Ring Flushing for Reduced Overload in Spanning Tree Protocol Controlled Ethernet Networks

Dániel Horváth, Gábor Kapitány, Sándor Plósz, István Moldován, and Csaba Lukovszki

Dept. of Telecommunications and Media Informatics
Budapest University of Technology and Economics
Budapest, Hungary
{horvathd,kapitanyg,plosz,moldovan,lukovszki}@tmit.bme.hu

Abstract. Flooding causes serious problems to the scalability of Ethernet networks. Recent proposals to overcome this problem, such as SEATTLE [5], usually require significant changes in different network layers, making the realistic chance of their deployment questionable. In this paper, we propose Ring Flushing, a practical method to reduce the burden of flooding during topology changes. The basic idea behind our approach is to locate stale forwarding information in an efficient way. Ring Flushing abolishes the broadcast-like spreading of topology change information thus shrinking the flushing domain. We implemented Ring Flushing in OMNeT++ simulation environment and evaluated its performance in different topologies and parameter settings. Our simulations show that the Ring Flushing has clear advantage over the approach of standard Rapid Spanning Tree Protocol (RSTP) in terms of throughput during network recovery. Furthermore, the Ring Flushing diminishes overall network overload during topology changes as the network size increases.

Keywords: ring flushing, flooding, spanning trees, rstp, Ethernet.

1 Introduction

The Ethernet became dominant technology in wired local area networks during the last two decades. This trend is supported by the development of new related standards. Ethernet has many advantages making it an appealing candidate in flat and homogeneously managed networks, such as Metropolitan Area Networks. The popularity of Ethernet technology resides in its low cost, ubiquity and easy management, without the need of error-prone manual configuration. A network topology is formed by connecting devices called bridges or switches.

The frame forwarding model in Ethernet did not change as the technology evolved; each bridge has a unique identifier called Media Access Control (MAC) address, used to address the device. The bridges have to learn the MAC addresses of the hosts and end devices in the network. Upon frame arrival a bridge stores the frame source address in conjunction with a timer and the port number the frame has arrived on, in a table called the Forwarding Database (FDB). Bridges rely only on

M. Oliver and S. Sallent (Eds.): EUNICE 2009, LNCS 5733, pp. 11–20, 2009.

the FDB at forwarding incoming frames. If the bridge has no information about the destination yet stored, it floods frames causing partially unnecessary traffic.

The purpose of the spanning tree protocols is to exclude redundant links from the active topology, which can otherwise form loops in the network. They construct one or more spanning trees in the network with a distributed algorithm. The elimination of loops is vital because the forwarding of frames is based on flooding and the lack of Time-to-Live field in the Ethernet header can cause the frames to remain in the network thereby congesting it. The Rapid Spanning Tree Protocol (RSTP) [1] is the most applied spanning tree protocol.

By the usage of a spanning tree protocol on Ethernet level the scope of this technology extends tremendously getting acceptance in more demanding networks, like backbone networks, where speed, reliability and easy manageability are primary requirements. The spanning tree protocols fulfill these requirements while keeping the benefits of the plain-old Ethernet technology. Drawbacks of such plain and well-distributed technology reveals while examining its capabilities in terms of scalability.

The classical Ethernet has several drawbacks. First of all, it does not forward traffic on the shortest path because of the use of a spanning tree (e.g. RSTP). This causes poor load balancing and inefficient resource usage. There are existing solutions for this problem, like Multiple Spanning Tree Protocol (MSTP) [2], Shortest Path Bridging [3] and Spanning Tree Alternate Routing (STAR) [4]. The other main problem with Ethernet is the scalability. The scalability of Ethernet is severely affected by the high number of broadcasts necessary for its operation. Kim et al. also considers scalability as the biggest problem of the Ethernet technology [5]. Proposals were made to eliminate spanning trees by applying link-state based algorithm instead of distance-vector algorithm, like CMU-Ethernet by Myers et al. [6].

