the requests in α_2 will be rejected, and all of the *v-links* are still free. As a consequence, all the requests in α_3 and α_4 can be routed. The total bandwidth routed by this strategy is 1 from α_1, 1/2 from α_3 and $(n-3)/2$ from α_4. ∎

Define the *scaled bandwidth* of a virtual circuit to be the bandwidth of the circuit times the minimum hop count between the circuit's source and destination. An argument similar to the one used in Theorem A.1 can prove an $\Omega(n)$ lower bound for the competitive ratio of on-line permanent virtual circuit routing algorithms with greedy admission control that seek to maximize the scaled bandwidth routed.

Theorem A.2 (Upper bound) *Let B be any algorithm for on-line routing of permanent virtual circuits with greedy admission control. Denote the bandwidth*

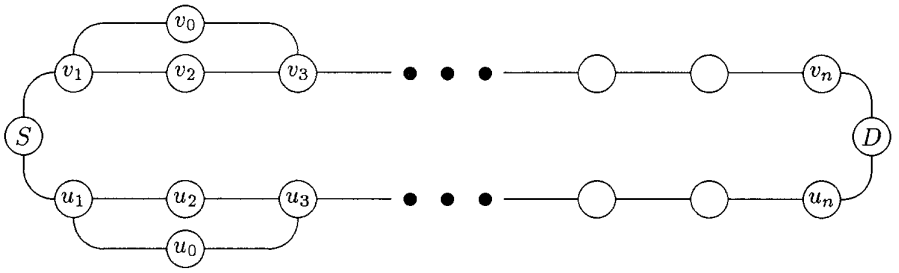

Figure 12: Network G with capacity 1 on each link.

requirement of the ith request of request sequence α by b_i. Assume that for every request sequence α and network G, there exists a constant $\beta > 1$ such that the capacity of the lowest capacity edge is at least β times the bandwidth requirement b_i for all i. Then the competitive ratio of B is $O(n)$, where n is the number of nodes in G.

Proof: Consider any request sequence α and any network with topology $G = (V, E)$. The capacity of link $e \in E$ is given by c_e. Assume that for all i, $\text{MIN}_{e \in E}(c_e)$ is at least βb_i. Let $\mathcal{A}$ be the set of virtual circuit requests in α accepted by algorithm B and let $\mathcal{R}$ be the set of virtual circuit requests in α accepted by the optimal off-line algorithm, but rejected by B. Furthermore, if B accepts the ith virtual circuit request, define P_i to be the path chosen for the ith request by B. Similarly, if the optimal off-line algorithm accepts the ith request, define P_i' to be the path chosen for the ith by the optimal off-line algorithm. Consider the following notation:

$A(i)$: The total capacity accepted by B for all requests up to the ith request: $\sum_{j \le i, j \in \mathcal{A}} b_j$.

$R(i)$: The total capacity accepted by the off-line algorithm but rejected by B for all requests up to the ith request: $\sum_{j \le i, j \in \mathcal{R}} b_j$.

$A_e(i)$: The total capacity accepted by B on link e for all requests up to the ith request: $\sum_{j \le i, j \in \mathcal{A}, e \in P_j} b_j$.

$R_e(i)$: The total capacity accepted by the off-line algorithm on link e for all requests up to the ith request, which B could not have accepted on e due to insufficient capacity: $\sum_{j \le i, j \in \mathcal{R}, e \in P_j', c_e - A_e(j-1) \le b_j} b_j$.

Since any path consists of at most n links, $\sum_{e \in E} A_e(i) \le nA(i)$ for all i.

Assume that the ith request is in $\mathcal{R}$. Since B uses greed admission control and B rejected the ith request, there exists a link $e \in P_i'$ such that $c_e - A_e(i-1) \le b_i$. Therefore, $\sum_{e \in E} R_e(i) \ge R(i)$ for all i.

The remainder of the proof assumes that α is finite. Using a simple limit argument the proof can be extended to infinite α. Let k be the last request in α. Consider any link e. Since the optimal off-line algorithm is also bound by capacity constraints, $R_e(k) \le c_e$. Furthermore, if $R_e(k) > 0$, there exists some request $i \le k$ such that $c_e - A_e(i-1) \le b_i$. Since $b_i \le c_e/\beta$, we conclude that $A_e(i-1) > c_e/\beta$. $A_e(i)$ is by definition monotonically increasing in i, so $A_e(i-1) > c_e/\beta$ implies that $A_e(k) > c_e/\beta$. Using the fact that $R_e(k) \le c_e$ and $A_e(k) > c_e/\beta$ when $R_e(k) > 0$ and the fact that $A_e(k) \ge 0$, we conclude that $R_e(k) \le \beta A_e(k)$. Thus,

$$R(k) \le \sum_{e \in E} R_e(k) \le \sum_{e \in E} \beta A_e(k) \le \beta n A(k).$$

The competitive ratio of B is upper bounded by

$$\text{MAX}_{\alpha, G} \left\{ \frac{A(k) + R(k)}{A(k)} \right\}.$$

Using the fact that $R(k) \le \beta n A(k)$ leads to the desired result. ∎