Evaluation of the Local State Fair Share Bandwidth Algorithm

Sven Krasser, Henry L. Owen,
David A. Barlow
Department of Electrical and
Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332-0250, USA
{sven, owen, barlow}@ece.gatech.edu

Jochen Grimminger, Hans-Peter Huth,
Joachim Sokol
Siemens AG, CT IC2 Corporate Technology
D-81730 Munich, Germany
{joachim.sokol, jochen.grimminger, hans-peter.huth}@mchp.siemens.de

Abstract

The purpose of this research is to further evaluate the Local State Fair Share Bandwidth (LSFSB) algorithm for traffic-engineering in an incomplex radio access network (RAN). LSFSB is a simple router-based algorithm using routers on the edges of the network make the decisions about which Multi-protocol Label Switched (MPLS) paths to place admitted traffic. LSFSB requires very little knowledge of global network state in order to function because it uses only local router state information. LSFSB was designed to work in an MPLS/DiffServ/HMIP environment in a RAN. Simulation is done using the network simulator ns2.

Keywords: Traffic-engineering, LSFSB, Mobile IP, MPLS, Differentiated Services, Radio Access Networks

1. Introduction

This research further evaluates the performance of the LSFSB algorithm as described in [1], [2], [3] in a simple RAN with a topology shown in Figure 1. We do not explain the details of the LSFSB algorithm here since they are presented in [1], [2], and [3]. The RAN routers incorporate MPLS to enable traffic-engineering for IP traffic, and DiffServ to make different Qualities of Service (QoS) available to the mobile nodes that connect to such a RAN. We assume that mobility support is provided by a protocol such as HMIP.

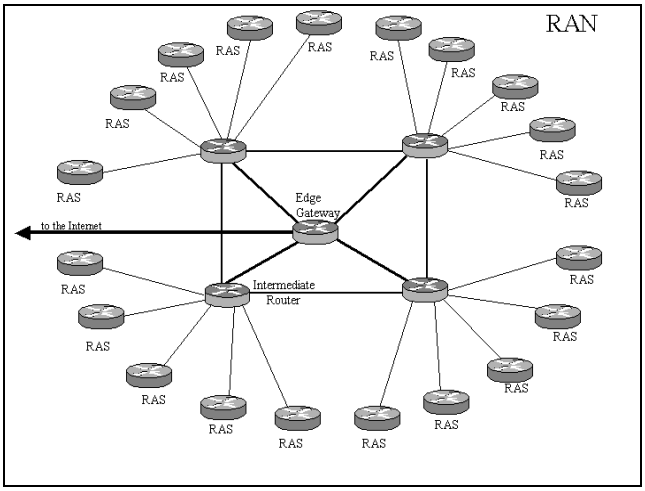

Figure 1: Radio Access Network

0-7803-7661-7/03/$17.00 ©2003 IEEE

The LSFSB algorithm was designed to be simple and independent of the distribution of network state information, hence it just relies on locally available data, namely the local link state and the share of the core bandwidth of a certain radio access router (RAS) as it makes decisions as to which existing MPLS path to place traffic on.

Every time a traffic request is issued to a radio access router (RAS) or edge gateway (EGW), it determines if there is enough bandwidth available on the local link. Since there is no global network state information available, every RAS/EGW determines also its share of the core RAN bandwidth. This data is incorporated by a widest shortest path (WSP) algorithm, which picks one of three label switched paths (LSPs) that have been set-up during network initialization to forward traffic on.

2. Simulation

Simulation is done using the simulator tools described in [2], which consist of the network simulator ns2 [4], version 2.1b6a, with MPLS Network Simulator v1 [5] and DiffServ for ns2 [6].

