Network Processors and their memory

Network Processor Workshop, Madrid 2004

Nick McKeown
Departments of Electrical Engineering and
Computer Science, Stanford University

nickm@stanford.edu
http://www.stanford.edu/~nickm

Outline

- What I was going to say
- Network processors and their memory
  - Packet processing is all about getting packets into and out of a chip and memory.
  - Computation is a side-issue.
  - Memory speed is everything: Speed matters more than size
- Remarks

General Observations

- Up until about 1998,
  - Low-end packet switches used general purpose processors,
  - Mid-range packet switches used FPGAs for datapath, general purpose processors for control plane.
  - High-end packet switches used ASICs for datapath, general purpose processors for control plane.
- More recently,
  - 3rd party network processors used in some low-end datapaths.
  - Home-grown network processors used in mid- and high-end.

Why NPUs seem like a good idea

- What makes a CPU appealing for a PC
  - **Flexibility:** Supports many applications
  - **Time to market:** Allows quick introduction of new applications
  - **Future proof:** Supports as-yet unthought of applications
- No-one would consider using fixed function ASICs for a PC

Why NPUs seem like a good idea

- What makes a NPU appealing
  - **Time to market:** Saves 18months building an ASIC. Code re-use.
  - **Flexibility:** Protocols and standards change.
  - **Future proof:** New protocols emerge.
  - **Less risk:** Bugs more easily fixed in s/w.
- Surely no-one would consider using fixed function ASICs for new networking equipment?

Why NPUs seem like a bad idea

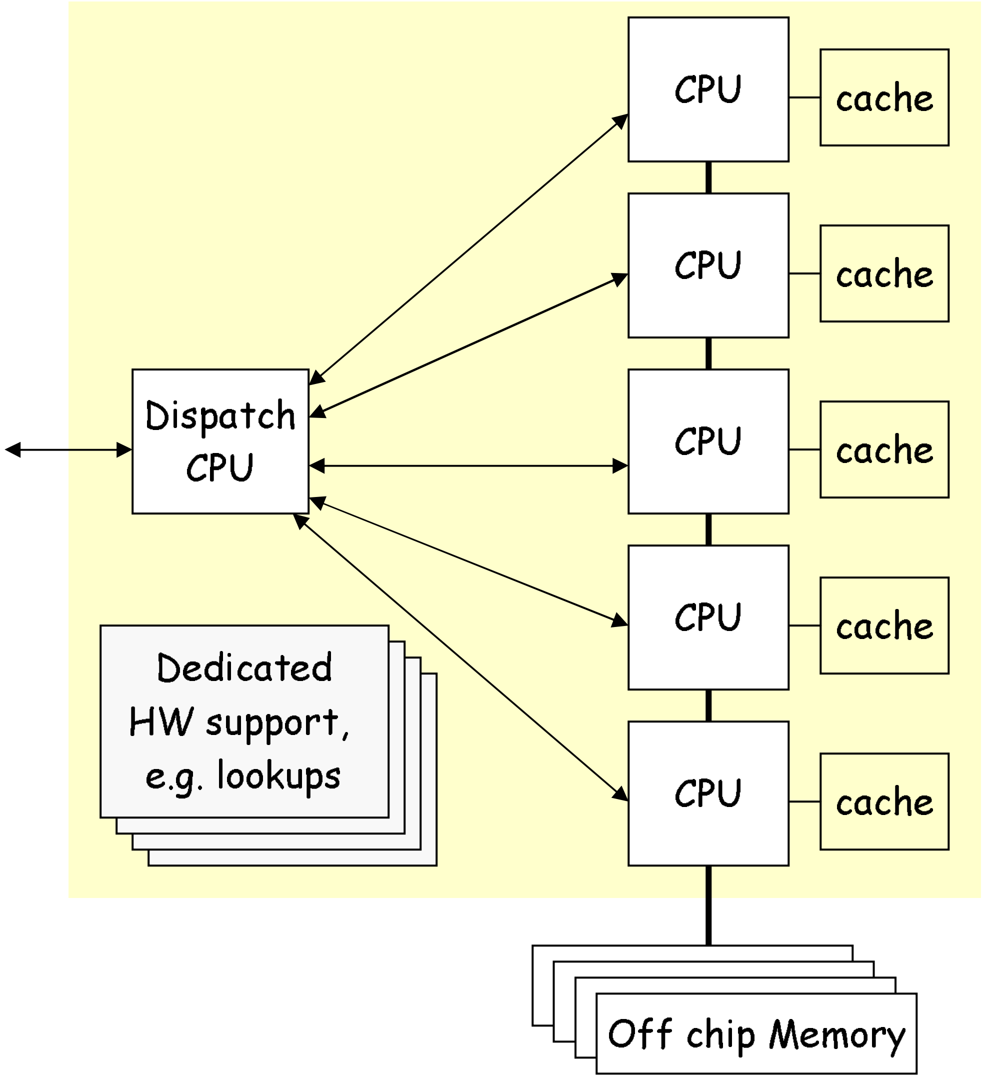
- Jack of all trades, master of none
  - NPUs are difficult to program
  - NPUs inevitably consume more power,
  - …run more slowly and
  - …cost more than an ASIC
- Requires domain expertise
  - Why would a/the networking vendor educate its suppliers?
- Designed for computation rather than memory-intensive operations

NPU Characteristics

- NPUs try hard to hide memory latency
  - Conventional caching doesn't work
    - Equal number of reads and writes
    - No temporal or spatial locality
    - Cache misses lose throughput, confuse schedulers and break pipelines
  - Therefore it is common to use multiple processors with multiple contexts

Network Processors

Load-balancing

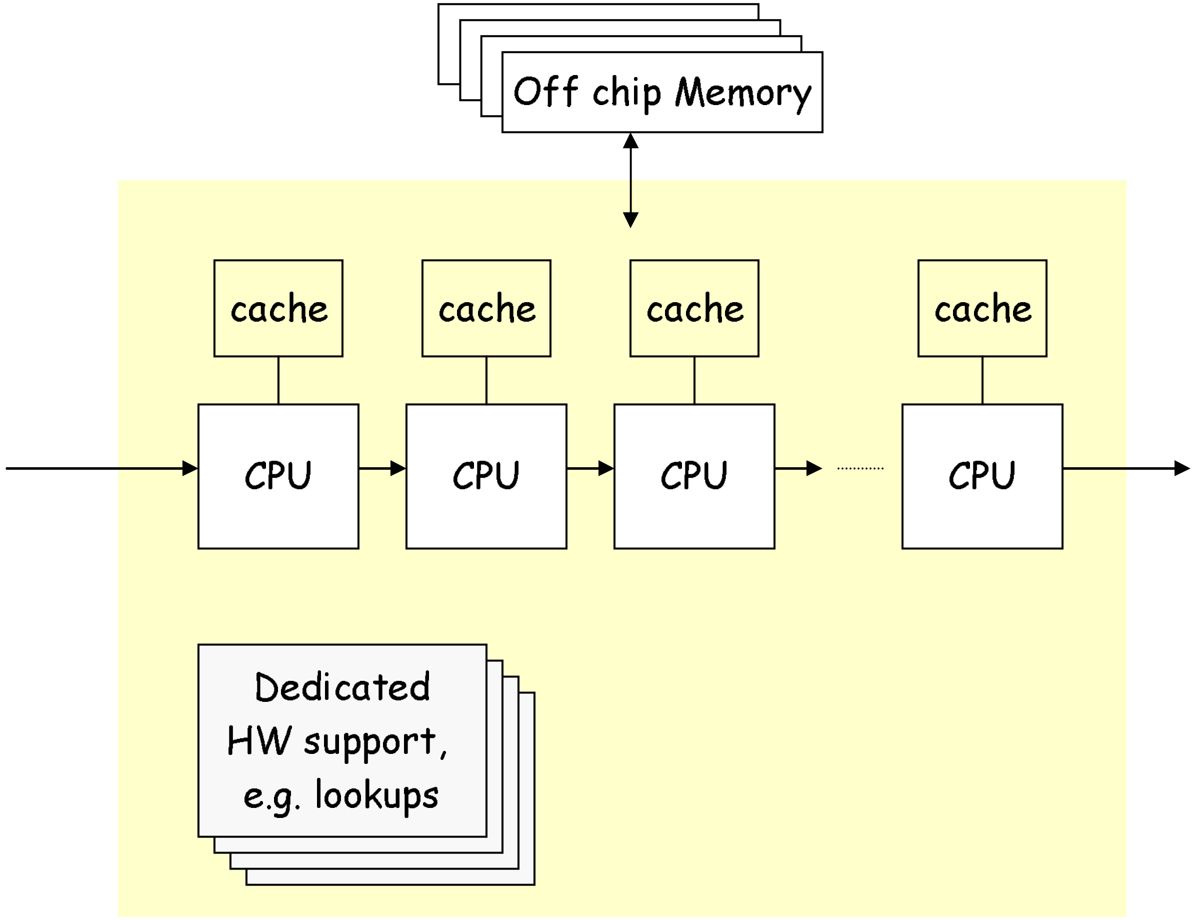


Incoming packets dispatched to:

1. Idle processor, or
2. Processor dedicated to packets in this flow (to prevent mis-sequencing), or
3. Special-purpose processor for flow, e.g. security, transcoding, application-level processing.

