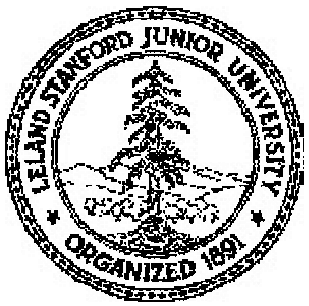
Load-Balancing and Parallelism for the Internet

Stanford University Ph.D. Oral Examination
Tuesday, Feb 18th 2003

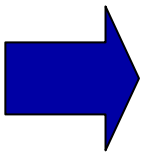


Sundar Iyer (sundaes@cs.stanford.edu)
Department of Computer Science
Stanford University,
http://www.stanford.edu/~sundaes

Motivation

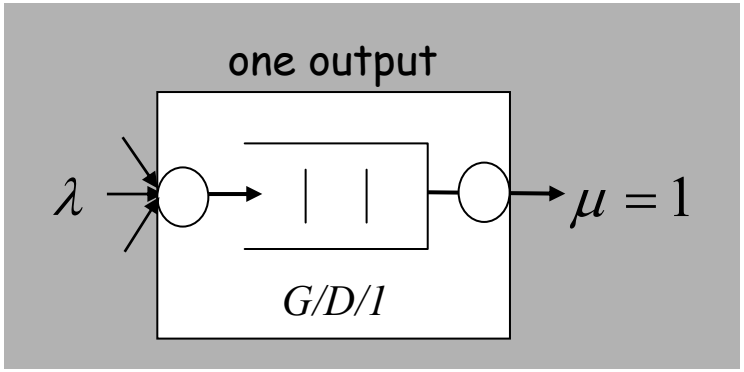
- Build routers with performance guarantees
- Guarantees are
  - Statistical (100% throughput)
  - Deterministic (work-conserving, delay guarantees)
- Hard to build big routers with guarantees
- This talk:
  - Deterministic guarantees
  - Use load-balancing and parallelism

Contents



1. Parallel routers: work-conserving
2. Parallel routers: delay guarantees
3. Parallel packet buffers
4. Summary of contributions

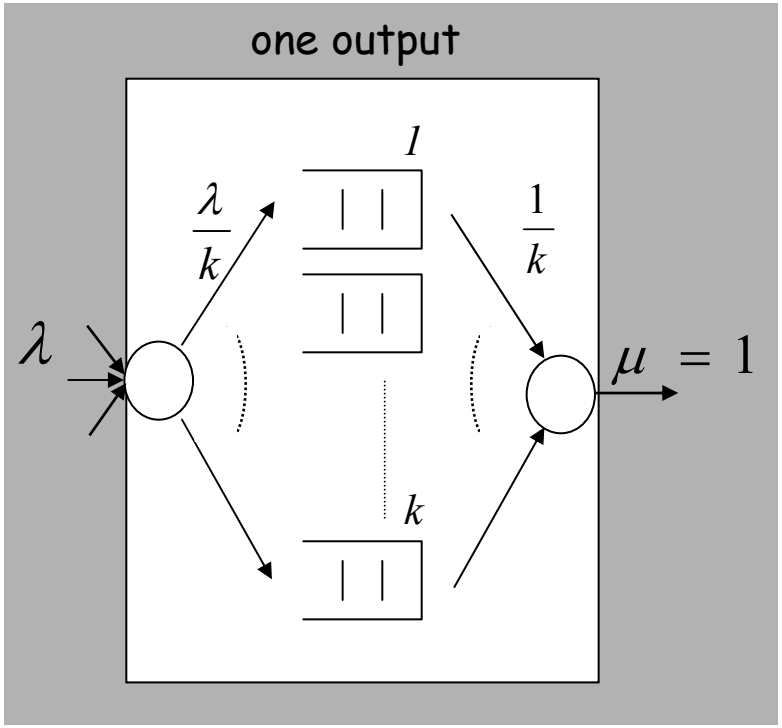
Output-Queued Router

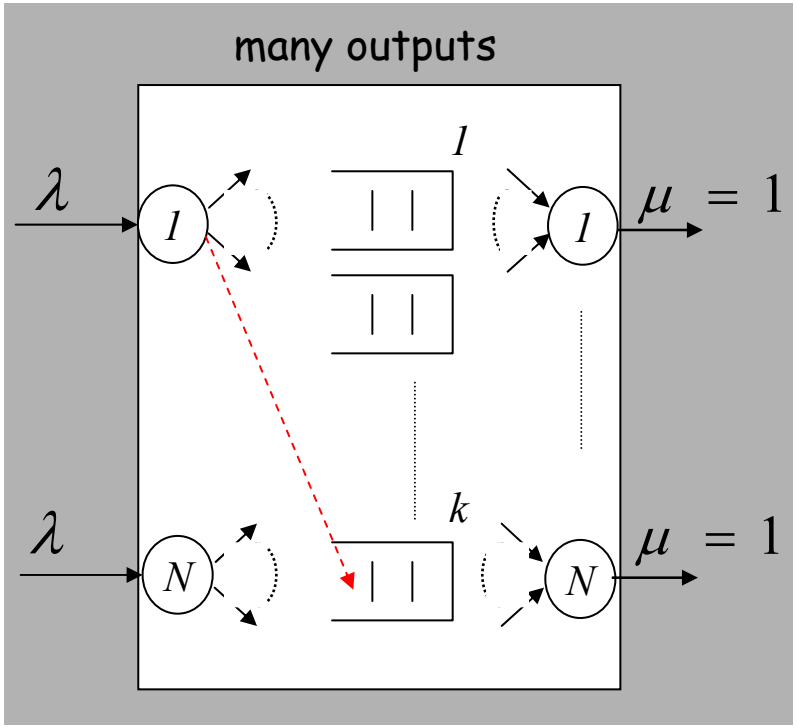


Gives 100% throughput if $\lambda < 1$ and output work conserving

- Work-conservation : $\Rightarrow$ 100% throughput, minimizes delay, delay guarantees possible
- Problem : Memory bandwidth not scalable
- Previous attempts : Very complicated or ad hoc
- My approach : Deterministic parallel work-conserving output-queued router

Parallel Output-Queued Router





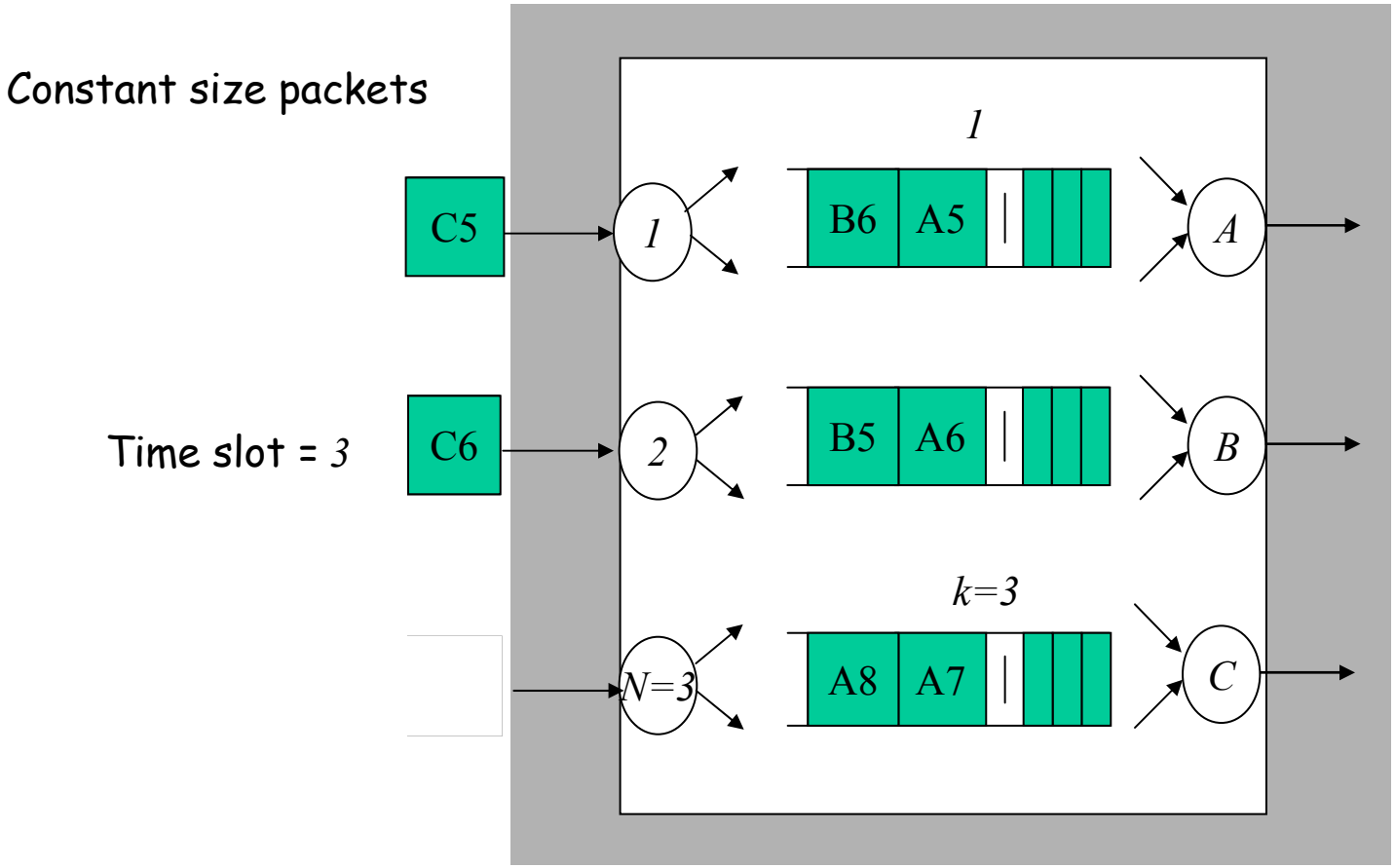
Gives 100% throughput if $\frac{\lambda}{k} < \frac{1}{k}$