We compare the performance of LSFSB to Open Shortest Path First as implemented in most IP networks and a reference algorithm called Global Widest Shortest Path (Global WSP, or also just referred to as Global). Global imposes the upper bound for the performance of LSFSB because it knows full state information about the entire network. However, it is just based on bandwidth accounted on distinct links for service requests. It does not incorporate any dynamic metric for routing decisions like delay or actual traffic demands of a source, e.g. due to shrinking the congestion window of a TCP sender. We expect an algorithm that incorporates these metrics to outperform Global WSP.

There are three different movement scenarios available in the simulator. The basic one is called *Random*. Each node randomly chooses a point to move towards and then moves towards that point as the simulation progresses. The *Linear* movement scenario introduces additional mobile nodes that move as a group, roughly synchronized, towards the same destination. This places a point load on the network. In the *Fixed* movement scenario, there are a number of randomly moving nodes like in the other two

scenarios. However, there are additional nodes connected to all the RASes in one corner of the RAN, which do not move.

The DiffServ traffic mix used consists of 40% expedited forwarding (EF) traffic, 20% assured forwarding (AF) traffic, and 40% best effort (BE) traffic. For EF we use only UDP senders, for BE only TCP senders. AF traffic consists of both TCP and UDP datagrams.

3. Results

We investigate the behavior of the LSFSB algorithm concerning the number of dropped packets in the expedited forwarding traffic class (EF). Simulations involve two different movement scenarios: *Random* and *Linear*.

In the random movement scenario nodes connect randomly to the RASes. This yields an evenly distributed traffic load for the network. To get an idea about a reasonable number of nodes we should simulate for this movement scenario, we started out to gather data about the relation between the number of random nodes to the EF drop percentage (Figure 2). The number of connections that we select to use in the following simulations should be in an interval in that congestion already takes place, but should still be low so that the simulation does not get computationally too expensive. We picked a number of 130 random nodes since the graph implies that all three algorithms have to deal with congestion at this value. Figure 3 shows a plot of the EF drop percentage for 130 randomly moving mobile nodes versus the amount of edge gateway traffic. We vary edge gateway traffic from 20% to 80%.

The linear movement scenario assumes a fixed number of nodes moving linearly in a group, and a number of random nodes that represent background traffic. The latter might be zero as well. In our previous work we used a number of 120 linearly moving nodes. However, this high number results in very high drop percentages for the expedited forwarding traffic class that do not occur in real networks – since the carrier of such a network would have had already taken countermeasures. Thus, we reduced the number to 40 in this work. In Figure 4 we present a plot of the total number of nodes versus the EF drop percentage. Based on Figure 4 we chose a total number of 80 nodes to show the impact of edge gateway traffic on the number of drops in Figure 5. If the number of random nodes is increased further, LSFSB performs worse than Global WSP, as you can see in Figure 6.

In both linear and random traffic scenarios all routing algorithms do rather well until a certain threshold of nodes

