Burstitication effect on the TCP Synchronization and Congestion Window mechanism

Kyriakos Vlachos
University of Patras
Email: kvlachos@ceid.upatras.gr

Introduction

- OBS is a new switching paradigm design but
- TCP is the de facto standard in transport protocols.
- TCP over OBS remains an open issue since :
  - burstification at the edge induce an unpredictable delay at the edge that affects TCP window
  - Burst losses in the core tends to synchronize flows.
- The impact on the individual flow performance is not investigated
- Average results conceal true performance

Overview of Assembly Schemes

- Size-based Assembly Schemes
  - burst is released as soon as a max (or min) burst size is reached
- Timer-based algorithms
  - Time-based algorithms impose a tight upper bound to the burst assembly time
- *Mixed-Algorithms*
  - Combination of size-based algorithm and timer-based algorithms. A burst is complete when either the size of the burst the size *B* or the T timer expires
- Adaptive Burst Assembly
  - They monitor traffic conditions (i.e congestion, burst losses, blocking) and provide feedback to adjust a timer-based or burst-size based scheme

TCP variants

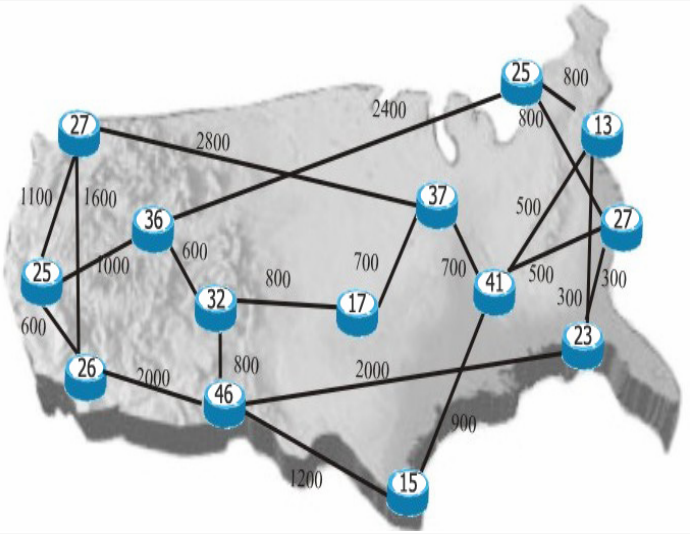
- Tahoe, Vegas, Reno, New Reno, SACK
  - TCP Reno refers to TCP with *Slow Start*, *Congestion Avoidance*, *Fast Retransmit* and *Fast Recovery* algorithms. It employs TO and TD losses.
  - New Reno retransmits one lost segment per round-trip-time upon receiving a partial ACK message
  - SACK Only the un-acknowledged segments are re-transmitted.

SACK performance clearly superior [Xiang Yu et al. "Traffic statistics and performance evaluation in optical burst switched networks", IEEE/OSA JLT]

Burstification effect in flow window

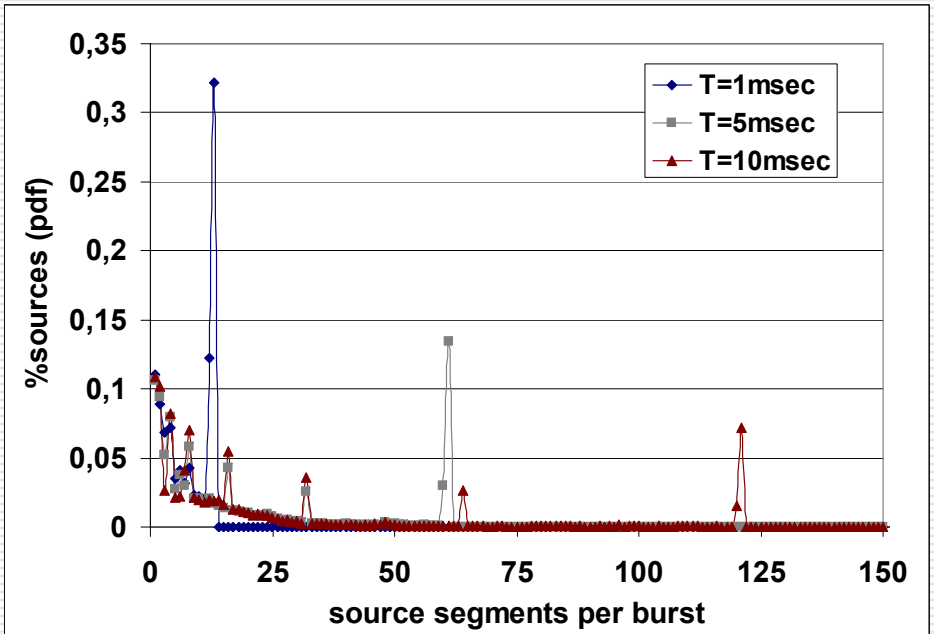
- Study SACK and timer-based schemes
- Built an efficient TCP-over-OBS (ns2) simulator