and output work conserving

Is this work conserving?

Parallel Output-Queued Router

(May not be work-conserving)



At most two memory operations per time slot: 1 write and 1 read

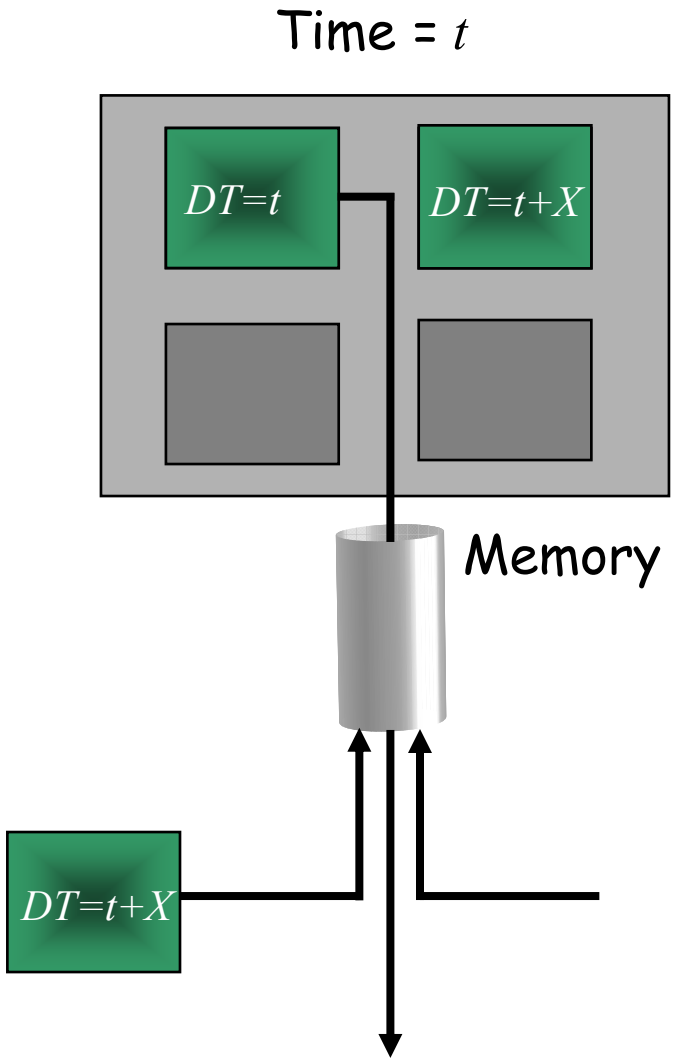
Problem

- Problem :
  - How can we design a parallel output-queued work-conserving router from slower parallel memories?
- Theorem 1: (sufficiency)
  - A parallel output-queued router is work-conserving with $3N-1$ memories that can perform at most one memory operation per time slot

Re-stating the Problem

- There are K cages which can contain an infinite number of pigeons.
- Assume that time is slotted, and in any one time slot
  - at most N pigeons can arrive and at most N can depart.
  - at most 1 pigeon can enter or leave a cage via a pigeon hole.
  - the time slot at which arriving pigeons will depart is known
- For any router:
  - What is the minimum K, such that all N pigeons can be immediately placed in a cage when they arrive, and can depart at the right time?

Intuition for Theorem 1



- Only one packet can enter a memory at time t
- Only one packet can enter or leave a memory at time t
- Only one packet can enter or leave a memory at any time

Proof of Theorem 1

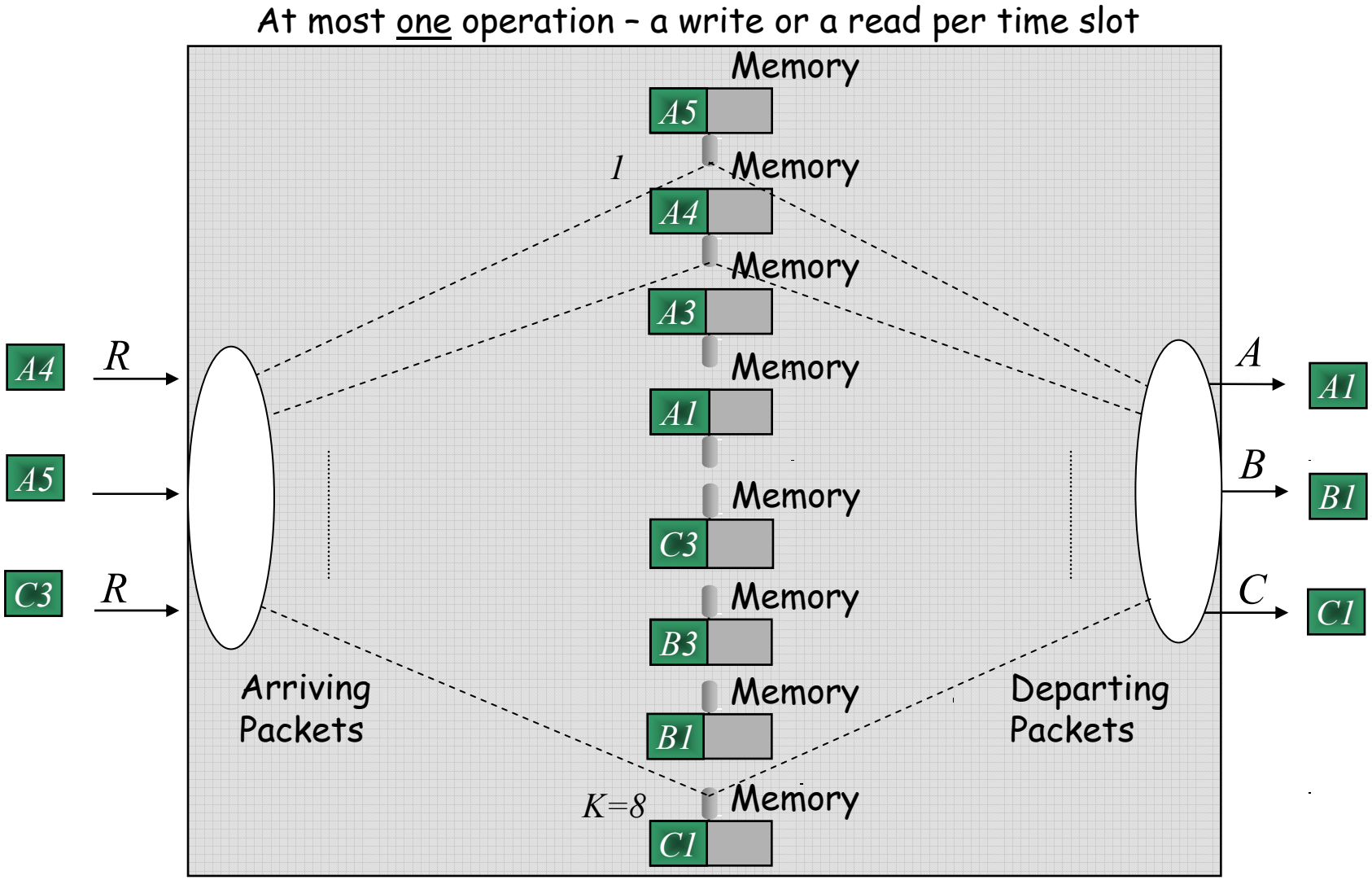
- When a packet arrives in a time slot it must choose a memory not chosen by

  1. The $N-1$ other packets that arrive at that timeslot.
  2. The N other packets that depart at that timeslot.
  3. The $N-1$ other packets that can depart at the same time as this packet departs (in future).