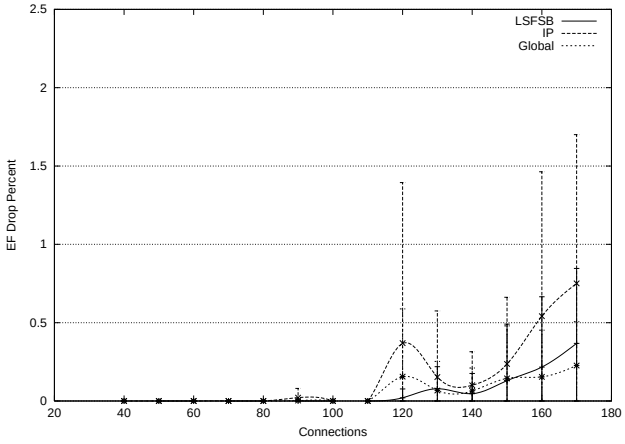

Figure 2: Random Movement, 50% EGW Traffic

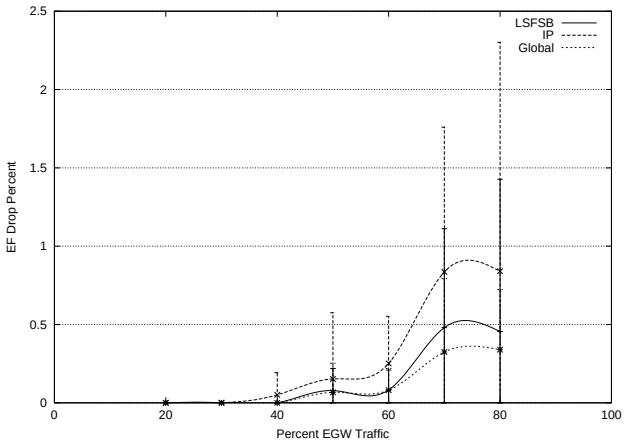

Figure 3: Random Movement, 130 Connections

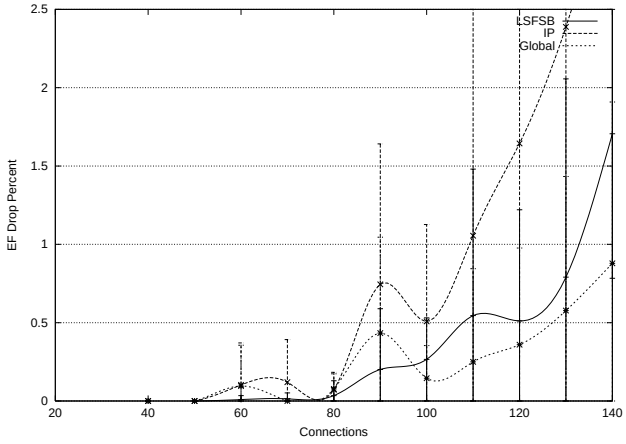

Figure 4: Linear Movement, 50% EGW Traffic

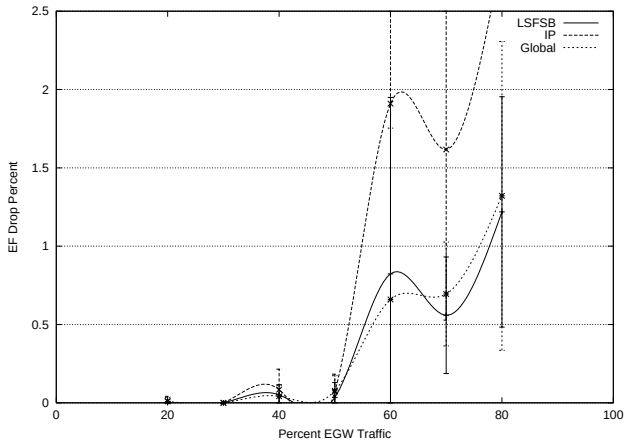

Figure 5: Linear Movement, 80 Connections

is reached that try to communicate. Since some nodes in the linear scenario move in groups, there are hot spots in the network and the traffic load is not evenly distributed. The traffic that occurs at one intermediate router in the

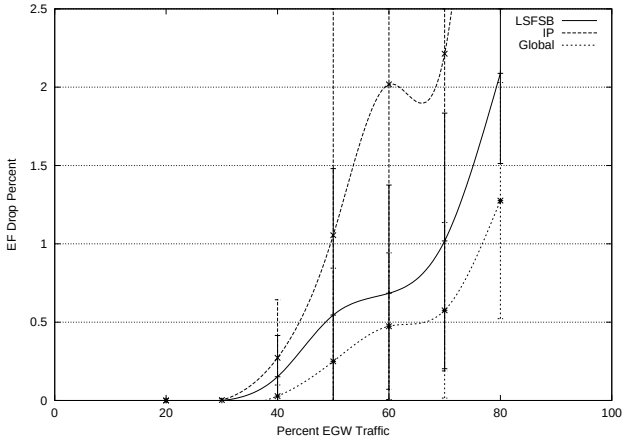

Figure 6: Linear Movement, 110 Connections

linear situation can be much larger than the bandwidth of all three neighboring links together. For this reason the drop rate is increased in this scenario, one region of the RAN might be very congested and experience lots of drops.

