Routers with a Single Stage of Buffering

Authors: Sundar Iyer
Rui Zhang
Nick McKeown

Published: SIGCOMM 2002

Presenter: Charlie Wiseman
Discussion Leader: Mike Wilson
Single-Buffered?

- Generic CIOQ architecture

- Generic SB architecture

Motivation

• Not all routers use CIOQ
  - Shared memory
  - Load balanced
• Some SB routers not well studied
  - Distributed Shared Memory
• SB routers could be “better”
Performance Metrics

- Speedup Needed?
  - Not well defined for SB routers

- Memory Bandwidth Needed
  - Limiting factor for router capacity
  - Limiting factor for power
    - Battery backup
    - Heat dissipation
SB Classes

- Randomized SB
  - Randomized(!) switching
  - Load balanced
  - Statistical analysis
- Deterministic SB
  - Deterministic(!) switching
  - Constraint set analysis
- Paper considers Deterministic SB
Constraint Sets

- Pigeon Hole Principle
  - N leaving pigeons
  - N-1 other arriving pigeons
  - N-1 other pigeons with same departure
  - need at least 3N-1 pigeon holes
Constraint Sets

- For each packet
  - Determine packet's departure time
  - Determine constraint on each resource
    - Buffer
    - Switch fabric
    - Link
  - Apply pigeon hole principle
Parallel Shared Memory

- Use $k$ DRAMs instead of one large shared memory

Figure from "Routers with a Single Stage of Buffering", Iyer, Zhang, McKeown, SIGCOMM 2002.
PSM Router

- A PSM router can emulate a FCFS shared memory router
  - Theorem 1
  - 3NR total memory bandwidth needed
- A PSM router can (nearly) emulate a PIFO WFQ shared memory router
  - Theorem 2
  - 4NR total memory bandwidth needed
  - Only emulates within k-1 time slots
Theorem 1

- Total memory bandwidth is SNR
  - Need to minimize $S$
- Divide each time slot into $N$ decision slots
  - At each decision slot, one output can start reading memory and one input can start writing memory
  - Each read/write takes $\lceil k/S \rceil$ decision slots
Theorem 1

- BWS(t) is the set of memories busy writing at time t
  - Each write takes \([k/S]\) so BWS(t) is the set of memories that started a write within the previous \([k/S]-1\) slots
  - So, \(|BWS(t)| \leq [k/S]-1\)
- BRS(t) similar
Theorem 1

- Example, $N=4, k=5, S=3$
Theorem 1

- Cell c arrives at time t, going to output j, departure time d
- Let m be the memory where c is written
  - m can't be in BWS(t)
  - m can't be in BRS(t)
  - m can't be in BRS(d)
- \( k - |BWS(t)| - |BRS(t)| - |BRS(d)| > 0 \)
  - \( k - 3(\lceil k/S \rceil - 1) > 0 \)
  - \( S \geq 3 \)
• Consider a single PIFO queue

  - Adding cells pushes other cells' departure time back

  - Changing departure time introduces possible conflicts with N-1 cells leaving before and after it

Figure from “Routers with a Single Stage of Buffering“, Iyer, Zhang, McKeown, SIGCOMM 2002.
PSM, PIFO WFQ

• Now consider N PIFO queues
  – Each output has its own PIFO queue
  – Now every time a cell's departure time changes it conflicts with two new cells (in other queues)
    – Potentially unbounded conflicts!

• Need to rearrange the departure order to avoid these conflicts
PSM, PIFO WFQ

• Original order
  - In each of k time slots, one cell departs from each output port

• New order
  - The first k cells for each output depart, one output at a time
    - Time slot meaning has changed

• Each ordering has same overall length, but many cells will be out of ideal PIFO WFQ order
**PSM, PIFO WFQ**

- Example, $N=3, k=5$

<table>
<thead>
<tr>
<th>Original Order</th>
<th>New Order</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Time 1</strong></td>
<td>a1</td>
</tr>
<tr>
<td></td>
<td>b1</td>
</tr>
<tr>
<td></td>
<td>c1</td>
</tr>
<tr>
<td><strong>Time 2</strong></td>
<td>a2</td>
</tr>
<tr>
<td></td>
<td>b2</td>
</tr>
<tr>
<td></td>
<td>c2</td>
</tr>
<tr>
<td><strong>Time 3</strong></td>
<td>a3</td>
</tr>
<tr>
<td></td>
<td>b3</td>
</tr>
<tr>
<td></td>
<td>c3</td>
</tr>
<tr>
<td><strong>Time 4</strong></td>
<td>a4</td>
</tr>
<tr>
<td></td>
<td>b4</td>
</tr>
<tr>
<td></td>
<td>c4</td>
</tr>
<tr>
<td><strong>Time 5</strong></td>
<td>a5</td>
</tr>
<tr>
<td></td>
<td>b5</td>
</tr>
<tr>
<td></td>
<td>c5</td>
</tr>
</tbody>
</table>