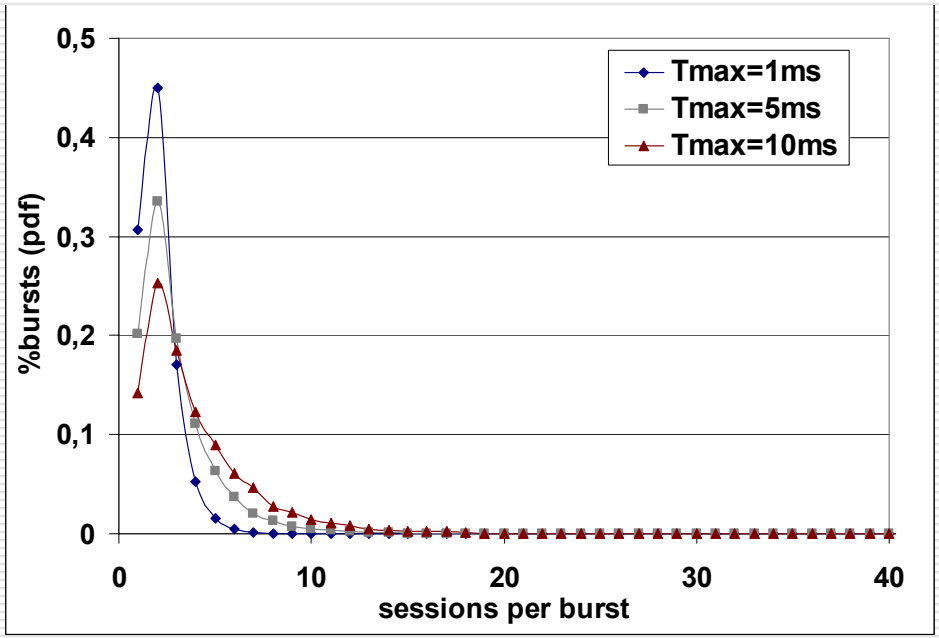
Simulation Setup:
NSF network with 14 nodes.
(8 edges and 6 core nodes).
10Gb/s link rate and 1λ per link
TCP arrivals at each edge node:
- Poisson distribution with mean λ=100arrivals/sec and
- exponential inter-arrival time with 1/μ=10msec.



TCP file size: Pareto distribution with variable mean and 40KByte min ON size.

