

A Novel Parallel Traffic Control Mechanism for Cloud Computing

Zheng Li, Nenghai Yu, Zhuo Hao

MOE-Microsoft Key Laboratory of Multimedia Computing and Communication
University of Science and Technology of China

Outline

- Introduction
- Weaknesses of HTB
- Parallel HTB
- Experiments

Outline

- Introduction
- Weaknesses of HTB
- Parallel HTB
- Experiments

Traffic Control in Cloud Computing

- ❑ Control the outbound bandwidth
 - ✓ require an effective bandwidth management
 - ✓ traffic scheduler & shaper
- ❑ Hierarchical Service
 - ✓ idea of cloud computing
 - ✓ different service level
 - ✓ an attempt of customized SLAs on bandwidth
- ❑ A Contradiction
 - ✓ different service levels vs. user experience
 - ✓ a possible solution : HTB

Hierarchical Token Bucket

□ HTB

- ✓ a traffic control algorithm
- ✓ currently implemented in Linux kernel
- ✓ a module of TC (Traffic Control)

□ Basic idea

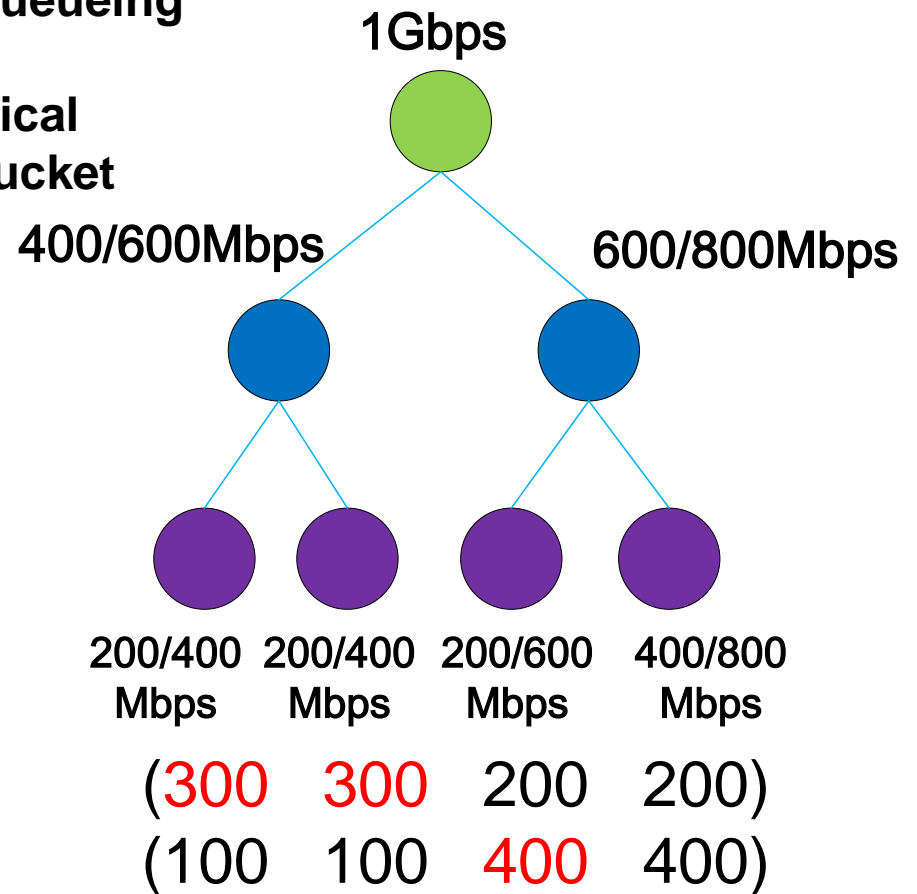
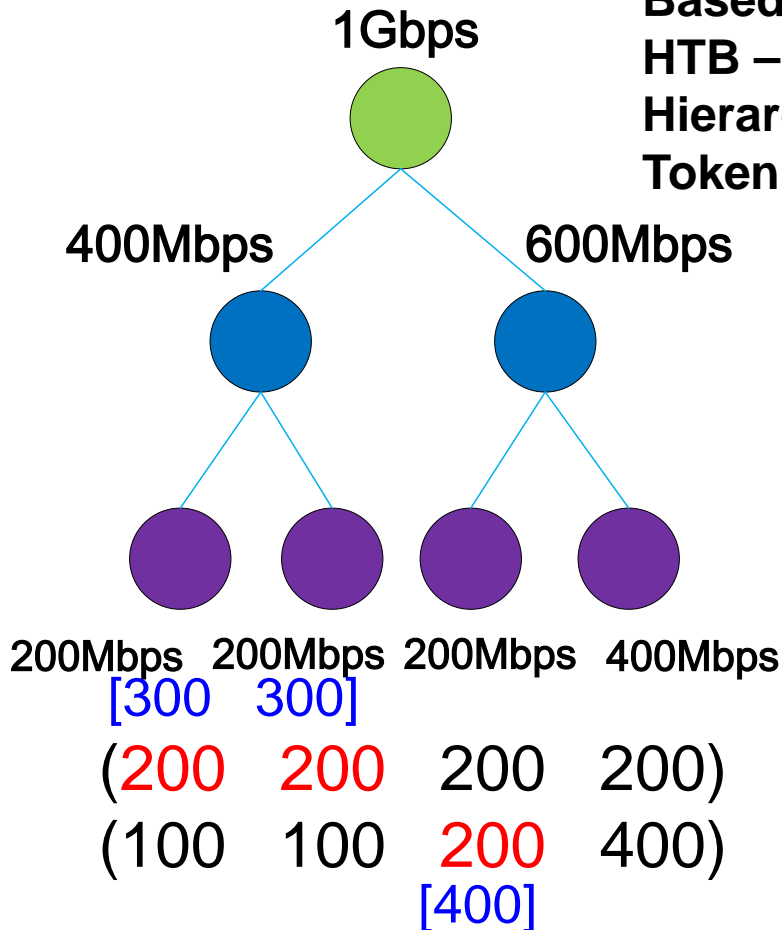
- ✓ bandwidth borrowing
- ✓ make full use of resource
- ✓ a solution for the contradiction
- ✓ hierarchical service & better user experience