- Proof:

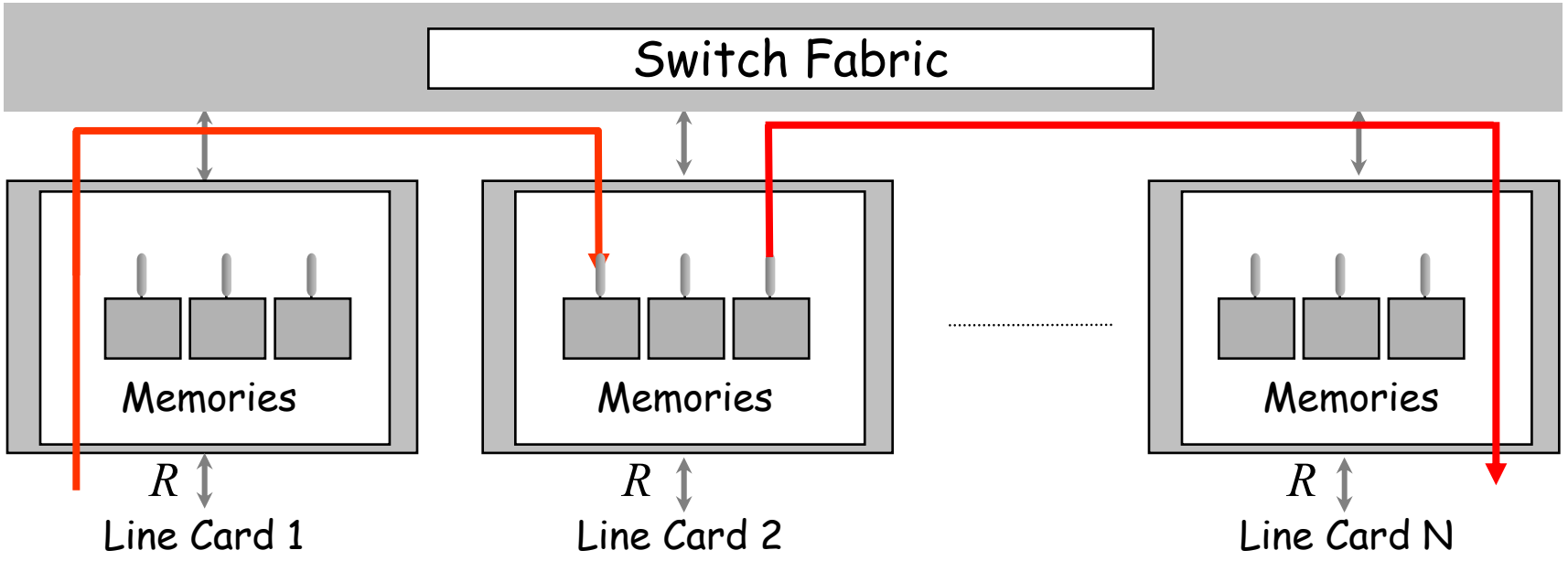
  - By the pigeon-hole principle, $3N-1$ memories that can perform at most one memory operation per time slot are sufficient for the router to be work-conserving

The Parallel Shared Memory Router



From theorem 1, $k = 7$ memories don't suffice .. but 8 memories do

Distributed Shared Memory Router



- The central memories are moved to distributed line cards and shared
- Memory and line cards can be added incrementally
- From theorem 1, $3N-1$ memories which can perform one operation per time slot i.e. a total memory bandwidth of $\cong 3NR$ suffice for the router to be work-conserving

Corollary 1

- Problem:
  - What is the switch bandwidth for a work-conserving DSM router?
- Corollary 1: (sufficiency)
  - A switch bandwidth of $4NR$ is sufficient for a distributed shared memory router to be work-conserving
- Proof 1:
  - There are a maximum of 3 memory accesses and 1 port access

Corollary 2

- Problem:
  - What is the switching algorithm for a work-conserving DSM router?
    - Bus : No algorithm needed, but impractical
    - Crossbar : Algorithm needed because only permutations are allowed
- Corollary 2: (existence)
  - An edge coloring algorithm can switch packets for a work-conserving distributed shared memory router
- Proof :
  - Follows from König's theorem - Any bipartite graph with maximum degree Δ has an edge coloring with Δ colors

Summary - Routers which give 100% throughput

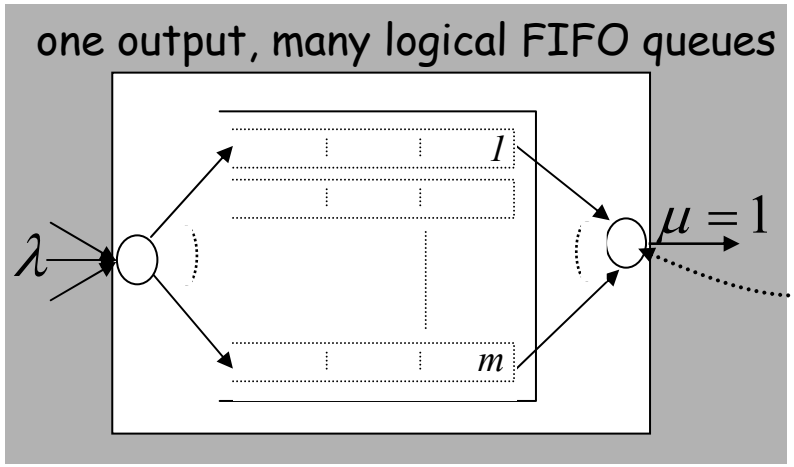
	Fabric	# Mem.	Mem. BW per Mem[1]	Total Mem. BW	Switch BW	Switch Algorithm
Output-Queued	Bus	N	(N+1)R	N(N+1)R	NR	None
C. Shared Mem.	Bus	1	2NR	2NR	2NR	None
Input Queued	Crossbar	N	2R	2NR	NR	MWM
CIOQ (Cisco)	**Crossbar**	2N	3R	6NR	2NR	Maximal
		2N	**3R**	**6NR**	**3NR**	**Time Reserve***
P Shared Mem.	**Bus**	**k**	**3NR/k**	**3NR**	**2NR**	**C. Sets**
DSM (Juniper)	**Xbar**	**N**	**3R**	**3NR**	**4NR**	**Edge Color**
		N	**3R**	**3NR**	**6NR**	**C. Sets**
		N	**4R**	**4NR**	**4NR**	**C. Sets**
PPS - OQ	**Clos**	**Nk**	**2R(N+1)/k**	**2N(N+1)R**	**4NR**	**C. Sets**
PPS –Shared Memory	**Clos**	**Nk**	**4NR/k**	**4NR**	**4NR**	**C. Sets**
		Nk	**2NR/k**	**2NR**	**2NR**	**None**

[1] Note that lower mem. bandwidth per memory implies higher random access time, which is better

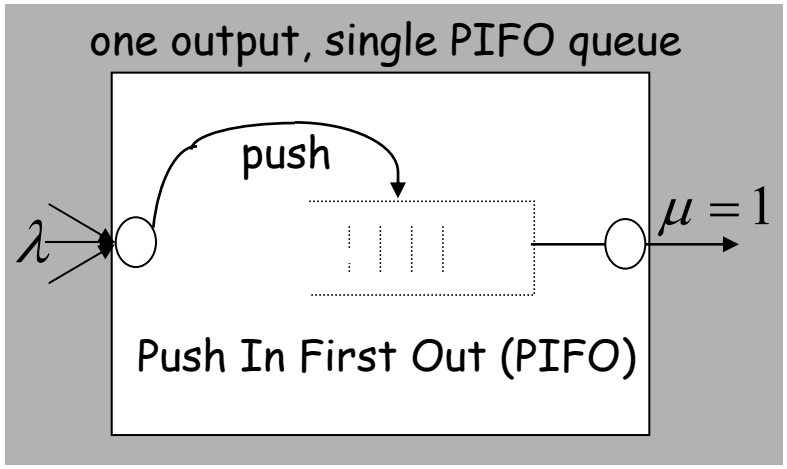
Contents

1. Parallel routers: work-conserving
2. Parallel routers: delay guarantees
3. Parallel packet buffers
4. Summary of contributions

Delay Guarantees



Weighted fair queueing sorts packets



PIFO models

- Weighted Fair Queueing
- Weighted Round Robin
- Strict priority etc.