- Note that cells departure time changes by at most $k-1$

- Also need buffers at each output!
Theorem 2

• Similar to Theorem 1

• Same constraints for time $t$
  - $m$ can't be in $BWS(t)$
  - $m$ can't be in $BRS(t)$

• New constraints for departure time
  - $m$ can't have $\left\lceil k/S \right\rceil - 1$ cells in front of $c$ in the PIFO queue for an output
  - $m$ can't have $\left\lfloor k/S \right\rfloor - 1$ cells following $c$ in the PIFO queue for an output

• 4 constraints $\Rightarrow S \geq 4$
Distributed Shared Memory

- Like PSM, but there are exactly N memories, one on each line card
  - Memories don't necessarily store packets going to or leaving from its associated line card
- Memory access control can be done in different ways
  - Bus based
  - Crossbar based
**Bus-based DSM**

- Bus-based DSM is exactly equivalent to PSM with N memories
  - All results for PSM apply
- Not practical
  - Bus demands too high

Figure from “Routers with a Single Stage of Buffering”, Iyer, Zhang, McKeown, SIGCOMM 2002.
**Crossbar DSM**

- NxN crossbar used to attach all outputs
  - Each packet traverses crossbar twice
  - Now also need crossbar scheduling algorithm
  - Otherwise, still like PSM with k=N

Figure from "Routers with a Single Stage of Buffering", Iyer, Zhang, McKeown, SIGCOMM 2002.
Crossbar DSM

• Crossbar-based DSM can emulate a FCFS shared memory router with 3NR total memory bandwidth and a crossbar speed of 6R
  - Consider crossbar schedule in two phases: departing and arriving packets
  - Theorem 1 applies
    • Need 3NR total memory bandwidth
    • Need 3R memory bandwidth per port (each port needs no more than 3 memory transfers per phase)
  - Crossbar must run at 3R/phase, 6R total
Crossbar DSM

- Crossbar-based DSM can emulate a PIFO WFQ shared memory router with total memory bandwidth 4NR, crossbar speed of 8R, within 2N-1 time slots
  - Same proof, with Theorem 2
- Can we get lower crossbar speeds?
Crossbar DSM

• Can still emulate FCFS shared memory router with 4R crossbar speed
  
  - Consider bi-partite graph from inputs to outputs
  
  - Only 3 memory operations for any port
  
  - Add 1 memory operation for any arriving or departing packets
  
  - Maximum degree of any vertex in graph is 4
Theorem 3

- Crossbar scheduling is equivalent to edge coloring the bi-partite graph
  - Each edge has unit weight
  - Max vertex degree is 4
  - König's Theorem shows that under these conditions, only 4 colors are needed
  - A crossbar schedule can be found that takes 4 rounds to complete request
  - Overall crossbar bandwidth is 4R
Crossbar DSM

- Can still emulate PIFO WFQ shared memory router with 5R crossbar speed, within 2N-1 slots
  - Combine ideas from Theorem 3 and Theorem 2 to get crossbar speed and added departure delay
Theorem 4

- Use the same idea of permuting the departure schedule as in PSM
  - Can't schedule the permutation as-is
    - N departures to one port per time slot
  - From Theorem 3, 4 memory accesses per port per time slot, or 4N per port over N time slots
  - 4N + N departures (arrivals) over N time slots
  - Max degree is 5N, or 5 per time slot
  - König, then leads to 5R crossbar speed
Theorem 4

- Maximum added delay is $2N - 1$
  - Break departure times up into batches of size $N$
  - When cell $a_1$ is about to leave, the next batch is fixed to be cells $a_1, a_2, \ldots, a_N$
  - If a new cell arrives that should go before $a_N$, it is pushed back into the next batch of $N$ departures
  - New 'a' could be pushed back to $a_{2N-1}$
Crossbar DSM

• Crossbar schedules from Theorems 3 and 4 aren't practical
  - Edge coloring is too complex
• Adding extra memory bandwidth can lead to more practical algorithms
  - May also need to slightly increase crossbar bandwidth
• Trade off between scheduler complexity and bandwidth
Crossbar DSM

