

Quantification of Traffic Burstiness with MAPI Middleware

Sven Ubik, Aleš Friedl, Stanislav Hotmar

CESNET, Prague, Czech Republic

Key words: network traffic dynamics, passive monitoring, packet bursts, traffic burstiness, delay and jitter

1 Introduction and motivation

Network load monitoring is useful for network planning and performance problem troubleshooting. We can monitor network load in different time scales ranging from long-term averages to short-term peak monitoring. Network traffic tends to be bursty for a variety of reasons including protocol design, user behaviour and traffic aggregation [1].

Consequently, load monitoring in fine time scales often reveals peaks of load much higher than in long-term averages. If the network is not provisioned for these peaks of load, packet queues in routers provide temporary buffers. In an extreme case when some queue overflows packet loss occurs. But even when packets fit into the queue, additional delay and jitter are introduced, which can negatively affect real-time video and audio application as well as behaviour of data transport protocols.

Link capacity has been recently precisely defined [2]. But this definition applies only to relatively long-term traffic dynamics. Metrics for traffic burstiness have not yet been defined and methods to monitor traffic burstiness are not well understood. In this paper we explore how traffic burstiness can be quantified, how it can be monitored transparently using various packet capture hardware and what are likely consequences of traffic bursts.

2 Link capacity and traffic burstiness

When observing network load, we can define several terms [2]:

Nominal physical link capacity is the theoretical maximum amount of data that the link can support measured at the physical layer. It includes any inter-frame gaps. For example, for Gigabit Ethernet it equals to 1^9 bits per second.

IP-layer link capacity is the maximum number of IP-layer bits that can be transmitted over the link during a specified time interval (usually per second).

IP-layer link usage is the actual number of IP-layer bits that are correctly received over the link during a specified time interval.

IP-layer link utilization is equal to the link usage divided by link capacity and is therefore between zero and one. It can also be multiplied by 100 and given as percentage.

IP-layer available link capacity is the complement of link usage to link capacity. It can also be defined for a network path as a minimum of available capacity on all links comprising the path.

While not specified in [2], some additional metrics are frequently used:

- All the above metrics can be also defined for a network path (in addition to the available link capacity). In that case we are interested in the maximum value over all links for usage and utilization and minimum value for nominal physical link capacity and IP-layer link capacity.
- The link with the smallest IP-layer link capacity is commonly referred to as the "narrow link" and the link with the smallest available capacity is often referred to as the "tight link".
- *Throughput* sometimes also called *bulk transfer capacity* or *goodput*, is the volume of additional data that can be transferred over the network path. It is measured at the transport layer (at the payload of a transport layer protocol) and depends on the transport protocols used to transfer data already carried over the network and newly added data. Elastic transport protocols, such as TCP, react to network overload (detected by packet drop) by reducing speed of sending data into the network.
- The term *bandwidth* is sometimes used interchangeably with capacity, although electrical engineers use bandwidth to denote the signal spectrum width.

The relationships between these metrics is illustrated in Fig. 1. While nominal physical link capacity is typically constant, the other metrics are varying in time. Available capacity cannot be measured directly, but we can measure link usage and compute available capacity as complement to link capacity.

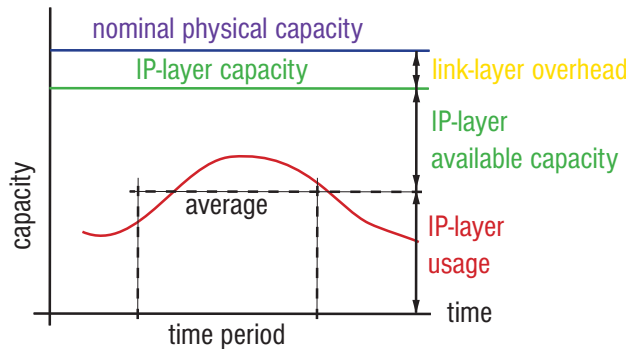


Fig. 1. Metrics for link load measurements

Different time periods of link usage monitoring commonly show different fluctuations measured over the same traffic. Long-term averages smooth out short peaks and drops, which are observable in short-term averages. We may ask what is the "right" time period? It depends on the purpose of monitoring. For accounting purposes, long-term averages can be sufficient. In order to detect peaks of load that can result in delay or loss in packet queues, the shorter time period, the more detailed view on traffic dynamics. However, when we get down to the packet transmission times, the link is always either fully loaded when a packet is currently being transmitted or fully unloaded when no packet is currently being transmitted. The fully loaded condition can span more packets when they are transmitted one after another, thus forming a packet burst. The situation is illustrated in Fig. 2.