Delay Guarantees

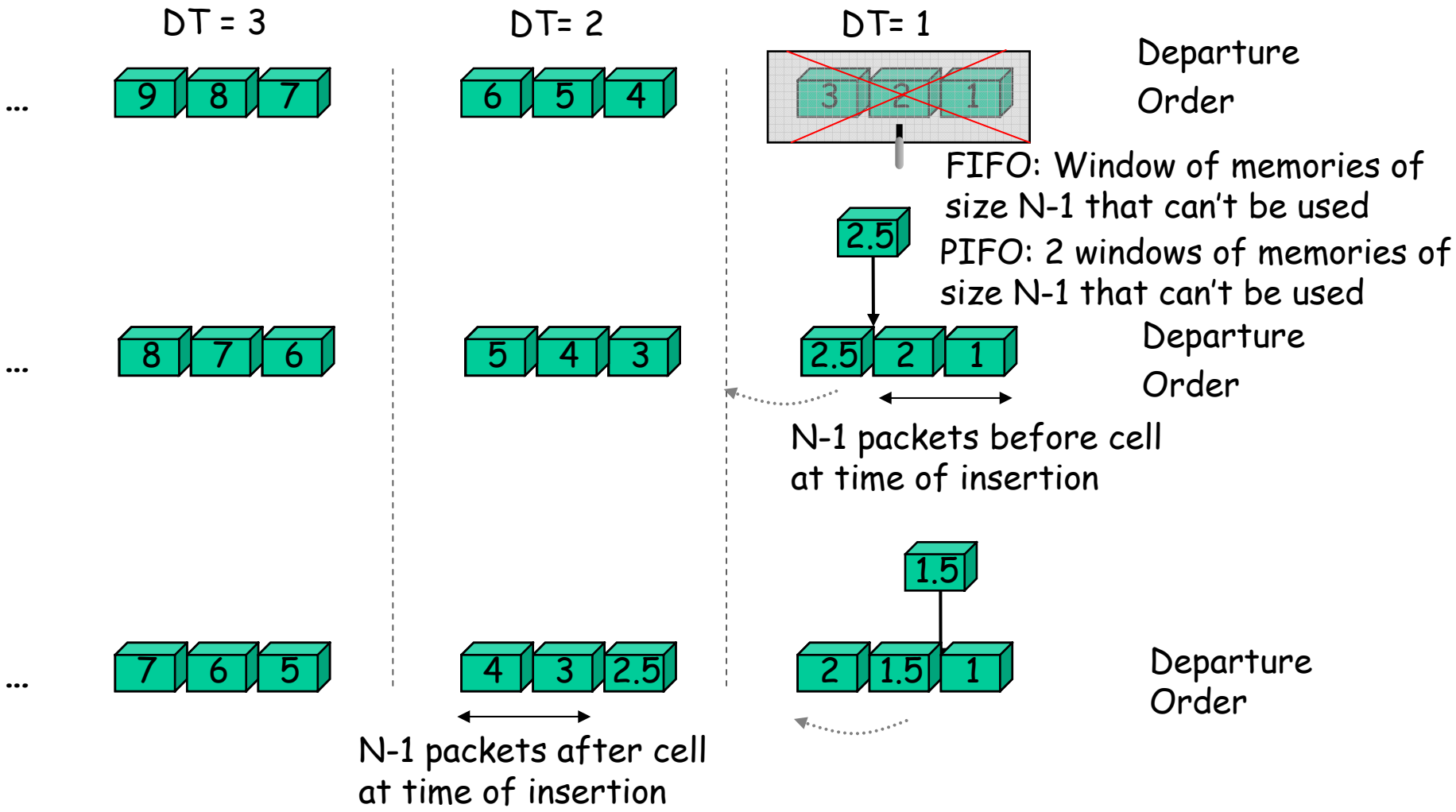
- Problem :
  - How can we design a parallel output-queued router from slower parallel memories and give delay guarantees?
- This is difficult because
  - The counting technique depends on being able to predict the departure time and schedule it.
  - The departure time of a cell is not fixed in policies such as strict priority, weighted fair queueing etc.

Theorem 2

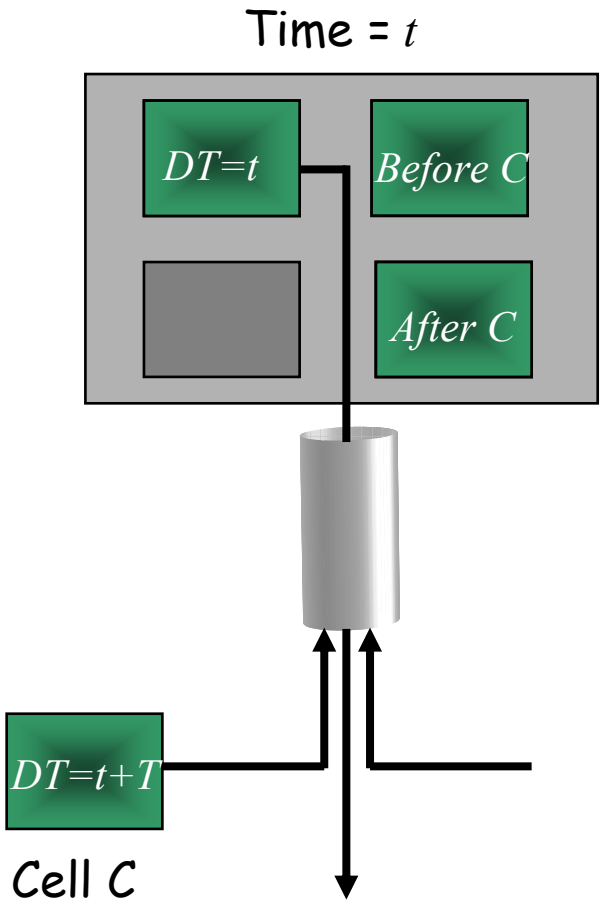
- Theorem 2: (sufficiency)
  - A parallel output-queued router can give delay guarantees with $4N-2$ memories that can perform at most one memory operation per time slot.

Intuition for Theorem 2

$N=3$ port router



Proof of Theorem 2



A packet cannot use the memories:

1. Used to write the $N-1$ arriving cells at t.
2. Used to read the N departing cells at t.

3. Will be used to read the $N-1$ cells that depart before it.
4. Will be used to read the $N-1$ cells that depart after it.

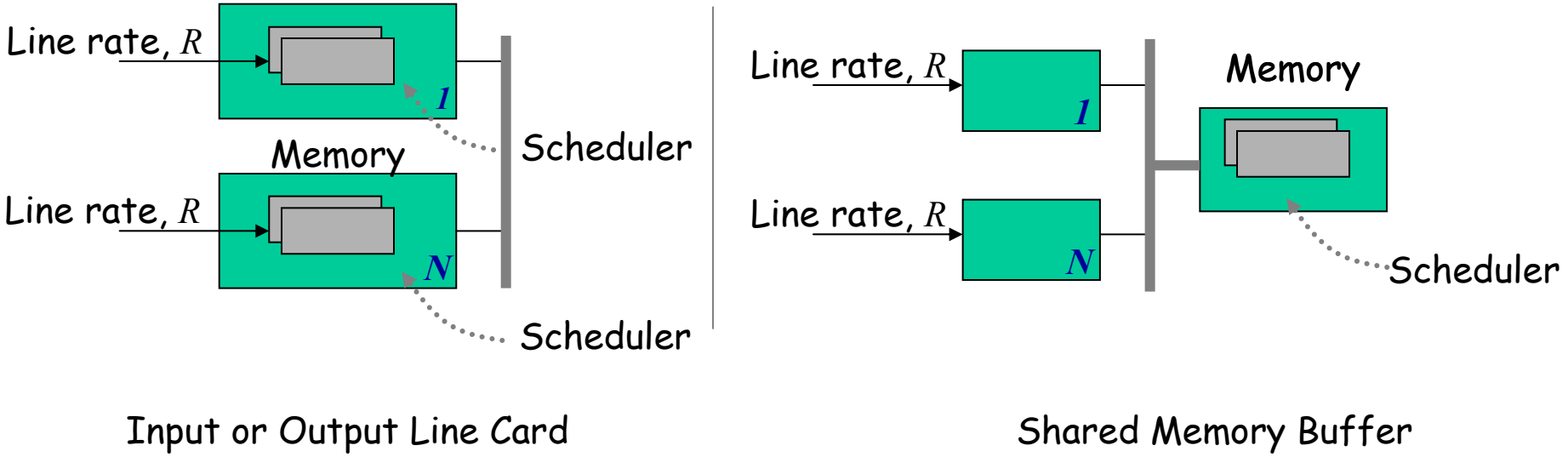
Summary- Routers which give delay guarantees