CBQ vs. HTB

CBQ (AR)

CBQ – Class
Based Queueing
HTB –
Hierarchical
Token Bucket

HTB (AR/CR)



HTB allows bandwidth borrowing to break AR!

Outline

- Introduction
- Weaknesses of HTB
- Parallel HTB
- Experiments

Weaknesses of HTB

□ Processing speed

- ✓ 500Mbps at most
- ✓ not eligible for cloud computing

□ Reasons

- ✓ the inherent limitation of sequential program
- ✓ usage of spin-lock in kernel

Outline

- Introduction
- Weaknesses of HTB
- **Parallel HTB**
- Experiments

Basic Idea

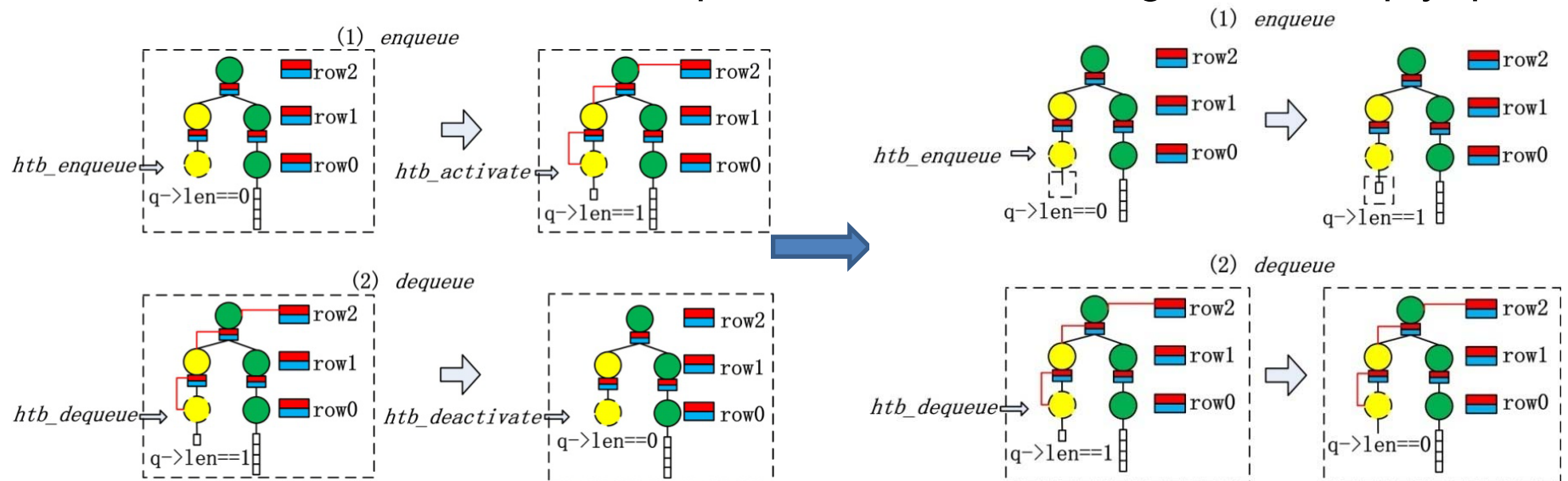
- Lock-free FIFOs based pipelining
 - ✓ port HTB from kernel to user space
 - ✓ based on multi-core architecture
 - ✓ try to eliminate necessity of using locks
 - ✓ reduce concurrency
 - ✓ selectively apply lock-free structures
 - ✓ make it run in a 1-way 2-stage pipeline fashion