In general the lower bound for the performance of LSFSB is given by IP shortest path routing, the upper one by the Global WSP algorithm. However, there are uncertainties due to e.g. statistical issues.

The data we will present in the following paragraphs for assured forwarding (AF) traffic is based on simulations

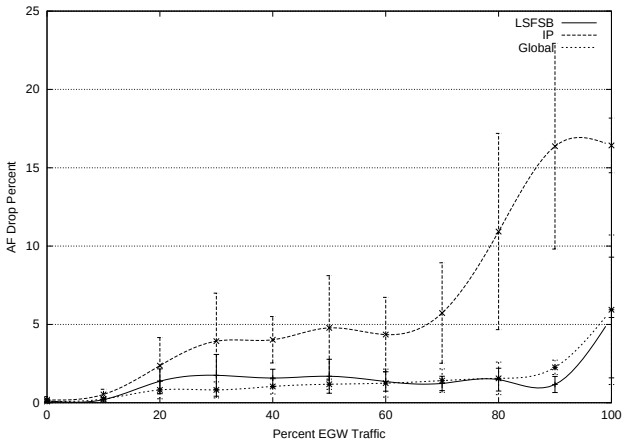

Figure 7: Linear Movement, 120 Connections

with 120 linearly moving nodes. As the name “assured” implies the drop rates are partly still at a reasonable value although the network experiences an extreme high load.

In Figure 7 we show drop rates in the case of just linearly moving nodes. Both traffic-engineering algorithms, LSFSB and Global Widest Shortest Path, do rather well. IP shortest path routing experiences an increased drop rate. For Figure 8 we introduce 100 additional nodes, which randomly connect to the RASes, and, in the case of Figure 9, a number of 150 additional nodes respectively. Global WSP shows increased drop rates, but those are at a reasonable value with regard to the high traffic load. The performance of LSFSB gets closer to

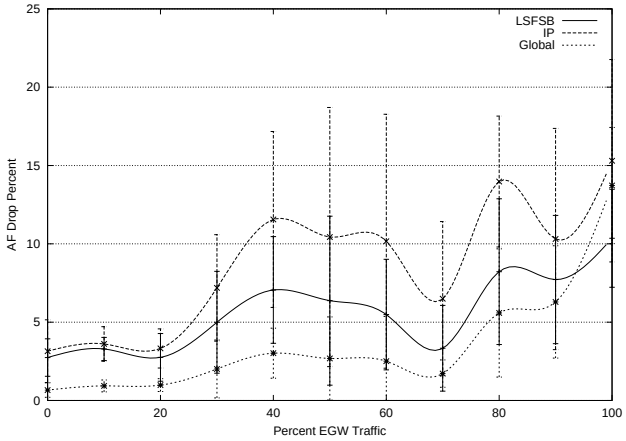

Figure 8: Linear Movement, 220 Connections

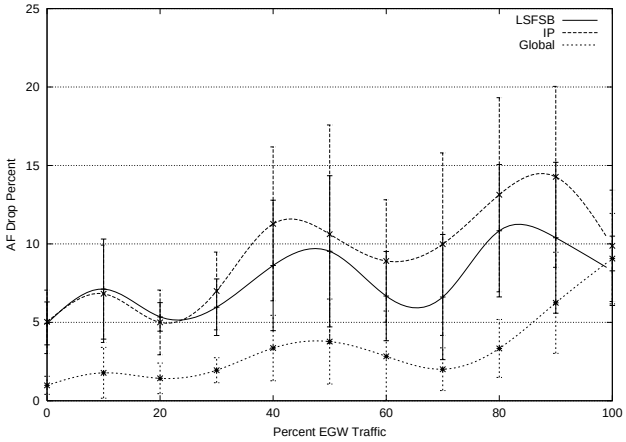

Figure 9: Linear Movement, 270 Connections

the one of standard IP routing when more random nodes are put into the simulation.