	Fabric	# Mem.	Mem. BW per Mem.	Total Memory BW	Switch BW	Switch Algorithm
Output-Queued	Bus	N	(N+1)R	N(N+1)R	NR	None
C. Shared Mem.	Bus	1	2NR	2NR	2NR	None
Input Queued	Crossbar	N	2R	2NR	NR	-
CIOQ (Cisco)	**Crossbar**	2N	3R	6NR	2NR	Marriage
		2N	**3R**	**6NR**	**3NR**	**Time Reserve**
P. Shared M	**Bus**	**k**	**4NR/k**	**4NR**	**2NR**	**C. Sets**
DSM (Juniper)	**Xbar**	**N**	**4R**	**4NR**	**5NR**	**Edge Color**
		N	**4R**	**4NR**	**8NR**	**C. Sets**
		N	**6R**	**6NR**	**6NR**	**C. Sets**
PPS - OQ	**Clos**	**Nk**	**3R(N+1)/k**	**3N(N+1)R**	**6NR**	**C. Sets**
PPS –Shared Memory	**Clos**	**Nk**	**6NR/k**	**6NR**	**6NR**	**C. Sets**
		Nk	2NR/k	2NR	2NR	-

Contents

1. Parallel routers: work-conserving
2. Parallel routers: delay guarantees
3. Parallel packet buffers
4. Summary of contributions

Packet Buffering



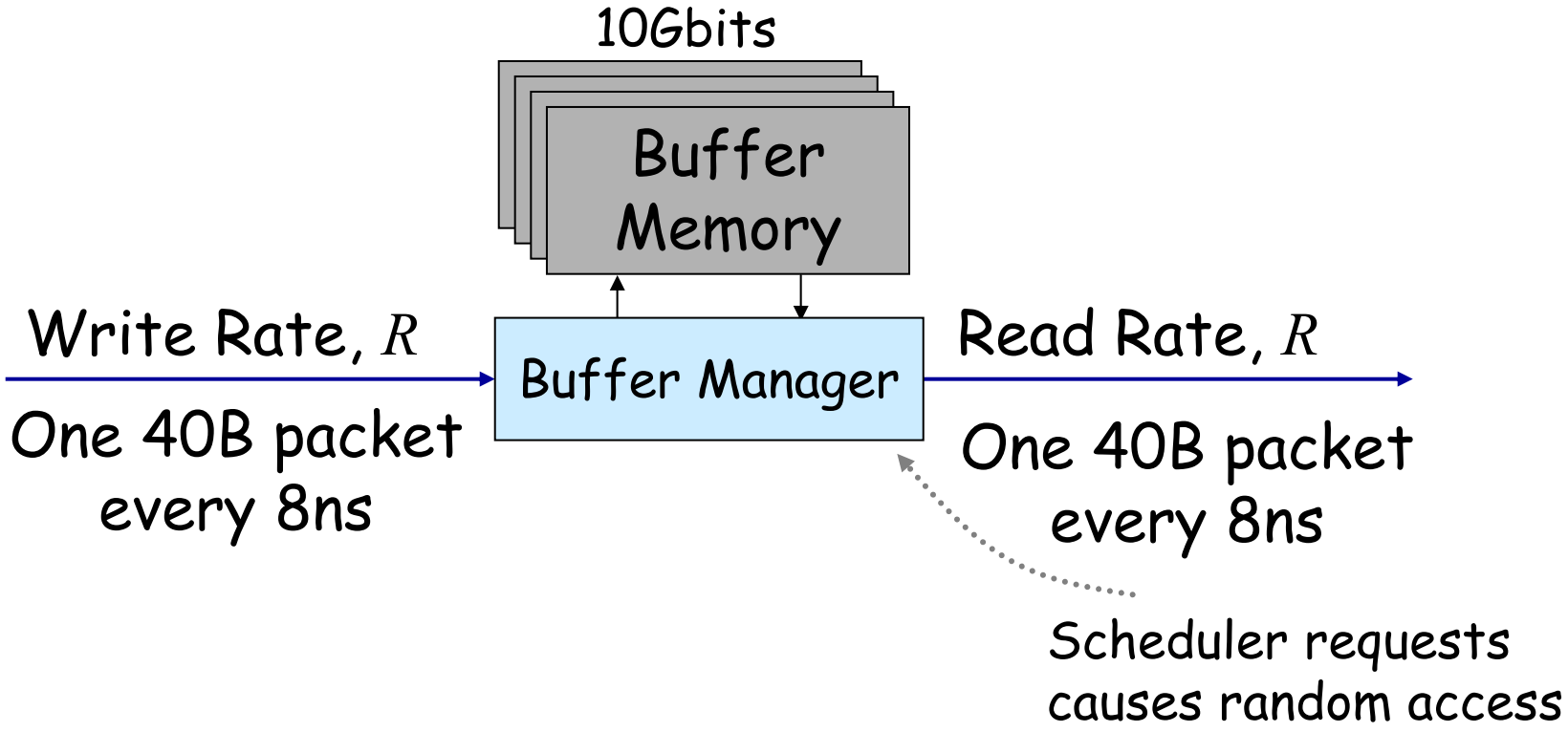
Input or Output Line Card

Shared Memory Buffer

- **Big:** For TCP to work well, the buffers need to hold one RTT (about 0.25s) of data.
- **Fast:** Clearly, the buffer needs to store (retrieve) packets as fast as they arrive (depart).

An Example

Packet buffers for a 40Gb/s line card



Problem is solved if a memory can be (random) accessed every 4 ns and store 10Gb of data

Available Memory Technology

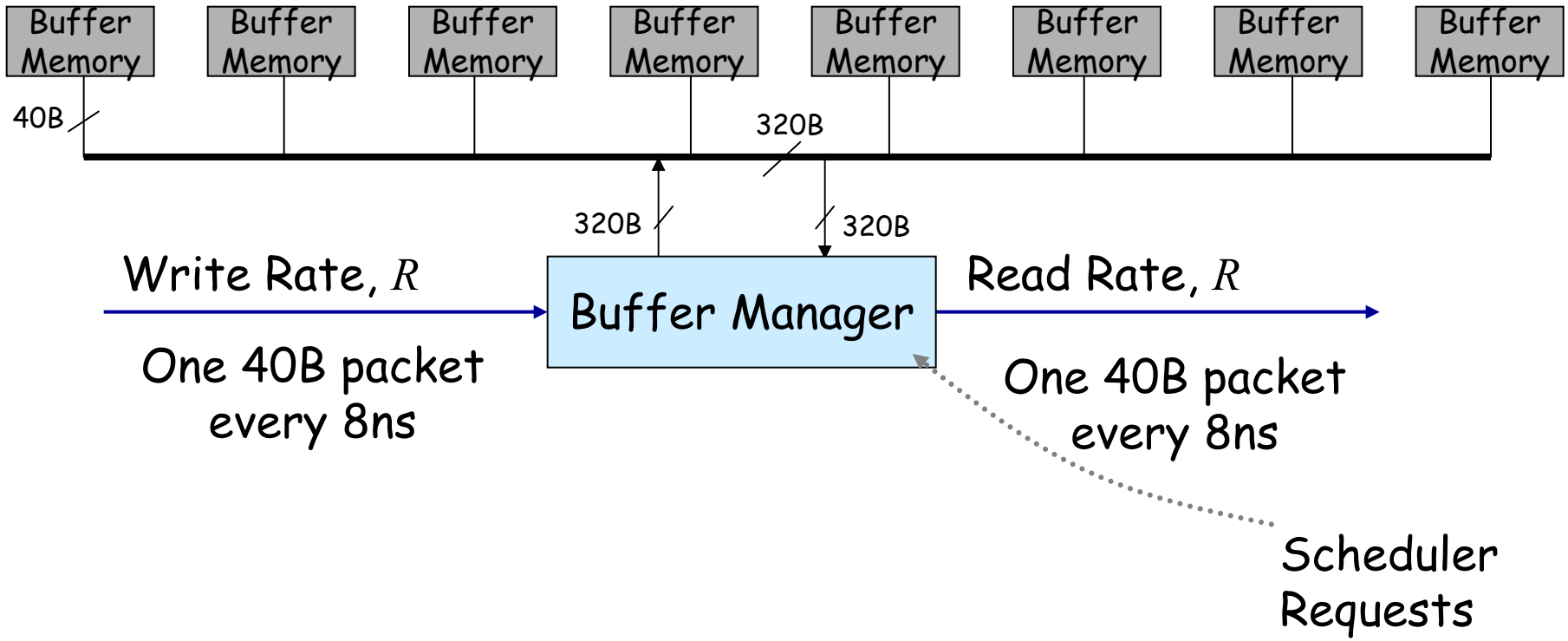
- Use SRAM?
  - \+ Fast enough random access time, but
  - \- Too low density to store 10Gbits of data.
- Use DRAM?
  - \+ High density means we can store data, but
  - \- Can't meet random access time.