Eliminate Locks

- Basic 2 operations of HTB: *enqueue* & *dequeue*
- Remove *htb_activate* and *htb_deactivate* in the 2 operations
- Critical region is reduced to only the packet queues
- A tradeoff: using locks but no empty queues

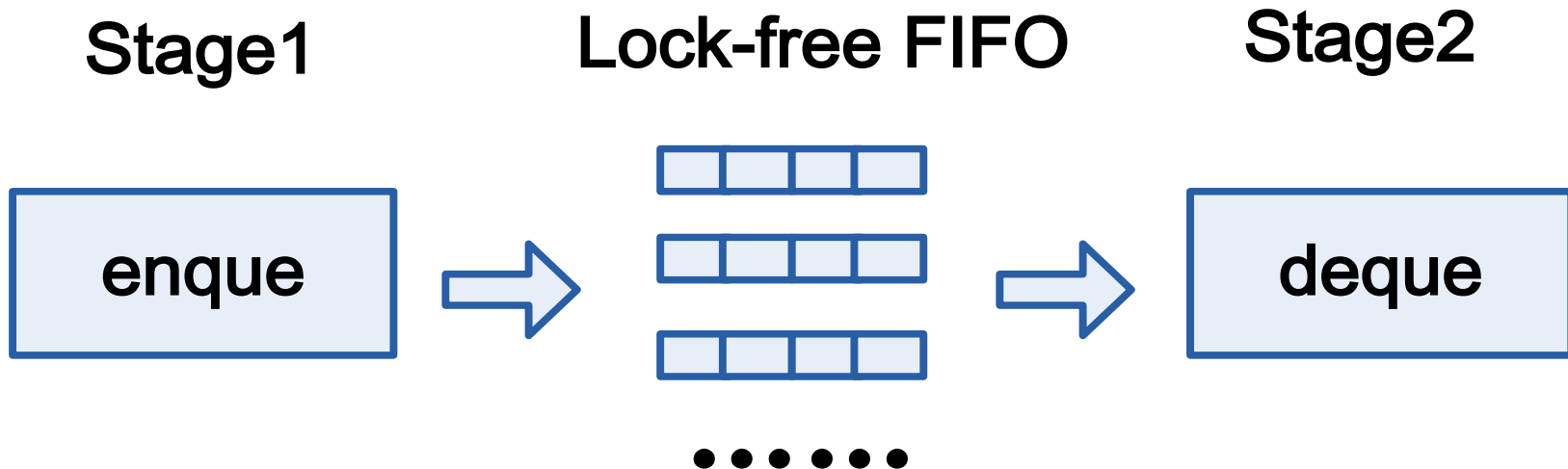
vs.

eliminate locks to parallelize HTB but might exist empty queues



Lock-free FIFOs

- ❑ Selectively used as the packet queue
- ❑ Eliminate time of lock/unlock operations
- ❑ Make it possible for HTB to run in a pipelined fashion
- ❑ We haven't adopted the advanced cache-line distance and cache-line aggregation techniques in [1], because unnecessary



