On the Scalability-Performance Tradeoffs in MPLS and IP Routing [*]

SELMA YILMAZ IBRAHIM MATTA
Computer Science Department
Boston University
Boston, MA 02215, USA
{*selma,matta*}*@cs.bu.edu*

Technical Report BUCS-TR-2002-013

Abstract

MPLS (Multi-Protocol Label Switching) has recently emerged to facilitate the engineering of network traffic. This can be achieved by directing packet flows over paths that satisfy multiple requirements. MPLS has been regarded as an enhancement to traditional IP routing, which has the following problems: (1) all packets with the same IP destination address have to follow the same path through the network; and (2) paths have often been computed based on static and single link metrics. These problems may cause traffic concentration, and thus degradation in quality of service. In this paper, we investigate by simulations a range of routing solutions and examine the tradeoff between scalability and performance. At one extreme, IP packet routing using dynamic link metrics provides a stateless solution but may lead to routing oscillations. At the other extreme, we consider a recently proposed Profile-based Routing (PBR), which uses knowledge of potential ingress-egress pairs as well as the traffic profile among them. Minimum Interference Routing (MIRA) is another recently proposed MPLS-based scheme, which only exploits knowledge of potential ingress-egress pairs but not their traffic profile. MIRA and the more conventional widest-shortest path (WSP) routing represent alternative MPLS-based approaches on the spectrum of routing solutions. We compare these solutions in terms of utility, bandwidth acceptance ratio as well as their scalability (routing state and computational overhead) and load balancing capability. While the simplest of the per-flow algorithms we consider, the performance of WSP is close to dynamic per-packet routing, without the potential instabilities of dynamic routing.

Keywords: Multi-Protocol Label Switching, IP Routing, Constraint-Based Routing, Multicommodity Flow Algorithms, Simulation.

[*]This work was supported in part by NSF grant ANI-0095988.

1. Introduction

The primary purpose of IP routing is to maintain connectivity in the presence of topology changes and network failures. IP routing typically chooses shortest paths to the destinations, based on simple metrics like hop count or delay. While the simplicity of this approach makes IP routing highly scalable, it does not provide any mechanism to enable optimization of resource utilization in the network. Shortest path destination based routing often leads to unbalanced traffic distribution across the network, creating hot spots, while other parts of the network have very light traffic loads [2]. The problems can be illustrated by the famous "fish" example (see Figure 1). Because of IP's shortest path destination based routing, all the traffic from R1 to R6 and R2 to R6 will follow the upper path. Consequently, the sub-path R3-R4-R5 may get over-utilized while the alternate (lower) path stays underutilized.

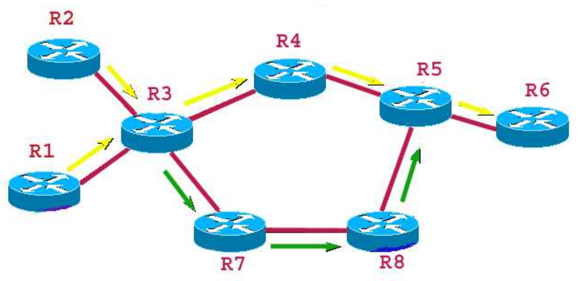

Figure 1. The Fish

In an earlier study [15], it has been shown that in 30-80% of the cases there is an alternate path with significantly superior quality, where quality is measured in terms of loss rate, bandwidth and round trip time (RTT). The non-optimal routing of the Internet can simply be improved by making use of these alternate paths.

In this context, several solutions are proposed to increase utilization of the network; among them are per-packet dynamic routing, and per-flow explicit routing. Although per-packet dynamic routing is a very effective way to balance network load, early attempts in the ARPANET [12] showed that it is hard to deploy because of potential routing oscillations. Even with the improvements in the definition of the link metrics [9], designing stable schemes for dynamic routing is shown to be fundamentally difficult in packet-based networks like the Internet [21]. Consequently, there has been considerable interest in traffic engineering using explicitly routed paths. While explicit routes can be chosen in many ways, most recent solutions, such as Minimum Interference Routing (MIRA) [7] and Profile-based Routing (PBR) [17], come in the context of MPLS-based routing. The challenge with these approaches is the need to keep state at the per-flow level, which limits the scalability of the solution. The other challenge is the requirement of having extra information in the form of possible ingress-egress pairs (for both MIRA and PBR) and traffic profiles between these pairs (in the case of PBR). In this study, we would like to examine the gains from this added complexity, i.e. how close the performance of these algorithms gets to the performance of optimal per-packet dynamic routing.

The rest of the paper is organized as follows: Section 2 gives a brief review of widest-shortest path routing (WSP), MIRA, PBR, and dynamic per-packet routing algorithms. Section 3 describes our experiments, and performance metrics. Results of the experiments are presented in Section 4. Section 5 revisits related work and finally Section 6 concludes the paper.

2. Review of Evaluated Algorithms

Bandwidth guaranteed paths can be computed dynamically using one of the following algorithms: Minimum Interference Routing (MIRA), Profile-based routing (PBR), and widest-shortest path routing (WSP). These algorithms are all source routing algorithms where explicit feasible routes are computed at the source node and resources are then reserved over the selected path before actual transmission begins. Furthermore, these algorithms do not split individual flow requests among multiple paths.

The most traditional algorithm of all is WSP [6]. WSP keeps track of residual bandwidth on the links, and when a request comes, a feasible shortest (min-hop) path with the largest residual path capacity is chosen. It has been shown that under heavy load, WSP achieves good performance (low blocking rate) by limiting resource consumption and balancing load when there are multiple shortest paths [10]. WSP uses neither ingress-egress pair information nor traffic profiles. Therefore, it is the least expensive of all and has computational complexity of Dijktra's shortest-path algorithm. However, not powered with ingress-egress pair information, it may end up accepting a request which creates a potential to block a large number of future requests.