While there is a metrics for burst loss [3,4], we currently do not have an official definition of a packet burst. We will therefore define a packet burst as: *a sequence of consecutive packets with*

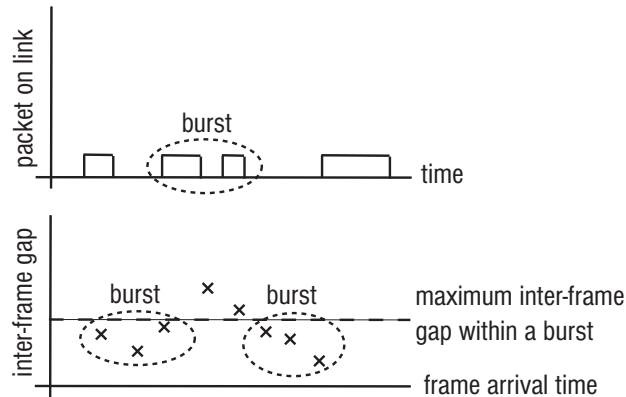


Fig. 2. Packet bursts

inter-frame gaps no greater than a specified parameter, while the inter-frame gap before and after this sequence of packets is greater than a specified parameter.

One possible inter-frame gap that can limit a packet burst is the minimum possible inter-frame gap on the given link type. For Ethernet links ranging from 10 Mb/s to 10 Gb/s the minimum inter-frame gap (IFG) measured from the last bit of the CRC field of one packet to the first bit of preamble of the next packet is equal to 12 byte slots at the given nominal physical link capacity. For example, for 10 Gigabit Ethernet 12 byte slots equal to 9.6 ns.

Packet bursts can be quantified by measuring their length in time, a number of packets or a number of bytes. We can also quantify the inverse of packet bursts, that is the space between bursts and burst arrival times, that is times between starts of consecutive bursts. Various statistics can be applied to quantified packet bursts, such as minimum, maximum, average or quantile of packet burst sizes, space between bursts and their arrival times.

Another possible reasoning is that an inter-frame gap so small that no valid packet can fit into it (including minimum possible inter-frame gaps before and after the packet) is a continuation of a burst. Alternatively, to assess the effect of packet bursts on delay and packet loss due to packet queue build-up, a packet burst can be considered as continuing when usable byte slots in an inter-arrival time between two packets destined for the same output router port are less than the space needed to send the first packet in a frame at the nominal physical link capacity of the output port. This can happen when the sum of nominal physical link capacities of all input ports exceed the nominal physical link capacity of the output port, which is a common case. This situation is shown in Fig. 3. During a packet burst, the packet queue builds-up. After a packet burst ends, the available link capacity (considering only currently arriving packets as link usage) is used to empty the queue. Of course it can happen that another burst arrives before the queue is emptied and it can even fill the queue to a higher level than the previous burst.

Although there was a work based on monitoring the volume of traffic that temporarily exceeds predefined or dynamically computed average link usage [7], it is difficult to capture and monitor traffic from all input router ports destined to particular output port. It is much easier to monitor the output port directly, such as a port to the network backbone on a border router with input ports aggregating user traffic.

Since the queue is emptied at the nominal physical link capacity, we can make a hypothesis that the size of a packet burst in bits measured on the router output port divided by the nominal physical link capacity is the upper limit of delay added due to queue build-up. Distribution of

packet burst sizes among the total volume of traffic is then a probability distribution of upper limits of delay that packets can encounter when passing through the router to the monitored link.

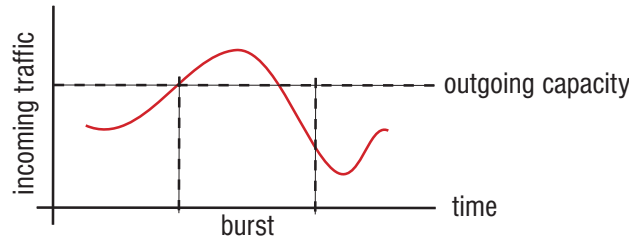


Fig. 3. Packet queue built-up due to a packet burst

3 Traffic burstiness monitoring with MAPI middleware

We designed a system for monitoring of distribution of packet bursts sizes. It can be used to assess effects of packet bursts in real network traffic on added delay, jitter and possibly packet loss. It is a passive monitoring system that works on captured packets and therefore it does not affect user traffic in any way. Only packet sizes and packet capture times are used in the system. User privacy is not affected, because no information inside packets is used. The system works also for encrypted user traffic.