Network Processors

Pipelining



Processing broken down into (hopefully balanced) steps,
Each processor performs one step of processing.

Question

Is it clear that multiple small parallel processors are needed?

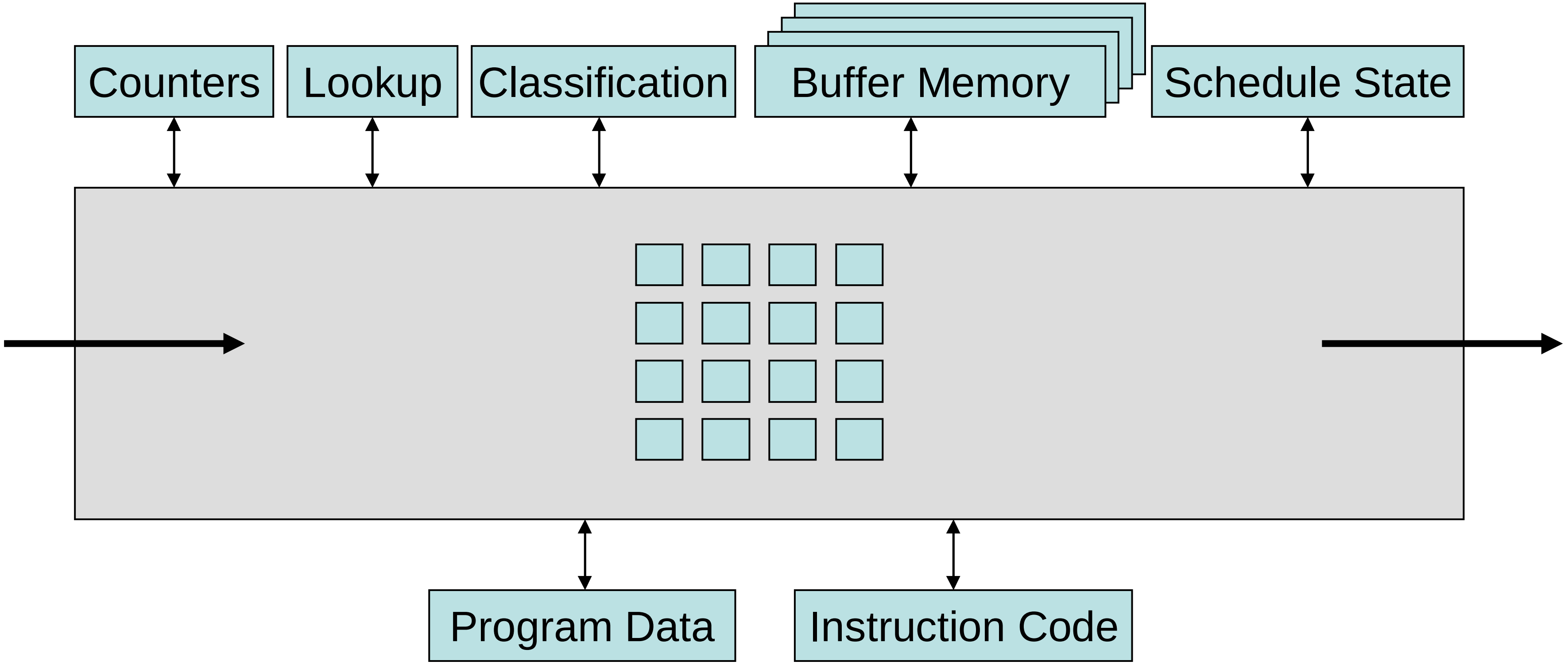
Doubts

- When are 10 processors at speed 1 better than 1 processor at speed 10?
- Network processors make sense if:
    - Application is parallelizable into multiple threads/contexts.
    - Uniprocessor performance is limited by load-latency.
- If general purpose processors evolve anyway to:
    - Contain multiple processors per chip.
    - Support hardware multi-threading.
- …then perhaps they are better suited because:
    - Greater development effort means faster general purpose processors.
    - Better development environments.

Outline

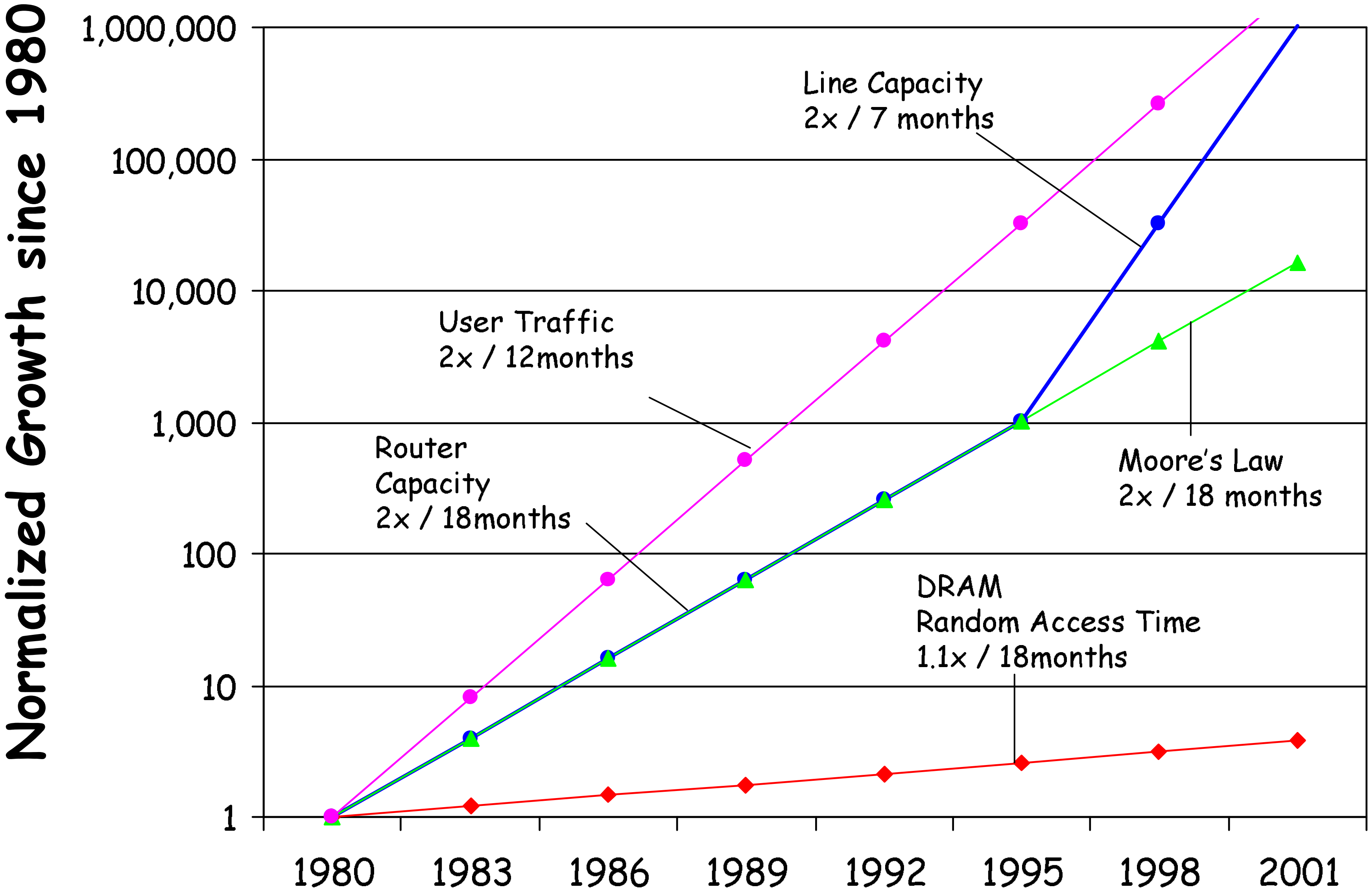
- What I was going to say
- Network processors and their memory
  - Packet processing is all about getting packets into and out of a chip and memory.
  - Computation is a side-issue.
  - Memory speed is everything: Speed matters more than size.
- Remarks

NPUs and Memory



Typical NPU or packet-processor has 8-64 CPUs,
12 memory interfaces and 2000 pins