To overcome the scalability shortage of Ethernet, service providers split the network to subnets by IP routers. By doing this efficiency increases because of the smaller broadcast domains and more optimal paths, on the other hand, the need for manual configuration arises. This approach does not solve the main disadvantage of Ethernet, simply avoids it. The flooding in Ethernet was also targeted by several papers in the recent literature. R. Perlman developed the Rbridges, a method of interconnecting links that combines the advantages of bridging and routing [7]. Moreover, the Scalable Ethernet Architecture for Large Enterprises (SEATTLE) network architecture by Kim et al. learns MAC addresses only on edge ports improving control plane scalability [5].

The main cause of flooding is the flushing of FDBs after a network failure. The solution for flooding applied in the original RSTP and MSTP standards is not optimal since it causes a large amount of unnecessary traffic on the network by flushing the FDBs of every bridge. This paper introduces Ring Flushing, which has a novel approach to flush ports containing stale entries only. In Ring Flushing the topology change information is propagated as unicast and not flooded. Moreover ports of a bridge are selectively flushed.

The paper is structured as follows. In Section 2, the RSTP is introduced, and its inefficiency problem is explained. In Section 3, we present our solution, the Ring Flushing Method. Simulation analysis and validation is shown in Section 4. Finally, in the Section 5, the paper is concluded.

2 Inefficient Address Removal of RSTP

The RSTP builds an overlay tree of the bridges in the network rooted at a bridge, called Root Bridge. The port roles calculations in the network are distributed. Each port providing the shortest path towards the Root Bridge is set to be Root Port. Each port of a bridge which provides the shortest path towards the attached LAN segment is elected to be a Designated Port. The roles of the rest ports are set to Alternate role.

The spanning tree protocols use link-local frames called Bridge Protocol Data Units (BPDU) to spread information to each bridge in the network.

Port state is assigned to each port besides port role on bridges. The Root Port and the Designated Ports are in forwarding state while the Alternate Ports are in discarding state in a stable topology. The forwarding ports transmit and receive all frames while discarding ports transmit and receive only BPDUs [1,9] thus pruning the network to a simple tree topology.

The most significant drawback of the spanning tree protocols is the inefficient address removal. The learnt addresses can be removed from the FDB via aging or flushing. Each entry in the FDB has an age attribute. An entry is removed on timeout unless its age is renewed by an ingress frame with the same source MAC address. This is called aging. Flush may be invoked on a bridge by RSTP after a change in the physical topology.

The current version of these protocols use a 1-bit flag called Topology Change (TC) bit to indicate the start of flushing. A BPDU with TC-bit set is called TC message. On reception of a BPDU the TC flag is polled. If it is set then all ports are flushed except the one which received the BPDU and those ports to where one end-device is attached only. The flush initiates a re-learning period. During the re-learning period all the devices forward packet by flooding therefore multiplying the traffic on the network. This is severely inefficient. The bigger the network the bigger the overload will be on certain links depending on the topology. These links can be overloaded, unable to transfer all data, resulting in packet losses. At present it is a major problem of Ethernet which diminishes its scalability.

3 Our Proposed Solution: The Ring Flushing Method

We start with a motivational example. In a topology which is redundant enough to remain connected after a topology change a ring can be identified where all the stale forwarding information reside. The idea is illustrated by an example topology in Fig. 1. The squares symbolize the bridges. The Root Bridge is marked with R. The cross indicates a failure of a link. The ports represented in grey indicate the old forwarding information to be stale. The out-dated forwarding information can be located easily. It resides in the ring defined by the recently failed link and the link which has become part of the active topology and the links connecting them in both directions. As a result, locating the stale information in an effective way would reduce unnecessarily flooding of messages. Any other possible end device attached to this topology would also have had stale forwarding information in this ring only. This ring always exists because the newly activated link has not been part of the previous topology.