We implemented our application using MAPI [8] middleware. MAPI is a library of functions for development of portable monitoring applications at a higher-level of abstraction. Application functionality is determined by a selection and order of predefined or user-defined monitoring functions applied on captured packet stream. A key benefit of MAPI is that allows applications to run transparently on different packet capture hardware. This is achieved by supporting each type of hardware by a separate implementation of monitoring functions. These functions can utilize whatever hardware acceleration is possible with the given hardware. Software implementation is provided in `stdflib` library that works for all hardware including regular Ethernet cards.

The architecture is shown in Fig. 4. A monitored link is tapped by an optical splitter or a mirroring port on a router. Packets are captured by an Ethernet card or a specialized monitoring card that can provide hardware acceleration, such as DAG [9] or COMBO [10] card.

We implemented two versions of a BURST monitoring function for MAPI: in `stdflib` library for regular Ethernet cards and in `combo6flib` library for a COMBO6X card. Both versions are designed for Gigabit Ethernet links. We are currently working on the implementation for our MTPP (Modular Traffic Processing Platform) hardware, which will operate on 10 Gigabit Ethernet at the full line rate.

The BURST function accepts the following set of arguments:

iftime The maximum inter-frame gap that is still considered as a continuation of a burst. For implementation purposes, it is specified including the packet preamble and the start-of-frame delimiter and it is expressed in nanoseconds. For example, the minimum value for Gigabit Ethernet is 160 ns.

min Bursts smaller than this value are counted together.

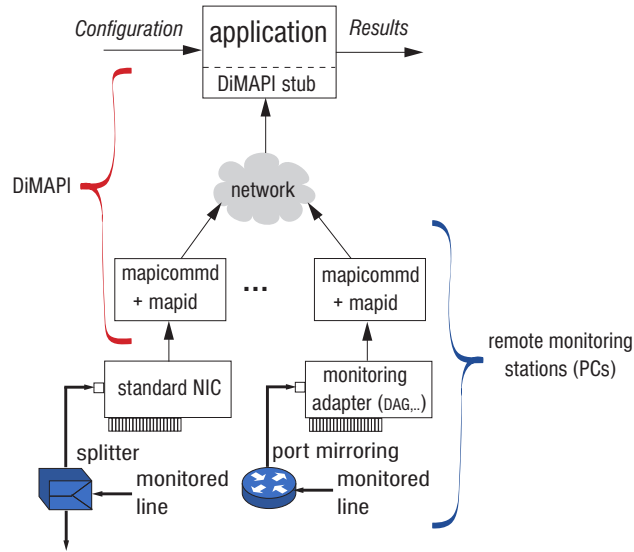


Fig. 4. Application architecture

max Bursts larger than or equal to this value are counted together.

step In between the minimum and maximum burst sizes specified by previous parameters, bursts are counted in classes separated by this step.

Min, max and step arguments determine categories where bursts and gaps between bursts will be sorted:

Category	Burst or gap size in bytes
0	<0, min)
1	<min, min+step)
...	...
254	<max-step, max)
255	>= max

is also available in the version where the size of each class can be specified individually. This allows for non-linear distribution of packet burst sizes. 256 classes are available including the two for small and large burst sizes.

Packet bursts are divided into classes and the following values are maintained for each class:

- Number of bursts
- Number of packets
- Number of bytes
- Number of gap bytes

The last value measures distribution of gap sizes between bursts and is available in MAPI and MTPP versions. All values are maintained in 64-bit counters.

4 Hardware implementation on COMBO6X card

The implementation for the COMBO6X card operates completely in hardware and MAPI is used only to retrieve measurement results.

COMBO6X is a PCI-X card with 1 Gb/s SFP optical transceivers for packet capture and a Xilinx Virtex II Pro FPGA for traffic processing.

We created an STU_BURST (Statistical Unit for Bursts) module for this card with the following key properties:

- Measurement can run continuously at the full line rate of 1 Gb/s for all packet and bursts sizes and for all four input ports simultaneously (results are provided for each port separately).
- Bursts sizes are classified into 256 bins whose width can be specified individually allowing for non-linear distributions.