MIRA is proposed to reduce blocking rate by having information regarding ingress-egress pairs. Since the maximum flow value between given ingress-egress pairs is the upper bound on the bandwidth that can be routed between them, the idea is to keep the minimum maxflow as big as possible to prevent blocking. Before routing each request, MIRA calculates weights for the links based on their criticality. The link is critical if the maxflow value of some ingress-egress pair decreases when a request is routed over it. The run time complexity of MIRA is $O(nVE^2)$, where n is the number of ingress-egress pairs, V is the number of nodes, and E is the number of edges.

PBR assumes both ingress-egress node pairs and the traffic profile between them are known. A traffic profile is the aggregate bandwidth requirement between a pair of ingress-egress nodes for a particular traffic class. In this study, we assume there is only one traffic class between any ingress-egress pair. In the offline phase of the algorithm, a multicommodity flow problem is solved, where each profile is a separate commodity. The objective is to route as much commodity as possible. The result of this phase is the pre-allocated link capacities for each commodity. The multicommodity flow problem is

$$minimize \sum (cost(e) \sum x_i(e)) \qquad (1)$$

subject to capacity and flow conservation (except at ingress and egress nodes) constraints. The output of this linear programming formulation is the values of variables $x_i(e)$ which denote the amount of commodity i that is routed through edge e. This phase provides static traffic engineering and cannot be done too often.

The on-line phase of the algorithm uses routed capacities from the offline phase as virtual capacities for each class. The on-line algorithm keeps track of the residual capacities along those virtual capacity links. To route a request, the shortest (min-hop) feasible path is chosen. The offline phase can be solved in polynomial time [8], and the on-line phase can be solved by a breadth first search algorithm which runs in $O(V + E)$. Since the offline phase is not intended to be done frequently, PBR will mostly run on-line. However, the offline phase has to be repeated from time to time to adjust to link failures, or changing traffic conditions. Because of the resulting new pre-allocated (virtual) capacities, re-routing of existing flows will be needed. The challenging part of the algorithm is gathering traffic matrix information, which adds extra complexity. The main advantage of profile-based routing is to provide some kind of admission control and prevent accepting a request that will put the network in a state where blocking probability gets very high.

In [17], example topologies and sequence of request arrivals are shown where MIRA leads to very high blocking probability as compared to PBR. The conclusion was that PBR can improve performance by $O(n)$, where n is the number of ingress-egress pairs. However, we present an example where PBR can perform much worse than MIRA. The topology is shown in Figure 2. Assume the capacities are as shown in the figure and there are 2 ingress-

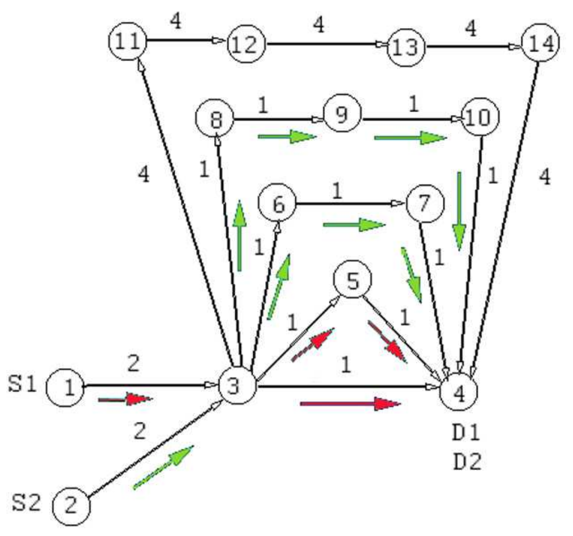

Figure 2. Rainbow Topology

egress pairs $(S1, D1)$, and $(S2, D2)$. Suppose there is one class between each ingress-egress pair. By using the notation in [17], profiles can be represented as quadruples of the form $(classID, ingress, egress, profile)$. Assume we have following profiles: $(class_1, S1, D1, 2)$, and $(class_2, S2, D2, 2)$. First, the offline phase will solve the multicommodity flow problem with these commodities. The solution of the multicommodity flow problem tries to minimize the cost, therefore it will choose routes along the shortest paths as much as possible. While doing this, it may split the individual profile of a class along multiple paths. In Figure 2, there are 5 paths available with increasing hop length. The top most path, 3-11-12-13-14-4, is the longest path and has enough capacity to satisfy both profiles. However, the solution of the offline phase will allocate capacities for each profile on the shorter paths between node 3 and node 4, which are 3-4, 3-5-4, 3-6-7-4, and 3-8-9-10-4. Since any one of these shorter paths cannot satisfy the individual profiles, a profile of a class will be split along two paths. Therefore, pre-allocated capacities will be 1 unit for each class on two different paths. A possible solution is shown in the figure: For profile $(class_1, S1, D1, 2)$, 2 units of bandwidth are reserved along the link 1-3, 1 unit is reserved along the link 3-4 and 1 unit is reserved along the path 3-5-4. Similarly, for the profile $(class_2, S2, D2, 2)$, 2 units of bandwidth are reserved along the link 2-3, 1 unit is reserved along the path 3-6-7-4 and 1 unit is reserved along the path 3-8-9-10-4. No bandwidth will be reserved on the top most expensive (longer) path. For the on-line phase, assume each request asks for bandwidth of 2 units. Since we are not allowed to split flows, PBR will reject all the requests. However, MIRA and even WSP will prune the links with capacity 1 since they are infeasible, and route a request along the top most path which is assumed to have large enough capacity. Thus, MIRA and WSP can perform better than PBR. We can also generalize the example for n ingress-egress pairs, in a similar topology. We can then show that MIRA and WSP can perform $O(n)$ better than PBR.