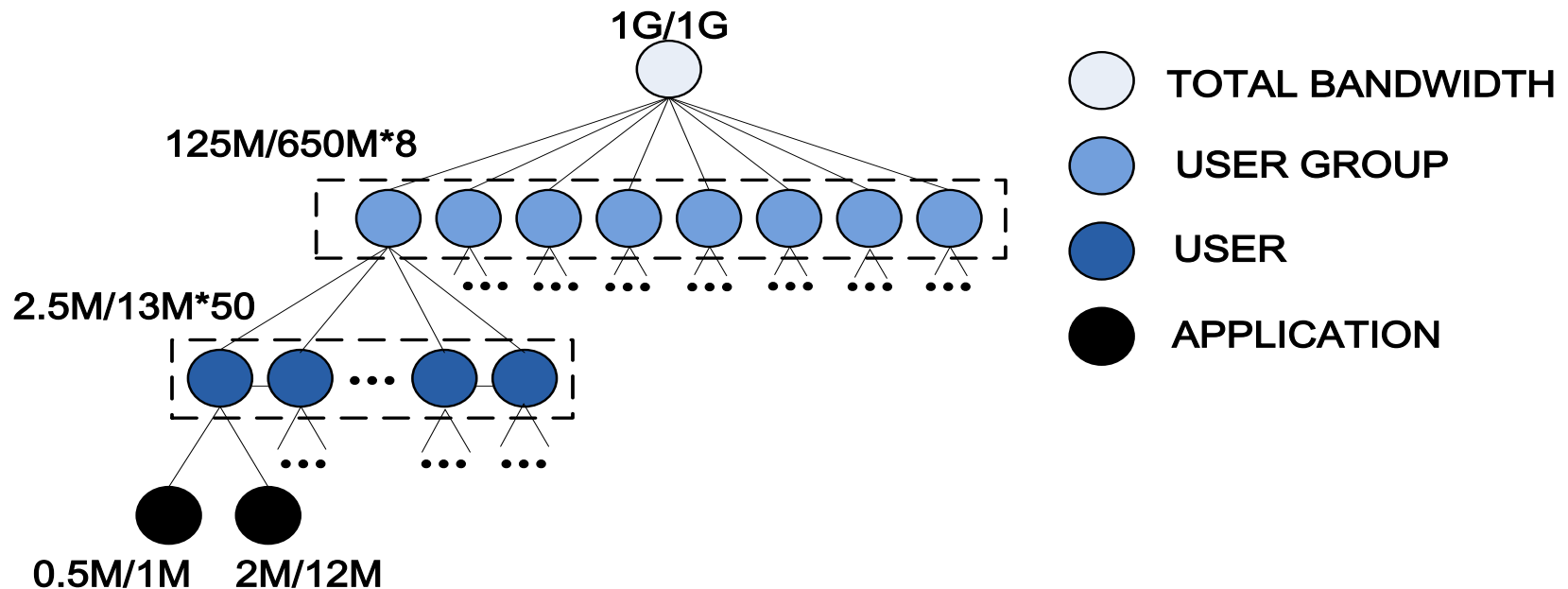
[1] J. Giacomoni, T. Moseley, and M. Vachharajani, "Fastforward for efficient pipeline parallelism: A cache-optimized concurrent lock-free queue", Proc. of PPoPP'08, New York, NY, USA, February 2008, pp.43-52

Outline

- Introduction
- Weaknesses of HTB
- Parallel HTB
- Experiments

Bandwidth Allocation

- ❑ 2 Scenarios: 1Gbps bandwidth & 2Gbps bandwidth
- ❑ The number of users of Scenario 2 are 2 times of that of Scenario 1
- ❑ Bandwidth for a user is 0.5Mbps/1Mbps and 2Mbps/12Mbps, for common service(require low band) and special service(require high band)
- ❑ Trace files are used in the experiments



