An Approach for Per-Flow Bandwidth Distribution in Routers (BDR)

Paulo Loureiro*, Saverio Mascolo**, Edmundo Monteiro ***
* Polytechnic Institute of Leiria
Leiria, Portugal
e-mail: loureiro@estg.ipleiria.pt

** Politecnico di Bari
Bari, Italy
e-mail: saverio.mascolo@gmail.com

*** University of Coimbra
Coimbra, Portugal
e-mail: edmundo@dei.uc.pt

Abstract **— In this paper we propose Per-flow Bandwidth Distribution in Routers (BDR). This is an approach which enables an efficient use of the available bandwidth in high or low speed networks. BDR gives routers capacities to detect the number of active flows that are crossing the routers' output interface. This is done without needing to maintain any per-flow state information. In the paper we describe BDR and discuss its evaluation results based on ns-2 simulations. The results show that BDR has capacities to discover the number of active flows and has capacities to distribute the resources between all flows.**

1. Introdution

In a recent publication [1] the author proposes a set of metrics to evaluate congestion control mechanisms. Among all the metrics proposed we chose two that point out some weaknesses of the congestion control mechanisms of the Transmission Control Protocol (TCP): aggressiveness and fairness. Aggressiveness is defined as "maximum increase in the send rate in one round-trip time, in packets per second, in the absence of congestion". Fairness "Fairness can be considered between flows of the same protocol and between flows using different protocols (e.g., fairness between TCP and a new protocol)".
In another publication [2], the authors have made several tests to see the competence of TCP Additive Increase and Multiplicative Decrease (AIMD) [3] congestion control algorithm. These tests were made in presence of a 1Gbps trans-atlantic path between Dublin (Ireland) and Chicago (United States), using just 1 flow and the propagation delay was 100 ms. The tests' results showed interesting behavior. For example, TCP needed 1200 seconds to recover, after a backoff, and the average throughput achieved, in that period, was only 218 Mbps of the 1 Gbps available. Taking in consideration these results, it is evident that TCP has difficulty to use network capacity in links that offer a high bandwidth. Despite the fact that these tests only used 1 flow is obvious the lack of aggressiveness in links with large bandwidth.
It is obvious that TCP is not prepared to be aggressive or to use the network capacity when it is available. This is critical because the Internet infrastructure is changing and it is common to have links with Gbps, such as links with unpredictable characteristics, for example, wireless networks in which link failures, frequent bandwidth changes, packet loss due to wireless channels or asymmetry paths between data and acknowledgment packets are common to occur.
Fair is defined [4] as "equal sharing of the resources". Thus for a single link, equal sharing of the resources means that all flows should receive an equal throughput. A scheduling algorithm that always gives all flows equal throughput is denoted as throughput-fair. Furthermore TCP doesn't have the capacity to provide an effective fairness because it ignores the congestion state of networks such as the flows that are competing to available bandwidth. In [4], authors arrive to the conclusion that with the exception of H-TCP [5], every proposal studied produced significantly greater round-trip times and unfairness between competing flows with diverse round-trip times.
Since the beginning of the internet that TCP has had an important role to control the congestion. The solution adopted until now was based on tuning previous TCP versions to adapt. These adjustments have been based on four algorithms: Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms [6]. From the results shown in previous paragraphs two conclusions may be drawn: first, TCP has difficulty to efficiently use the network resources; second, the four TCP mechanisms mentioned above are unable to adapt to actual network conditions.

A relevant point about TCP is that it only works at end-systems. TCP sees network as a black box and only receives feedbacks about packet loss. If a packet is lost TCP receives the same feedback, in spite of the characteristics of network. For example, TCP should not have the same response from a network with few nodes, small delays and reduced bandwidth or a long distance network, with high delay and a large bandwidth. Characteristics of the two networks are substantially different and TCP should produce a different response. If TCP only receives packet loss feedback then the capacity of making the best decisions is limited. It is clear that TCP must be helped from other network components. This increases its capacities to make good decisions about the network use in a large and different network environment.