Problem

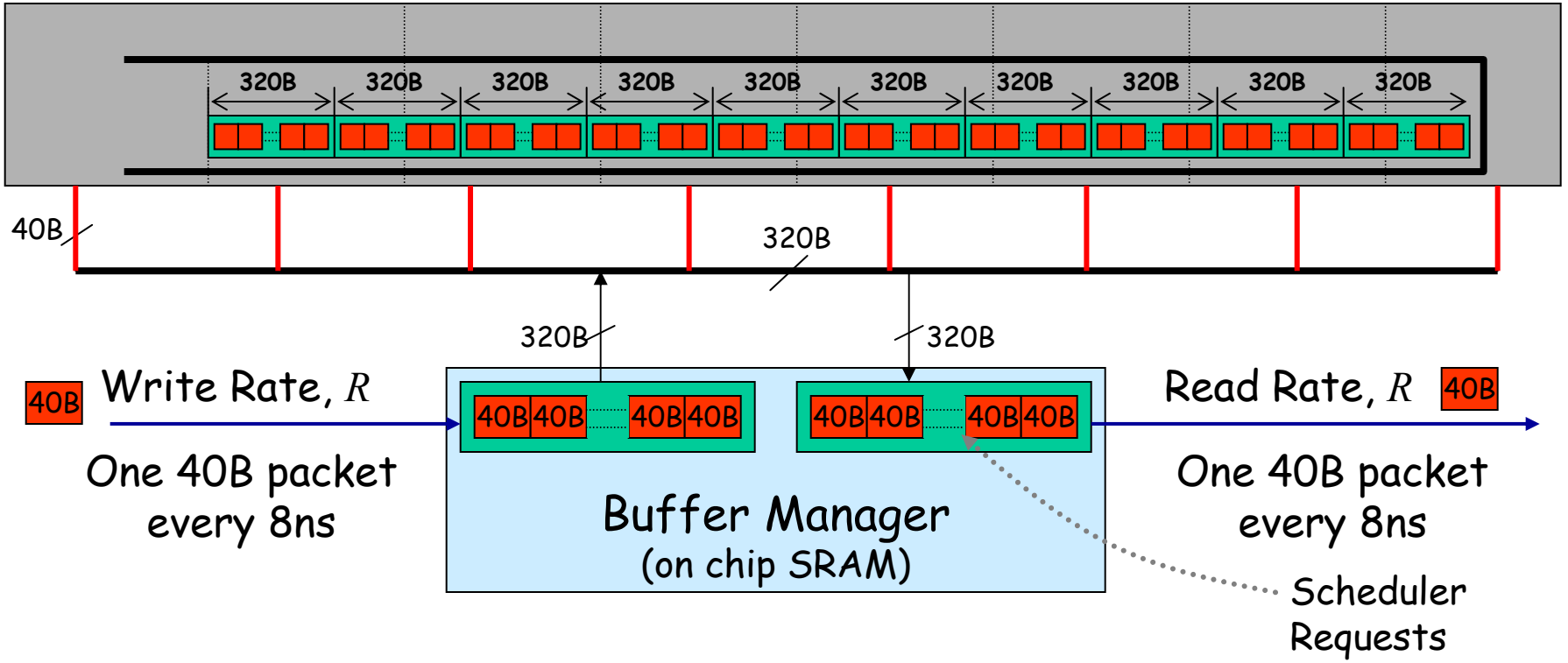
Problem:

How can we design high speed packet buffers from commodity available memories?

Can't we just use lots of DRAMs in parallel?



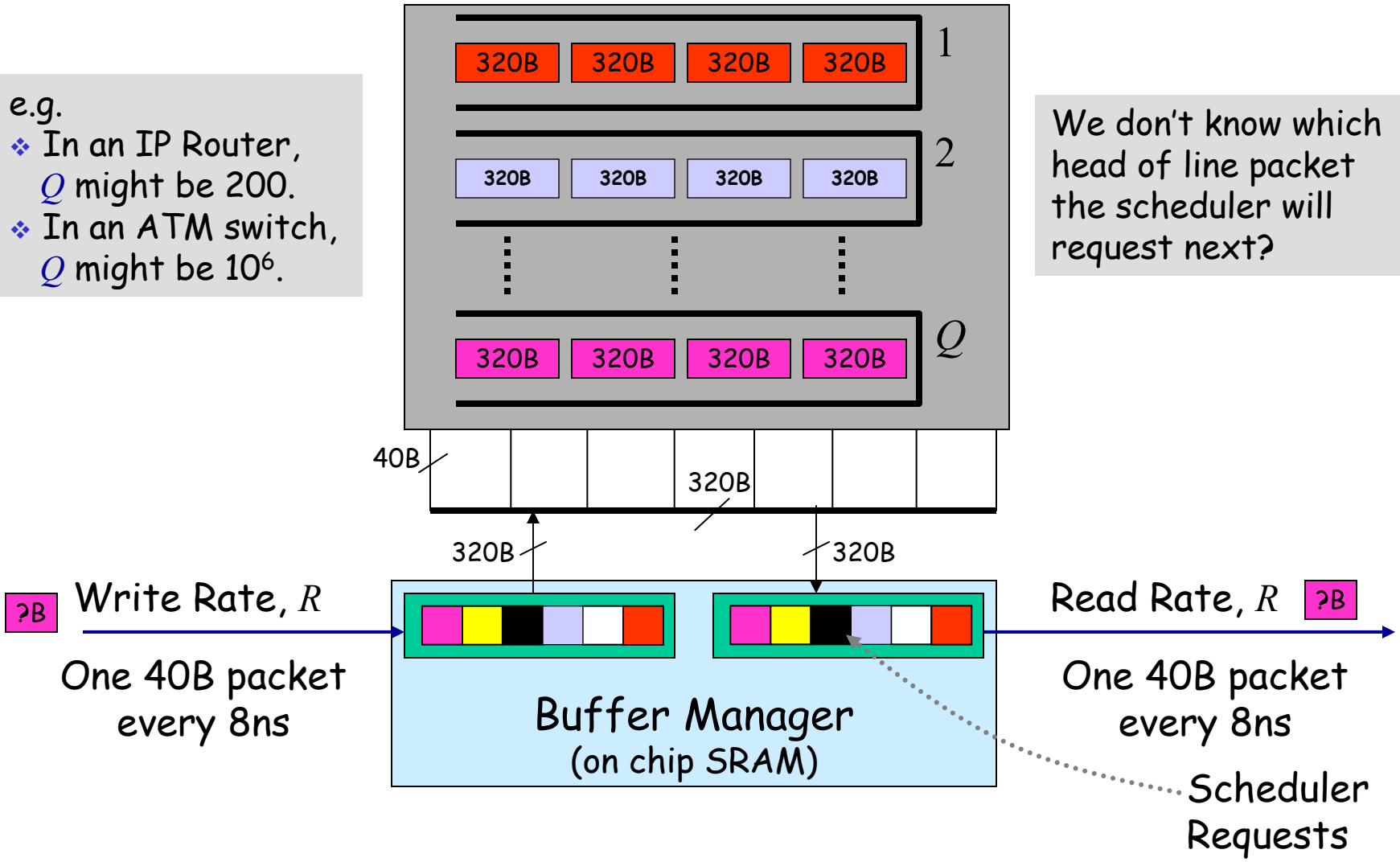
Works fine if there is only one FIFO queue



Aggregate 320B for the queue in fast SRAM and read and write to all DRAMs in parallel

In practice, buffer holds many FIFOs

e.g.
- In an IP Router, Q might be 200.
- In an ATM switch, Q might be 10^6.



We don't know which head of line packet the scheduler will request next?

Problems

- Problems

  1. A 320B block will contain packets for different queues, which can't be written to, or read from the same location.

  2. Suppose we write packets for different queues to different locations. How do we know that the memory will be available for reading when we need to retrieve the packet?