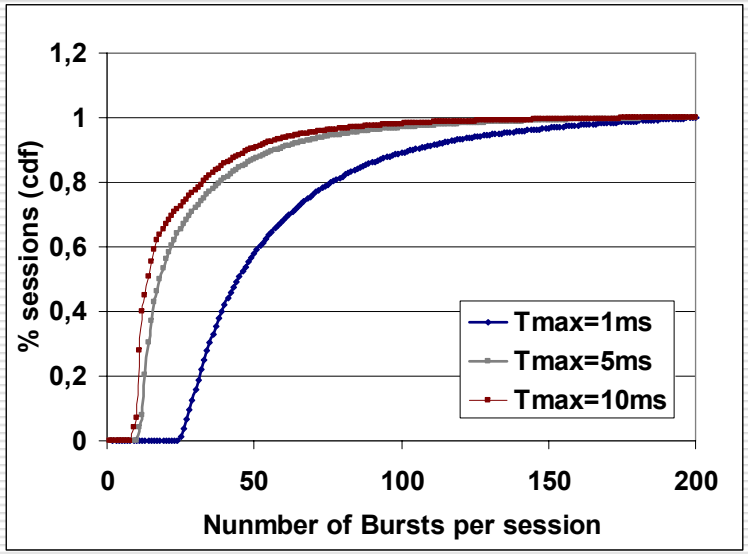
Segment and Flow distribution

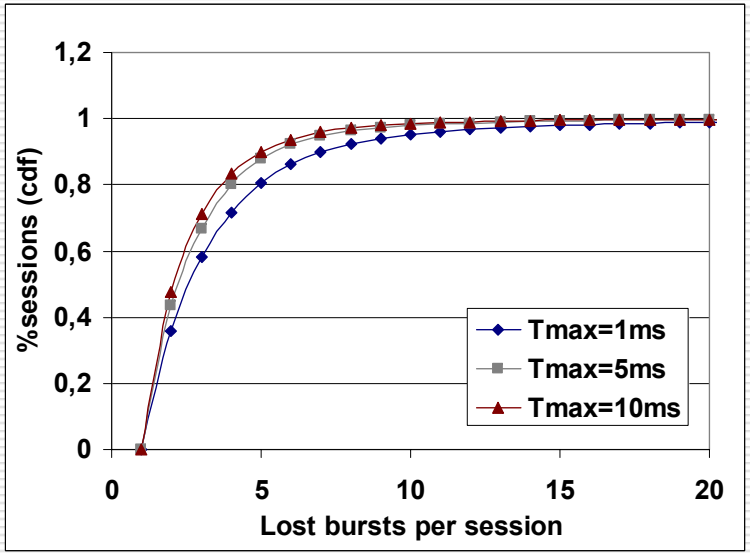




A mean file size of 700KB was selected, which correspond to 600 - 800 active sources and a burst loss ratio of <2%.