In general, all algorithms do better for less edge gateway traffic and perform more poorly for very high amounts of edge gateway traffic.

The drop rates for all routing algorithms for the best effort (BE) traffic class are relatively equal as long as there is no background traffic due to additional randomly moving sources (Figure 10). Now looking at throughput in Figure 11, the throughput for IP is slightly smaller than for the traffic-engineering algorithms. This makes sense since all BE traffic in the simulation uses TCP as the transport layer protocol. A dropped packet implies congestion to the TCP sender, and therefore the sender throttles down if it notes a drop. However, the bandwidth that is freed by that might now be grabbed by a TCP sender in the AF traffic class, so there is no benefit for the BE traffic source – it just loses bandwidth, congestion is not reduced, and therefore it yields throughput to other traffic classes. The BE drop rates stay constant, but the throughput goes down.

In Figure 12 and Figure 13 we show the corresponding graphs for the same simulation setup with 100 additional randomly moving nodes as background traffic. IP surprisingly outperforms in most cases both traffic-engineering algorithms in this scenario. However, this does not imply that traffic engineering fails for BE traffic. This is more related to the way DiffServ works and the simulation is set up. If lots of EF or AF packets are

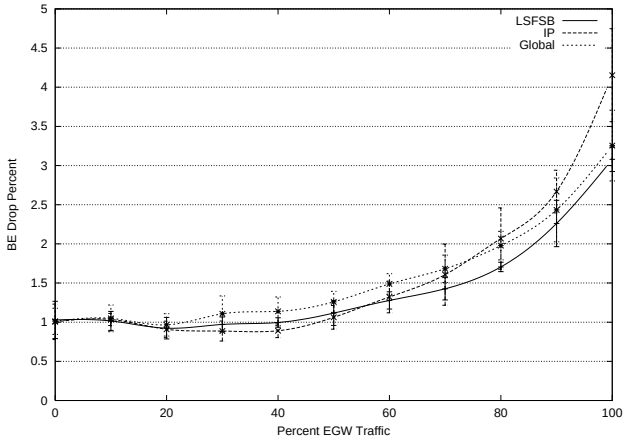

Figure 10: Linear Movement, 120 Connections

dropped, there are on average more slots available for BE traffic. If the EF traffic and the AF traffic are using all their scheduler slots, then just 10% of the bandwidth is granted to BE. This is due to the fact that we set the weighted round robin scheduler for the three DiffServ queues to use weights of 6 for EF, 3 for AF, and 1 for BE. Unused slots by one traffic class are yielded to the next lower one. The reason that IP performs well for BE traffic might thus be related to the fact that it does not perform well for the prioritized traffic classes.

For the reasons stated above, namely the interrelations between the performance of EF or AF traffic and the available BE bandwidth, we do not consider the BE performance as an important metric to evaluate the different routing algorithms.

Additionally, we visualize the link utilization in the network core. During the interpretation of many different simulation results it turned out that it would be helpful to have means to visualize the actual link usage data of the core links that is gathered by the simulator. The network core consists of four intermediate routers, I_1 to I_4, and the edge gateway (*EGW*). The traffic scenario used assumes 40 fixed nodes that connect through I_1 to the edge gateway or to corresponding nodes that are randomly distributed over the other intermediate routers (*Fixed* with no randomly moving sources). By using the fixed scenario we generate more static traffic demands so that it is easier to evaluate routing decisions by looking at link utilizations.

