Implementation and Evaluation of Proportional Integral controller Enhanced Algorithm in ns-3

Mohit P. Tahiliani, Shravya K. S., Smriti Murali

Wireless Information Networking Group,

National Institute of Technology Karnataka, Surathkal, India

tahiliani@nitk.ac.in

Outline of the presentation

- Introduction: *Bufferbloat* and PIE
- Motivation
- Main contributions
- Implementation details
- Model evaluation
- Functional verification
- Conclusions and the next goals
- Acknowledgement

Introduction: *Bufferbloat*

- ❑ Inexpensive memory.
- ❑ *Side effect*: Bloated buffers at routers!
- ❑ *Bufferbloat*: large queueing delays
- ❑ *Potential solution*: deploy AQM algorithms to control queue delay

Popular AQM algorithms under research:

- ❑ RED / Adaptive RED [S. Floyd, V. Jacobson, …]
- ❑ CoDel / Fair Queue CoDel [K. Nichols, V. Jacobson, …]
- ❑ PIE [R. Pan, P. Natarajan, …]
- ❑ COBALT (CoDel + BLUE) [J. Morton, discussions on codel mailing list]

Introduction: PIE

- ❑ PIE = RED + CoDel. Extension of the original PI algorithm [1]

Four major components of PIE:

- ❑ Random dropping
  - based on drop probability calculation
- ❑ Drop probability calculation
  - happens at a regular interval
- ❑ Average departure rate estimation
  - only when there is sufficient amount of data
- ❑ Burst allowance calculation
  - allows short bursts to pass through successfully

[1] Hollot, C. V., Misra, V., Towsley, D., & Gong, W. B. (2001). On designing improved controllers for AQM routers supporting TCP flows. In INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE (Vol. 3, pp. 1726-1734). IEEE.

Motivation

- Latency of 300ms appears to be "slow" [1]
- *Bufferbloat* makes the situation worse.

Why implement PIE in ns-3:

- PIE algorithm: is being studied against ARED, CoDel (e.g.: MADPIE)
- PIE models: available in Linux and ns-2 (ns-2.36.rc1 only)
- ns-2 support and maintenance has stopped!
- ns-3: several new features compared to other simulators.

[1] Grigorik, I. (2013). High Performance Browser Networking: What every web developer should know about networking and web performance. "O'Reilly Media, Inc.".

Main contributions

- Developed a new model for PIE in ns-3 (version 3.24) [1].
- It is based on the ns-2 model of PIE, implemented by its authors.
- Preliminary verification by writing test cases in ns-3.
- Detailed evaluation by comparing results obtained from ns-2 and ns-3.
- Directions to reproduce the results [2].

[1] https://codereview.appspot.com/277610043

[2] https://github.com/mohittahiliani/reproduce-pie-paper

Implementation details

Source location:

src/network/utils/pie-queue{.h, .cc}

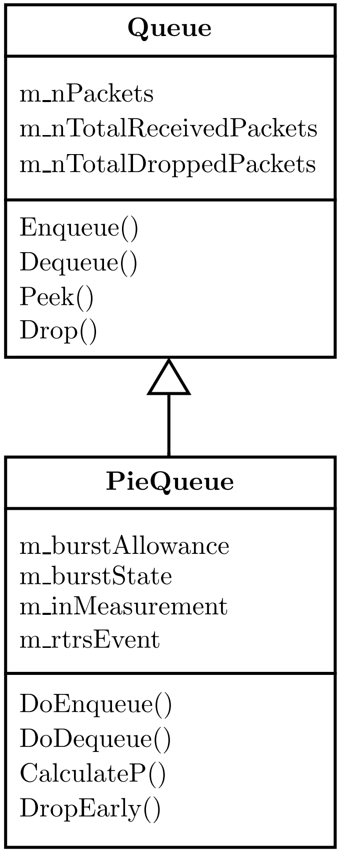


Figure: Class diagram for PIE model in ns-3

Implementation details

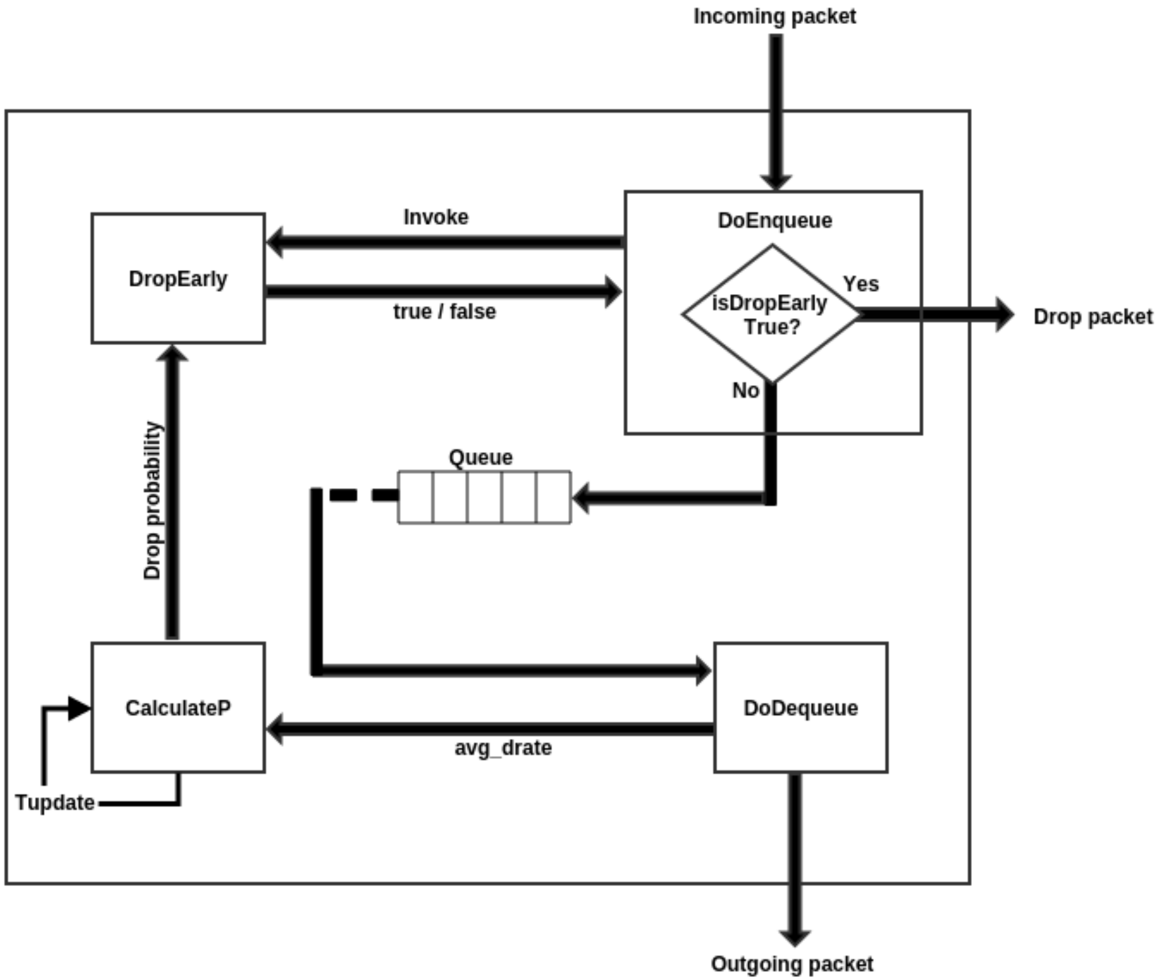


Figure: Interaction between the core methods of PIE

Implementation details

Table 1: PIE parameters to calculate *p*.

PIE parameter	ns-3 variable
Tupdate	m_tUpdate
cur_del	m_qDelay
old_del	m_qDelayOld
ref_del	m_qDelayRef
α	m_a
β	m_b
avg_drate	m_avqDqRate

Table 2: PIE parameters to estimate *avg_drate*.