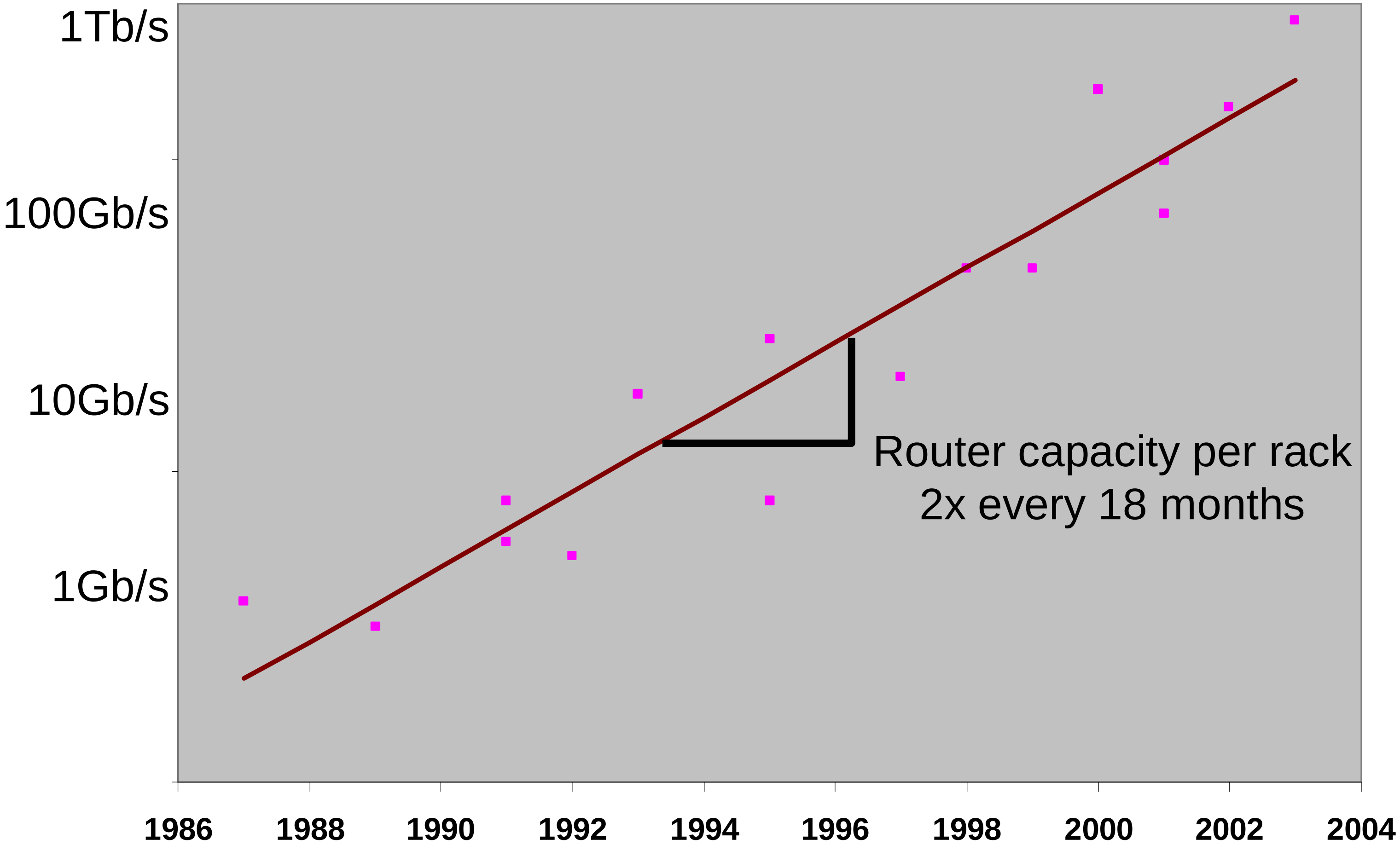
Trends in Technology, Routers & Traffic



Memory gets further away

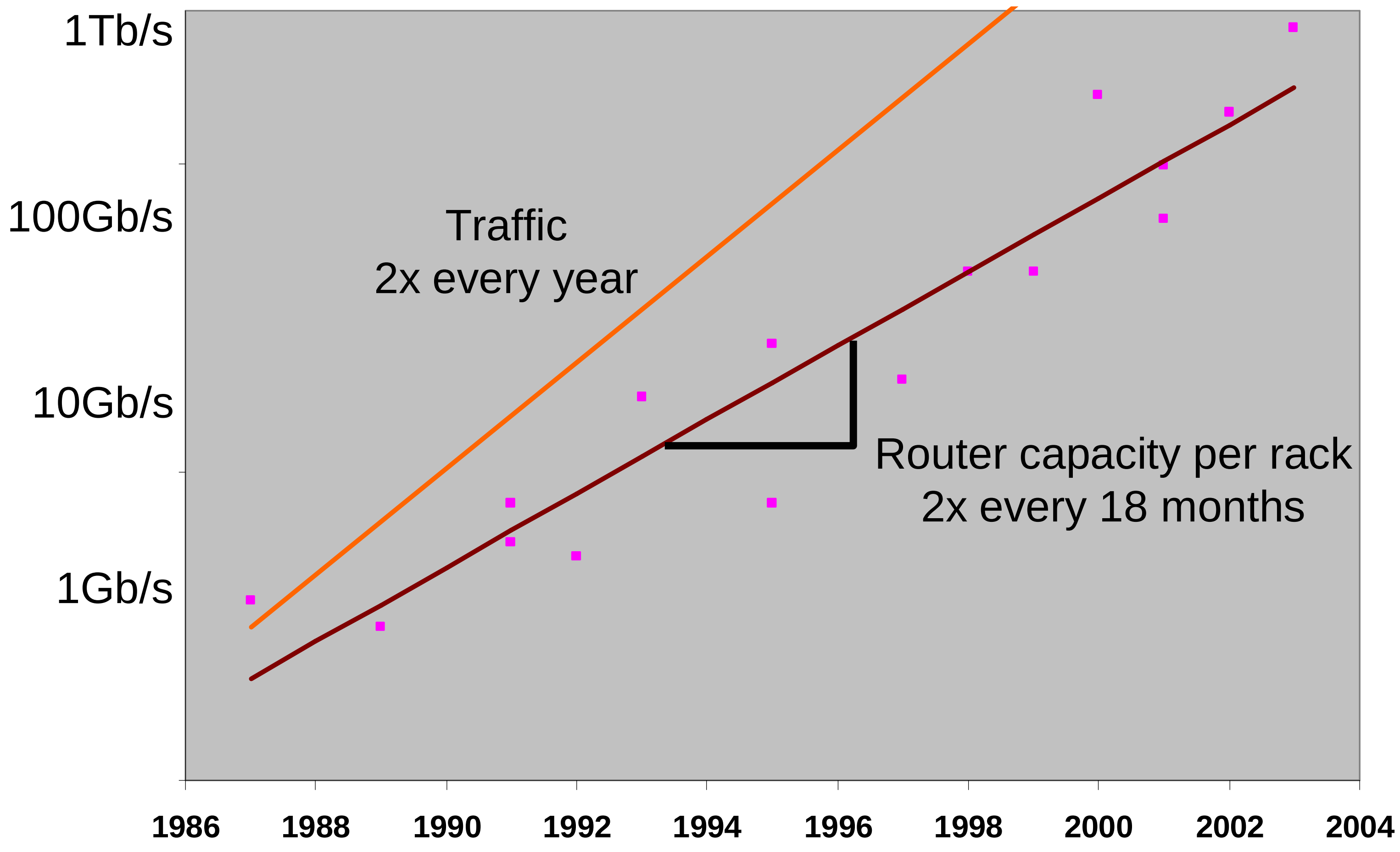
- Accessing memory becomes twice as expensive every 18 months.
- CPUs
  - Bigger caches
  - Larger refill blocks and faster pins
  - Better pre-fetching algorithms
- NPUs
  - More CPUs…?