Our ideal per-packet dynamic routing refers to the algorithm where instantaneous link states are known when packets are routed. The routing decision is done hop-by-hop at each intermediate node. This algorithm gives the optimal performance and in this study it is used as a benchmark. The properties of dynamic routing can be listed as (1) hard to implement in practice, because of potential oscillation problems; (2) stateless; (3) computationally simple; (4) leads to high communication overhead; (5) since packets of the same flow may follow different paths, packets can be reordered (this will cause performance degradation for applications using TCP-like protocols [14]); and (6) doesn't require any extra knowledge like ingress-egress pairs and/or traffic matrix or profiles.

Table 1 shows the properties of the algorithms. We can easily see that dynamic per-packet routing is the most scalable of all. Assuming distance-vector routing, it maintains only distance vectors at each node, and uses distributed route computation. In addition, it maintains neither per-flow nor per-class state. Among the others WSP is the only one which does not use extra information in the form of ingress-egress pair matrix or traffic profile matrix, which makes the algorithm less complex (both time and space), easier to implement, and more scalable as compared to MIRA and PBR. Although time complexity of PBR (in the on-line phase) is smaller than MIRA, its space complexity is the highest. Also, extra effort is needed to gather traffic profiles and the offline phase needs to be run from time to time. Overall, while trying very hard to improve performance, PBR sacrifices scalability.

3. Experiments and Simulation Set-up

A traffic profile is represented by the quadruple $(class_i, s_i, d_i, B_i)$, where $class_i$ is the traffic class between ingress node s_i and egress node d_i. B_i is the aggregate traffic expected for this class between s_i and d_i (profile). Throughout the study, we have assumed there is only one class between an ingress and an egress node. A request belonging to some class is defined by the quadruple (id, s_i, d_i, b_i) where b_i is the bandwidth requirement of the request. In this study, we assumed that flows belonging to the same class have the same b_i, and b_i's are selected randomly from [1,50] according to a uniform distribution. Flows from a class arrive according to a Poisson process of rate λ_i. Flow holding times are modeled using a Pareto distribution with mean μ_i and shape parameter set to 2.5 to capture the heavy tail nature of connection durations while still producing a distribution with finite variance to have smaller confidence intervals [16]. The total load is defined as $\rho = \sum_{i=1}^{K} \lambda_i/\mu_i$, where K is number of classes. Given

	Widest-Shortest Path [6]	**Minimum Interference Routing [7]**	**Profile-based Routing [17]**	**Dynamic per-packet routing**
Solves	Guaranteed bandwidth	Guaranteed bandwidth	Guaranteed bandwidth	Best Effort
Routing Strategy	Source	Source	Source	Hop-by-hop
Type	Link State	Link State	Link State	Link State or Distance Vector
Time Complexity	O(ElogV) [3]	$O(nVE^2)$ n is number of ingress-egress pairs	O(V+E) for on-line phase +offline phase	O(1)
Communication Complexity	O(k) to distribute link states k is number of neighbors	O(k) to distribute link states k is number of neighbors	O(k) to distribute link states k is number of neighbors	O(k) to distribute link states if Link State k is number of neighbors O(V) if Distance Vector
Space Complexity	O(E) for network state	O(E) for network state $O(V^2)$ for ingress-egress pair matrix	O(E) for network state $O(V^2)$ for ingress-egress pair matrix $O(V^2)$ for traffic profile matrix	O(E) for network state if Link State O(kV) for path state if Distance Vector k is number of neighbors
Maintained State	per-flow	per-flow	per-class and per-flow	none
Extra Information Used	none	ingress-egress pair matrix	ingress-egress pair matrix traffic profile matrix	none

Table 1. Algorithms and their properties.

λ_i, b_i, and μ_i, B_i is computed by the following equation: $B_i = (\lambda_i/\mu_i)b_i$. With changing load, profile values are changed. For each simulation, to be able to have the largest reservations at the pre-processing (offline) phase of PBR, we have scaled up the B_i values such that the solution of the multicommodity flow problem remains feasible. Thus, PBR can achieve its best performance.

To simulate dynamic per-packet routing with optimal performance, we have solved the multicommodity flow problem given in Equation (1) each time a flow arrives or departs. In this case, the commodities are the requests that are active at the time of the event. By solving the multicommodity flow problem at each event occurrence, we get different amounts of commodity (flow) routed along the links each time. Thus, a flow can not only be routed on different paths, but also get different bandwidth during its lifetime at different times. Since at each event occurrence (flow departure or arrival), we are computing the optimal routes for the existing set of flows without putting any restrictions on splitability, we get optimal overall performance. There is one aspect that needs explanation: Since satisfying all bandwidth requirements is not always possible, we have added extra edges, as suggested in [17], between ingress and egress nodes of active flows. These extra edges are called *excess edges* and not part of the original graph, but added to route the excess flow along them so that the problem always has a feasible solution. Excess edges are assigned infinite capacity to force the routing of flows through original edges as much as possible (original edges have cost of 1). We can easily see how much of the request can be accommodated by simply checking how much of it is not routed over the original graph, which is the amount routed along the corresponding ingress-egress excess edge. This can be seen from Figure 3. The figure shows the resulting bandwidth assignment for a flow with requested bandwidth b. Since $b4$ of b is routed along the excess edge, the routed amount is simply $b - b4$. We think of $b4$ as congestion-induced performance loss for that flow.