PIE parameter	ns-3 variable
qlen	m_packets / m_bytesInQueue
dq_threshold	m_dqThreshold
dq_count	m_dqCount
start	m_dqStart
dq_int	tmp
ϵ	fixed to 0.5

Table 3: PIE parameters to calculate *burst_allow*.

PIE parameter	ns-3 variable
burst_allow	m_burstAllowance
max_burst	m_maxBurst
Tupdate	m_tUpdate
cur_del	m_qDelay
old_del	m_qDelayOld
ref_del	m_qDelayRef

Model Evaluation

- ❑ A test suite for evaluating the working of PIE algorithm

  - verifies the attribute settings of PIE parameters

  - basic enqueue / dequeue of packets

- ❑ Compare PIE results obtained from ns-2 and ns-3 for same scenarios

- ❑ Performance metrics under observation:

  - Queue delay

  - Throughput

  - Number of packet drops

Functional verification

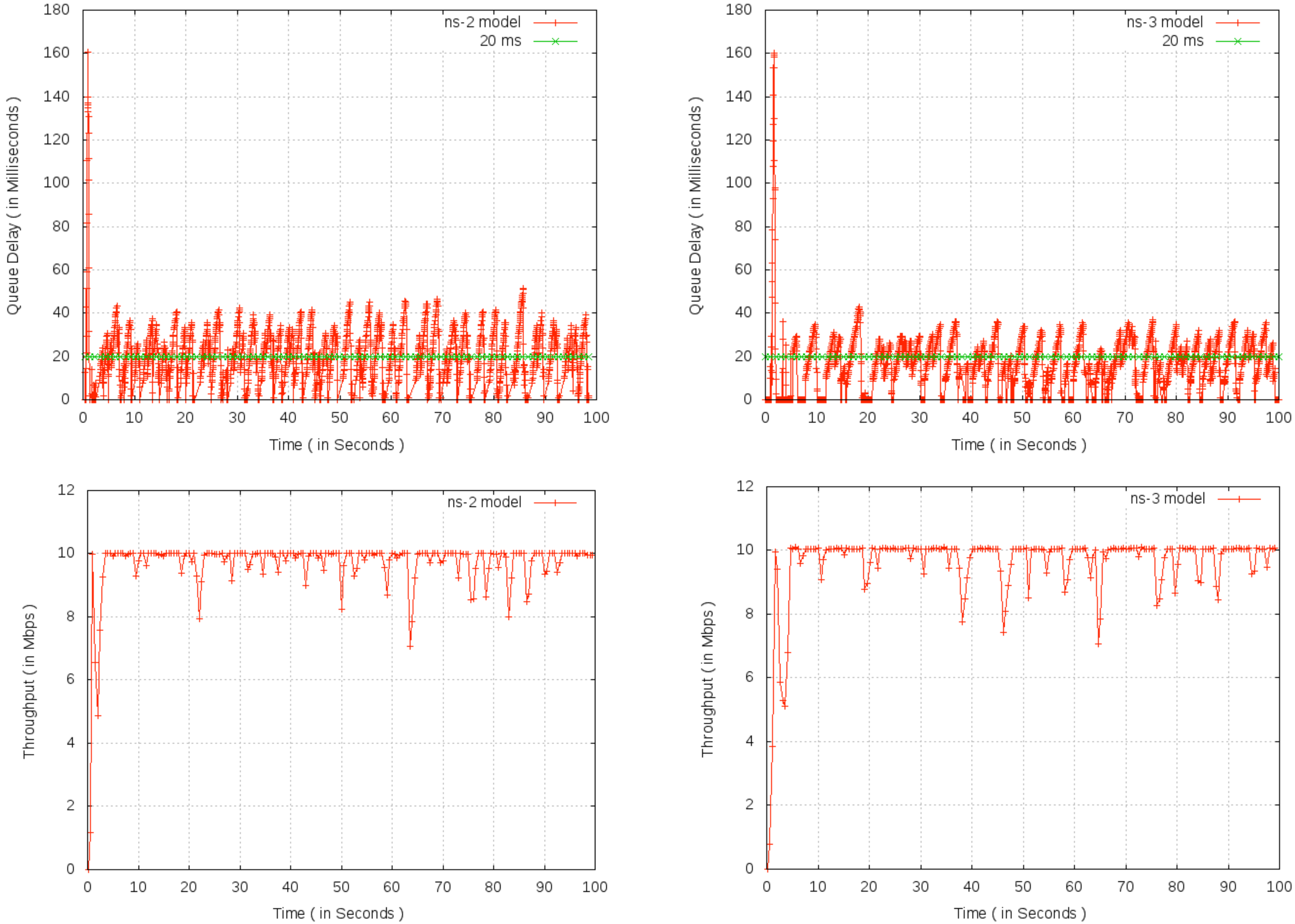
Four simulation scenarios:

1. Light TCP traffic
2. Heavy TCP traffic
3. Mix TCP and UDP traffic
4. Bursty UDP traffic

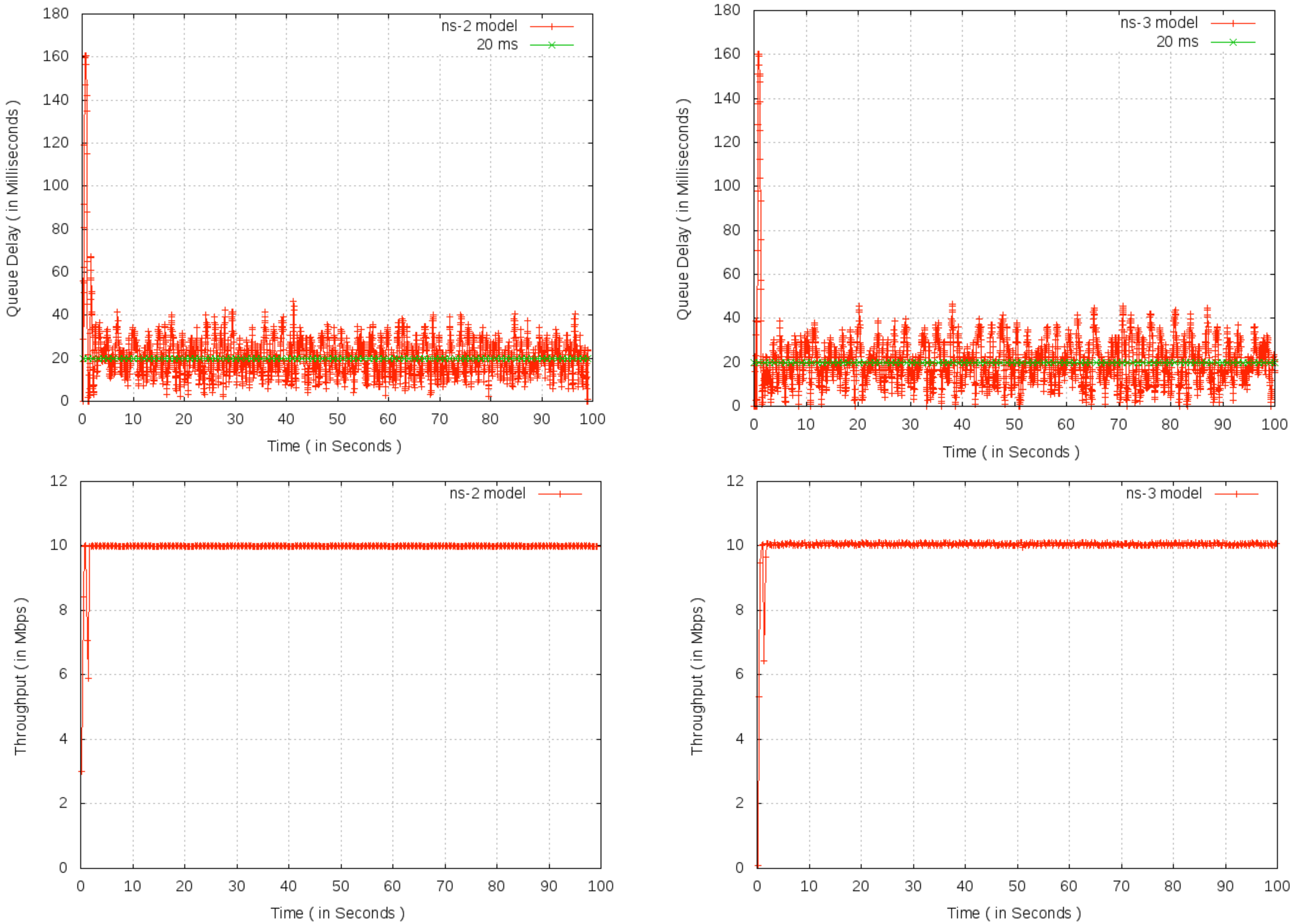
Table 4: Simulation setup.