Backbone router capacity



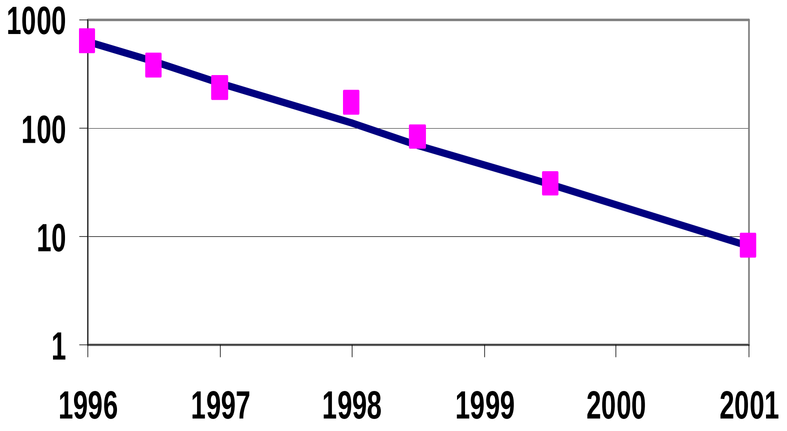
Router capacity per rack
2x every 18 months

Backbone router capacity

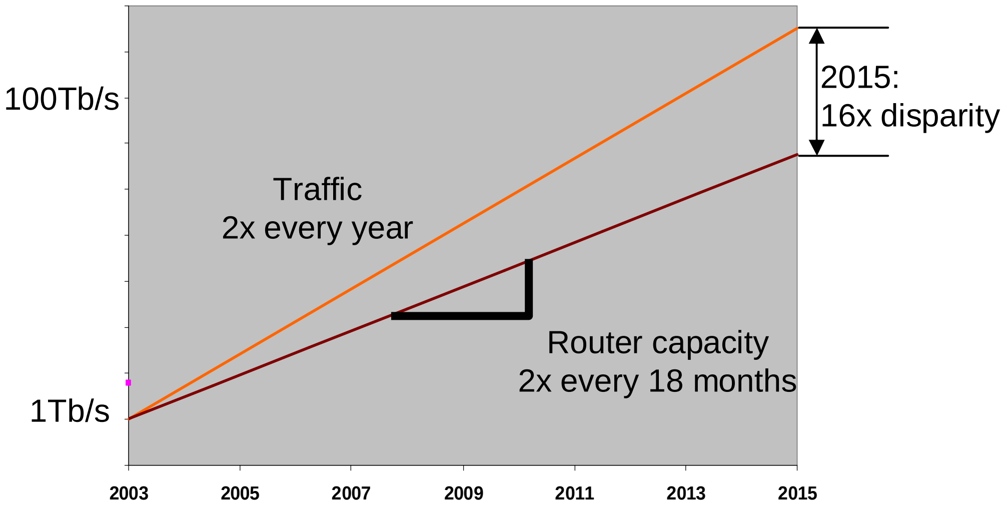


Trends and Consequences

1 CPU Instructions per minimum length packet



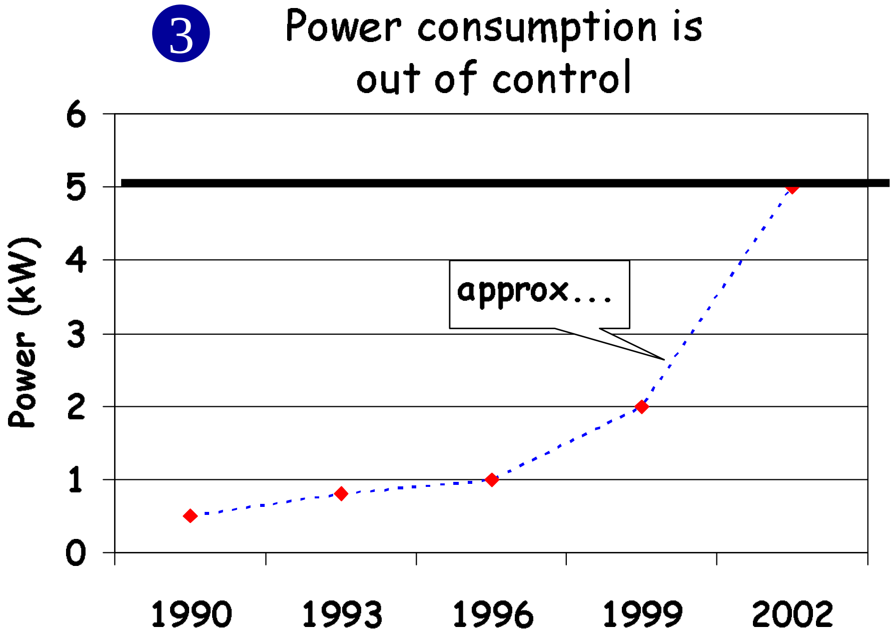
2 Disparity between traffic and router growth

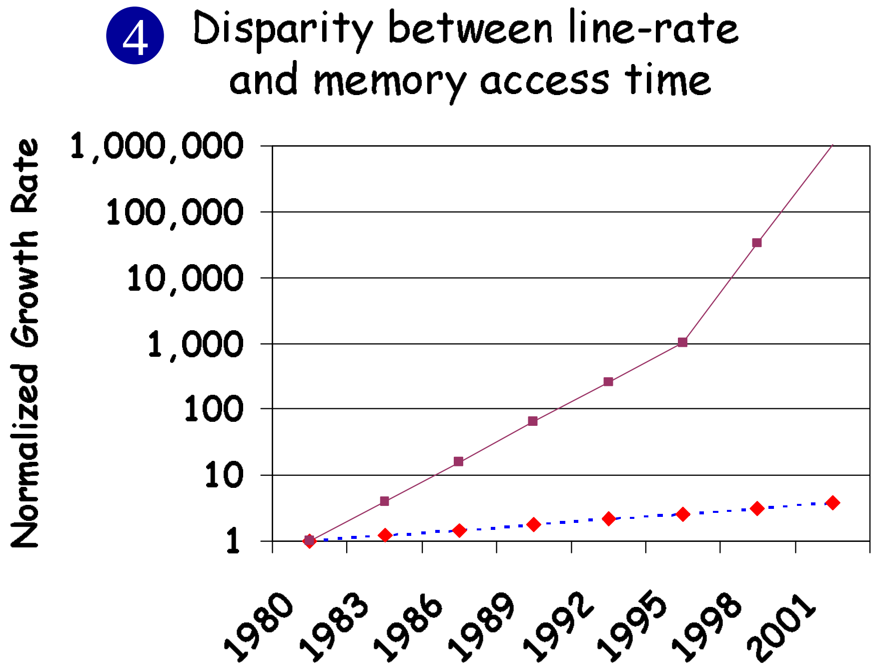


Consequences:

1. Per-packet processing is getting harder.
2. Efficient, simple processing will become more important.
3. Routers will get faster, simpler and more efficient. (Weren't they supposed to simple in the first place?)

Trends and Consequences (2)





Consequences:

1. Power efficiency will continue to be important.
2. Memories will seem slower and slower.
   Are we just going to keep adding more parallelism?

Predictions (1)

- Memory speed will matter more than size
  - Memory speed will remain a problem.
  - Waiting for slow off-chip memory will become intolerable.
  - Memory size will become less of an issue.
- Memory Size
  - Packet buffers: Today they are too big; they'll get smaller.

Memory Size

- Universally applied rule-of-thumb:
  - A router needs a buffer size: $\boxed{B = 2T \times C}$
    - $2T$ is the round-trip propagation time
    - C is the capacity of the outgoing link

- Background
  - Mandated in backbone and edge routers.
  - Appears in RFPs and IETF architectural guidelines.
  - Has huge consequences for router design.
  - Comes from dynamics of TCP congestion control.
  - Villamizar and Song: "High Performance TCP in ANSNET", CCR, 1994.
  - Based on 16 TCP flows at speeds of up to 40 Mb/s.

Example

- 10Gb/s linecard or router
  - Requires 300Mbytes of buffering.
  - Read and write new packet every 32ns.
- Memory technologies
  - SRAM: require 80 devices, 1kW, $2000.
  - DRAM: require 4 devices, but too slow.
- Problem gets harder at 40Gb/s
  - Hence RLDRAM, FCRAM, etc.