The obvious expectation is that MIRA, PBR and WSP will all perform worse than optimal dynamic routing algorithm in terms of network utilization. The reason is that these algorithms prevent individual flows from being split. On the other hand, the multicommodity flow problem given in Equation (1), is an optimization problem, and the traffic of a source-destination pair can be split along multiple

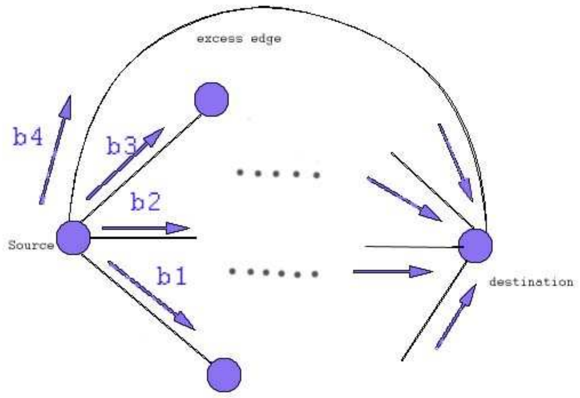

Figure 3. A possible bandwidth assignment for a flow under dynamic per-packet routing. The flow asks for bandwidth b and gets only $(b1 + b2 + b3)$

paths. If splitting of flows is not allowed, the solutions obtained will be sub-optimal [19]. This version of the multicommodity flow problem is exactly the same as given in Equation (1), but the values of x_i are restricted to b_i or 0.

The other expectation is that since WSP does not use any extra information like ingress-egress pairs or traffic profiles, its performance *may* be worse. If there are no alternate paths, MIRA and WSP algorithms should perform similarly. In this case, both are forced to choose the same route while the admission control mechanism of PBR due to pre-allocated (virtual) capacities makes the usage of this route optional.

To solve the multicommodity flow problem, we have used the PPRN package [13].

3.1. Performance Parameters and Measures

Three metrics are used to compare the performance of the algorithms: *Bandwidth acceptance ratio*, *utility* and *maximum utilization*. Although mostly circuit switched routing performance is measured in terms of connection blocking ratio, this is not an accurate measurement in our study. First, bandwidth requirements of the flows are not the same. Second, under dynamic per-packet routing, a flow during its lifetime can get different amounts of bandwidth at different times. Therefore, we have used *bandwidth acceptance ratio* as a performance measure defined as:

$$\frac{\sum_{i=1}^{n}(bandwidth\_accepted\ \Delta t_i)}{\sum_{i=1}^{n}(bandwidth\_requested\ \Delta t_i)}$$

where $T = \sum_{i=1}^{n} \Delta t_i$, and Δt_i's are time intervals over which there is no flow arrival or departure.

Note that our dynamic per-packet routing solves the optimization problem given in Equation (1). We take $Cost(e)$=1, thus the solution gives preference to shortest paths, filling them up first. In other words, our dynamic per-packet routing attempts to *pack* shortest paths first so as to increase the *bandwidth acceptance ratio* at the expense of load balancing [11].

Our second metric is *utility*, which is defined as

$$(\sum_{f \in F}(\sum_i \Delta t_i b_i / T b_f))/N$$

where F is the set of flows that arrive during the simulation time T. During each Δt_i, the flow f gets bandwidth b_i, while its requested bandwidth is b_f. N is the size of set F. For MIRA, PBR, and WSP, since no flows are split, and a flow is either admitted or rejected, this metric reduces to $number\_of\_flows\_accepted/N$. *Utility* is used to measure how well the user needs are met.

Our last metric is *maximum utilization*, which is defined as

$$(\sum_{i=1}^{n}(max\_link\_utilization_i\ \Delta t_i))/T$$

This metric is used to measure how well the algorithms balance the load.

4. Results

For the simulations, the topologies shown in Figure 2, Figure 4, and Figure 11 are used. As expected, for all topologies, as total load increases the *utility* and *bandwidth acceptance ratio* get smaller and *maximum utilization* gets higher.

For the parking lot topology shown in Figure 4, the number of ingress-egress pairs are 6: $(S_1, D_1), \cdots, (S_6, D_6)$. All the link capacities are 200. The *utility* plot is shown in Figure 5, *bandwidth acceptance ratio* in Figure 6, and *maximum utilization* in Figure 7.

For *utility* and *bandwidth acceptance ratio*, dynamic per-packet routing performs best, while profile-based routing performs worst. This is expected for *bandwidth acceptance ratio*, because dynamic per-packet routing optimizes this metric. However, because of high accepted bandwidth ratio, more requests are satisfied. Thus, utility is also highest under per-packet dynamic routing.

Since there are no alternate paths, all of the per-flow algorithms choose the same (only) path. However, the performance of PBR is lowest because MIRA and WSP do not pre-allocate (virtual) capacities for future requests as