Parameter	**Value**
Topology	Dumbbell
Bottleneck RTT	100ms
Bottleneck buffer size	200KB
Bottleneck bandwidth	10Mbps
Bottleneck queue	PIE
Non-bottleneck RTT	10ms
Non-bottleneck bandwidth	10Mbps
Non-bottleneck queue	DropTail
Mean packet size	1000B
TCP	NewReno
ref_del	20ms
Tupdate	30ms
α	0.125Hz
β	1.25Hz
dq_threshold	10KB
max_burst	100ms
Application start time	0s
Application stop time	99s
Simulation stop time	100s

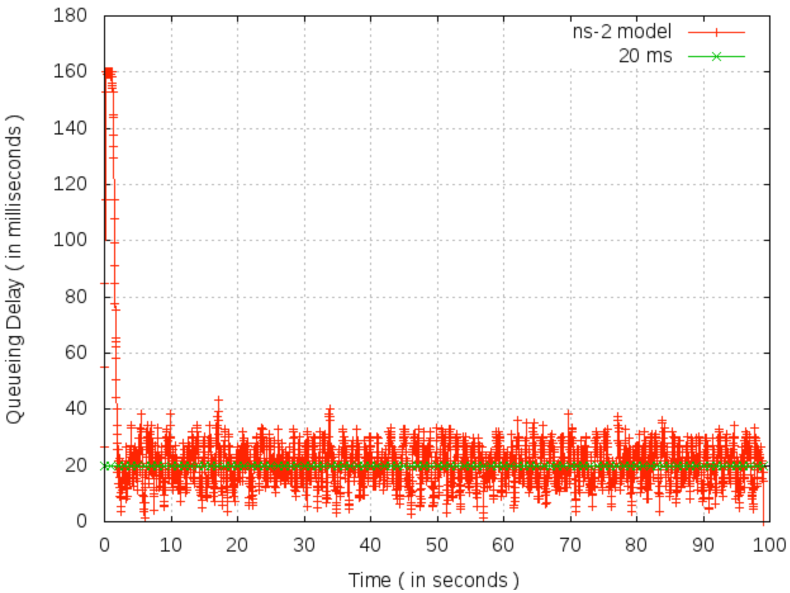
Functional verification: Light TCP traffic