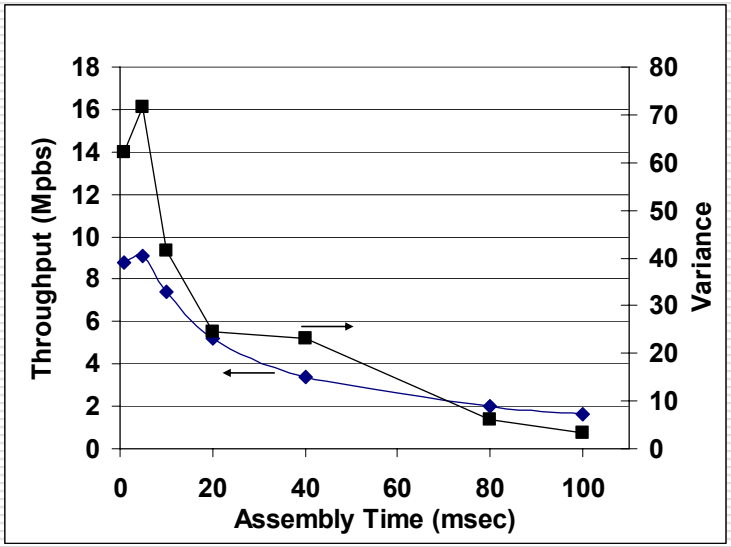
Burst per TCP flow

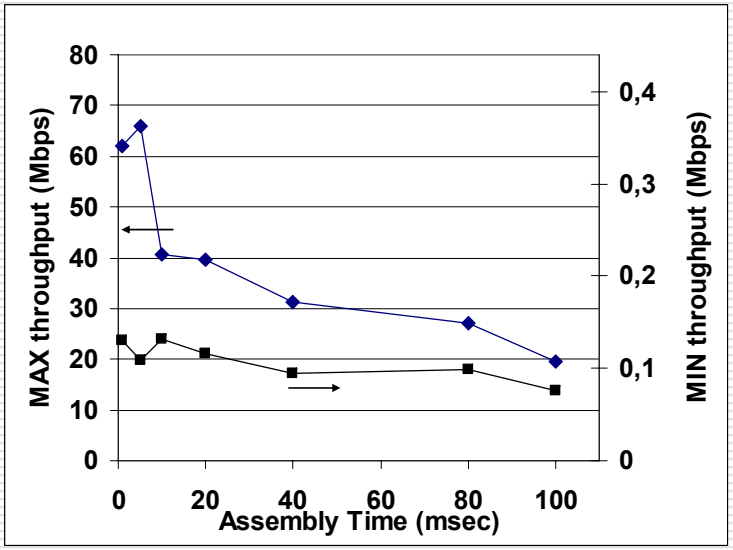




Number of bursts needed to complete a TCP session and number of burst lost per session for T = 1, 5, 10msec assembly time.

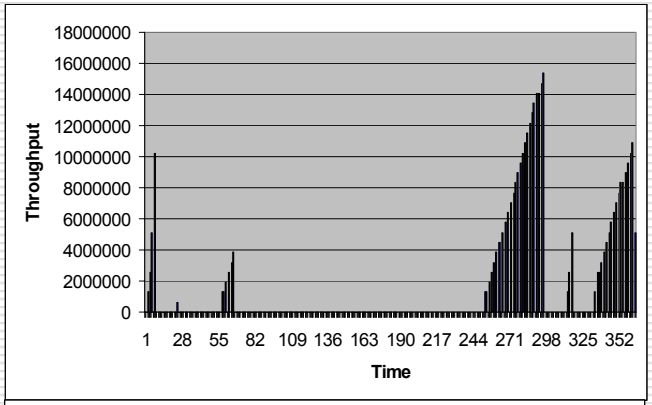
Throughput Stats

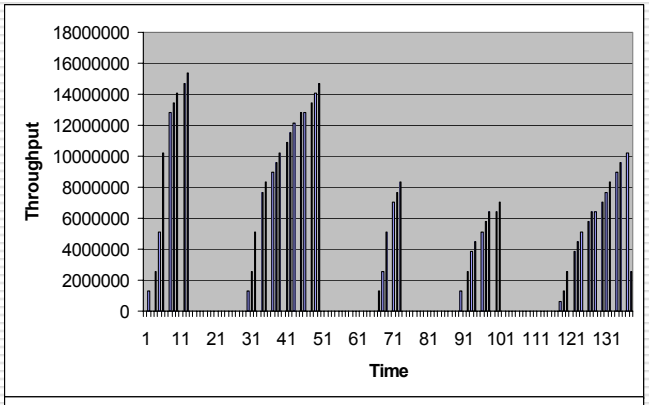


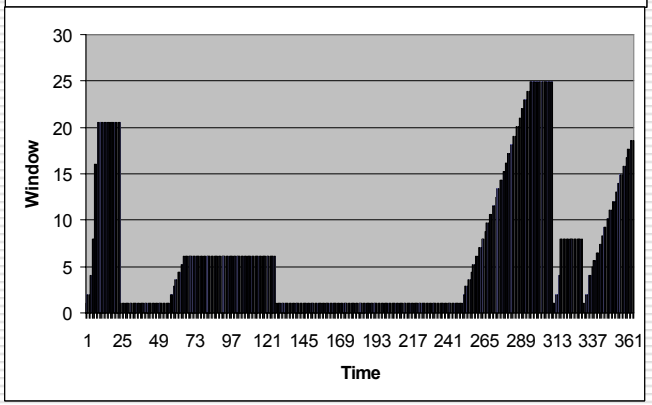


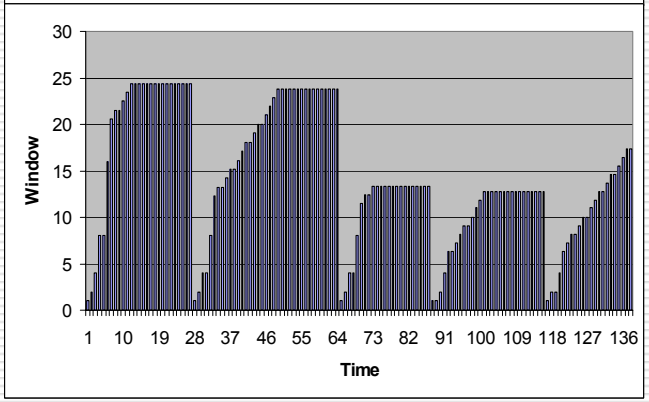
(LEFT) Average and variance of throughput of all flows versus burst aggregation time.
(RIGHT) Corresponding maximum and minimum values measured.