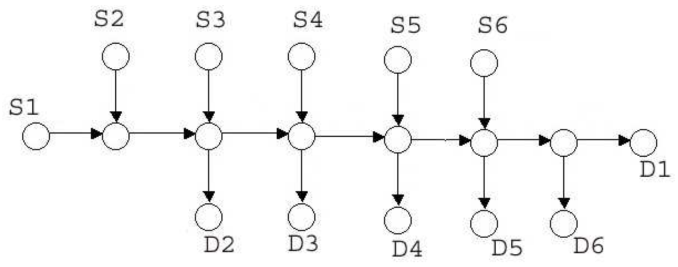

Figure 4. Parking Lot Topology

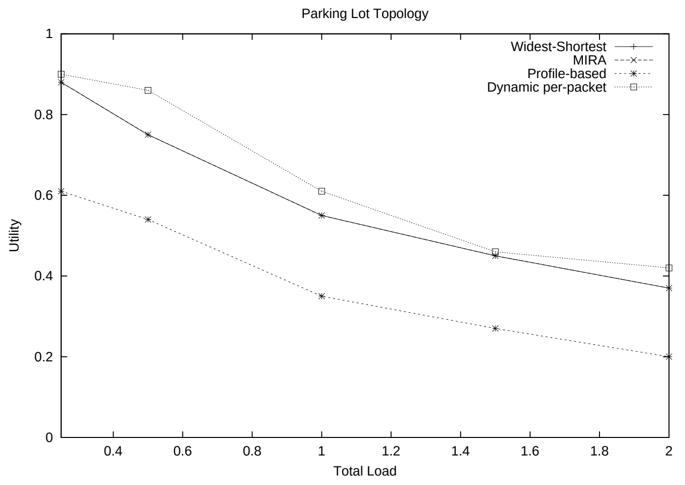

Figure 5. Utility for Parking Lot Topology

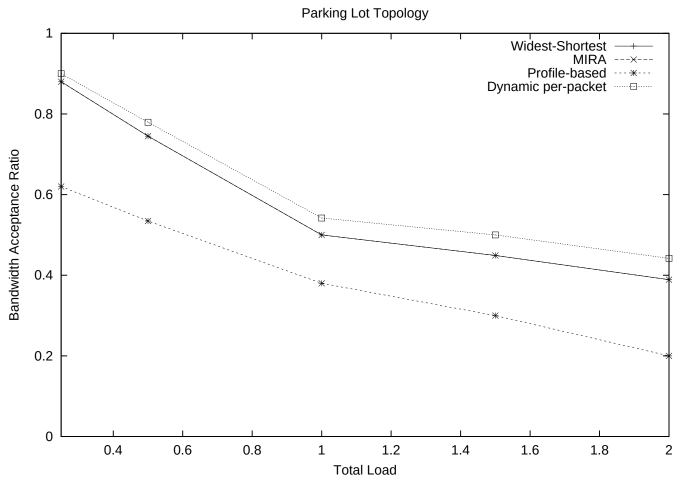

Figure 6. Bandwidth Acceptance Ratio for Parking Lot Topology

PBR. In the case of MIRA, link weights are computed based on criticality, but since there are no alternate paths, MIRA keeps filling the shortest paths as WSP does. On the other hand, PBR is reserving resources beforehand (as outcome

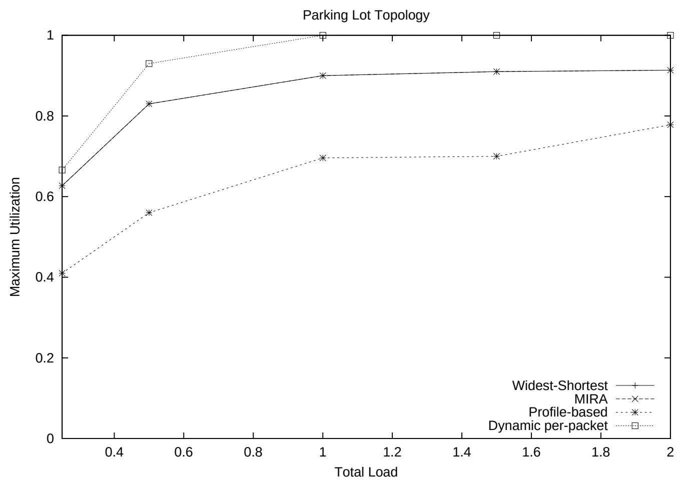

Figure 7. Maximum Utilization for Parking Lot Topology

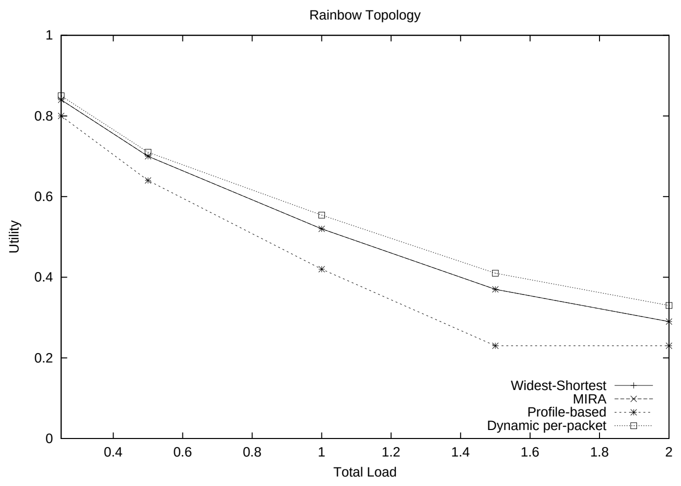

Figure 8. Utility for Rainbow Topology

of the offline phase) and not using them even if they are not currently being used by the corresponding ingress-egress pair(s). That is why PBR begins to reject flows earlier than the other algorithms. At any given time, both MIRA and WSP may end up using all the available resources to meet the current demand, which leads to higher utility and bandwidth acceptance ratio.

For *maximum utilization*, dynamic per-packet routing has the highest value. This metric is not optimized by Equation (1), and the solution of the optimization problem allows to route as much bandwidth as possible preferably along the shortest paths. Therefore, links on the shortest paths become relatively more loaded, resulting in higher maximum utilization. PBR has lowest *maximum utilization*. This is because the carried bandwidth under PBR is lowest, and corresponding resources are underutilized.

For the rainbow topology, shown in Figure 2, the number of ingress-egress pairs are taken to be 2: (S_1, D_1), and (S_2, D_2). The link capacities are 200 for links 1-3 and 2-3, for links on the top most path (3-11-12-13-14-4) the capacities are 400, and the capacities of the remaining links are 100. The *utility* plot is shown in Figure 8, *bandwidth acceptance ratio* in Figure 9, and *maximum utilization* in Figure 10.