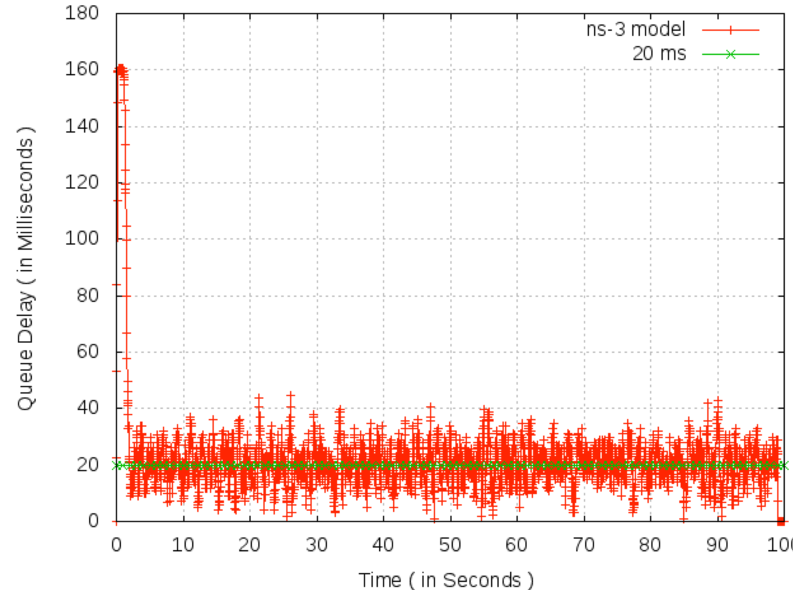


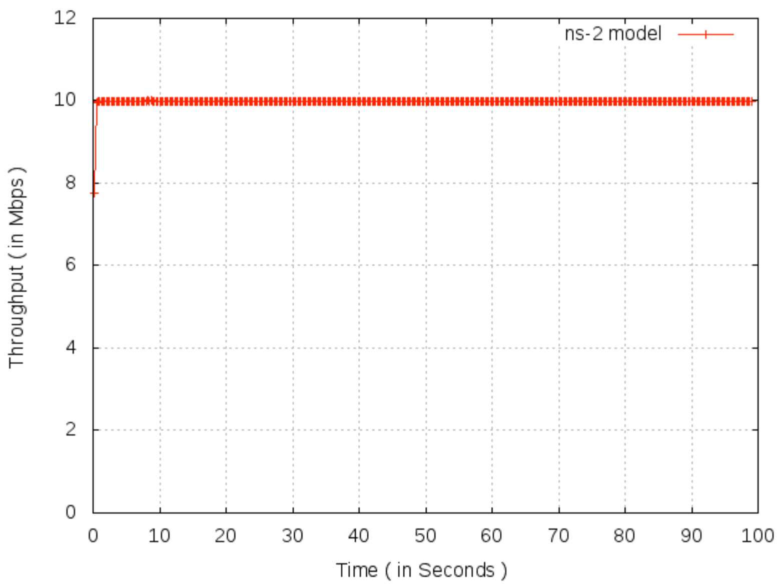
Functional verification: Heavy TCP traffic