Efficient Throughput and Congestion Window



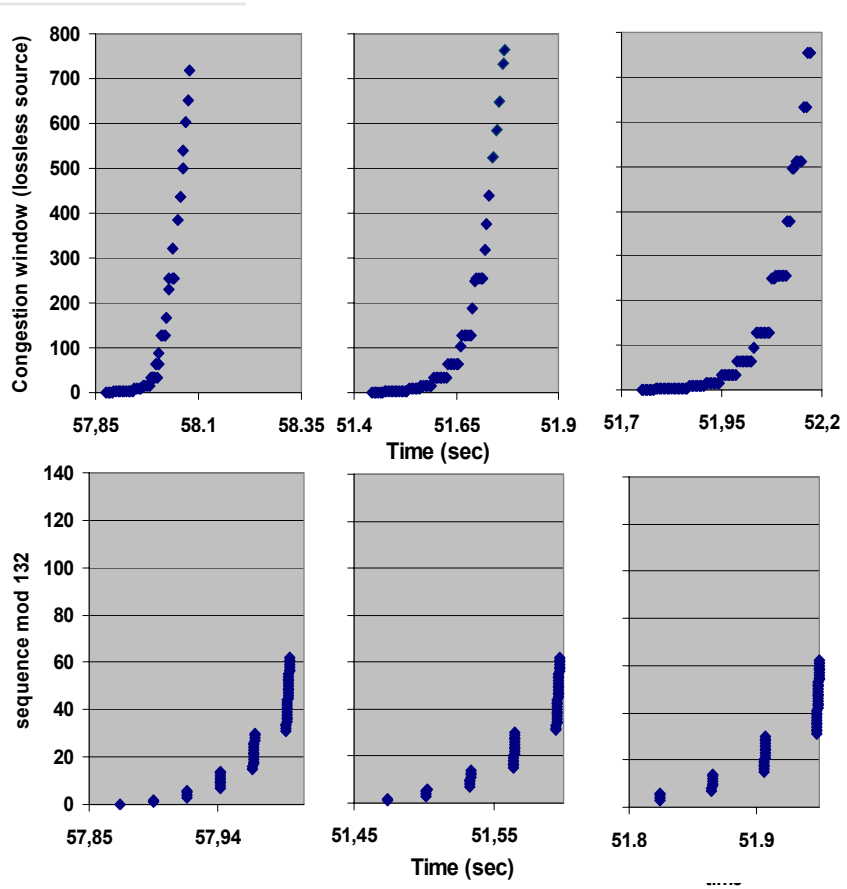






Typical TCP Throughput and congestion windows. Measurements were taken for time-units of 13msec for the whole duration of the experiment

Throughput Stats



Congestion window and Sequence number of the transmitted segments for TMAX = 1*msec* (left column), 5*msec* (middle column) and 10*msec* (right column) assembly time.

It can be seen that for small assembly times, window rises faster but the long term throughput remains the same