As we can see from the results, PBR does not behave terribly worse as it did for the parking lot topology. Interestingly, the *bandwidth acceptance ratio* of both MIRA and WSP gets closer to dynamic per-packet routing, which is optimal. For this topology, we have several alternate paths, and both of the algorithms can make use of all alternate paths. However, as we explained earlier, PBR only preallocates bandwidth (during the offline phase) along particular paths and gets stuck to them even if alternate paths are available. The effects of the amount of accepted bandwidth can be seen from *maximum utilization*. Per-packet dynamic routing has the highest amount of accepted bandwidth at the expense of load imbalance that results in highest maximum utilization. On the contrary, PBR has the lowest maximum utilization due to its lowest accepted bandwidth. The ranking of the algorithms is the same with respect to all metrics as for the parking lot topology. However, the difference between the algorithms is reduced.

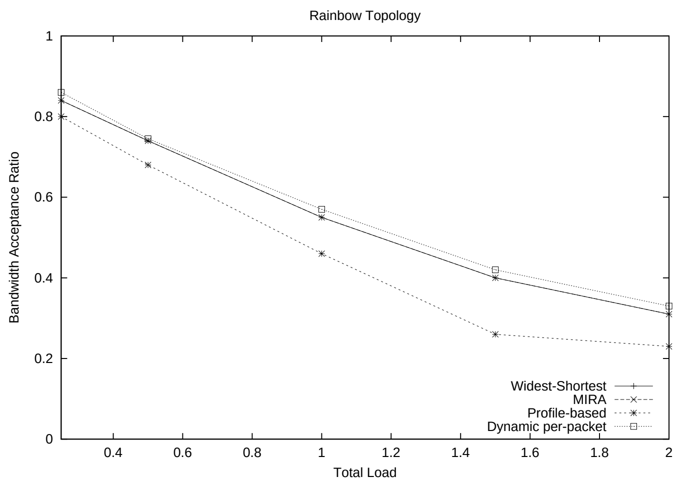

Figure 9. Bandwidth Acceptance Ratio for Rainbow Topology

We study the behavior of the algorithms on a more regular topology shown in Figure 11. The number of ingress-egress pairs are 3, and all the link capacities are 150. The *utility*, *bandwidth acceptance ratio* and *maximum utilization* plots are shown in Figures 12, 13, and 14, respectively. For this topology, MIRA and WSP deviates more from dynamic per-packet routing for both *utility*, and *bandwidth acceptance ratio*. PBR still has the worse performance. For *maximum utilization*, dynamic per-packet routing has the highest value as expected. WSP achieves good load bal-

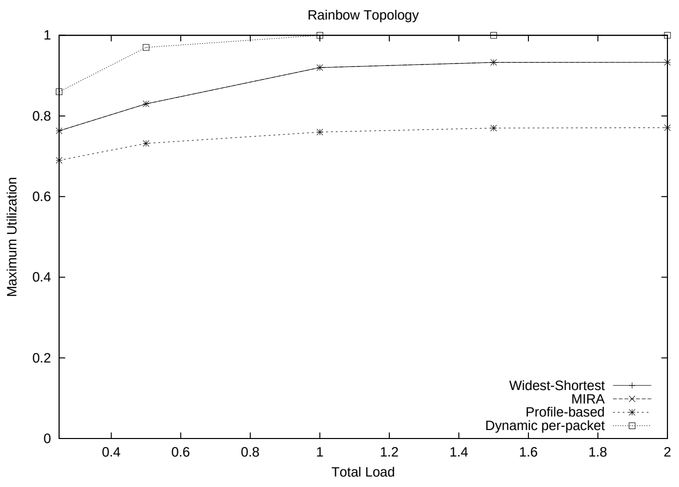

Figure 10. Maximum Utilization for Rainbow Topology

ancing, better than MIRA, while providing the same *bandwidth acceptance ratio*. This is because the main concern of WSP is to balance load, while with MIRA, the main goal is to maximize open capacity of some other ingress-egress pairs. This effect manifests itself in this topology because it has many alternate paths. PBR also has very low maximum utilization, merely because it lets links go underutilized.

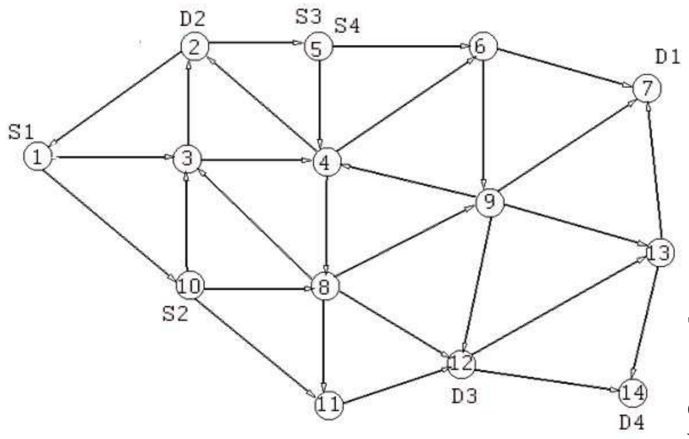

Figure 11. Regular Topology

Overall, we see that for different topologies, PBR cannot achieve the performance of MIRA and WSP, because of loss of statistical multiplexing. WSP performs as well as MIRA, and better balances load. Furthermore, while the simplest of the per-flow algorithms we consider, the performance of WSP is close to dynamic per-packet routing, without the potential instabilities of dynamic routing.

