Prediction-Based Routing in IP-MPLS Networks

Eva Marín-Tordera, Xavier Masip-Bruin, Sergio Sánchez-López, Jordi Domingo-Pascual

Advanced Broadband Communications Lab
Universitat Politècnica de Catalunya, UPC
Vilanova i la Geltrú, Barcelona, Catalunya, Spain
{eva, xmasip, sergio}@ac.upc.edu

Ariel Orda

Department of Electrical Engineering
Technion I.T.T., Haifa, Israel

ariel@ee.technion.ac.il

Abstract: There are many problems closely involved with the QoS deployment in the current Internet. One of significant relevance focuses on the Qos routing scalability concerns. In this paper we propose a new QoS routing mechanism, named Prediction-Based Routing, mainly based on a precomputation scheme. The main characteristic of such a precomputation scheme is the prediction concept introduced to reduce the scalability issues. Because of this prediction concept links and routes availability are predicted according to a processing system which take routing decisions regardless of the link state information allocated in the nodes databases. As a consequence, updating messages are not required, so overhead is reduced and scalability is enhanced.

Keywords: QoS Routing, Prediction-Based Routing, updating mechanism, routing inaccuracy

1. INTRODUCTION

One of the most significant problems when talking about QoS routing is in fact the scalability. Some of the proposals dealing with this problem are mainly based on either implementing better QoS routing algorithms, introducing a hierarchical structure or using a precomputation approach. There are many contributions in the literature using a precomputation scheme to address such scalability issues. The well known hot potato routing [1] 'predicts' which will be the best route to a destination based on the delay information coming from such destination node. Authors in [2] propose to predict the future traffic load in a link, based on past measured samples of the traffic load in that link. In [3] authors present a dynamic variation of the hot potato routing. Unlike these proposals our contribution focusses on predicting link and route availability instead of predicting incoming traffic load. The *Prediction-Based Routing* (PBR) mechanism has already been presented in [4] as a *Routing and Wavelength Assignment* (RWA) mechanism to be applied to optical transport networks.

Qos routing algorithms look for the "optimal" route between source-destination nodes pair based on the network state information obtained from the network state databases. Despite the fact that some update policies are included in the routing protocol to keep updated routing information, in highly dynamic networks it is extremely difficult to maintain accurate routing information on all network nodes. It has been widely shown in the literature the main causes motivating this inaccuracy, such as the non-negligible delay in propagating the update messages, the aggregation process inherent to hierarchical networks and the mechanisms used to reduce the signaling overhead produced by the need of updating. Hence, assuming the unavoidable existence of such an inaccuracy, new routing algorithms taken into account this parameter must

be sought [5-9]. The PBR mechanism also impacts on the signaling overhead. One of the main skills of the PBR is that by using such an approach update messages are not needed, therefore reducing the signaling overhead. Assuming source routing, nodes do not select routes based on their network state information but also on predicted information. The basic idea of the PBR focuses on the training effort developed by the source nodes to find out the "optimal" route for every traffic request. In short, the prediction is obtained from some information kept in new prediction tables on every source node. The decision of which path is selected is performed by reading these prediction tables.

2. APPLIYING THE PREDICTION-BASED ROUTING IN IP/MPLS NETWORKS

The prediction-based routing is based on the well-known ideas of branch prediction developed in computer architecture [10]. In this area the main target boils down to find out whether a branch instruction will be taken or not before being processed. This is done to speed up the processor. The concepts used in branch prediction can be applied to a network scenario whenever substantial changes are included. The main components of our proposal are described in the next sub-sections.

2.1. Route Registers

Unlike branch prediction where the branch prediction outcomes history is stored in a register, in a network scenario, it is necessary to keep the network state from the point of view of the source node. In order to achieve it the PBR mechanism registers the amount of bandwidth that every source node allocates to every route from such a source node. In order to make understanding easier we assume that the information about both available and used bandwidth are expressed in terms of a percentage of the total capacity of the end-to-end route. We suppose that all the links has the same 100% of bandwidth capacity. There is one register per route on every source node. These registers are updated with information about assigned bandwidth from the point of view of these source nodes. One of the main skills of the PBR mechanism is that the registers updating process is achieved without flooding the whole network with messages containing the network state information. Because of the lack of updating messages the bandwidth allocated in the registers in the source nodes cannot be the real bandwidth occupation.