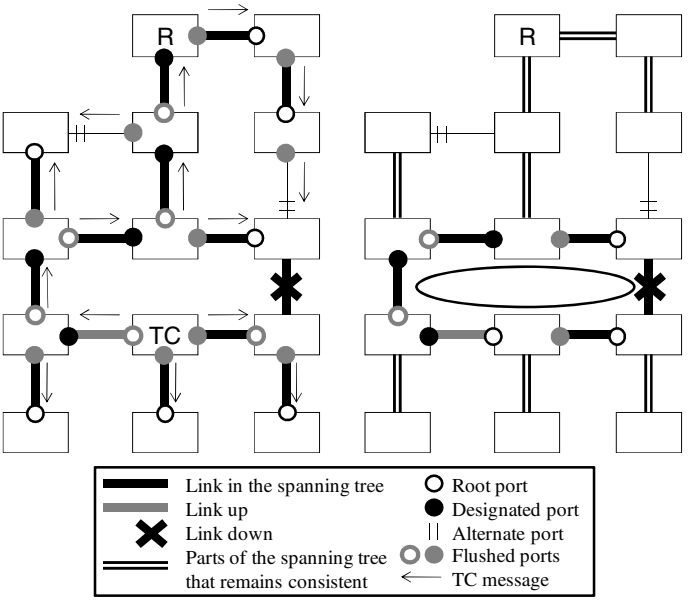

Fig. 1. Flushed ports by the standard flushing method are shown on the left figure. The ports that are needed to be flushed reside along a ring identified by the recently inactivated and activated links.

We state that bridges along the ring are affected by the change only. Indirectly, suppose a bridge which is not in the ring has at least one stale forwarding entry. Suppose also that there is no link state change outside the ring. There are two kinds of ports on bridges which are not part of the ring: ones which forward frames towards the broken link, and the ones which forward frames in the opposite direction. The forwarding entries in the port of the first kind can point to a destination whose path is changed. The path of the entries, whose destination's reachability is changed, traverses a part of the ring. The ring contains the old and the new path to that destination therefore the entries are valid. The forwarding entries of the port of the second kind are always point to a destination whose path has not changed. Therefore these entries are valid also. Then all entries are valid, which is a contradiction therefore implying that the considered bridge should have been the part of the ring..

Even in the ring not all entries in the FDB are needed to be deleted. The stale forwarding information is present in ports which are in the direction of the failed link in the original spanning tree. Therefore only this subset of the ports on the ring should be flushed, which can be seen on the right side of Fig. 1. The standard flushing method is presented on the left side. The number of ports unnecessarily affected by the standard ring flushing can be seen in this example network in Fig. 1.

3.1 How It Works

As illustrated above, the basic idea of our approach is the efficient removal of the entries. The current standard prescribes the removal of all entries with some

exceptions whose validity can be verified by rules. These rules are not fully comprehensive. The standard leaves the ports untouched by the flush, which receives the TC message and the ones to which only an end-station is attached. Our approach is to remove the stale entries only by isolating the affected part of the topology. We introduce the notation Branching Bridge for the closest bridge to the Root Bridge on the ring defined by the link-down and the link-up event. If the Root Bridge resides in the ring then the Root Bridge is the Branching Bridge. Three sectors of the ring can be distinguished. Each sector is an alternating series of bridges and links which starts and ends with a bridge {$bridge_a$, $link_{ab}$, $bridge_b$ … $bridge_m$, $link_{mn}$, $bridge_n$}. Every link in these series is part of the active topology. The sectors of the ring in the example and the relevant port roles in the final topology are shown in Fig. 2.