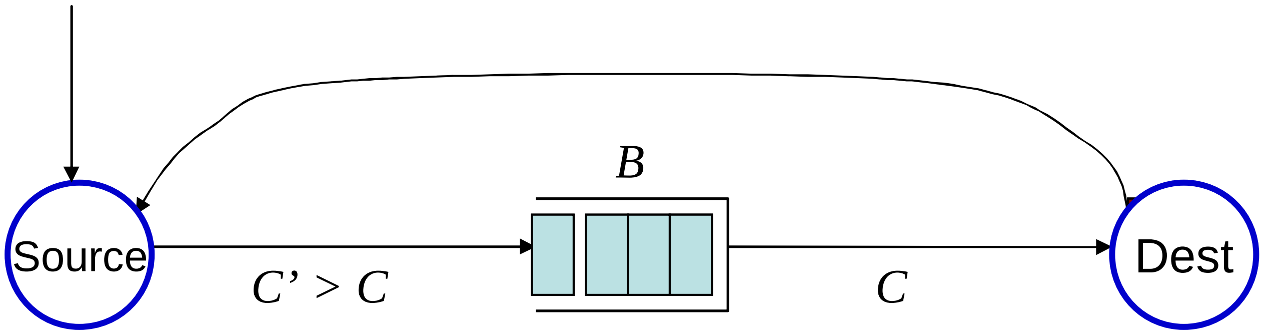
Rule-of-thumb

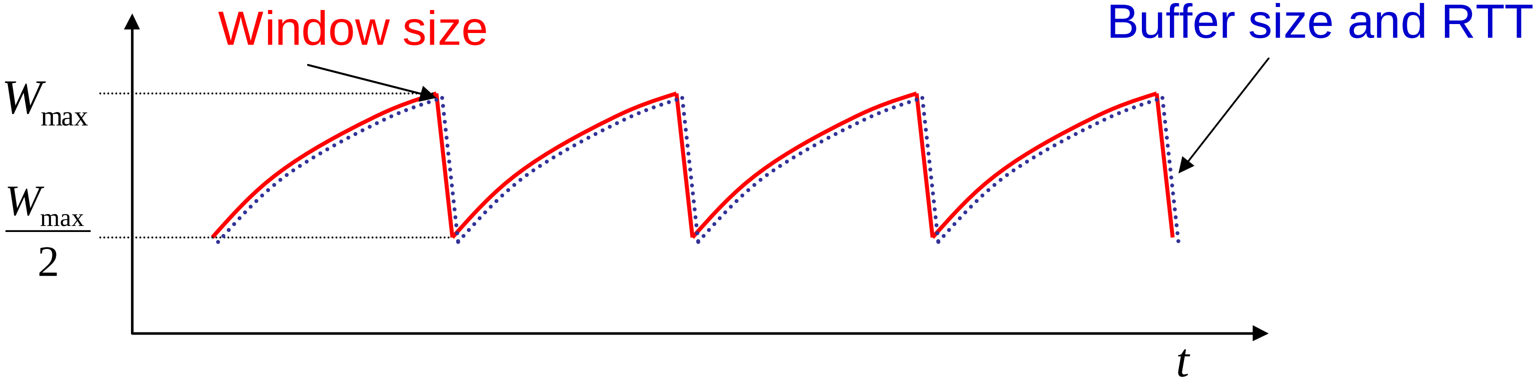
- Where did the rule-of-thumb come from?
- Is it correct? (No)

Joint work with Guido Appenzeller and Isaac Keslassy, Stanford

Single TCP Flow

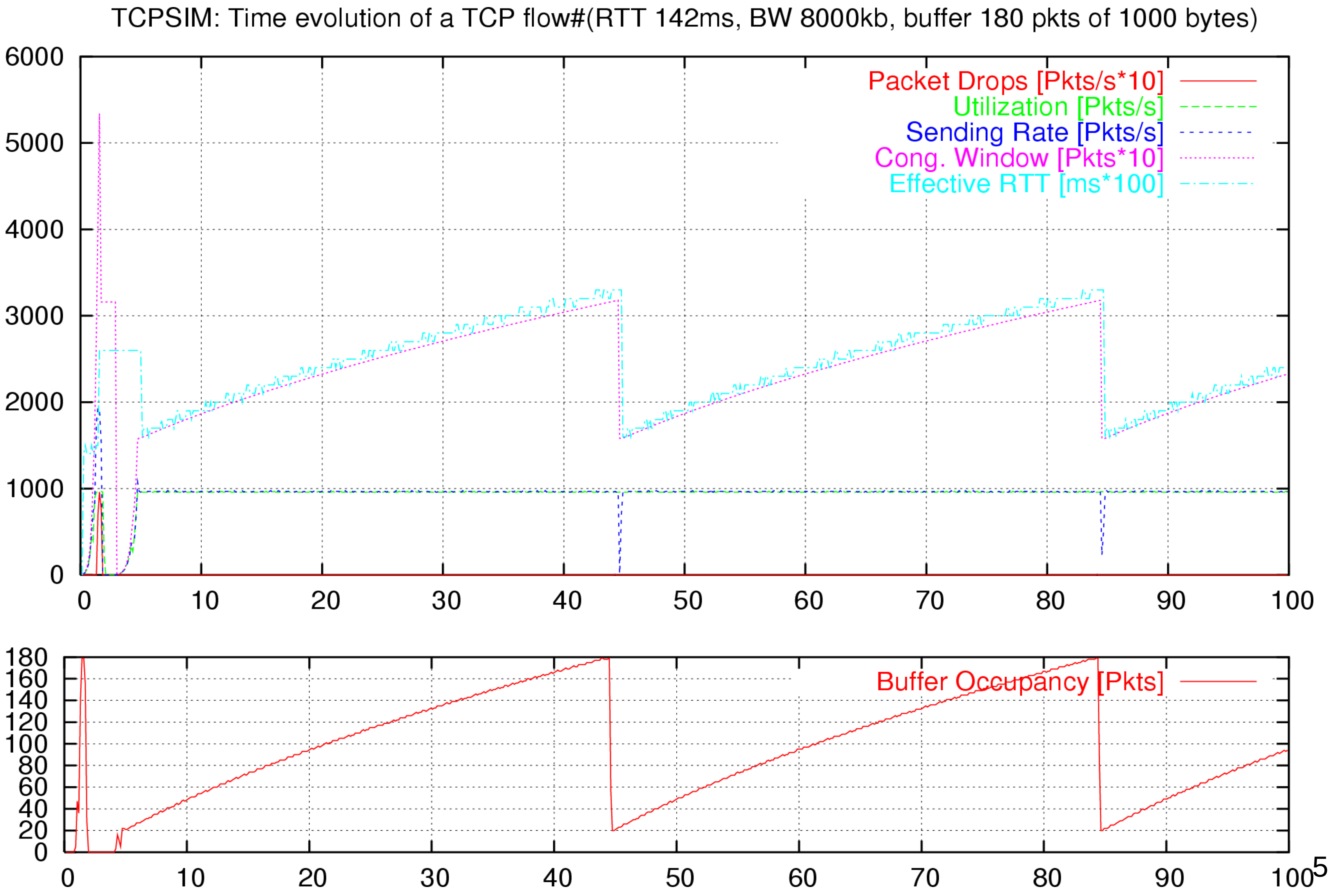
For every *W* ACKs received,
send *W*+1 packets