Since the information about assigned bandwidth is used to access some tables (so-named prediction tables), this information must be digitalized in order to build the index of such prediction tables. Just as an example if the number of bits used to digitalize the bandwidth information is 1, we can assign '0' to the index when the used bandwidth in the path is bigger or equal than the 50%, and otherwise we assign '1'. Instead, if the number of bits used to digitalize the bandwidth information is 2 then 0 (00 in binary) stands for an used bandwidth bigger than 75%, 1 (01 in binary) stands for an used bandwidth between 75-50%, 2 (10 in binary) stands for an used bandwidth between 50-25%, and finally 3 (11 in binary) stands for an used bandwidth lower than 25% and so on depending on the number of bits. These indexes express more or less the available bandwidth.

2.2. Prediction Tables

In the source nodes there is one prediction table for every feasible route from that node. Every route register has its corresponding prediction table. The prediction tables

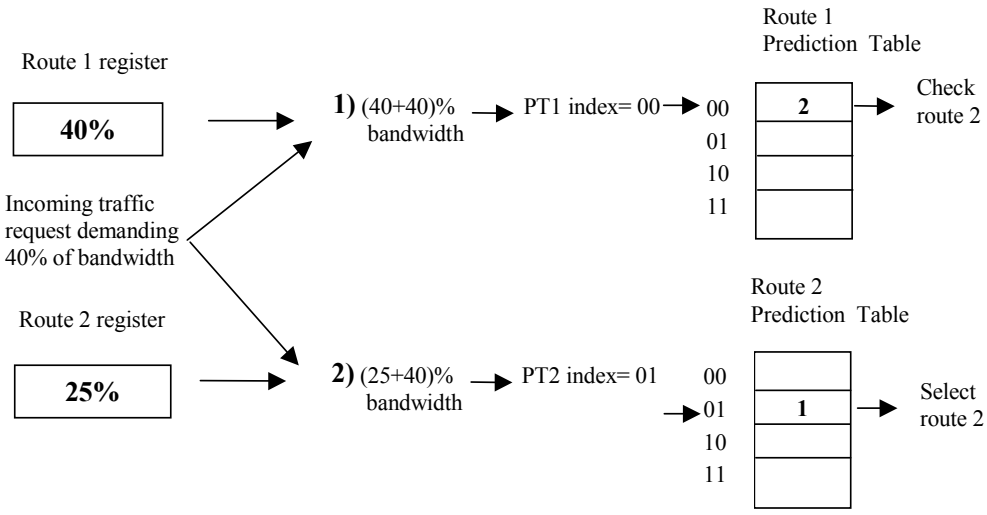

Figure 1. Showing the PBR performance

have different entries, each one keeping the information about a different pattern by means of a two bit counter. The use of two values to account for the availability or the unavailability has been well studied in the area of branch prediction. As it is presented in [10] a two bit counter gives better accuracy than a one bit counter. It is also exposed in [10] that going to counters larger than two bits do not necessarily give better results. This is due to the "inertia" that can be built up with a large counter. A two-bit counter can have 4 values: 0, 1, 2 and 3. The prediction is done reading the value of such a two bit counter. If the value is 0 or 1 (less than 2) the prediction is to select that route, otherwise, if the prediction is 2 or 3 the prediction is that the route is unavailable and it is not selected then. The number of entries of the prediction tables depends on the number of bits of the route registers. For example, if the route registers keep information about the occupied bandwidth in the route with two bits; the number of entries of the prediction tables will be 4.