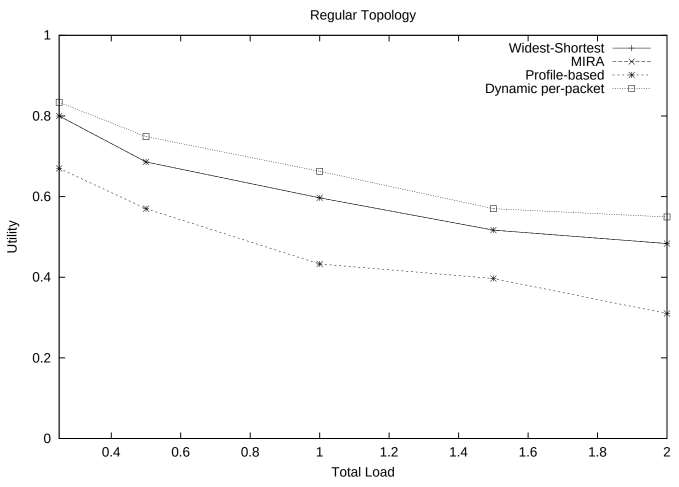

Figure 12. Utility for Regular Topology

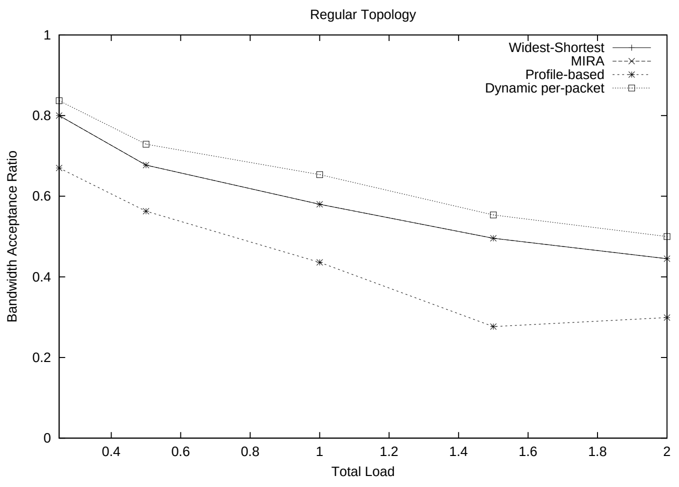

Figure 13. Bandwidth Acceptance Ratio for Regular Topology

5. Related Work

To overcome the shortcomings of traditional static shortest path IP routing and to increase the utilization of the network, several solutions have been proposed. One approach is to use distributed dynamic routing [9, 5, 18]. These algorithms make local measurements of the links, distribute the information, and update the routing tables. They may converge slowly, and may suffer from oscillations.

Another approach is to use the traffic matrix rather than measured link congestion. Using this information, studies in [4, 20, 19, 17] either propose an extension of IP routing to manipulate weights (e.g. of OSPF or IS-IS routing) in such a way that traffic is preferentially routed along underutilized links, or use the traffic matrix to efficiently map demands to the resources in the form of precomputed set of

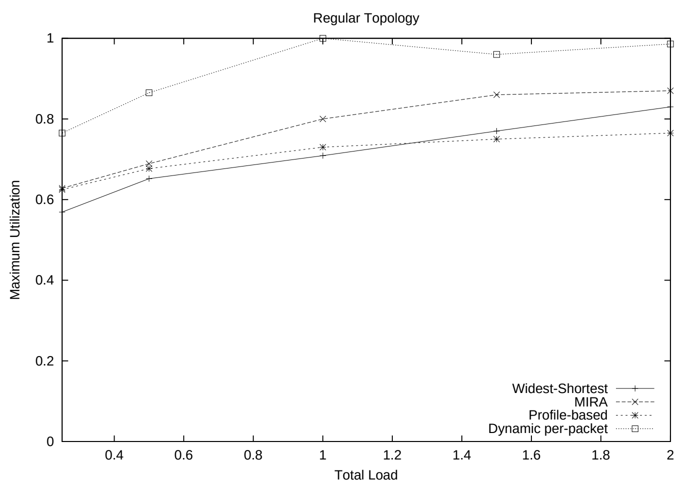

Figure 14. Maximum Utilization for Regular Topology

routes. Other studies [7, 17, 22] only assume the knowledge of an ingress-egress pair set.

While all the above algorithms (along with so many others) are proposed to increase the efficiency of routing in IP networks, the performance of these algorithms against optimal routing has not received much attention. The papers above mainly show that the proposed solutions are better than static shortest path routing algorithms. In this study we tried to fill in this gap, and see how close we can actually get to the optimal. Answering this question is of utmost importance because of the tradeoff between added complexity and performance.

In this context, the only study that has been done and we are aware of is [1]. Our study was done simultaneously and independently of [1]. The paper evaluates relative performance of several algorithms: static shortest path (min-hop and weighted shortest path where the link weight is set as the inverse of link bandwidth), linear programming formulation that optimizes maximum link utilization (min-max), and Gallager's optimal routing that minimizes the queuing delay. They have also evaluated performance of practical approximations of these optimal routing algorithms. One of them extends the first optimization problem where the total path length is also minimized (bi-optimal) and the second one is an approximation of Gallager's algorithm, where the distribution of a flow is readjusted to prevent arbitrary fractional assignment. The algorithms are evaluated for average delay, utilization, and path length, for which Gallager's algorithm, min-max, and bi-optimal are optimal, respectively. The paper concludes that dynamic routing algorithms outperform static routing algorithms for different topologies and workloads.

In this study, we have evaluated different per-flow (MPLS) dynamic algorithms: Minimum Interference (MIRA), Profile-based (PBR), and Widest-shortest path (WSP) routing against dynamic per-packet routing using different metrics. By doing that we have tried to answer the question whether or not the added complexity in terms of increased knowledge about the network's traffic characteristics helps us to reach optimal performance.

6. Conclusion and Future Work

In this study, we have evaluated the tradeoff between performance and complexity of recently proposed routing algorithms: MIRA and PBR. We have compared the performance of these algorithms to WSP and dynamic per-packet routing over different topologies.