In this paper we introduce an approach that makes way for the efficient use of network resources as well as guarantees fairness between flows. The proposed approach, which we call BDR, gives routers important information. The number of flows that are using the output interface can be used by routers to distribute the available bandwidth. With this mechanism routers can effectively control the traffic that is sent to networks. The sender is only informed by routers about the bandwidth that this application can use. Note that this mechanism does not need to maintain any per-flow state information. What makes this mechanism different from other proposals, that use explicit congestion control is that BDR guarantees an effective use of the available bandwidth, fairness between flows, scalability and all this is done without needing to maintain any per-flow state information and with minimum additional CPU processing.

The remainder of this paper is organized as follows. In section 2 we provide related work on congestion control mechanisms based in the cooperation among end-systems and intermediate nodes. Section 3 describes characteristics and the design to the proposed scheme. Results of the evaluation using ns-2 [7] simulations are presented in Section 4. Section 5 presents the conclusions and some directions for future work.

2. Background and Related Work

Over the past few years, several solutions have been proposed to give TCP better and more network feedback, beyond packets loss information and RTT variations. In addition, the research community has been specifying alternative solutions to the TCP architecture. As the BDR, some of these models are classified in category of "modification of the network infrastructure". They are briefly explained as follows.

The ECN [8], [9] and the Quick-Start [10] are two solutions that use the router collaboration to address the congestion control problem. With the ECN, the router collaboration is done by detecting congestion situations and by informing the end systems about this situation. With the Quick-Start, the collaboration of routers is used to decide the value of the initial congestion window. Along the path routers accept or not an initial congestion window proposed by the end system. These two mechanisms are nevertheless used by the traditional congestion control mechanisms. This means that the algorithms slow start, congestion avoidance, fast retransmission and fast recovery [11] are still used and therefore the problems associated to theses mechanisms remain.

Explicit Control Protocol (XCP) [12], [13] is designed to work well in networks with large bandwidth-delay products. This Internet congestion control protocol outperforms TCP in conventional environments and remains efficient, fair, scalable, and stable in high bandwidth-delay product networks. The XCP generalizes the Explicit Congestion Notification proposal (ECN). In addition, the XCP introduces the new concept of decoupling utilization control from fairness control. Routers provide feedback, in terms of incremental window changes, to the sources in multiple round-trip times, which works well when all flows are long-lived. However, in a dynamic network' environment, the XCP can increase the duration of each flow beyond the ideal and can contribute to maintain more active flows in the network [14].

3. Per-Flow Bandwidth Distribution in Routers

Per-Flow Bandwidth Distribution by Routers (BDR) is based on three main concepts: 1) intermediate nodes know the number of flows that cross each output interface; 2) intermediate nodes can interact with each flow created at any end-system; 3) intermediated nodes can distribute the output bandwidth of any interface between all flows that are crossing the interface.

As explained below, these concepts allow routers to provide end systems with important congestion control information. Another relevant point is that routers do not need to maintain any per-flow state information.

This mechanism relies on changes in TCP behavior as well as in intermediate nodes. First along the router in the path BDR identifies the number of flows in each output interface. Then it evaluates the available bandwidth and makes decisions regarding bandwidth distribution per-flow. If a router detects an increase in the number of flows at any output interface then router requests all the flows to adjust their transmission rate. Also, if there is a large available bandwidth and the number of flows is reduced, routers can suggest a quick increase of transmission rate to flows.

The information provided by routers is inserted in a special packet which is sent to the receiver. That can then send it back to the sender end-system. At this point TCP should accept the recommendation provided and adjusts its congestion window. Mark that in a path with multiples routers only the bottleneck router recommendation will arrive at TCP sender. The information used by BDR mechanism is exchanged between routers and end-systems. This information is put inside a special packet, which transports data like all others, but has new skills. The packet used to transport the information is named designated packet and, there is only one packet in transit per flow. Fig. 1 presents the architecture of BDR.

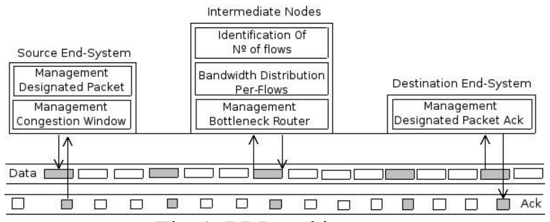
Intermediate Nodes
Identification Of Nº of flows
Bandwidth Distribution Per-Flows
Management Bottleneck Router
Source End-System
Management Designated Packet
Management Congestion Window
Destination End-System
Management Designated Packet Ack
Data
Ack

Fig. 1. BDR architecture

A. Source End-System

Source end-system has tow main responsibilities: to manage the chosen designated packet and create all the information about a particular flow, which is necessary so that routers may discover the number of flows that cross an output interface; to accept and implement the recommended bandwidth for each flow.