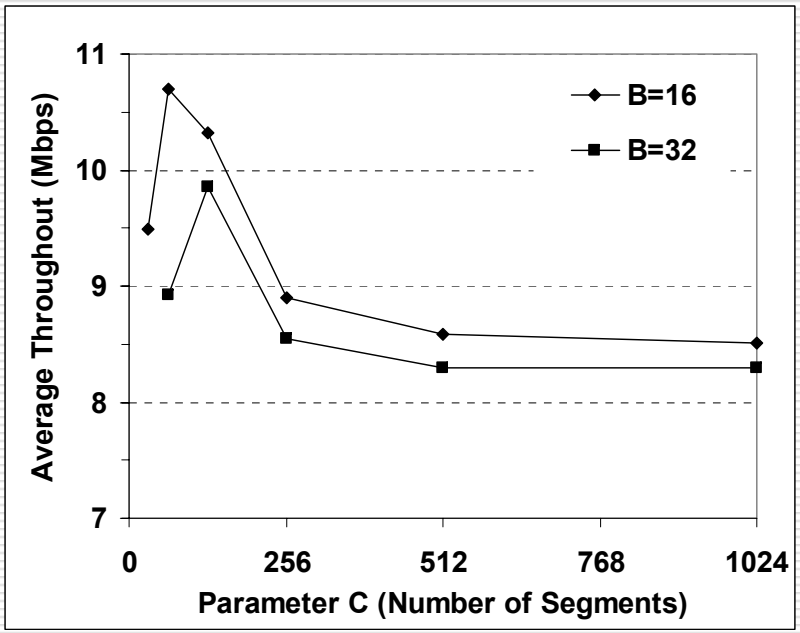
Window-based Burst Assembly Scheme

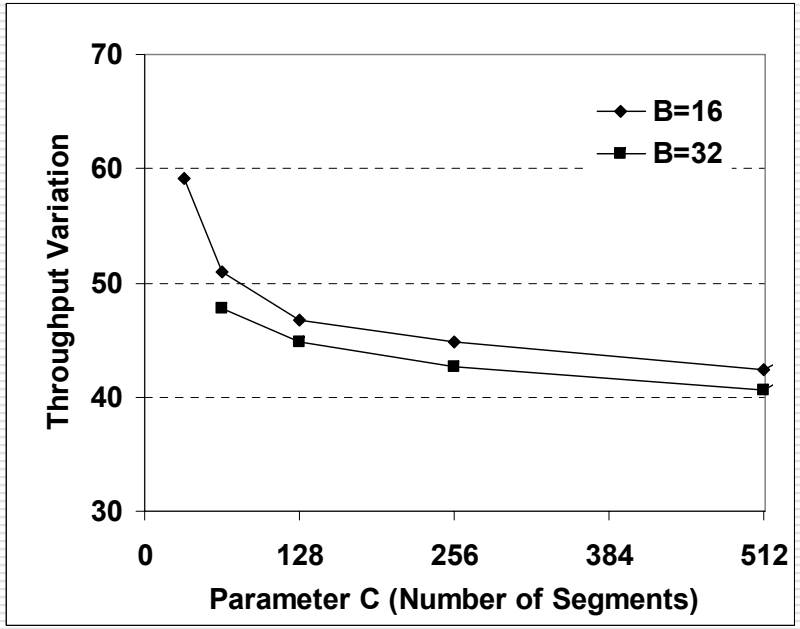
- We propose a new burst assembly scheme that assigns different burstification delays to flows based on their congestion window size. In particular, we define assembly time as follows

$$T = \begin{cases} 1\,msec & if\ 1 \leqslant congestion\ \ window < B\ segments \\ 5\,msec & if\ B \leqslant congestion\ \ window < C\ segments \\ 10\,msec & if\ C \leqslant congestion\ \ window \end{cases}$$

- A flow is defined as
  - *Slow,* when its window is < B segments.
  - *Medium,* when its window is < C segments and
  - *Fast*, when its window is >C segments

Average throughput and variance





the particular case of "*B=16*" and "*C=64*" exhibits the highest throughput (10.7Mbps) while variation is significantly improved to only 51.

Conclusions

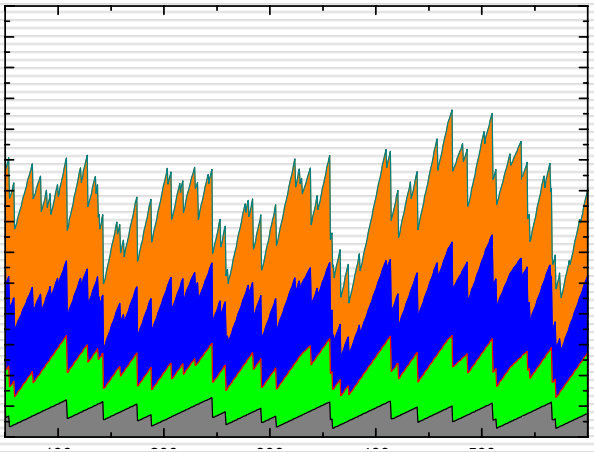
- Average throughputs are not indicative for TCP performance.
- Fixed assembly times do not provide maximum performance but only optimal performance for individual flows with similar characteristics (i.e file size, size of window, blocking, etc.).
- Instant congestion window is a metric to be considered for determining optimum assembly time.
- A multi-queue burst assembly scheme with different timers can provide maximum performance with a notion of fairness.