- Crossbar-based DSM can emulate a FCFS shared memory router with total memory bandwidth 4NR and crossbar speed 4R
  - Theorem 5

- Crossbar-based DSM can emulate a PIFO WFQ router with total memory bandwidth 6NR and crossbar speed 6R within N-1 time slots
  - Theorem 6

- For each, crossbar first schedules all arriving packets, then all departing packets
Theorem 5

- Similar to Theorem 1
  - $S_w$ is the number of switch schedules devoted to writing cells each slot
  - $S_R$ is the number of switch schedules devoted to reading cells each slot
  - $BVWS(t)$ is the set of ports writing cells at time $t$
    - $|BVWS(t)| \leq \left\lfloor (N-1)/S_w \right\rfloor$
  - $BVRS(t)$ is the set of ports reading cells at time $t$
    - $|BVRS(t)| \leq \left\lfloor (N-1)/S_R \right\rfloor$
Theorem 5

- Cell c arrives at time t, going to output j, departure time d
- Let x be the line card memory where c is written
  - x can't be in BVWS(t)
  - Reads are separate, so no constraint
  - x can't be in BVRS(d)
  - As before, two constraints, so $S_R$ and $S_W$ must both be at least 2
  - Total crossbar speed must be 4R
  - Total memory bandwidth must be 4NR
Theorem 6

- Similar to Theorems 5 and 2
  - x can't be in BVWS(t)
  - Reads are separate, so no constraint
  - x can't have $\lceil N/S_R \rceil - 1$ cells in front of c in the PIFO queue for an output
  - x can't have $\lceil N/S_R \rceil - 1$ cells following c in the PIFO queue for an output
  - Three constraints (two read, one write)
    - $S_R \geq 3$, $S_W \geq 3$ or $S_R \geq 4$, $S_W \geq 2$
    - Either way, total crossbar bandwidth is 6R, total memory bandwidth is 6NR
Crossbar DSM

• Best analysis pointed to needed memory bandwidth of 3NR, crossbar bandwidth of 4R

  - There is a practical algorithm that meets these bounds, using a wave-front arbiter

  - Bi-partite graph has many useful properties

  - Given those and a request matrix for the crossbar, WFA can find a conflict free crossbar schedule

  - Lemma 3 and Theorem 7
Crossbar DSM

- Lemma 3
  - Ordering request matrix for WFA
  - Iterative method that maintains invariants over the matrix

- Theorem 7
  - Use the properties of the ordered request matrix with the properties of a bi-partite graph
  - Proof by contradiction
Combined PSM and DSM

• Basic DSM router, but each line card now has a set of memories
  - Line cards now function like PSM

• Theorem 8
  - $2h-1$ memories with rate $R/h$ emulate one memory of rate $R$ for a FCFS DSM router

• Theorem 9
  - $3h-2$ memories with rate $R/h$ emulate one memory of rate $R$ for a PIFO WFQ DSM router
Practical Considerations

- Is it feasible to build a router (PIFO WFQ DSM) with so many memories?
  - CIOQ router with speedup 2 needs 6NR total memory bandwidth, 2N memories
  - PIFO WFQ DSM needs 4NR total memory bandwidth, N memories

- What about crossbar bandwidth?
  - CIOQ router only needs 2R
  - PIFO WFQ DSM needs 5R
  - This is a problem. Increasing 'speedup' is not cheap
Practical Considerations

• Need to know packet departure times
  - Same problem for CIOQ as for DSM
  - Would like a distributed mechanism

• Complexity of memory management
  - Another problem here
  - Lots of centralized information needed
  - $O(N)$ operations to schedule $N$ new arrivals
Practical Considerations

- Reduce complexity with batch scheduling
  - Instead of operating on a packet (cell) level, let packets collect into frames and schedule frames
  - Reduces number of decisions made and so reduces overall time spent scheduling

- Does CIOQ or DSM require larger buffers?
  - DSM buffers share all flows, so no one can be a bottleneck
  - Can lead to smaller individual buffers
Conclusion

- PIFO WFQ DSM has overall lower memory requirements than PIFO CIOQ
  - Fewer memories, smaller memories
- PIFO WFQ DSM has higher crossbar speed requirements than PIFO CIOQ
- PIFO WFQ DSM has practicality problems
  - Too much global state needed
  - Schedulers are sequential
Discussion

• Questions?
• Problems?
• Issues?