The designated packet is a packet that is chosen by the end-system source from all packets belonging to the same TCP congestion window. It transports until intermediate nodes all information necessary so that routers can make their decisions. This solution enables routers to have information about flows without needing to keep it.

Another important issue is that routers do not need to process all packets from every flow to extract flow information. Only a reduced number of packets need to be analyzed. At the router the CPU processing is minimized. Furthermore, there is a guaranty that senders don't receive duplicated information because, per flow, there is only one designated packet travelling at a time. In fact, the designated packet is not send before the packet that acknowledges the previous one has been received.

Once more, this designated packet can be at any router belonging to the path. This solution dilutes the additional processing by all routers.

In the BDR mechanism the main task of routers is to discover the number of flows that cross any output interface. To make this decision, routers must receive from end-systems the size of the last congestion window (*size_cwin*) and the time needed to send all packets within the last congestion window and receive the corresponding acknowledgments (*time_per_cwin*). This information is provided to routers inside the designated packet. It is important to take into account that these variables are put inside the IP header.

The other task of the source end-system is to accept the recommendations received from intermediate nodes. These nodes suggest the bandwidth that should be used by the flow. These recommendations are received in the packet that acknowledges the designated packet. Source end-system converts the bandwidth to congestion window size and implements these recommendations.

Note that the TCP sender does not execute the traditional TCP algorithms: slow-start and congestion avoidance. The sender TCP just implements the recommendations from intermediate nodes. Moreover, the detection of packet loss is not the key to find the adequate transmission rate. Packet loss is only used to reduce immediately the transmission rate, but if intermediate nodes continue to suggest a high congestion window, source end-system must implement the recommendation. In this mechanism the transmission rate is based on the available bandwidth that is distributed by intermediate nodes.

B. Intermediate nodes

The BDR mechanism is responsible for managing and distributing the bandwidth by all flows that are in any output interface. The distribution is done without needing to maintain per-flow state information. Routers only receive from end-system sender two pieces of information from each flow: the size of the congestion window; and the time needed to send successfully all packets belonging to that congestion window.

Bearing this information, routers must perform two tasks: 1) discover the number of flows that are using a specific output interface; 2) distribute the available bandwidth among all the flows.

To discover the total of flows that are in the output interface, routers must have two variables per output interface, *router_time* and *total_time_flows*.

The variable *router_time* represents a time space that corresponds to the router's life time. *total_time_flows* is calculated by the sum of all *time_per_cwin*, received from designated packets. So, when a new designated packet arrives routers must read the content of *time_per_cwin* variable and sum it to *total_time_flows*. At any time the variable *total_time_flows* contains the sum of all *time_per_cwin* received since the beginning of the process.

For each output interface, router has the variables *router_time* and *total_time_flows*. With this information it is possible to discover the number of flows that are crossing the interface. This is done by expression (1).

$$\text{number\_flows} = \text{total\_time\_flows} / \text{router\_time} \qquad (1)$$

The next step consists of calculating the bandwidth used by each flow. *bw_per_flow* is the variable used and it is calculated by expression (2).

$$\text{bw\_per\_flow} = (8 * \text{packet\_size} * \text{size\_cwin}) / \text{time\_per\_cwin} \qquad (2)$$

The bandwidth depends on the packet size (*packet_size*) in bytes, the number of packets per congestion window (*size_cwin*) and the time needed to send all packets and receive the acknowledgements of the congestion window (*time_per_cwin*). To obtain *bw_per_flow* in bps we must multiply it by 8.