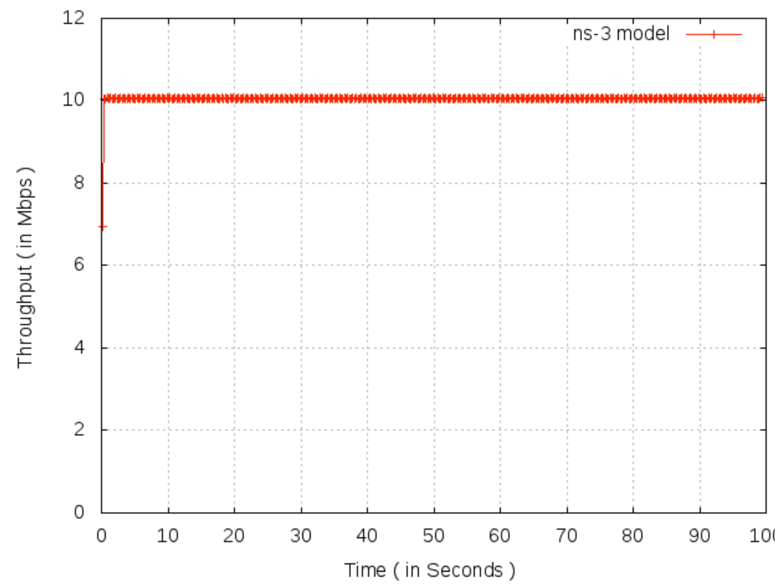


Functional verification: Mix TCP and UDP traffic

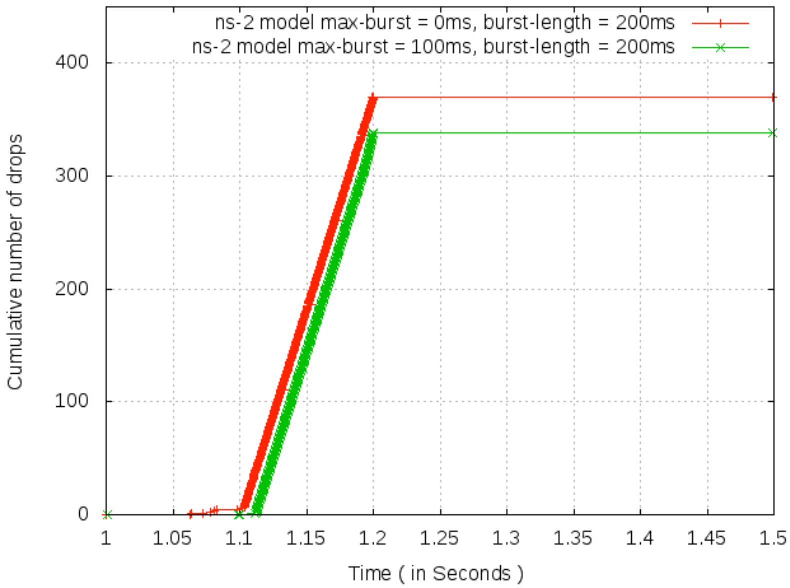


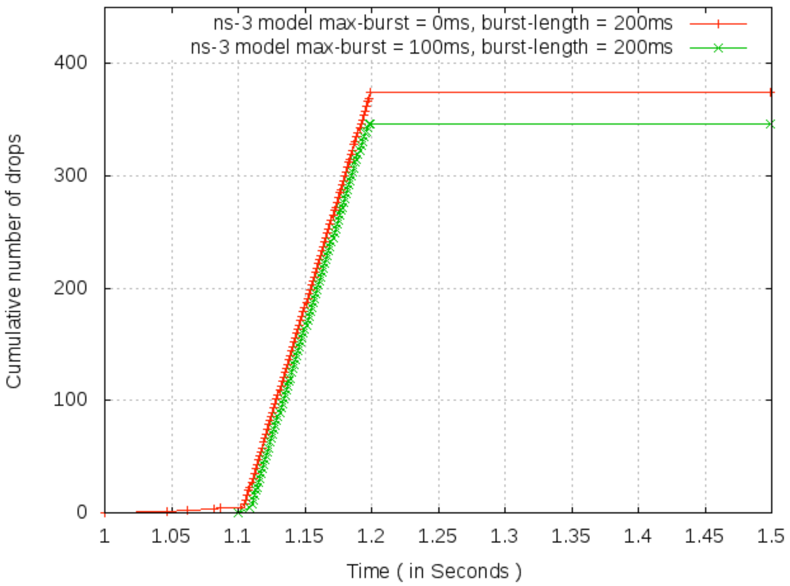






Functional verification: Bursty UDP traffic





Conclusions and the next goals

- A ns-3 model for PIE has been implemented and evaluated.
- Results obtained from ns-3 have been compared to those of ns-2.
- Steps to reproduce the results have been provided.

Next goals:

- A new traffic-control layer has been added since ns-3.25
- Port the current model to work with traffic-control layer (Completed!)
- Address the suggestions from the reviewers
- Merge it into the main line of ns-3

Acknowledgement

- Hajime Tazaki and all the reviewers, for the highly effective reviews!
- Tommaso Pecorella, for reviewing the PIE code and identifying bugs.
- Tom Henderson, for his excellent support and advice.
- Stefano Avallone, for reviewing our latest PIE code (works with tc layer).
- The entire ns-3 community.
- Our research group at NITK Surathkal.

Thank you.