The COMBO6 family of cards uses a separate timestamp card to provide precise packet arrival timestamps. We did not have a working timestamp card available when developing the STU_BURST module. Therefore we used a counter driven by the 125 MHz clock of the GMII interface and used the value of this counter to determine both relative timestamps of packet arrivals and packet sizes. It was sufficient for this application, absolute (wall clock) timestamps are not needed. The timestamp is 64-bit long, where 32 bits are used for seconds and the other 32 bits are used for the fraction of a second. The rest of the design runs at the 100 MHz clock driven by an on board oscillator.

The structure of the STU_BURST module is shown in Fig. 5. The BURST_-SAMPLER unit computes the inter-frame gap between every two consecutive packets A and B using the formula:

$$IFG = ts_B - ts_A - len_A$$

where ts_A is a relative timestamp of packet A , ts_B is a relative timestamp of packet B and len_A is the length of packet A .

The BURST_CLASSIFIER unit increments one of 256 counters of packets, bytes and bursts that corresponds to the bin where the current burst is classified. Bursts are classified into bins in two sequential steps. In the first step, the size of burst is compared to 7 limits that divide the space of 256 bins into 8 parts. These limits are precomputed after the user uploads configuration of bin sizes and the comparison to all 7 limits takes place in parallel. In the second step, the size of burst is compared step-by-step with bin sizes within the part that was identified in the first step. This solution is a compromise between comparing to all 256 limits step-by-step, which would be slow, or in parallel, which would require more FPGA resources.

At any time one bank is active and is used to count the number of packets, bytes and bursts in 256 bins. The other bank is inactive. When the user wants to read measured results, the inactive bank becomes the active bank and the now inactive inactive bank is available for reading. In this way the user can read results safely at any speed and values from all counters are valid for the same moment in time while monitoring can continue. Separate counters are provided for each input port. The total number of counters is bins * characteristics * banks * input ports = $256 * 3 * 2 * 4 = 6144$. The counters are implemented using BlockRAM in FPGA.

The required FPGA resources are summarized in Table 1 along with the total resources available in FPGA.

5 Laboratory measurements

We first tested the STU_BURST module by simulation, more details can be found in [12]. Then we did a stress test by sending full 1 Gb/s to all four ports simultaneously in 1518 and 64-byte

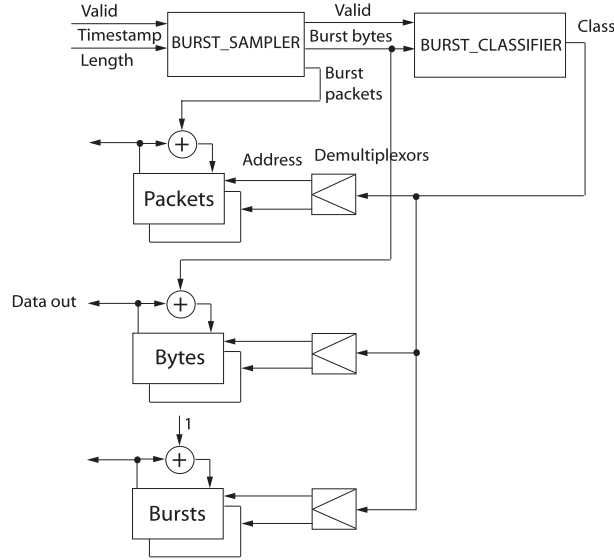


Fig. 5. Structure of STU_BURST unit

Table 1. Required FPGA resources

	STU_BURST	FPGA capacity
Slice	1509	23616
Flip-flops	1665	47232
4-input LUTs	2214	47232
BlockRAM	17	232

packets from Ixia 250 hardware packet generator. The module counted all packets correctly and could operate continuously. In the next test, we checked if bursts are correctly classified into bins according to their sizes by sending bursts of various lengths.

In the final test, we checked precision of end-of-burst recognition. We sent packets with various interframe gaps and for each gap size we measured the minimum value of iftime parameter where all packets were still counted as one burst. Ideally, these two numbers should be equal. As we can see in measured results in Table 2, we needed to set iftime to approximately 3 bytes more than the theoretical value. When we sent packets between two ports on the generator connected back-to-back, we found that the generator exhibits certain fluctuations in packet dispatch times, which can be one of the reasons while we needed to set iftime to slightly higher value. Another reason can be impressions in timing within the STU_BURST unit. In measurements on real traffic, we can safely accept this 3-byte difference, because interframe gaps of just a few bytes cannot be utilized by inserting a new packet and have no practical value.