Results

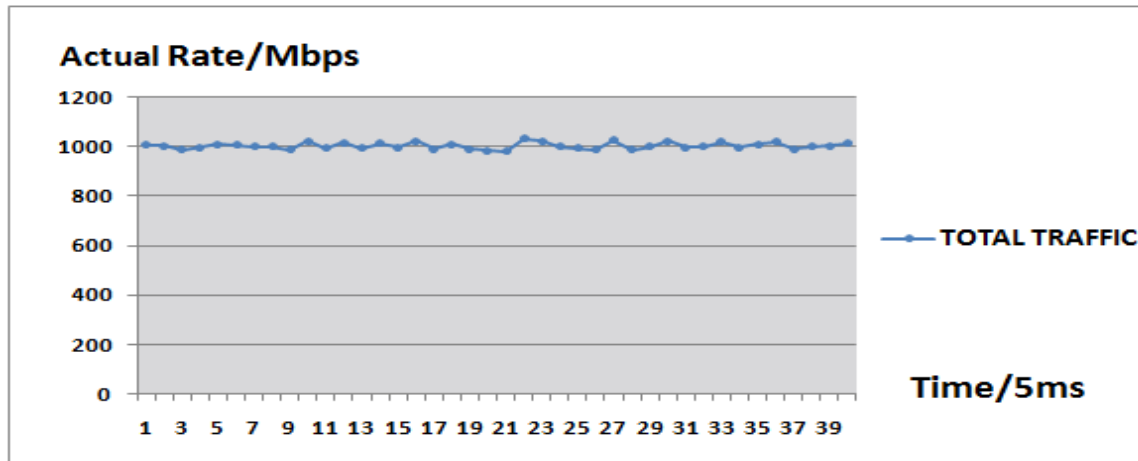
- Exp.1 ~ Exp.4: 1Gbps. Exp.5 ~ Exp.6: 2Gbps
- Exp.1: all users have traffics. Exp.2: 2/3 of users have traffics
- Exp.3 ~ Exp.4: 64B pkt len. Exp.3: use parallel HTB, Exp.4: use HTB
- Exp.5 :all users have traffics. Exp.6: 2/3 of users have traffics

FILE	#Packets	#Pkt Len.	#Max Len.	#Min Len.	#Traffic
File-1	2,397,696	782	1500	64	800
File-2	2,397,696	782	1500	64	533
File-3	9,765,925	64	64	64	800
File-4	4,795,392	782	1500	64	1600
File-5	4,795,392	782	1500	64	1067

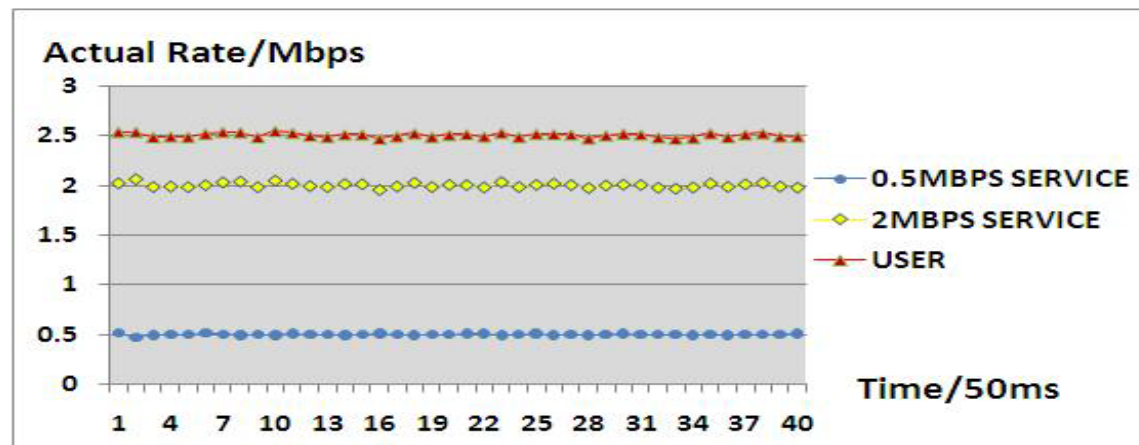
Exp.	#Trace	#MPPS	#Mbps	#Enq.	#Deq.
1	File-1	1.29	1008	0.39	0.54
2	File-2	1.29	1006	0.39	0.57
3	File-3	14.1	941	0.39	0.53
4	File-3	6.7	427	0.64	1.11
5	File-4	2.60	2033	0.39	0.54
6	File-5	2.59	2026	0.39	0.58

Parallel HTB can reach 2Gbps for common packet lengths, 300% improvement of the traditional HTB

Results



Output traffic rate of the total traffic



Output traffic rate of a selected user

THANKS!

QUESTIONS ARE

WELCOME