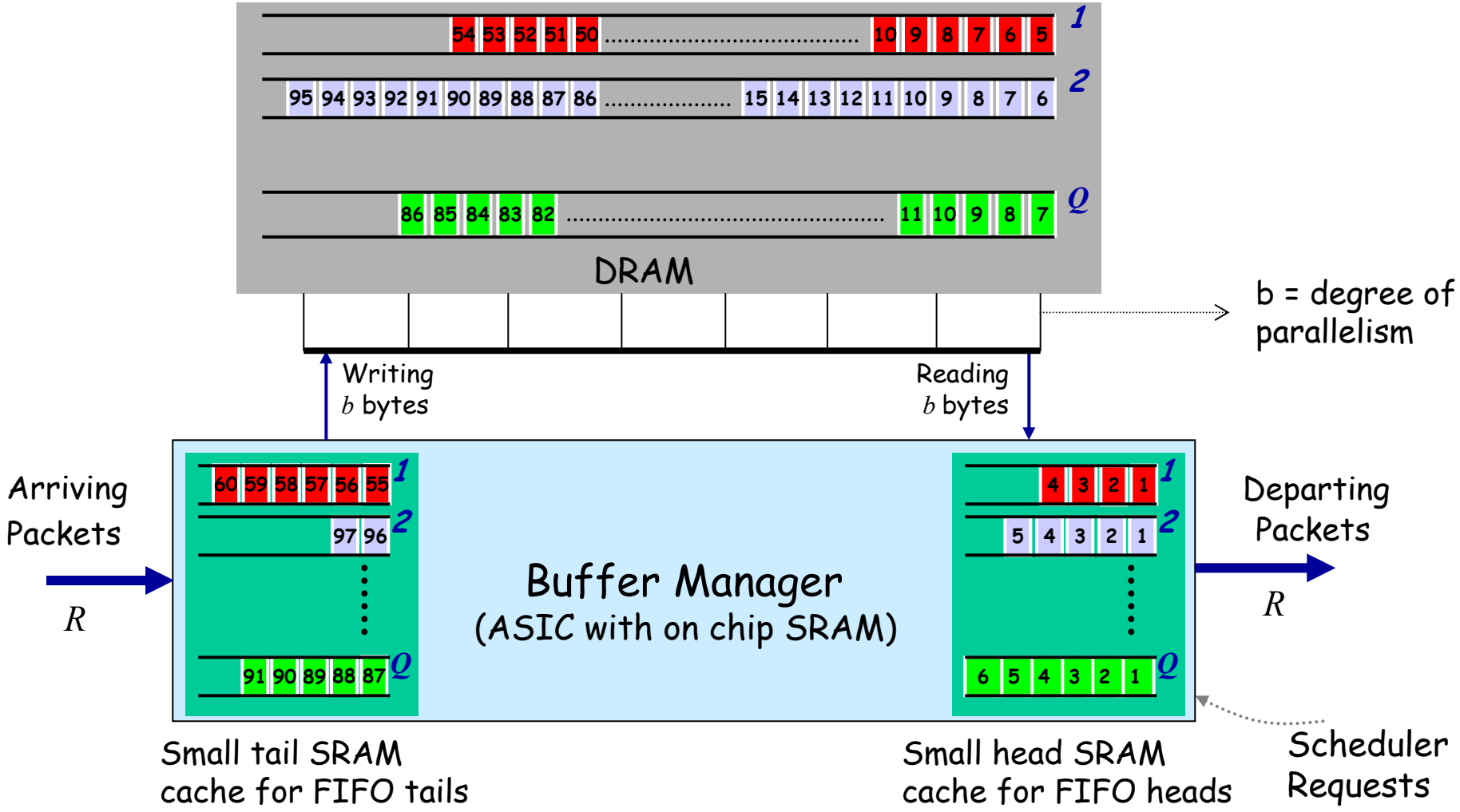
- Requirement

  1. We will have to aggregate 320B for every queue and read and write to all DRAMs in parallel

Parallel Packet Buffer

Hybrid Memory Hierarchy

Large DRAM memory holds the body of FIFOs

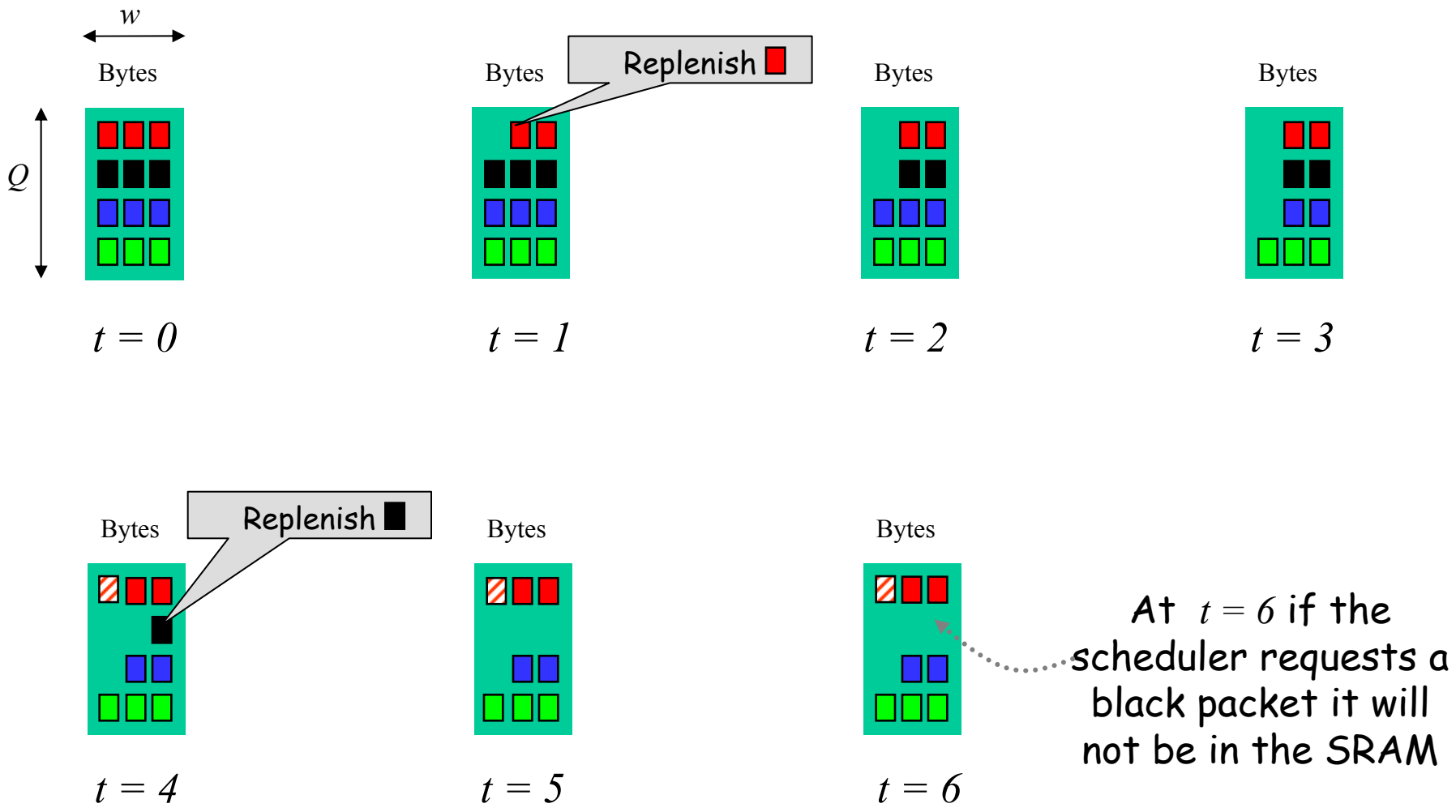


Re-stating the Problem

- Problem:
  - What is the minimum size of the SRAM needed for the parallel buffer so that every packet is available immediately in the SRAM when requested?
- Theorem 3: (Necessity)
  - An SRAM size of size $Qw = Q(b-1)(2+\ln Q)$ bytes is necessary.
    - w is the size of each queue

Why do we need a large SRAM?

Q = 4, w = 3, b = 3



Theorem 4

- Theorem 4: (sufficiency)
  - An SRAM cache of size $Qw = Qb(2 + \ln Q)$ bytes is sufficient so that every packet is available immediately in the SRAM when requested
- Discussion:
  - The algorithm replenishes the queue with the most urgent need of replenishment
  - It is almost optimal

Intuition for Theorem 4

- The maximum number of un-replenished requests for any i queues w_i, is the solution of the difference equation -

$$w_i \geq (w_{i-1} - b) + \frac{(w_{i-1} - b)}{i - 1}; \quad i \in \{2, 3, \ldots Q\}$$

- with boundary conditions $w_q < Qb$

Examples:

1. 40Gb/s line card, b=640, Q=128: SRAM = 560kBytes
2. 160Gb/s line card, b=2560, Q=512: SRAM = 10MBytes

Theorem 5

- Problem:
  - What is the minimum size of the SRAM so that every packet is available in the SRAM within a bounded pipeline latency when requested?
- Theorem 5: (necessity and sufficiency)
  - An SRAM cache of size $Qw = Q(b-1)$ bytes is both necessary and sufficient if the pipeline latency is $Q(b-1)+1$ time slots.

Intuition for Theorem 5

- The algorithm replenishes the queue which is going to be in the most urgent need of replenishment
  - If we use a lookahead buffer to know the requests "in advance", we can identify the queue which will empty earliest
  - This increases the pipeline latency from when a request is made until the byte is available.