2.3. Algorithm

The *Predictive Selection of Route* (PSR) is the routing algorithm inferred from the PBR we suggest in the paper. Its performance is shown in the example of Fig.1. We assume there are two routes computed between every source-destination nodes pair (the algorithm is able to compute more than 2 routes) as well as the assigned bandwidth is stored in the route registers by two bits. The PSR algorithm checks the 2 shortest routes in a computed order, according to the availability of their links. In the case of PSR there are not any updates of the network state information, that is, the information about the availability of the links does not represent the current picture of the network. Indeed, without updating, every node only knows how routes and links the node has assigned in the past. This information dictates the order by which the PTs are checked. For the example of Figure 1, we suppose that the first route to be checked is Route 1 and the second one is Route 2. When a new request demanding 40% of bandwidth reaches the source node, the first route is examined. The last information in this first route is that the used bandwidth is 40%. This used bandwidth is incremented by the requested bandwidth, i.e. 40%+40%. If the resultant bandwidth is lower than 100 % then the prediction table of the first route is checked, that is the counter of the corresponding entry read; otherwise the next prediction table would be checked. In our example the

total bandwidth is 80% (>75%) so that the index used to access the first prediction table is 0 (00 codified in 2 bits). With this index, the prediction table of the first path is accessed and the counter is read. According to the Fig.1 the value obtained after accessing the prediction table is a 2, so that the prediction turns out to avoid the first route. Hence the second route is examined.. In this second path the used bandwidth is 25%, so that the resultant bandwidth will be 40%+25%=65%. This means an index of 1 (01 codified with 2 bits). The prediction table of the second route is accessed with this indexed obtaining a value of 1. According to this counter value, the algorithm selects this second route since prediction states that this route will not be blocked. It is necessary to specify that the algorithm checks both the counter value of the prediction table and the availability of the node's outgoing link towards the route 1 or the route 2. This is done because the nodes always have updated information of availability of their outgoing links. In Fig. 2 there is a summary of the algorithm. We call the functions that check the availability of route 1, Check(Route1), and for route 2 Check(Route2). In the example, after checking the prediction tables of the two routes, if the algorithm had not selected any route because it predicted that the two routes will be blocked, the algorithm would select the route that has availability in their outgoing link. That is the algorithm selects the route only checking the availability of the outgoing links of the node. In the summary of the algorithm these functions are called CheckF(Route1) and CheckF(Route2).

New request demanding an X% of bandwidth.
Check(Route 1):
The new bandwidth is added to the bandwidth kept in the route1 register (Y%). The total bandwidth is X+Y%.
If (X+Y)% <=100% the PT of the first route is checked
If(PT counter<2) and there is availability in the outgoing link the algorithm selects the route1
Else Check(Route 2).
Else Check(Route 2)
Check(Route 2)
The new bandwidth is added to the bandwidth kept in the route2 register (Z%). The total bandwidth is X+Z%.
If (X+Z)% <=100% the PT of the second route is checked
If (PTcounter<2)) and there is availability in the outgoing link the algorithm selects the route2
Else CheckF(Route 1)
Else CheckF(Route 1)
CheckF (Route 1):
The new bandwidth is added to the bandwidth kept in the route1 register (Y%). The total bandwidth will be X+Y%.
If (X+Y)% <=100%
If there is availability in the outgoing link the algorithm selects the route1
Else CheckF(Route 2).
Else CheckF(Route 2)
CheckF (Route 2):
The new bandwidth is added to the bandwidth kept in the route1 register (Z%). The total bandwidth will be X+Z%.
If (X+Z)% <=100%
If there is availability in the outgoing link the algorithm selects the route2
Else No route is assigned
Else No route is assigned