Each link is represented as a line between to nodes, and the thickness of the line corresponds to the link utilization. Since all links are bi-directional there are two thicknesses associated with each line. The part of a line next to a node represents the traffic going out of this node to the neighbor, while the thickness of the part of the line that is further away from the node is chosen accordingly to the amount of

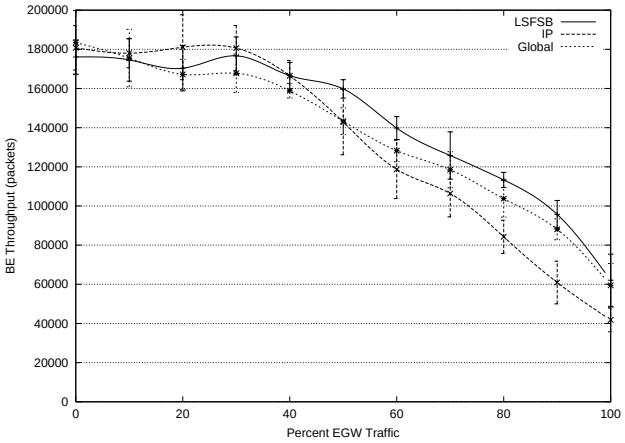

Figure 11: Linear Movement, 120 Connections

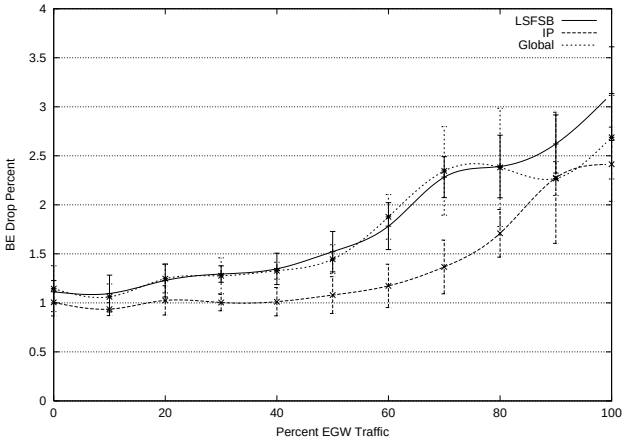

Figure 12: Linear Movement, 220 Connections

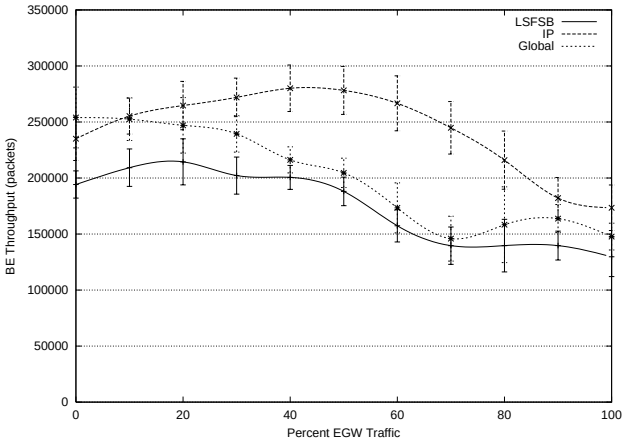

Figure 13: Linear Movement, 220 Connections

traffic the neighboring node sends back on that link.

Figure 14 shows the link utilization for IP routing if all nodes communicate to other mobile nodes in the network. Note that *mobile* does not refer to the movement of the node during the simulation, since all *mobile* nodes are

fixed in the movement scenario chosen. There is no big difference in the link utilization for the LSFSB or the Global WSP algorithm due to the fact that all corresponding nodes are evenly distributed.

If all traffic is edge gateway traffic, traffic engineering becomes interesting. Instead of having just one link to the edge gateway from I_1, there are also two alternate paths over I_2 and I_4. We show the link utilization for IP shortest path routing in Figure 15, for LSFSB in Figure 16, and for Global WSP in Figure 17 for the case of 100% edge gateway traffic. It is interesting to note that the overall throughput for LSFSB is bigger than the overall throughput of Global WSP in this scenario. However, the line thickness is not *directly* related to the throughput since also retransmissions and packets that are dropped later on are taken into account for the link usage calculation.