At this point, routers already know the number of flows that are crossing the output interface and the actual bandwidth used by each flow. Thus the next step of this process is to decide the bandwidth recommendation for a specific flow. Routers know the bandwidth used by each flow; they also know which are the used bandwidth and the available bandwidth in the output interface. In hold of this information it is possible to estimate, per-flow, a gain that corresponds to the increase or decrease of the actual transmission rate. This gain is calculated by expression (3). It depends on available bandwidth and is expressed by number of packets. This means that the flow must increase or reduce its congestion window in gain number of packets.

gain=(bw_available/number_flows)/(bw_per_flow/size_cwin) (3)

The gain produced for a specific flow is only put in the designated packet if this is the first router in the path or if this gain is less than others suggested by previous routers in the path. This is done by module management bottleneck router.
Finally routers insert the recommendation in the designated packet and resend it to the next hop in the path.

C. Destination End-system

When a packet arrives at destination, TCP creates another packet to acknowledge the first. The same process is done when a designated packet arrives. Additionally, inside the acknowledgement packet is the recommendation received from routers.
This packet will arrive at end-system sender and the congestion window will be adjusted to implement the recommendation.

4. Evaluation

To evaluate the BDR mechanism, we have created simulations on the ns-2 simulator [7]. Fig. 2 shows the network topology used in the simulations (this topology is known as dumbbell network). The bottleneck link bandwidth is set to 100 Mbps or 1 Gbps. The links that connect the senders and the receivers to the routers have a bandwidth of 2.4 Gbps. The round-trip time is set to 100 ms. Routers have the queue size set to 5000 packets, which is near to one third of the delay-bandwidth product of the bottleneck link, in the case of bottleneck link being set to 1 Gbps. The BDR mechanism is implemented by modifying the TCP Reno agent [7], TcpSink agent [7] and the DropTail queue [7]. The traffic used in the simulation is Poison flow arrival and flows size are Pareto distribution with the mean equal to 12.5 packets (1000 bytes / packet) and the shape equal to 1.8. Either 2000 new flows are created per second, when the bottleneck link is 1 Gbps, or 400 new flows are created per second when the bottleneck link is 100 Mbps. The RTT is equal to 100 ms. We have done tests in scenarios with more routers but the results were the same presented in this publication, because in BDR mechanism only the bottleneck router controls the transmission rate of a flow. So the number of routers in the path does not influence the results of the BDR mechanism.

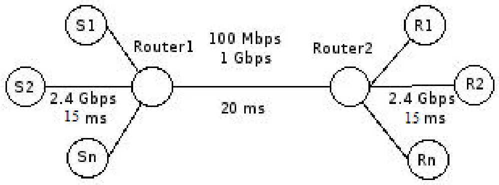

Fig. 2. Simulation topology

A. Discovery of the number of active flows

The BDR mechanism is supported in the capacity of routers to identify the number of flows that are crossing any output interface. Using this information, routers can distribute its output interface bandwidth by all flows. This distribution can be done quickly and efficiently.
Fig. 3 and Fig. 4 show the number of flows discovered by BDR mechanism. As can be seen in Fig. 3 and Fig. 4 the exact number of flows that are crossing the output interface, in the bottleneck link, was discovered and is always near the exact number of active flows (Active flows line) present in the test scenario. These results show the capacity of BDR mechanism to identify the number of active flows. The efficiency of BDR mechanism is based on this capacity.

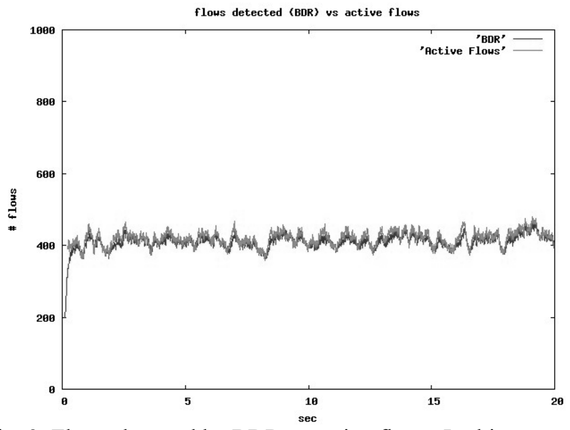

Fig. 3. Flows detected by BDR vs active flows. In this scenario the bottleneck link is 1 Gbps