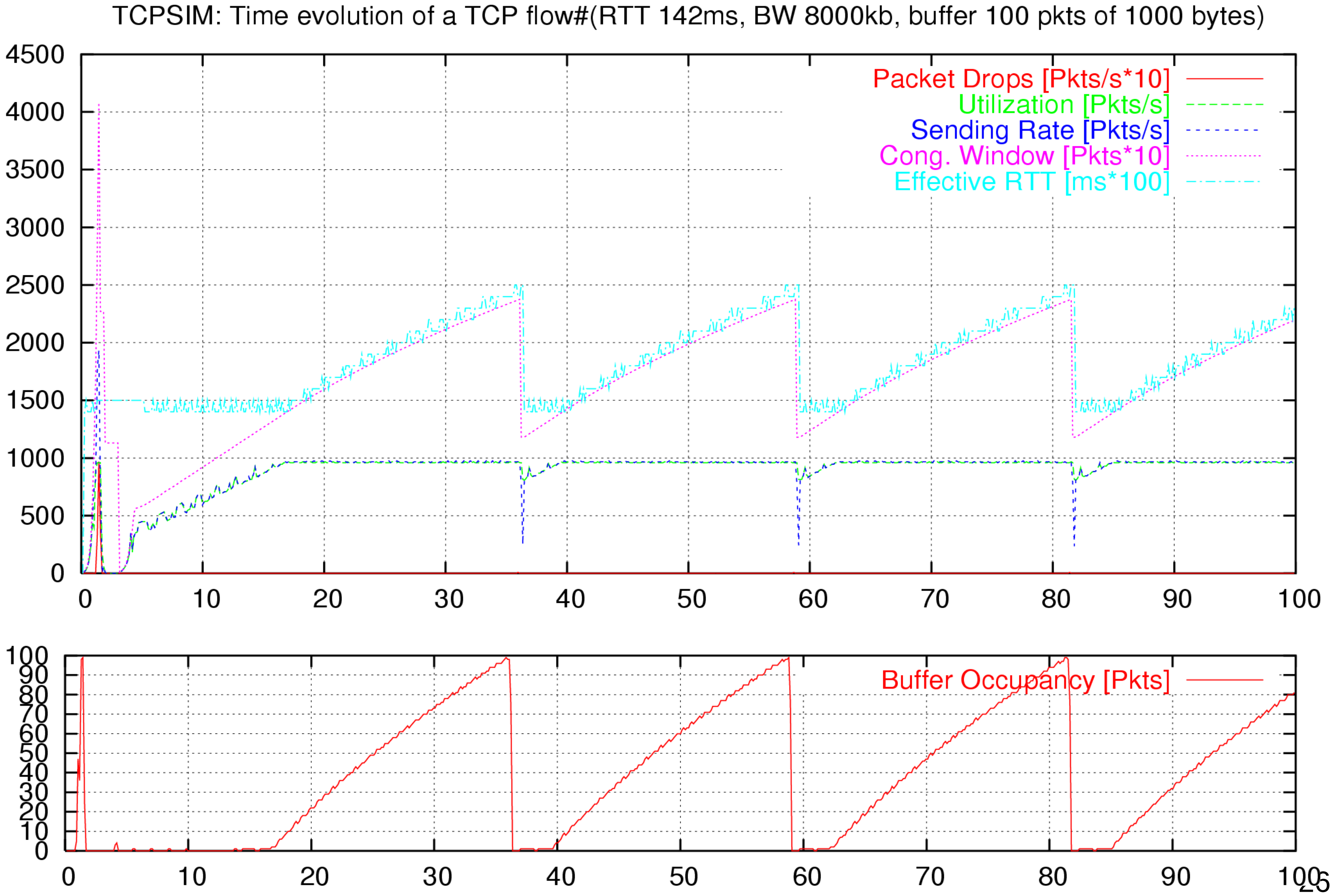

Window size

Buffer size and RTT

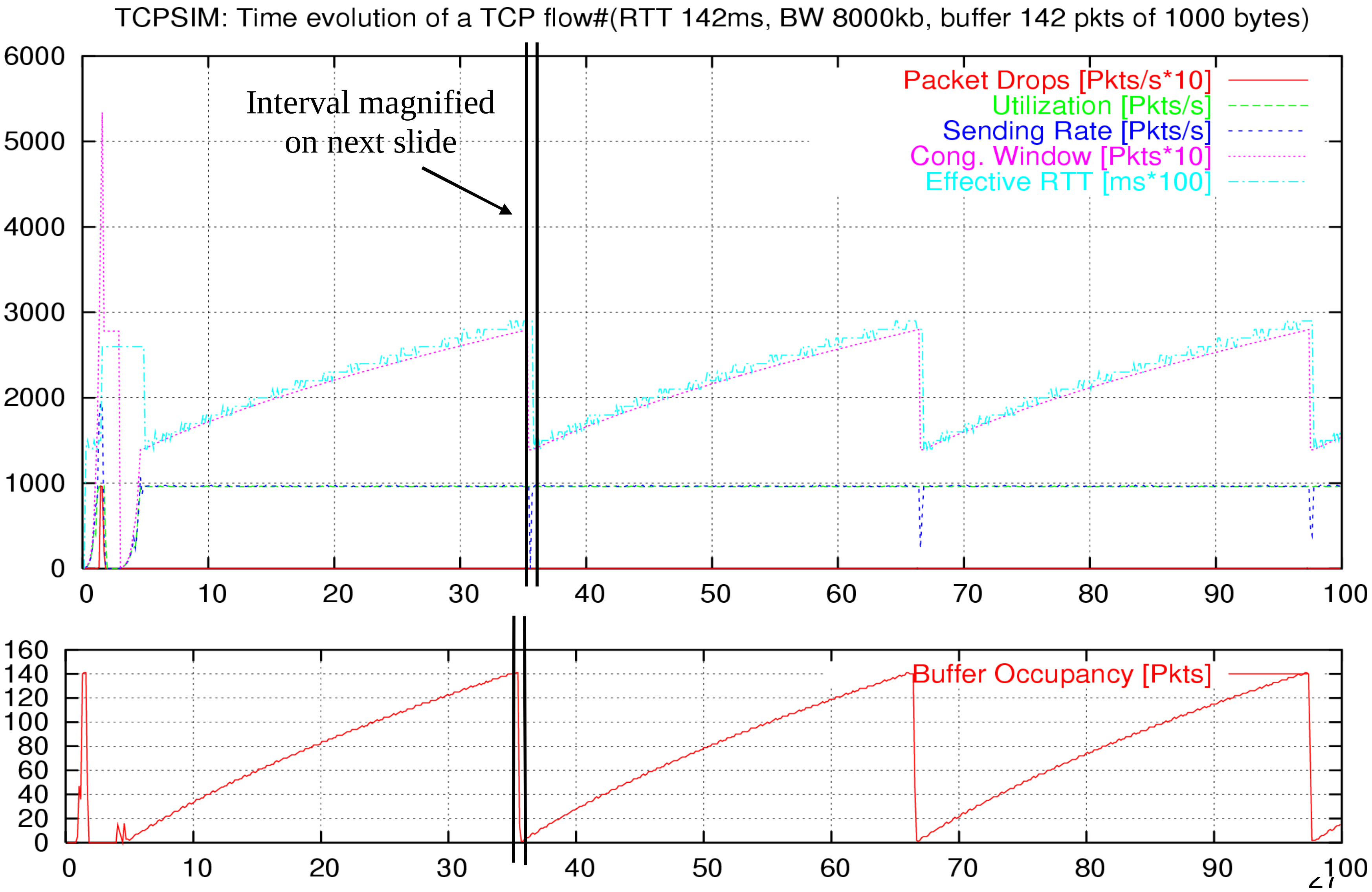
Over-buffered Link



Under-buffered Link

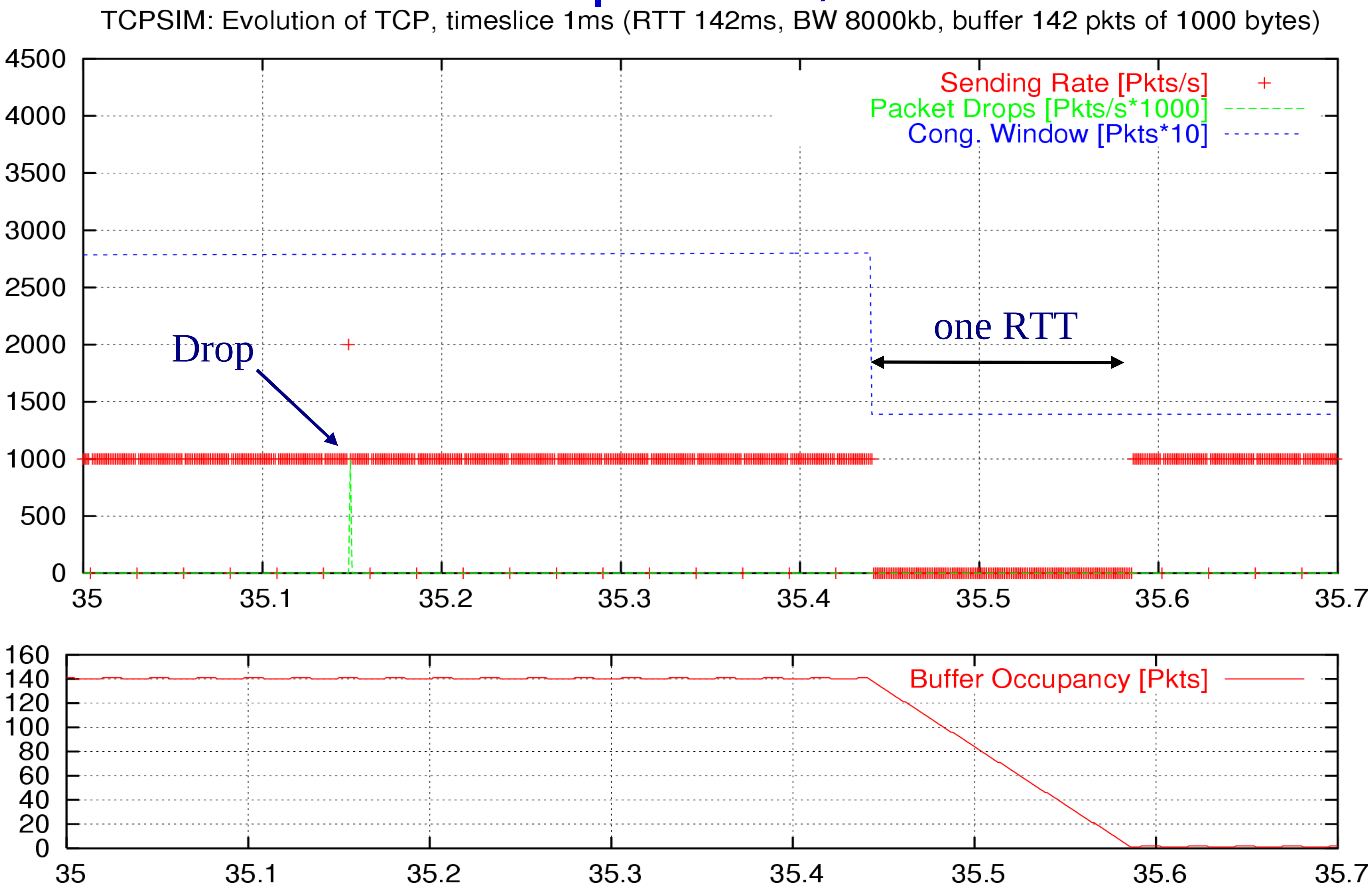


Buffer = Rule-of-thumb

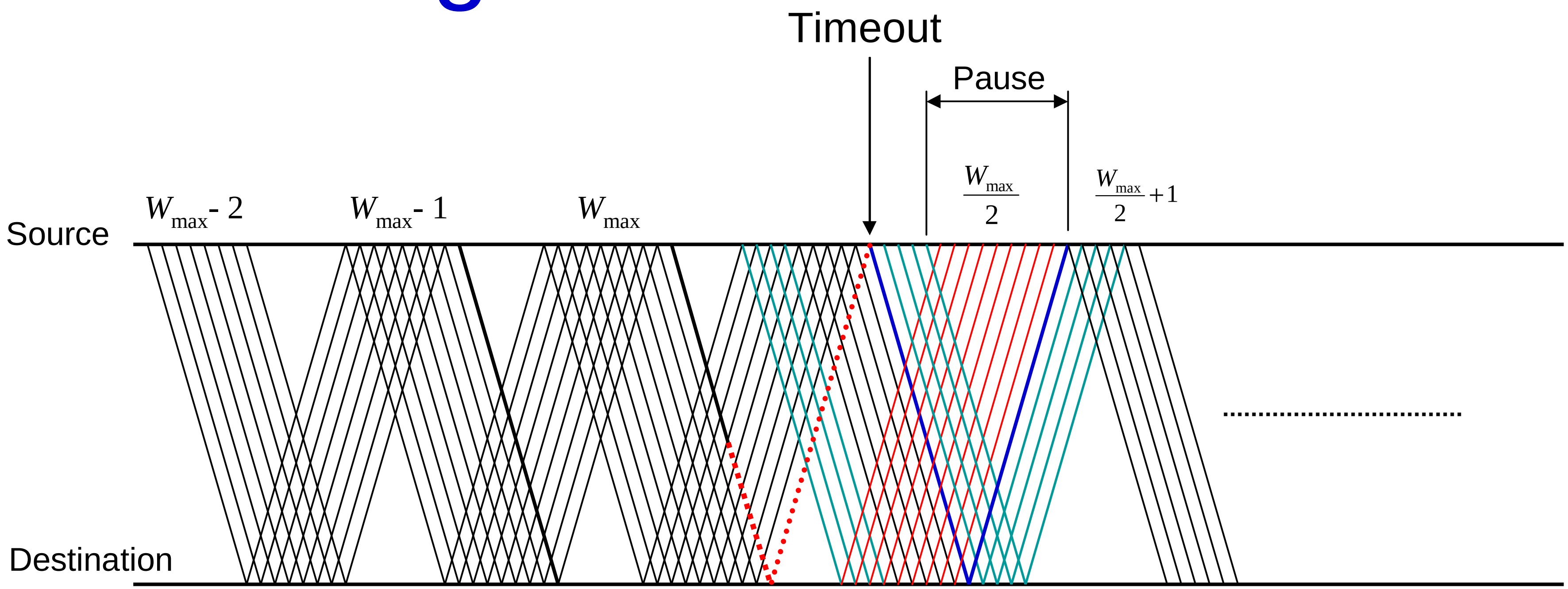


Microscopic TCP Behavior

When sender pauses, buffer drains



Origin of rule-of-thumb



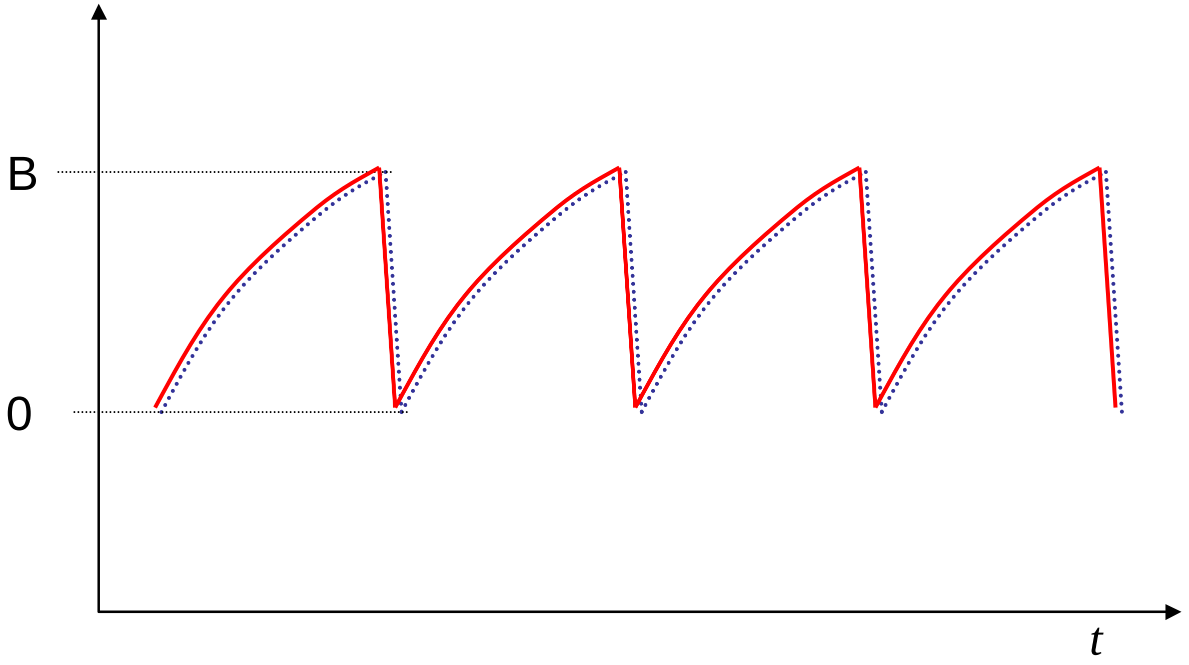
- While Source pauses, buffer drains
