

Scheduling multicast packets in switches with multiple input- queues

Shashank Gupta Adnan Aziz
Andiamo UT Austin

Roadmap:

- Introduction: implement multicast
 - Motivation: performance, don't let switch be the bottleneck
 - Prior work: suffers from HOL blocking
- Proposed solution: VOQs
 - Scheduling algorithm
 - Queue assignment
- Results
 - Simulations on different sizes, traffic



Motivation

- Multicast traffic is increasing
 - Multimedia: A/V conferencing, broadcasting
 - Storage: replicating data
 - Cluster computing: shared data
- Existing protocols for network-level multicast
 - Multicast trees – minimize physical layer usage
- Multicast within a switch
 - Output-queuing infeasible (memories too slow)
 - Multiple unicast approach inefficient

9



Previous work: multicast

- [H+-TC-91] One-shot scheduling
 - terrible performance => must use fanout splitting
- [P+-JSAC-97] WBA
 - I/p-queued crossbar-based fabric
 - request, grant phases (details later)
 - Single queue at input => HOL blocking
- [C+-TON-00] window-based scheduling
 - Examine first K packets in queue
 - High complexity, less diversity => poor performance

9



Previous work: multicast

- [M+-II-01] Even with VOQ for each multicast address, can't achieve 100% throughput
 - Contrast with unicast case
- [A+-II-99] Computing good schedules for multicast is computationally hard
 - Online/offline, Exact/approx
- Above results largely of theoretical interest
 - Pathological traffic
 - $O(n)=NP=2\text{-EXPSPACE}=\text{nonprimitive recursive}$ for HW implementation

9



Previous work: unicast

- Input-queued switch
 - VOQs: one queue for each output at every input
 - Overcomes HOL blocking
- iSLIP scheduler
 - Idea: build "matching" from inputs to outputs
 - Each cycle: Request-Grant-Accept phases
 - Multiple iterations to increase throughput
- VOQs infeasible for multicast: 2^N queues

9



WBA (Multicast scheduling)

- Target Architecture
 - Input-queued crossbar-based switch
 - One queue for multicast packets at each input
- Algorithm
 - HOL packet sends **requests** to each output in its fanout set
 - weight: function of age and fanout
 - Each output **grants** input with highest weight
 - ties broken randomly
 - Note there's no accept phase!
- Problem: HOL blocking

9



Proposal

- Target architecture: Input-queued crossbar switch
 - Each input: small fixed set of queues
 - incoming packet inserted in a single queue
 - Each cycle: schedule one packet
 - can go to multiple destinations
- Need to resolve following technical issues:
 - How to schedule based on queue head state
 - How should packets be assigned to queues

9



Meta-algorithm

- **Request**

- Weight of HOL packet = $age - 2 * fanout$
 - Age: +ve coefficient for fairness
 - Fanout: -ve coefficient for residue concentration
- HOL packet in **each** queue: generate weighted requests for all its outputs
- Input port: for each output, forward **largest** weight request

- **Grant**

- Each output port selects largest weight request

9



Meta-algorithm

- **Accept**

- Each input port determines outputs that can be serviced
- Packet selected for transfer: maximizer of $N(outputs\ serviced) / N(outputs\ requested)$
- The outputs for this packet are *matched* by the input

- **Multiple iterations**

- Input may not accept all received grants
- Can repeat for better throughput

19



Example

Input 1	Q1a	Q1b	Input 2	Q2a	Q2b
	3 5 7	1 3 7		3 5	3 4 5 7
Age	2	1		2	1
Weight	-4	-5		-2	-7
(a - 2*f)					

Request

Outputs	1	2	3	4	5	6	7	8
Input1	-5		-4		-4		-4	
Input2			-2	-7	-2		-7	

Grant

Input1: 1, 7 Input2: 3, 4, 5

11



Example

Input 1	Q1a	Q1b	Input 2	Q2a	Q2b
	3 5 7	1 3 7		3 5	3 4 5 7
Age	2	1		2	1

Accept

- Input1 (grants = 1, 7)
 - Q1a : Service = 1, Request = 3, Ratio = 0.34
 - Q1b : Service = 2, Request = 3, Ratio = 0.67
 - Select Q1b, Accept outputs 1, 7
- Input2 (grants = 3, 4, 5)
 - Q2a : Service = 2, Request = 2, Ratio = 1.00
 - Q2b : Service = 3, Request = 4, Ratio = 0.75
 - Select Q2a, Accept outputs 3, 5

19



Queue assignment

- Random
 - Packets to same multicast group may get serviced out of order
 - Burst can block multiple HOLs
- Hash packets to queue based on address
 - Minimize HOL blocking: Send packets with similar destination addresses to same queue

19



Queue assignment

- “Majority” hash function
 - Statically assign a subset of outputs to each queue: “mask”
 - Packet is assigned to queue whose mask it best matches
- Mask selection (offline)
 - Size of mask close to average fanout
 - Distribute outputs randomly
 - Alternatively, based on *knowledge* of traffic
- Multi-level masking
 - To break ties use multiple masks
 - Can be done in parallel

19

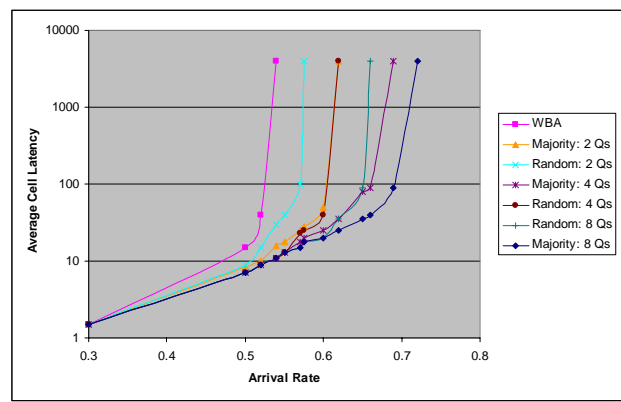


Summary of experiments tried

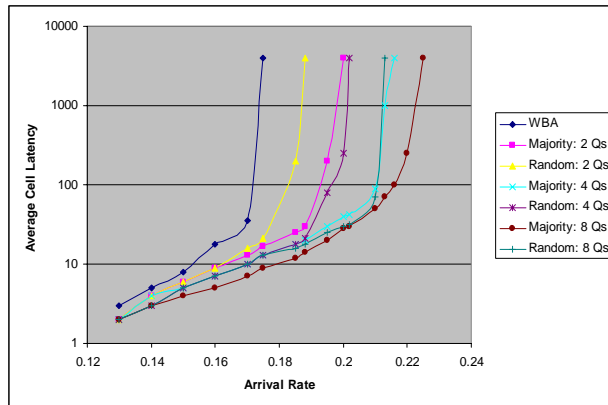
- Parameters:
 - Traffic patterns
 - Number of queues
 - Queue assignment schemes
 - Iterations of scheduling algorithm
- Results
 - 16% gain in throughput over WBA for 2x8 switch with 8 VOQs (for bursty traffic)
 - 20% gain in throughput over WBA for 8x8 switch with 8 VOQs (for bursty traffic)



2x8 switch



8x8 switch



99

99

19

Summary

- Can get significant improvement in performance
 - Small number of iterations provide good throughput
 - Even a small number of VOQs give high benefit
 - Majority scheme performs best
- Hardware implementation
 - Can implement efficiently
 - Logic-level description in MS writeup

19



Future Work

- *Adaptive* mask selection
- Integration with unicast scheduling
- Formal analysis and worst case performance