Our results show that while dynamic per-packet routing achieves the best performance for both *utility* and *bandwidth acceptance ratio*, PBR shows the worst performance. Since MIRA, PBR and WSP keep per-flow state, dynamic per-packet routing is the most scalable because it is a distributed and stateless solution. Although dynamic per-packet routing cannot provide service guarantees, it achieves good overall *utility*, better than the other three.

Another interesting observation is that WSP performs as well as MIRA. WSP is less costly than both MIRA and PBR. It does not use any extra information in the form of traffic matrix or ingress-egress pairs, and runs as fast as Dijkstra. This makes the algorithm more scalable than MIRA and PBR, and a preferred candidate for deployment. MIRA needs to compute weights on the links each time a new request arrives, and the complexity of this operation is polynomial in the number of ingress-egress pairs. To do this MIRA also needs to know (and store) possible ingress-egress pairs. This makes MIRA less scalable and more complex, in both time and space.

PBR is the least scalable and most complex of all. During the offline phase, the algorithm finds the optimal distribution of profiles. However, because of the unsplitability of flows and static preallocation of capacity, its performance suffers.

For future work, different dynamic routing algorithms optimizing different metrics may be used to evaluate relative performance. PBR and MIRA can also be improved to overcome their inefficiencies.

7. Acknowledgment

We would like to thank Solom Heddaya for initial discussion of this work.

References

[1] E. Anderson, T. Anderson, S. Gribble, A. Karlin, and S. Savage. "A Quantitative Evaluation of Traffic-Aware Routing Strategies". In *Proceedings of ACM SIGCOMM*, San Diego, CA, August 2001. Poster.

[2] D. Awduche, J. Macholm, M. O'Dell, and J. McManus. "Requirements for Traffic Engineering over

MPLS". In *Internet Draft, IETF, draft-awduche-mpls-eng-00.txt*, April 1998.

[3] T. Cormen, C. Leiserson, and R. Rivest. *"Introduction to Algorithms"*. MIT Press, Cambridge, MA, 1990.

[4] B. Fortz and M. Thorup. "Internet Traffic Engineering by Optimizing OSPF Weights". In *Proceedings of IEEE INFOCOM*, Tel-Aviv, Israel, March 2000.

[5] R. Gallager. "A Minimum Delay Routing Algorthm Using Distributed Computation". *IEEE Transactions on Communications*, 25, 1977.

[6] R. Guerin, A. Orda, and D. Williams. "QoS Routing Mechanisms and OSPF Extensions". In *Proceedings of Globecom*, Phoenix, AZ, November 1997.

[7] K. Kar, M. Kodialam, and T. Lakshman. "Minimum Interference Routing of Bandwidth Guaranteed Tunnels with MPLS Traffic Engineering Applications". In *Proceedings of IEEE INFOCOM*, Tel-Aviv, Israel, March 2000.

[8] L.G. Khachiyan. "A Polynomial Time Algorithm for Linear Programming". *Dokl. Akad. Nauk SSSR*, 244, 1979.

[9] A. Khanna and J. Zinky. "The Revised ARPANET Routing Metric". In *Proceedings of ACM SIGCOMM*, Austin, Texas, September 1989.

[10] Q. Ma and P. Steenkiste. "On Path Selection for Traffic with Bandwidth Guarantees". In *Proceedings of ICNP: IEEE International Conference on Network Protocols*, Atlanta, Georgia, October 1997.

[11] I. Matta and A. Bestavros. "A Load Profiling Approach to Routing Guaranteed Bandwidth Flows". In *Proceedings of IEEE INFOCOM*, March 1998. Extended version in *European Transactions on Telecommunications - Special Issue on Architectures, Protocols and Quality of Service for the Internet of the Future,* March-April 1999.

[12] J. McQuillan, I. Richer, and E. Rosen. "The New Routing Algorithm for the ARPANET". *IEEE Transactions on Communications*, 28(5), May 1980.

[13] The PPRN Package. http://www-eio.upc.es/ ˜ jcastro/pprn.html.

[14] V. Paxson. "End-to-End Routing Behavior in the Internet". In *Proceedings of ACM SIGCOMM*, Stanford, CA, August 1996.

[15] S. Savage, A. Collins, and E. Hoffman. "The End-to-End Effects of Internet Path Selection". In *Proceedings of ACM SIGCOMM*, Cambridge, MA, Sptember 1999.

[16] A. Shaikh, J. Rexford, and K. Shin. "Evaluating the Overheads of Source-Directed QoS Routing". In *Proceedings of ICNP: IEEE International Conference on Network Protocols*, 1998.

[17] S. Suri, M. Waldvogel, and P. Warkhede. "Profile-based Routing: A new Framework for MPLS Traffic Engineering". *Quality of Future Internet Services, Lecture Notes in Computer Science*, 2156, September 2001.

[18] S.Vutukury and J. J. Garcia-Luna-Aceves. "A Simple Approximation to Minimum-delay Routing". In *Proceedings of ACM SIGCOMM*, Cambridge, MA, September 1999.

[19] Y. Wang and Z. Wang. "Explicit Routing for Internet Traffic Engineering". In *8th International Conference on Computer Communications and Networks* , Boston, MA, October 1999.

[20] Y. Wang, Z. Wang, and L. Zhang. "Internet Traffic Engineering without Full Mesh Overlaying". In *Proceedings of IEEE INFOCOM* , April 2001.

[21] Z. Wang and J. Crowcroft. "Analysis of Shortest-path Routing Algorithms in Dynamic Network Environment". *ACM Computer Communication Review*, 22(2), April 1992.

[22] Y. Yang, J. Muppala, and S. Chanson. "QoS Routing Algorithms for Bandwidth-Delay Constraint Applications ". In *Proceedings of ICNP: IEEE International Conference on Network Protocols* , 2001.