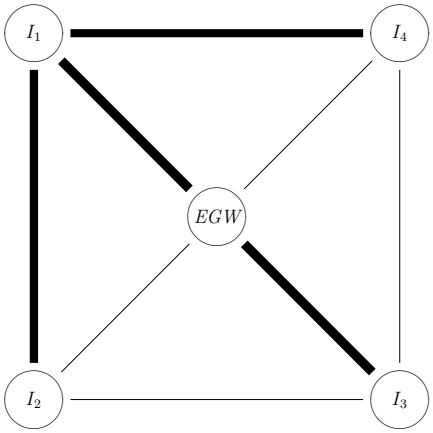

Figure 14: IP Shortest Path, 0% EGW Traffic

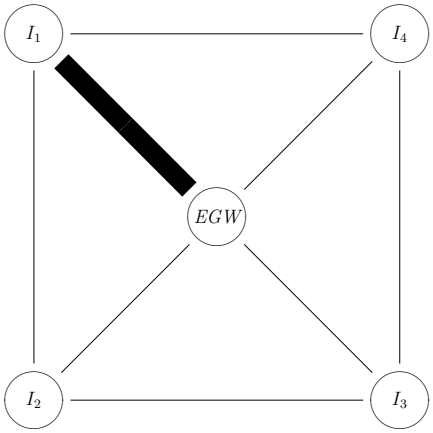

Figure 15: IP Shortest Path, 100% EGW Traffic

4. Conclusions

We saw that the LSFSB algorithm generally performs better than standard IP shortest path routing. In most situations it performs equally well or worse than our reference algorithm that incorporates global network state knowledge.

However, the RAN we assumed for our evaluations is rather small. In such a RAN it would not be a huge overhead to distribute network state data between the nodes. Thus, the main advantage of the LSFSB algorithm, namely that it does not rely on network state information from other nodes, is not as significant for this topology as it may be for other topologies. However, if the network gets bigger, we expect LSFSB to perform worse since the ratio of the information available to one single router compared to the overall information a globally working algorithm could use gets smaller and smaller. In other words, the portion of the network state known to a single router that keeps track of its local state decreases with an increasing number of routers in the RAN.

A direct improvement to the algorithm with regard to this fact is to do traffic-engineering decisions on the intermediate routers on the rim of the core network. These routers can incorporate more information, namely state information of their links into the core. The trade-off here is less scalability.

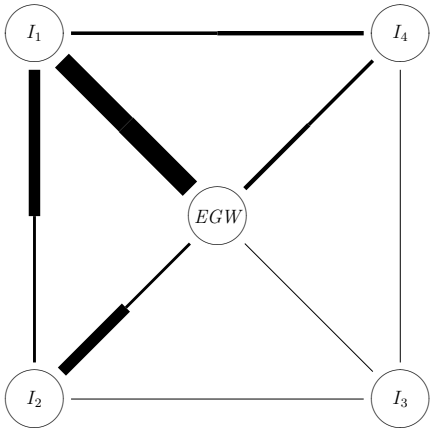

Figure 16: LSFSB, 100% EGW Traffic

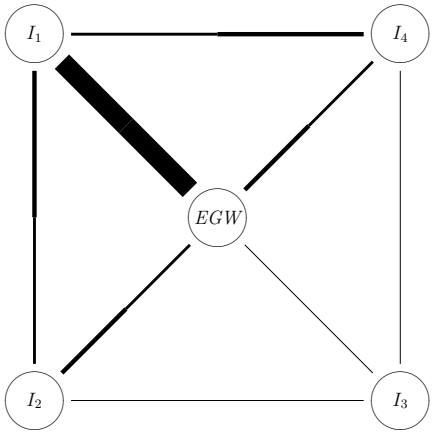

Figure 17: Global WSP, 100% EGW Traffic

However, local state information might not be sufficient in every situation. There are some situations where there is a hot spot in the network but a neighboring router does not know about this and starts sending more packets over LSPs that lead through this hot spot.