Figure 2. Summarizing the PSR algorithm

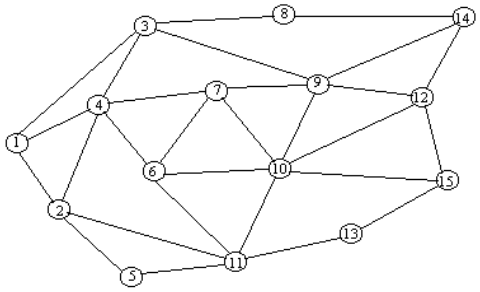

Figure 3. Topology used in the simulations

2.3. Updating procedure

The route registers are updated with the information about the used bandwidth for the source node in every route. In the example above when the algorithm selects the second route the new bandwidth occupied by this node in this second route will be 65%. It is important to note that this used bandwidth is only the bandwidth known by the node which might be quite different from the real occupation since other source nodes may allocate other bandwidths in links of the same route. Due to the lack of update messages this source node will not be aware of such a bandwidth changes.An important issue to be considered is that only the prediction table of the selected route is really updated. Hence, if the connection can be setup the corresponding counter on the prediction table is decreased, otherwise if the connection is blocked the counter is increased. In the above example if the connection is established the counter of the entry 01 in the prediction table of route 2 will be 0, but if the connection is blocked the counter will be 2. The fact of trying to select routes only checking the outgoing availability when any route is assigned (by means of functions CheckF(Route1) and CheckF(Route2)) is done to unblock the PT counters. When both routes might not be selected because the PT counters of route 1 and route 2 are bigger than 1, the PSR algorithm runs the function CheckF(Route1), and if the route cannot be assigned the algorithm runs CheckF(Route2). If the route is selected by means of function CheckF(Route1) or CheckF(Route2) and the connection can be setup, then the corresponding PT counter of route 1 or route 2 is decreased therefore unblocking it.

3. PERFORMANCE EVALUATION

In order to evaluate our proposal we first measure the percentage of blocked connections produced by the PSR algorithm when varying the number of bits used to digitalize the requested bandwidth. Then, we compare the performance of the PSR algorithm with a well-known QoS routing algorithm, the *Widest Shortest Path* (WSP) [11] (which requires the updating process). The WSP dynamically selects for every new incoming request, the route with more available bandwidth among the shortest ones. We assume in our evaluation that shortest routes are link disjoint. Finally we have carried out a set of simulations to check the learning process of the PSR algorithm. All the performed simulations are obtained by applying the PSR and the WSP to the NSF net topology shown in Fig.3. We assume nodes (1, 2, 11, 12, 14 and 15 in Fig.3) acting as source and destinations node. A Poisson distribution models the connection arrivals, and all the links have the same available bandwidth, which is normalized to 100%. Each arrival connection requires a percentage of the total bandwidth.

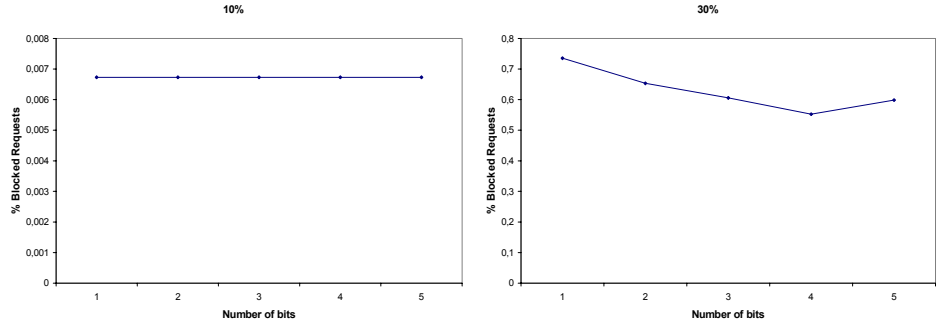

Figure 4. % of Blocked Request of the PSR algorithm for two different traffic loads

3.1. Number of bit to codify the requested bandwidth

In the first evaluation we test the performance of the PSR algorithm when the number of bits used to digitalize the bandwidth is changed for different traffic loads. Notice that in every one of the 6 possible source nodes there are one prediction table, PT, and one route register for every possible route from such a source node. If there are 5 possible destinations, and there are 2 possible routes for every one, there are 10 route registers and 10 prediction tables in every source node. The length of these route registers and the number of entries of the prediction tables depend on the number of bits used to codify the bandwidth. For example, if the number of bits is 1, in the source nodes there will be 10 route registers of 1 bit, and 10 prediction tables of 2 entries each one (remind that in every entry of the PT there are a 2 bit counter). But if the number of bits used to codify the bandwidth is 5, the route registers will have a length of 5 bits, and the PT will have 32 entries each one. The holding and arrival times of the incoming requests are measured in units of time. Besides, all different traffic loads have a holding time of 10 units of time, and also an arrival time of 10 units of time. In order to change the traffic load we change the average requested bandwidth demanded by the incoming request from 10% to 30 % of average bandwidth. In Fig.4 we can see the results of these simulations, that is the percentage of blocked connections versus the number of bits used to codify the bandwidth requested for different traffic loads. From the obtained results we observe that the optimum number of bits depends on the pattern of traffic load, for 10% of average bandwidth for all the different number of bits we obtain the same results, but for 30% of average bandwidth the best results are when the bits used to digitalize the bandwidth is 4. Being aware the lower the number of bits the lower the hardware cost, we only present results for 1, and 4.