Example:

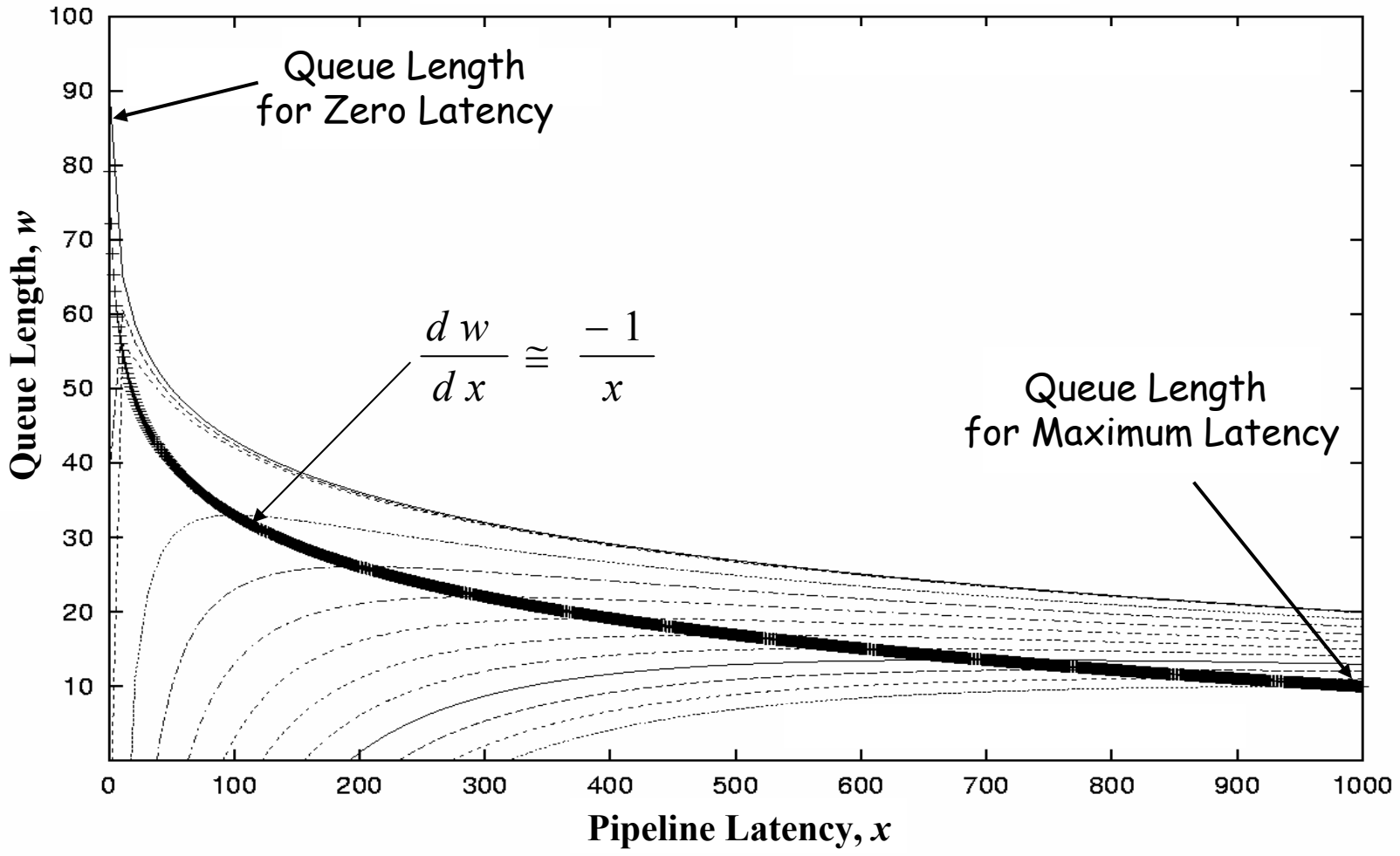
1. 160Gb/s line card, b=2560, Q=128: SRAM = 160kBytes, latency is 8μs.

Theorem 6

- Problem:
  - What is the minimum size of the SRAM needed so that every packet is available in the SRAM within a bounded pipeline latency x in the range $(0, Q(b-1)+1)$ when requested?
- Theorem 6: (sufficiency)
  - An SRAM cache of size $Qw \cong Qb(2+\ln Qb/x)$ bytes is sufficient.

Discussion of Theorem 6

Q=1000, b = 10



Contents

1. Parallel routers: work-conserving
2. Parallel routers: delay guarantees
3. Parallel packet buffers
4. Summary of contributions

Summary of Contributions

Covered in this talk

1. Parallel routers:

   - S. Iyer, R. Zhang, N. McKeown, "Routers with a Single Stage of Buffering", Proceedings of *ACM SIGCOMM*, Pittsburgh, Pennsylvania, Sep 2002. Also in *Computer Communication Review*, vol. 32, no. 4, Oct 2002.

2. Parallel packet buffers:

   - S. Iyer, R. R. Kompella, N. McKeown, "Designing Packet Buffers for Router Line Cards", Under review in *IEEE/ACM Transactions on Networking,* Oct. 2002.

   - S. Iyer, R. R. Kompella, N. McKeown, "Analysis of a Memory Architecture for Fast Packet Buffers", *IEEE - High Performance Switching and Routing*, pp. 368-373, Dallas, May 2001. A preliminary version appeared in *IEEE GBN*, Alaska, April 2001.

Summary of Contributions

Not covered in this talk

3. Other switches (load-balancing):

   - S. Iyer, N. McKeown, "Analysis of the Parallel Packet Switch Architecture", to appear in *IEEE/ACM Transactions on Networking*, Apr. 2003.
   - S. Iyer, A. Awadallah, N. McKeown, "Analysis of a Packet Switch with Memories Running Slower than the Line Rate", Proceedings of *IEEE INFOCOM*, pp. 529-537, Tel Aviv, March 2000.
   - S. Iyer, N. McKeown, "Making Parallel Packet Switches Practical", Proceedings of *IEEE INFOCOM*, Alaska, , vol. 3, pp. 1680-87, April 2001.
   - S. Iyer, N. McKeown, "Using Constraint Sets to Achieve Delay Bounds in CIOQ Switches", to appear in *IEEE Communication Letters*, 2003.

4. Parallel packet switch (multicast):

   - S. Iyer, N. McKeown, "On the Speedup Required for a Multicast Parallel Packet Switch", *IEEE Communication Letters*, vol. 5, no. 6, pp. 269-271, June 2001.

Summary of Contributions

Not covered in this talk

5. Parallel architecture (statistics counters):

   - D. Shah, S. Iyer, B. Prabhakar, N. McKeown, "Maintaining Statistics Counters in Router Line Cards", *IEEE Micro*, pp. 76-81, Jan-Feb, 2002. Also appeared as "Analysis of a Statistics Counter Architecture" in *IEEE Hot Interconnects*, Stanford University, Aug. 2001.

6. Parallel architecture (packet state):

   - S. Iyer, N. McKeown, "Maintaining State in Router Line Cards". In preparation for *IEEE Communication Letters*.

7. Parallel packet classification:

   - S. Iyer, R. R. Kompella, A. Shelat, "ClassiPI: An Architecture for Fast and Flexible Packet Classification", *IEEE NETWORK*, Special Issue on Fast IP Packet Forwarding and Classification for Next Generation Internet Services, Mar-Apr. 2001.

   - S. Iyer, A. Desai, A. Tambe, A. Shelat, "ClassiPI: A Classifier for Next Generation Policy Based Engines", *IEEE Hot Chips*, Stanford University, Aug 2000.

Summary of Contributions

Not covered in this talk

8. Parallel switching algorithms (Buffered crossbars):

   - Jointly with Da Chuang, N. McKeown, "Attaining Statistical and Deterministic Switching Guarantees using Buffered Crossbars", In preparation for *IEEE/ACM Transactions on Networking*.

   - Jointly with Da Chuang, N. McKeown, "Designing Practical Routers using Buffered Crossbars", In preparation for *IEEE/ACM Transactions on Networking.*

9. Load-balancing (IP backbone):

   - S. Iyer, S. Bhattacharyya, N. Taft, C. Diot, "An Approach to Alleviate Link Overload as Observed on an IP Backbone", Proceedings of *IEEE INFOCOM*, San Francisco, March 2003.

10. Greedy switching algorithms:

    - S. Iyer, N. McKeown, "Maximum Size Matchings and Input Queued Switches", Proceedings of the 40th *Annual Allerton Conference on Communication, Control, and Computing*, Monticello, Illinois, Oct 2002.

Acknowledgements

1. Nick
2. Balaji
3. Members of my committee – Sunil, Mark and Rajeev
4. Industry:
   - Cisco - Flavio
   - Sprint Labs – Christophe, Supratik, Nina and the IP group
   - SwitchOn/PMC-Sierra - Ajit, Moti and the whole team
5. Research "co-conspirators"
6. Members of the research group
7. Members of network-group
8. Department staff
9. Friends, friends & friends from all over the globe …
10. The couch on the 5th floor, the basement vending machine, …
11. Mom

- I was told: "A good way to judge how well you are doing, is to measure the quality of people with whom you interact.
- By that criteria, I could not have asked for more ….