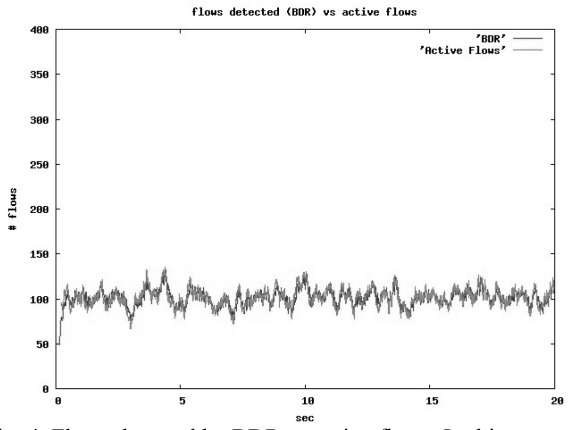

Fig. 4. Flows detected by BDR vs active flows. In this scenario the bottleneck link is 100 Mbps

B. Flow completion time

In the BDR mechanism the congestion control is a responsibility of routers. Moreover, these elements know the output interface bandwidth and they make a fair bandwidth distribution for all flows. Additionally routers never make a recommendation that overflows the output interface. Fig. 5 shows the flow average completion time of one tests when the bottleneck link is configured to 1 Gbps. In these scenarios the delay is 100 ms.
It is visible, from Fig. 5, that short flows (until 30 packets, in this scenario) were completed in 2 RTTs. At bottleneck router, this model identified the number of active flows and distributed the available bandwidth by these flows. In this case each flow received at least bandwidth to send 30 packets per RTT. Flows larger than 30 packets needed

more RTTs to be completed. In this case, the number 30 packets per RTT was obtained by router and corresponds to the bandwidth distributed to each flow. This bandwidth is a function of the number of active flows, capacity of output interface and RTT.
Another important conclusion is that all flows always have a life time proportional to their sizes. This means that all active flows receive an equal share of output interface capacity, and the BDR model implements fairness among flows.

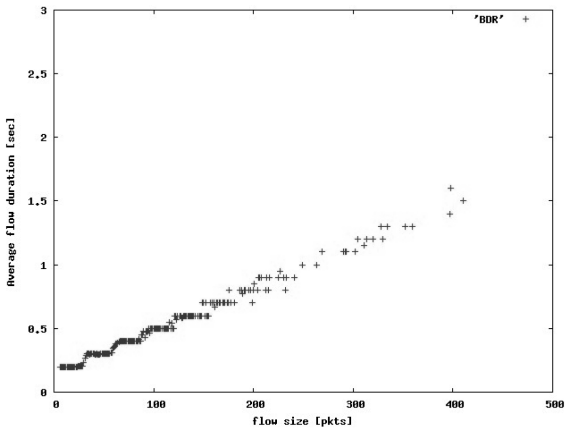

Fig.5. Average completion time. In this scenario the bottleneck link is configured to 1 Gbps

C. *Comparison between BDR, XCP and TCP Reno*

The BDR mechanism can be used in highspeed networks as well as in networks with speeds of few megabits per second. This happens because routers know the output interface capacity and distributes it by all flows in some round-trip times.
On the other hand, TCP is a protocol that does not use efficiently the available bandwidth. This is critical in highspeed links where TCP needs many round-trip times to fill the channel. Also we have done tests using XCP protocol.
This evaluation consists of 3 tests using bottleneck link of 1 Gbps and the RTT is equal to 100 ms. Flows arrival are Poison distribution (2000 new flows per second) and flows size are Pareto distribution with the mean equal to 12.5 packets (1000 bytes per packet) and the shape equal to 1.8.
We can see from Fig. 6, BDR mechanism has good results to complete short flows. Long flows need more time to be completed than TCP or XCP. This happens because BDR distributes the interface capacity equality by all active flows, so the average completion time is proportional to flows size. On the other hand, for example, TCP is a protocol that allows congestion window size to increase by 1 per each acknowledge packet received (slow start phase). Longs flows can reach high congestion window sizes. With long flows, TCP and XCP show better results than BDR and BDR shows better results with short flows.