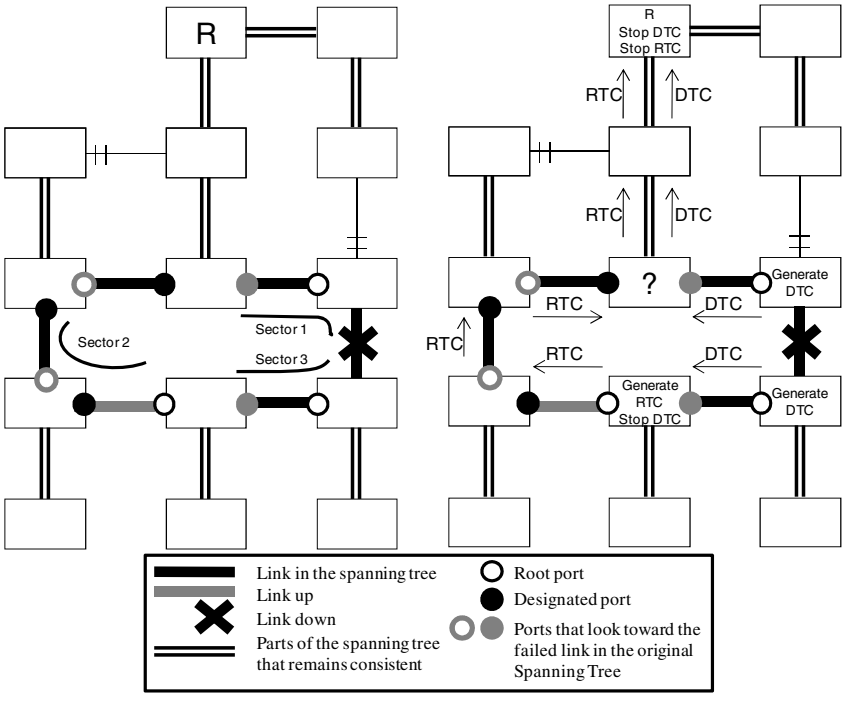

Fig. 2. The three sectors of the ring are shown in the left figure and the spreading of the RTC and DTC messages in the Ring Flushing implementation is presented in the right figure

The first sector is the path from the bridge having a link which is recently inactivated, to the Branching Bridge. It does not contain the newly enabled link. In this sector only Designated Ports should be flushed. The second sector starts at a bridge which has a forwarding port that has been discarding before the topology change and ends at a bridge which has the Branching Bridge as its Designated Bridge. In this sector, only Root Ports should be flushed. At last, the third sector is between the recently inactivated and activated links. In this sector, only Designated Ports should be flushed.

A sector of a ring may contain one bridge only. Nor the second nor the third sector contains the Branching Bridge. The second and the third sectors may overlap in their first and last bridge, respectively. The newly inactivated and activated ports do not need explicit flushing. The newly inactivated ones are flushed anyway at inactivation time, and the newly activated ones do not contain any entries.

3.2 Practical Issues on Implementation

Port table flushing is needed only on the ring. Additional control messages are required in order to flush the appropriate port tables. We define Root Topology Change (RTC) and Designated Topology Change (DTC) messages. These are handled the following way. A bridge receiving either an RTC or a DTC propagates it on its Root Port. If the received message is a DTC then the port table of the receiver port is flushed. The role of the receiver port is always Designated because the sender port is always a Root Port or an Alternate Port. Further on, if the message is an RTC message then the port table of the Root Port is flushed.

With the help of the RTC and DTC messages, and the definitions of the sectors, one can easily see, where to generate which kind of messages, and where to terminate them. Fig. 2 shows that one RTC message should be generated at the link-up event, and should be propagated until the penultimate bridge before the Branching Bridge. Two DTC message should be generated at both side of the inactivated link. Either should be propagated until the Branching Bridge and the other should be propagated until the end of the third sector.