6 Measurements on real traffic

We created a set of PHP scripts for graphical user interface, which is illustrated in Fig. 6. Distribution and cumulative distribution functions of bursts, packets and bytes in individual burst

Table 2. Measuring precision of burst end recognition

IXIA 250	STUB
Sent IFG	Required iftime
12 B	15 B
100 B	103 B
1.000 B	1.003 B
10.000 B	10.003 B
65.530 B	65.534 B

size categories can be plotted for specified time intervals in 2D graphs or as evolution in time in 3D graphs. For example, a cumulative distribution function of the number of bytes in different burst sizes over a period of one hour on the link between the European Géant network and the SWITCH network in 2D and 3D graphs (showing evolution in 1 minute increments) is shown in Fig. 7 and Fig. 8, respectively. This measurement was obtained as part of the SA3 activity [5] of the GN2 project.

7 Related work

Monitoring of packet burst sizes in categories was first used in [6]. Software implementation with a Gigabit Ethernet DAG card to capture packets was realized. Measured results were however not interpreted regarding their effect on data transfer characteristics and only the case of back-to-back packets forming a burst was considered.

Hardware implementation using a specialized hardware of link load deviation from the predefined or dynamically determined average load was presented in [7].

Considerations about possible standardization of packet bursts and burstiness were presented in [13]. This work builds on a configurable limit of inter-frame gaps to end a burst including proposal of some statistics on burst sizes.

Network capacity can be monitored with other methods than passive monitoring with certain limitations. Monitoring based on reading of router interface byte counters by SNMP is a common and reliable method, which also does not affect network traffic. However, it requires access to routers and can provide only relatively long-term averages (approx. > 20 seconds), because routers tend to update their counters with varying delay of several seconds. Stress-type active monitoring (e.g., iperf, bwctl) measures throughput over a network path, which is a different metrics than link usage. It also affects user traffic and is affected by end host performance. Lightweight active monitoring (e.g., pathrate, pathload) estimates used or available capacity in a network path or per link. However, these tools provide unreliable results in high-speed networks.

8 Conclusion

We proposed how packet bursts can be quantified and monitored. We discussed what is the effect of packet bursts on delay and jitter introduced by router queues. We described software and hardware implementations of a system for precise and non-intrusive monitoring of packet bursts in high-speed networks.

In our future work we plan to do a hardware implementation for a 10 Gb/s version of the MTPP platform and conduct more experiments to further investigate dependence of delay and jitter in real network traffic on packet bursts introduced in routers.