- Source pauses for $2T + B/C – W_{max}/2C$ seconds
- Buffer drains in B/C seconds
- Therefore, buffer never goes empty if $B > 2T$ x C
- We can size B to keep bottleneck link busy

Rule-of-thumb

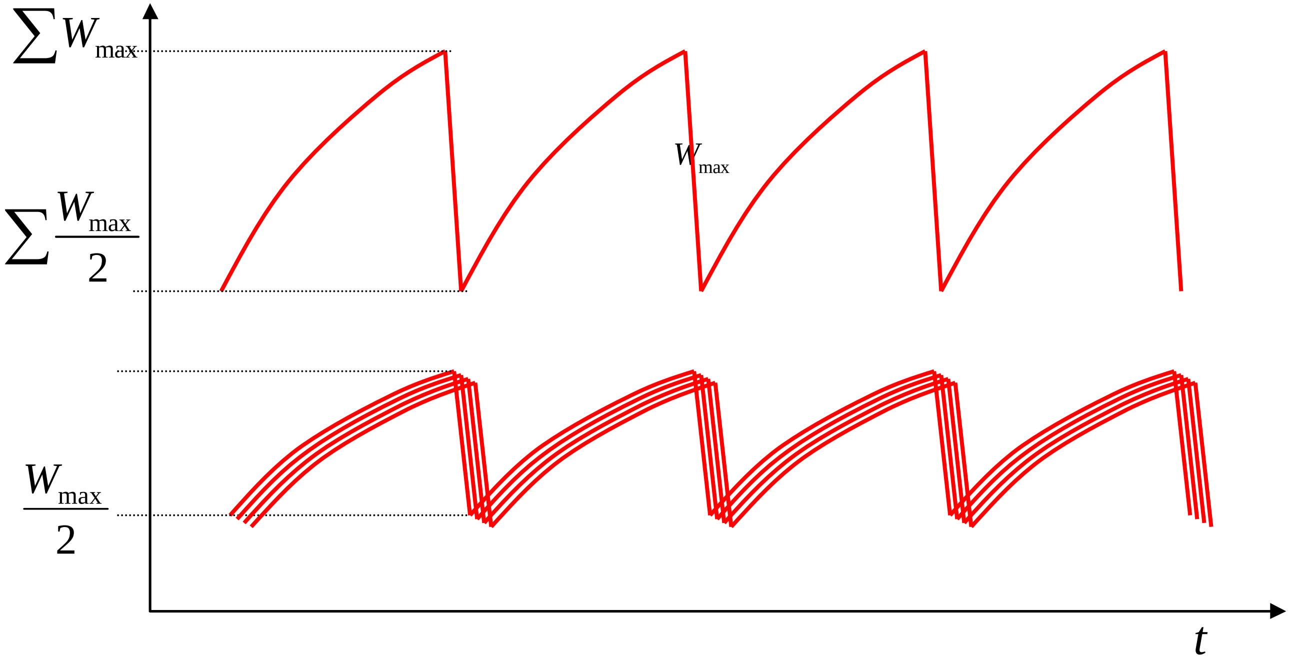
- Rule-of-thumb makes sense for one flow
- Typical backbone link has > 20,000 flows
- Does the rule-of-thumb still hold?

- Answer:
  - If flows are perfectly synchronized, then Yes.
  - If flows are desynchronized then No.