3.2. PSR versus Widest Shortest Path

In Fig.5 we present results obtained for the number of blocked connections for the *Widest Shortest Path* and the PSR algorithm versus the time between the update messages are flooded in the network, for different traffic loads. Note that the PSR algorithm does not vary with the time of updating because it does not need update messages. In general, the *Widest Shortest Path* behaves better than the PSR only when the time between update messages is 1 unit of time. Even, for 10% the PSR behaves better than the *Widest Shortest Path* when the time between update is 1 unit the time. But it is important to note that one update message every single unit of time is unaffordable from the point of view of the signaling overhead. In fact, in our simulation sending an update message every single unit of time implies (in average) approximately

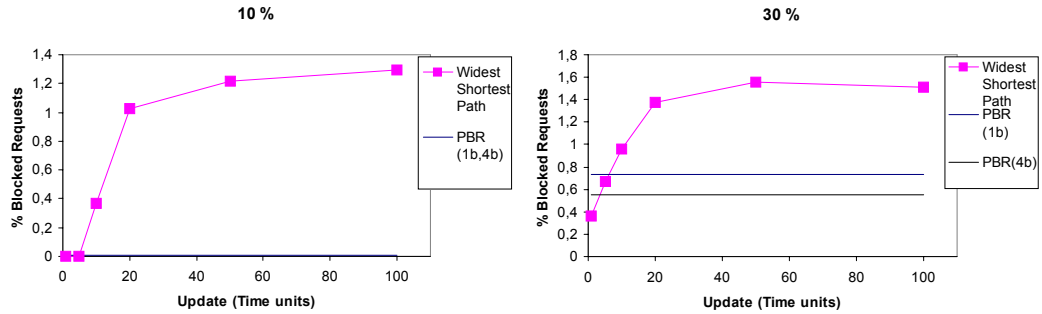

Figure 5. % of Blocked Requests for the *Widest Shortest Path* and PSR algorithm for two different traffic loads

3 set-up connections per updating; or if updating is every 5 units of time it implies 15 set-up connections per updating.

The results of the PSR algorithm show that the routing based on prediction is a valid option because of both its capacity of learning how to assign routes and the complete reduction of the signaling overhead owing to flooding update messages (only topology update messages are required).

3.3. Learning time of the PSR algorithm

Finally we evaluate the time required to train the prediction tables, i.e., the learning time for the PSR algorithm. Results on Fig.6 show the number of blocked connections versus the number of connections requests for the PSR and the Widest Shortest Path algorithm. Results are presented for 2 different traffic loads: when the average demanded bandwidth is 10 % and 30 % respectively. For traffic load of 10 % in the x-axis the number of connection requests ranges from 0 to 20000 new requests. In this graphic we can observe that initially the PSR behaves worse than the Widest Shortest Path. However, when the number of requests is approximately 8000 the PSR has similar number of blocked requests than the Widest Shortest Path with Updating=5 units of time. Finally, for 15000 requests the PSR reaches the results of the Widest Shortest Path with Updating=1 unit of time. For this traffic load we can consider that the PSR has a "time of learning" of 8000 or 15000 request, depending on what we define as learning time. On the other hand we also present results for a traffic load of 30% of average bandwidth requested. In this case the PSR never reaches the results of the *Widest Shortest Path* algorithm with Updating=1. The PSR has similar performance than the *Widest Shortest Path* algorithm with Updating=5, and clearly behaves better from