One DTC message is needed to traverse until the Branching Bridge, one RTC message should be terminated at a neighbor of the Branching Bridge. However, the Branching Bridge cannot be located in topology change time, where a DTC should be terminated. Nor its neighbor, which is the endpoint of the second sector, can be located, where the RTC should be terminated. A bridge could only find out of being the Branching Bridge if received both an RTC and a DTC message from different directions. It is unlikely that these messages met in the Branching Bridge. Without the demand to make a bridge wait for the other message this recognition is not possible. Applying timer however would only slow down the transmission in both directions thereby increasing the delay without any gain. The flushing domain is therefore extended to the path between the Branching Bridge and the Root Bridge. The Root Bridge does not have Root Port therefore these messages are terminated there. The RTC and DTC messages traversing to the Root Bridge are presented on Fig. 2.

The flushing domain is a set of bridges which have to flush at least one port table. The flushing domain of the RSTP is by definition the whole topology except when the topology got partitioned during the topology change. The flushing domain of Ring Flushing is the ring and the path between the Branching Bridge and the Root Bridge.

The advanced removal is enabled by the existing 2-bit TC flag of the Bridge Protocol Data Unit (BPDU) headers. We use the TCAck flag of the BPDU header as well which is unused by RSTP. One of the 2-bit flag indicates RTC and the other indicates DTC. Setting both bits is possible which makes the receiver bridge behave like it received an RTC and DTC message independently. No additional fields required in the BPDU header but kept for compatibility with STP, therefore the size of the BPDUs remains the same.

RTC and DTC messages are triggered by network events. Transitions between RSTP port states initiate the RTC and DTC message by means of the following rules. A bridge that has a Discarding Port changing to Forwarding transmits an RTC message on its (new) Root Port. A bridge losing its Forwarding Port transmits a DTC message on its (new) Root Port.

The proposal-agreement mechanism temporarily sets some forwarding ports to discarding and later back to forwarding. This causes a lot of unnecessary DTC and RTC messages unless we detect and bypass these in the Ring Flushing algorithm.

We were intent on minimizing the modification needed on the implementation advocated by the standard thereby contributing to the easy integration of our method into existing architectures.

4 Performance Evaluation

In order to be able to measure the performance of RSTP protocol, we implemented it in a simulation environment following the standards. Measurements of properties of real devices like PDU construction and processing times were added to authenticate our simulator. To verify its proper operation several scenarios were worked out to deploy the simulator against real networks. These protocols are deterministic besides some exceptional transient behavior hence allowing mathematical verification in simple scenarios. Ring Flushing method has been added in the RSTP implementation followed by verification. The simulations are implemented in the OMNeT++ environment [8].

We implemented an MSTP module and used existing modules slightly modified to create an Ethernet based bridge. Interconnecting these bridges and a number of hosts we could easily measure differences between the standard and our proposal. The simulation scenario presented in this paper uses the following properties.

- Traffic:
  - Packet size is 1500 byte (including all headers)
  - Packet transition frequency is 0.01s
  - Each Host communicate with one other host
- Delays:
  - Each bridge has a BPDU construction time which is uniformly distributed between 0.1ms and 1ms
  - Each bridge has a BPDU processing time which is uniformly distributed between 1ms and 10ms
- Other RSTP specific settings are set to default according to 802.1D

An example network we used during investigation consists of a main circle (link 1, 2, 15, 19) and sub trees that connect to this circle. The left side of Fig. 3. shows the spanning tree (thick line), the link failure (cross on a line), the new link (grey thick line), as well as ports flushed. The standard flushes ports marked with black or grey. Flushing of all these ports is not necessary. The black ports are flushed unnecessarily. The Ring Flushing removes entries only in the ports marked with grey. In this scenario, each bridge has one host attached that sends traffic into the network.