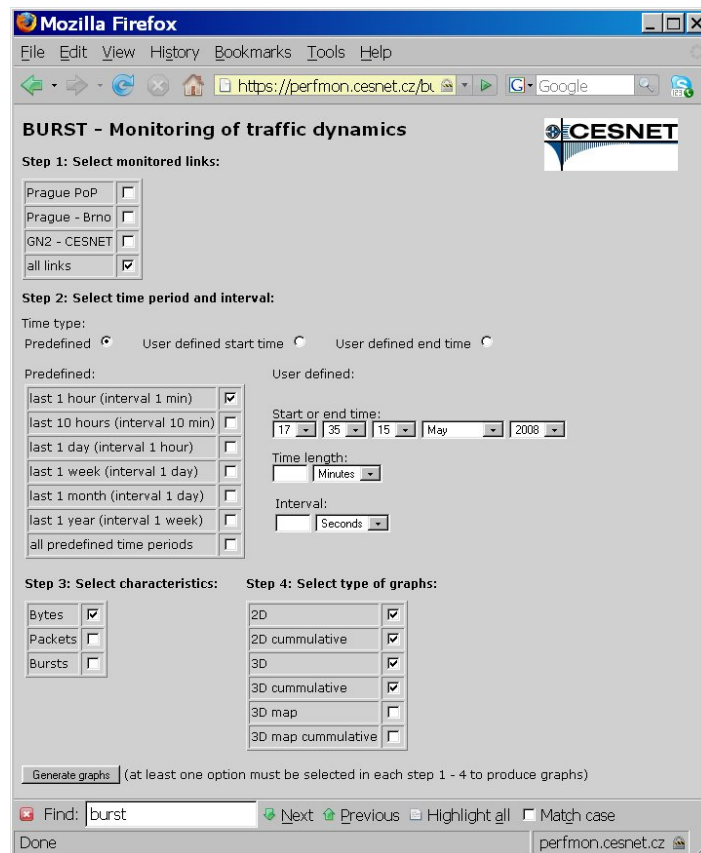


Fig. 6. User interface

References

1. Constantine Dovrolis, Hao Jiang. *Why is the Internet traffic bursty in short time scales?*, ACM SIGMETRICS Performance Evaluation Review, Vol. 33, No. 1, June 2005.
2. P. Chimento, J. Ishac. *Defining Network Capacity*, RFC5136, IETF, February 2008.
3. T. Friedman, R. Caceres, A. Clark. *RTP Control Protocol Extended Reports (RTCP XR)*, RFC3611, IETF, November 2003.
4. R. Koodli, R. Ravikanth. *One-way Loss Pattern Sample Metrics*, RFC 3357, IETF, August 2002.
5. GN2 Project, FP6 Contract No. 511082, <http://www.geant2.net>.
6. Klaus Mochalski. *A Real-Time Packet Burst Metric*, TNC2004, Rhodos, June 2004.
7. Ryosei Takano, Yetsu Kodama, Tomohiro Kudoh, Motohiko Matsuda, Fumihoro Okazaki. *Realtime Burstiness Measurement*, PFLDnet 2006, Nara, Japan, February 2006.
8. MAPI – Monitoring Application Programmable Interface, <http://mapi.uninett.no>.
9. DAG cards, Endace company, <http://www.endace.com>.
10. COMBO cards, <http://www.liberouter.org>.
11. SCAMPI project, <http://www.ist-scampi.org>.

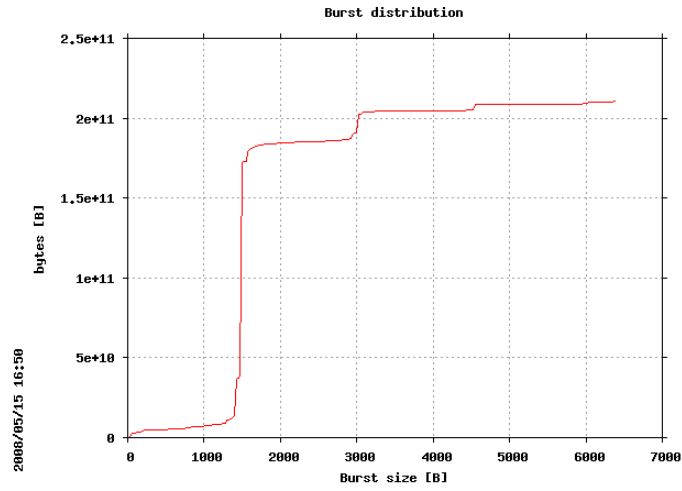


Fig. 7. Byte distribution in bursts

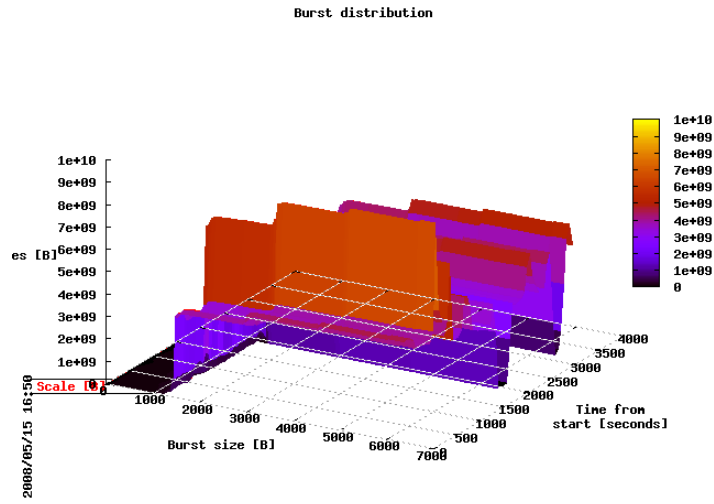


Fig. 8. Evolution of byte distribution in bursts

12. Stanislav Hotmar, *Implementace VHDL modulu pro sledování dynamiky síťového provozu*, M.Sc. Thesis, Computer Science Department, Faculty of Electrical Engineering, Czech Technical University, May 2007.
13. R. Krzanowski. *Burst (of packets) and Burstiness*, 66th IETF Meeting, Ontario, Canada, October 2006.