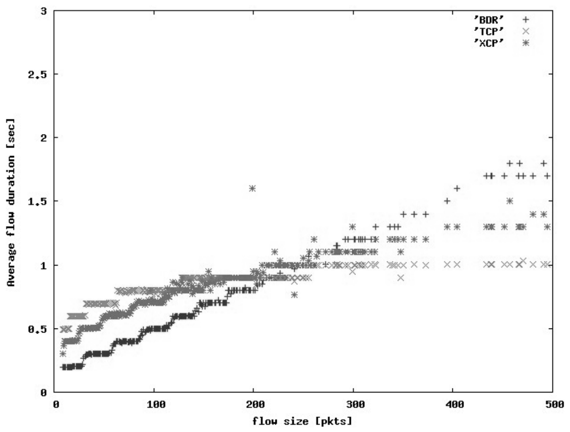

Fig.6. Average flow completion time of BDR, TCP and XCP

5. Conclusion and future work

In this paper we propose TCP Per-Flow Bandwidth Distribution in Routers (BDR). This is a new approach to enable an efficient use of the available bandwidth by all flows in high or low speed networks. BDR adds to routers the capacities to detect the number of active flows that are crossing the output interface. This is done without needing to maintain any per-flow state information. With this information BDR can make a fair distribution of the available bandwidth between all flows. Equally BDR can use all available bandwidth in a few round-trip times. In this mechanism TCP merely receives recommendations about the size of congestion window and implements these recommendations.
We have shown through analysis and experimental evaluation that BDR has capacities to discover the number of flows that are using any output interface, in static or dynamic network scenarios. Also, it is visible that BDR is scalable and uses efficiently the available bandwidth in networks with high or low speed links as well as provides fairness between flows.
Currently we are in the process of evaluating BDR with other schemes developed to be used in high speed networks as well as with protocols that are classified in "Modifications to the network infrastructure" approach, like XCP.
As part of our future work, we plan to investigate the possibility of each application informing routers about its satisfaction level with the recommended transmission rate received by routers.

References

[1] Floyd, S. "Metrics for the Evaluation of Congestion Control Mechanisms". *Internet Draft, IETF*, August 2005.

[2] Li, Y., Leith, D., Shorten, R. "Experimental Evaluation of TCP Protocols for High-Speed Networks". Technical Report, Hamilton Institute, NUI Maynooth, June 2005.

[3] Chiu D., Jain R. "Analysis of the Increase / Decrease Algorithms for Congestion Avoidance in Computer Networks". *Journal of Computer Networks and ISDN*, Volume 17, Number 1, June 1989.

[4] Norlund, K., Ottosson, T., Brunstrom, A.” TCP fairness measures for scheduling algorithms in wireless networks”. *Second International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QShine)*, Orlando Florida USA, August 2005.

[5] Leith, D., Shorten, R., Li, Y. “H-TCP: A framework for congestion control in high-speed and long-distance networks”. *Technical Report, Hamilton Institute*, August 2005.

[6] Stevens, W. “TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms”. IETF, RFC 2001.

[7] Ns-2 network simulator (version 2). LBL, URL: http://www.isi.edu/nsnam/ns/.

[8] J. Hadi Salim, U. Ahmed. s.l. “Performance Evaluation of Explicit Congestion Notification (ECN) in IP Networks”: RFC 2884, 2000.

[9] K. Ramakrishnan, S. Floyd, D. Black. s.l.”The Addition of Explicit Congestion Notification (ECN) to IP”. RFC 3168, 2001.

[10] S. Floyd, M. Allman, A. Jain, P. Sarolahti. s.l.”Quick-Start for TCP and IP”. RFC 4782, 2007.

[11] . Stevens, W. s.l.”TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms”. RFC 2001.

[12] A. Falk, Y. Pryadkin, D. Katabi. s.l. “Specification for the Explicit Control Protocol (XCP)”. Internet-Draft, Expires: May 9, 2007.

[13] Dina Katabi, Mark Handley, and Charles Rohrs. “Internet Congestion Control for High Bandwidth-Delay Product Networks”. *In Proceedings of ACM Sigcomm 2002*, August, 2002.

[14] Nandita Dukkipati, Masayoshi Kobayashi, Rui Zhang-Shen and Nick McKeown. “Processor Sharing Flow in the Internet”. *Thirteenth International Workshop on Quality of Service (IWQoS)*, Passau, Germany 2005.