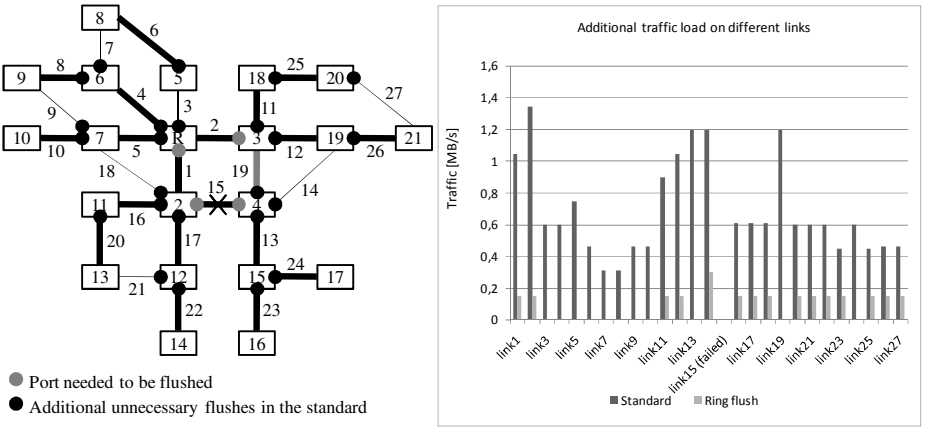

Fig. 3. An example network can be seen on the left. On the right the overhead caused by the flooded traffic is shown during re-learning which is smaller when Ring Flushing is used.

The right side of Fig. 3. shows the overhead of the traffic caused by flooding in each link after the failure. It can be seen that the less port flushed the less the traffic overhead will occur during the relearning period. In this example, most of the traffic avoids the main circle, which gives reason for the big difference of the amount of flooded traffic during the relearning period.

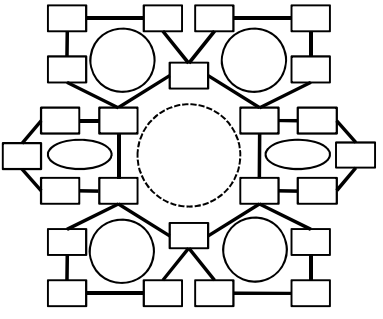

Fig. 4. An example topology of inner circle of 6 bridges and outer circles of 5 bridges. This means an additional 3 bridges per outer circles, and a total of $6 + 3 \times 6 = 24$.

Additionally a topology was investigated which had an inner circle, shown as a dashed circle in Fig. 4., and several outer circles, indicated with solid circles, connected to the inner circle. Simulations were made with different number of bridges. Each bridge has one host that sent traffic to the Root Bridge which is located in the inner circle. Each host transmitted traffic at a constant rate. We measured the peak value of the traffic load on each link after the failure. The network traffic is the sum of the link loads at a given time instant. We compared the peak value of the network load to the constant value of the network load after the relearning period.

Table 1. The simulation results shows diminishing of overhead during broadcast storm

Traffic load comparison: inner-outer ring topology				
Number of bridges			*Additional traffic load on network*	
In the inner circle	*Per outer circle*	*total*	*Standard flushing*	*Ring flushing*
3	1	6	64%	43%
3	2	9	48%	16%
4	3	16	50%	31%
5	4	25	60%	13%
6	5	36	50%	11%
7	6	49	55%	4%
8	7	64	67%	9%
9	8	81	69%	1%
10	9	100	67%	1%

For example when the traffic load on the network is 320Mbps before link failure, 600Mbps the peak value, and 400Mbps after the relearning then the additional traffic load on the network is (600-400) / 400 × 100%= 50%. The mean values of the additional traffic load of 30 simulations are presented in Table 1. As can be seen, using the Ring Flushing method the traffic overload caused by flooding diminishes in each scenario. Increasing the number of bridges in the network the standard solution greatly overloads the network, while Ring Flushing decreases the overload.