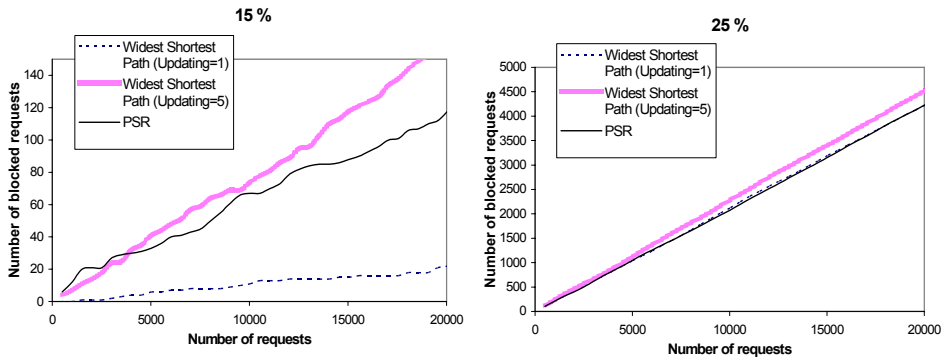

Figure 6. PSR process of learning vs the WSP

10000 requests. Therefore, we can argue that the "time of learning" of the PSR algorithm strongly depends on the traffic pattern.

4. CONCLUSIONS

We have presented a precomputation approach for QoS routing based on the prediction concept to address the scalability concerns in QoS routing. Our proposal is based on predicting links and routes availability not according to the network state information but according to its capacity of learning. One of the main skills of this approach is that update messages are not needed, so reducing the signaling overhead. Simulation results show that the algorithm inferred from the PBR mechanism behaves better than usual routing algorithms such as the *Widest Shortest Path* for reasonable network scenarios, that is, when the updating frequency is affordable the *Widest Shortest Path* has a better performance than the PSR.

ACKNOWLEDGEMENTS

This work was partially funded by the MCyT (Spanish Ministry of Science and Technology) under contract FEDER-TIC2002-04531-C04-02 and the CIRIT (Catalan Research Council) 2001-SGR00226 and the European Commission through the Network of Excellence E-NEXT under contract FP6-506869.

REFERENCES

[1] P. Baran, "On Distributed Communications Networks", IEEE Transactions on Communications, pages 1-9, 1964.

[2] T.Anjali, C.Scoglio, J.de Oliveira, L.C. Chen, I.F. Akyldiz, J.A. Smith, G.Uhl, A.Sciuto, "A New Path Selection Algorithm for MPLS Networks Based on Available Bandwidth Estimation", QofIS 2002.

[3] C. Busch, M. Herlihy, R.Wattenhofer, "Routing without Flow Control", ACM Symposium on Parallel Algorithms and Architectures 2001.

[4] E. Marín-Tordera, X.Masip-Bruin, S.Sánchez-López, J.Solé-Pareta, J.Domingo-Pascual, "A New Prediction-Based Routing and Wavelength Assignment Mechanism for Optical Transport Networks", QofIS 2004.

[5] R.A.Guerin, A.Orda, "QoS routing in networks with inaccurate information: theory and algorithms", IEEE/ACM Transactions on Networking, Vol.7, nº.3, pp. 350-364, June 1999.

[6] D.H.Lorenz, A.Orda, "QoS routing in networks with uncertain parameters", IEEE/ACM Transactions on Networking, Vol.6, nº.6, pp.768-778, December 1998.

[7] G.Apostolopoulos, R.Guerin, S.Kamat, S.K.Tripathi, "Improving QoS routing performance under inaccurate link state information", Proc. ITC'16, June 1999.

[8] S.Chen, K.Nahrstedt, "Distributed QoS routing with imprecise state information", Proc.7th IEEE International Conference of Computer, Communications and Nettworks, 1998.

[9] X.Masip, S.Sánchez, J.Solé, J.Domingo, "QoS Routing Algorithms under Inaccurate Routing Information for Bandwidth Constrained Applications", Proc. IEEE ICC, Anchorage, Alaska, May 2003.

[10] J.E. Smith, "A study of branch prediction strategies", In Proc. of 8th International Symposium in Computer Architecture, Minneapolis 1981.

[11] R. Guerin, A.Orda and D. Williams, "QoS Routing Mechanism and OSPF Extensions", in Proceedings of 2nd Global Internet Miniconference (joint with GLOBECOM'97), Phoenix, USA, November 1997.