All this makes LSFSB a good algorithm – as opposed to IP shortest path routing – if one has to deal with two special scenarios. It can deal well with traffic bursts in a network that is evenly utilized because the bursts can be balanced easily over the network with a low risk to redirect them to a congested area, and it shows a good performance for a network with hot spots and low background traffic. The latter fact assures that there is not a huge amount of additional traffic routed through the hot spot region.

However, LSFSB is not the algorithm of choice for traffic-engineering to generally avoid hot spots in a network that does not fit to the two constraints given above. This is confirmed by the performance evaluation of the random movement scenario compared to the linear one with additional random sources. We saw that the algorithm performs well in the *Random* scenario and is able to distribute traffic requests on the distinct LSPs, even for a high number of mobile nodes (Figure 2). In the *Linear* scenario, which includes linearly moving nodes and background traffic modeled by randomly moving nodes, LSFSB performs worse than Global WSP if more random nodes are introduced (Figure 4, and Figure 7 compared to Figure 9). Therefore, LSFSB might be applied to the highest level in the network hierarchy in areas with homogeneous traffic demands or in areas with low traffic demands in order to improve overall service quality.

Nevertheless it is to mention that a final evaluation of LSFSB is complicated due to the huge amount of parameters that can vary. We saw that the simulator acts even very sensitive to changes in the random number generator seed, as implied by the large confidence intervals. And this is true for the actual simulation parameters as well. Additionally, some results are counter-intuitive and appear therefore rather confusing or contradictory on the first quick look, e.g. the good performance of IP shortest path routing (Figure 12 and Figure 13).

6. Future Work

Future research on traffic engineering algorithms for radio access networks with topologies similar to the one we assumed during this research should exploit more of the properties of such networks. Namely these are small size, symmetry, and simplicity. For example, probe packets could be sent to gather the state of an LSP like suggested in [7]. Measurements could be delay, jitter, and the packet loss rate. A bandwidth request can then be serviced by a path that has the best properties for the requested traffic class, e.g. low drop rate for AF or low delay for EF. All this intelligence should be implemented on the rim of the core. This reduces the amount of work the RASes have to do and makes them simpler and cheaper, which is desirable since our set-up assumes a lot of more RASes than there are intermediate routers in the RAN core.

This probe method has the advantage that it is much more scalable than an approach that distributes network state information to every router. The probe packets can be rather small so that they do not interfere too much with the actual traffic on the network. This algorithm combines the advantages of more global network knowledge with the simplicity of rendering decisions based on locally available metrics.

Additional ideas for such an algorithm are to dynamically change the weights of the DiffServ round robin scheduler, or to introduce *light-weighted* signaling between the routers, e.g. a request to not reroute any more traffic from a router to a neighboring router – to act rather than react if congestion is expected, for example in the case that an intermediate router has to deny local service requests due to a high load introduced by rerouted traffic of its neighbor.

References

[1] Barlow, D., “Router-Based Traffic-Engineering in MPLS/DiffServ/HMIP Radio Access Networks”, Proceedings of Wireless and Optical Communications Conference 2002, Banff, Canada, July 2002.

[2] Barlow, D., “Simulating Router-Based Traffic-Engineering in Radio Access Networks using ns-2”, accepted Communications, Internet, and Information Technology 2002, St. Thomas, November 2002.

[3] Barlow, D., “Router-Based Traffic Engineering in MPLS/DiffServ/HMIP Radio Access Networks”, Ph.D. Thesis, Georgia Institute of Technology, 2002

[4] The USB/LBNL Network Simulator – ns2, http://www.isi.edu/nsnam/ns.

[5] Ahn, G. and Chun, W., "Overview of MPLS Network Simulator: Design and Implementation," Department of Computer Engineering, Chungnam National University, Korea, 2000.

[6] Murphy, S., "The ns/MPLS/DiffServ patch," Dublin City University, Ireland, 2000

[7] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao, “Overview and Principles of Internet Traffic Engineering”, *IETF RFC 3272*, May 2002