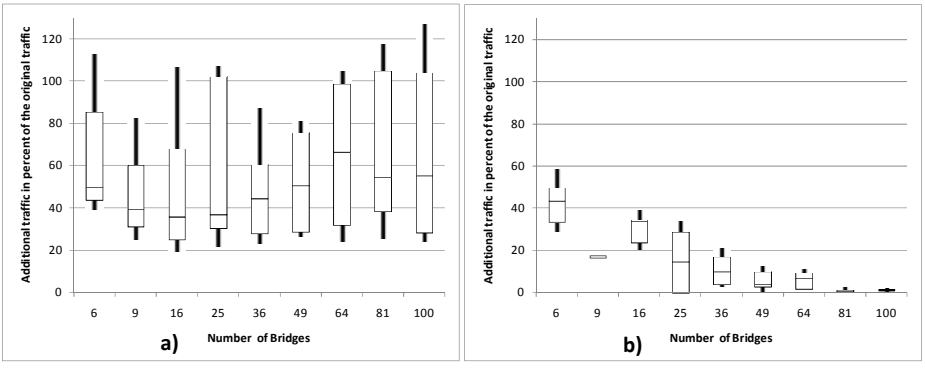

Fig. 5. The simulation results for 30 runs with different random delays. a) shows the maximum traffic load on the network during the relearning period using the standard method. b) shows the maximum traffic load on the network during the relearning period using the Ring Flushing method.

The results of simulations can be compared in Fig. 5. Thirty simulations were made for the standard method (shown on the left side), and for the ring flushing method (shown on the right side) for each size of the network. The box-plots show the 10 percentiles, first quartiles, the medians, the third quartiles and the 90 percentiles of the maximum traffic overload during relearning period. The median values of the standard method (50%, 39%, 35%, 37%, 44%, 50%, 66%, 54% and 55%) do not have obvious trend and are approximately constant. The values for the Ring Flushing

method (43%, 16%, 33%, 11%, 10%, 1%, 6%, 1% and 1%) show a downtrend with the increasing network size, therefore the burden of the traffic overload diminishes.

To summarize, the standard method operates with the same efficiency usually generating flood of more than 50% overhead. Meanwhile, the efficiency of Ring Flushing method is increasing with the increasing size of the network because the flushing domain to whole network ratio diminishes causing proportionally less FDB entry removal. In larger networks more link will be unaffected by the relearning in the ring flush method, while in the standard, always all links are affected.

5 Conclusion

We presented Ring Flushing, a practical method for reduced overload in spanning tree controlled Ethernet networks. By keeping all the benefits of Ethernet while dealing locally with events previously considered to be network-wide, our proposal flushes fewer ports in most cases but still slightly more than necessary. We implemented the algorithm into OMNeT++ simulation environment to investigate the performance of our Ring Flushing algorithm. The presented results showed significant improvement over the approach of standard RSTP in different network topologies and parameter settings.

Acknowledgement

We would like to express gratitude to Ericsson Hungary Ltd. for supporting our work. Moreover, we would like to thank to the reviewers for their useful advices and comments which helped us to improve this paper.

References

1. IEEE Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Bridges, 802.1D (2004)
2. IEEE Standard for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks, 802.1Q (2005)
3. Virtual Bridged Local Area Networks — Amendment 9: Shortest Path Bridging, IEEE draft standard 802.1aq-D0.3 (2006)
4. Lui, K., Lee, W., Nahrstedt, N.: STAR: A Transparent Spanning Tree Bridge Protocol with Alternate Routing. ACM Sigcomm Computer Communications Review (2002)
5. Kim, Ch., Ceasar, M., Rexford, J.: Floodless in SEATTLE: A Scalable Ethernet Architecture for Large Enterprises. ACM SIGCOMM Computer Communication Review (2008)
6. Myers, A., Ng, T.S.E., Zhang, H.: Rethinking the Service Model: Scaling Ethernet to a Million Nodes. In: Proceedings of HotNets III (2004)
7. Perlman, R.: Rbridges: Transparent Routing. In: INFOCOM (2004)
8. OMNeT++ Community Site, `http://www.omnetpp.org`
9. Pallos, R., Farkas, J., Moldovan, I., Lukovszki, C.: Performance of rapid spanning tree protocol in access and metro networks. In: Access Networks & Workshops (2007)