Burstification effect on TCP synchronization

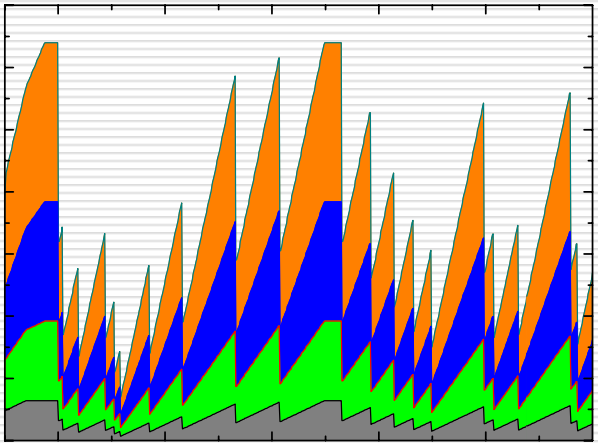
in collaboration with:
Carla Raffaelli, Anna Guidotti (Univ. of Bologna) and
Óscar González de Dios (Telefonica)

Burstification effect on TCP synchronization

- TCP synchronization occurs when flows simultaneously increase/decrease their window.
- Burst losses is the prime cause for synchronization



NO Synchronization



Synchronization

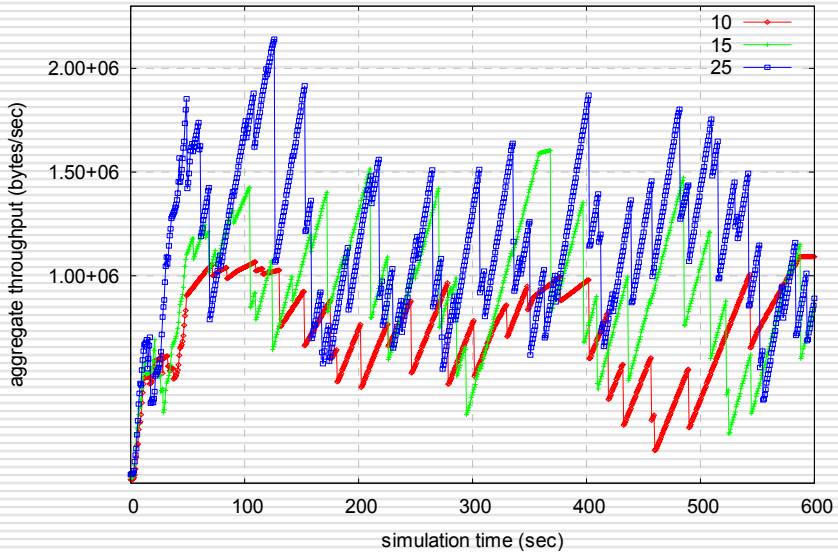
Burstification effect on TCP synchronization

- Synchronization effect depends on the number of flows per burst and
- The number of burst losses per TCP flow.
  - Small assembly delays may lead to a strong synchronization of a small number of flows (within 2-3 burst losses)
  - Large assembly times may lead to a smaller synch. effect but for a higher number of flows.
- TCP-Synch. effect differs per assembly strategy.

TCP throughputs variation

Aggregated throughput of TCP flows with different number of flows being assembled into a burst.

Standard deviation differs significantly.



1queue for all flows
(normal case)

Dynamic Multi-Queue Burst Assembly

- a dynamic allocation would hamper the continuous aggregation of the same sources over the same bursts that cane be
  - *a priori* by aggregating together different flows per assembly cycle based on a predefined flow allocation algorithm OR
  - *a posteriori* by avoiding aggregating together only that flows that suffered from a segment loss in the same burst

Flow allocation algorithm

The allocation algorithm is modeled by bounding alternate trials with *n* possible outcomes. In the simple case of employing, *n=2* queues, a flow is allocated to 1st queue with a probability equal to: $p_1=p_2=1/2$

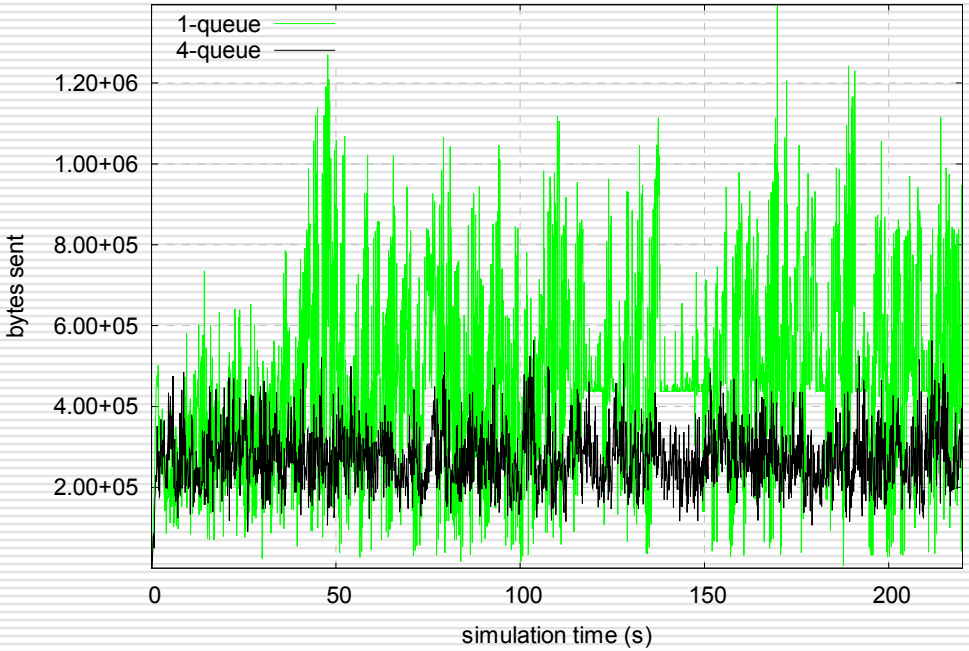
In the general case of *n* queues, the probability P_i^j of *j* flow, with $j \in \{0 \ldots N-1\}$ to be assigned to queue $i \in \{0 \ldots n-1\}$ is:

$$P_i^j = \rho_i^{j \bmod (n-1)}$$

where

$$\begin{pmatrix} \rho_0^0 = p_0 \\ \rho_1^0 = p_1 \\ .. \\ \rho_{n-1}^0 = p_{n-1} \end{pmatrix}, \begin{pmatrix} \rho_0^1 = p_0 \bullet \left(1 - \rho_0^0\right) \\ \rho_1^1 = p_1 \bullet \left(1 - \rho_1^0\right) \\ .. \\ \rho_{n-1}^1 = p_{n-1} \bullet \left(1 - \rho_{n-1}^0\right) \end{pmatrix}, \ldots, \begin{pmatrix} \rho_0^{n-1} = 1 - \rho_0^{n-2} \\ \rho_1^{n-1} = 1 - \rho_1^{n-2} \\ .. \\ \rho_{n-1}^{n-1} = 1 - \rho_{n-1}^{n-2} \end{pmatrix}$$

Performance of DMQ



Aggregated throughput of TCP flows for dynamic flow allocation in a **four-assembly** queues

Synchronization is weakened: standard deviation has been reduced by 37% in the case of two-queues and 40% in the case of four queues.

Acknowledgement

This work has been supported by European Commission through the NoE E-Photon/ONe+ project via the Joint Project on *Optical Burst Switching* (JP-B).

THANK YOU