Buffer size is height of sawtooth



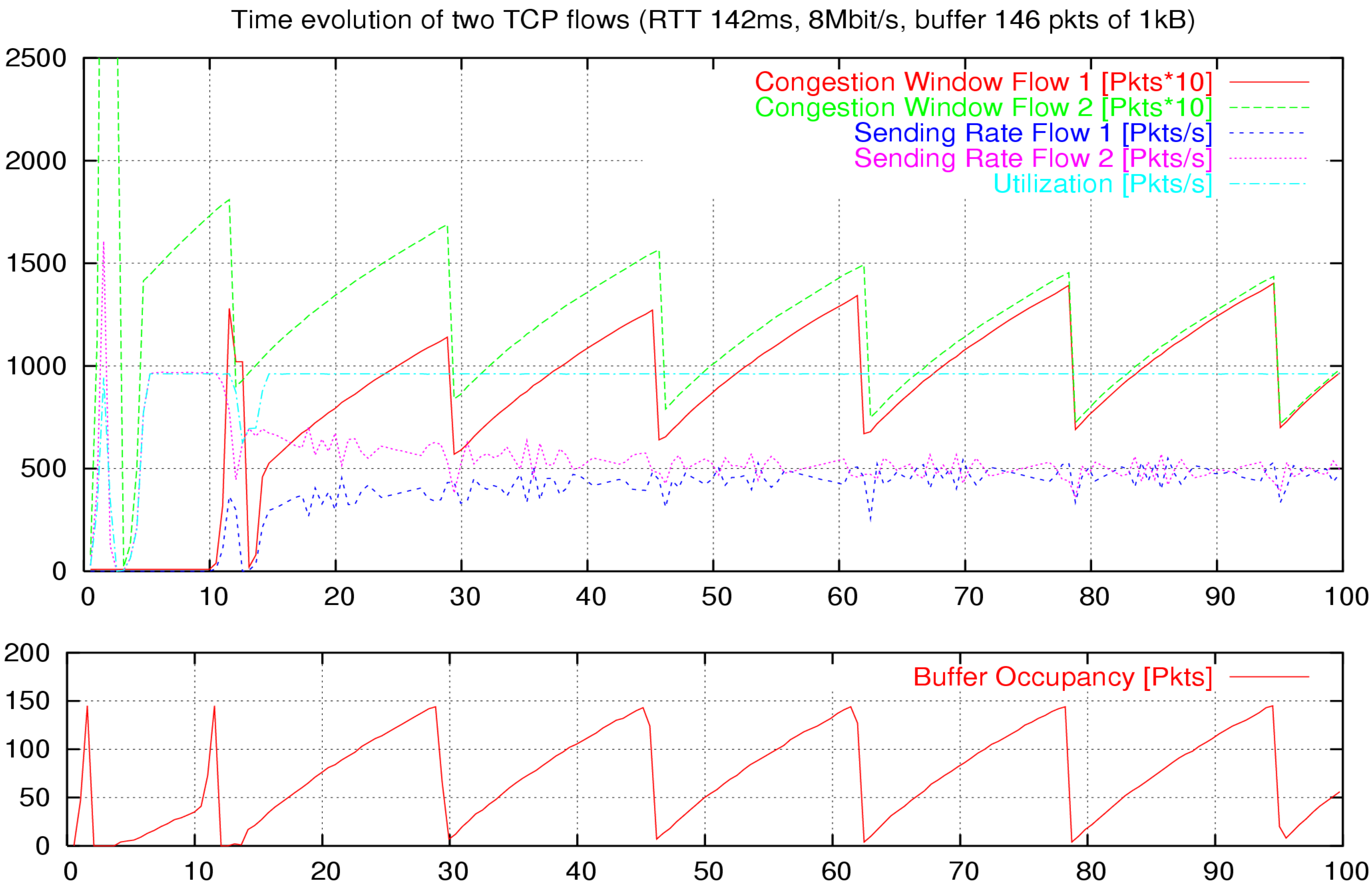
If flows are synchronized



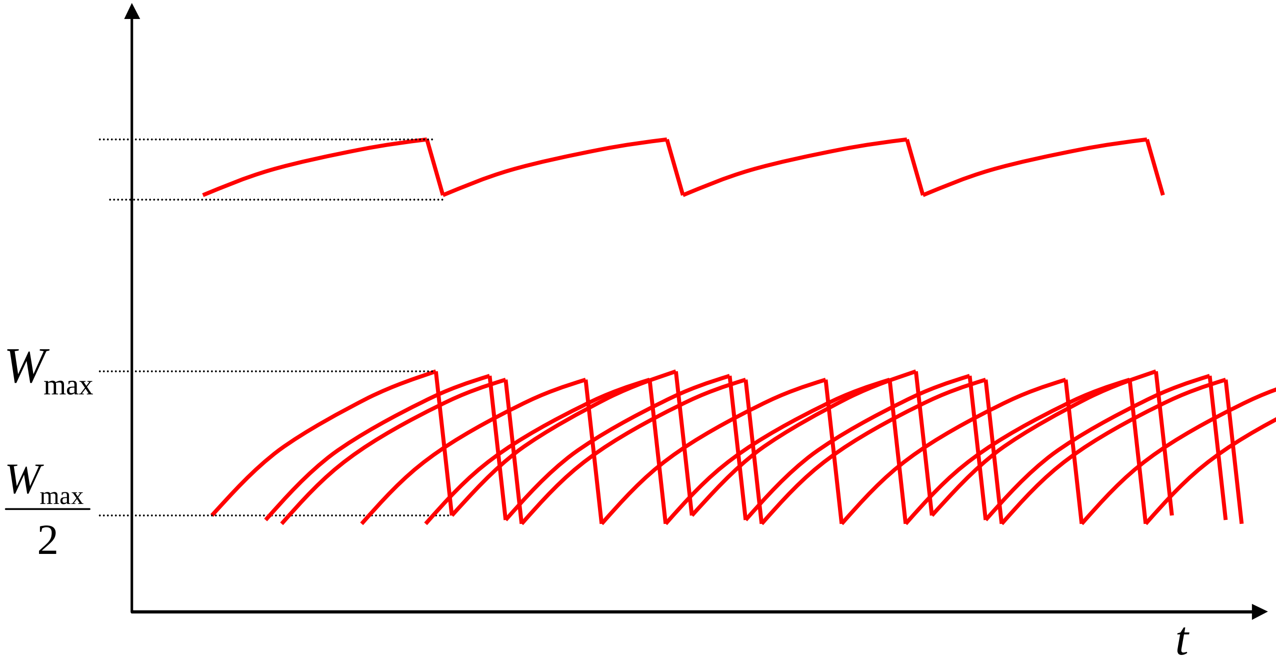
- Aggregate window has same dynamics
- Therefore buffer occupancy has same dynamics
- Rule-of-thumb still holds.

Two TCP Flows

Two TCP flows can synchronize



If flows are not synchronized

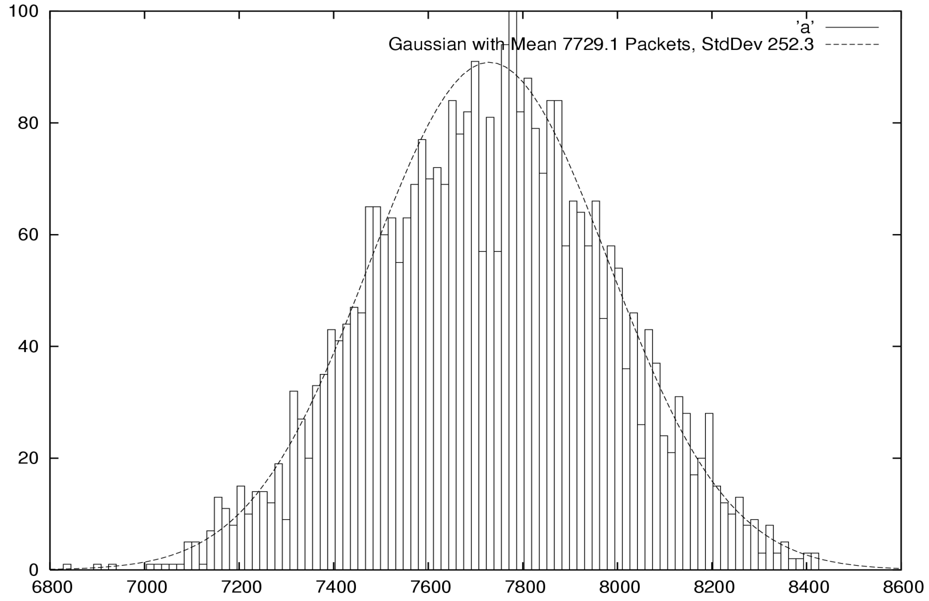


- Aggregate window has less variation
- Therefore buffer occupancy has less variation
- The more flows, the **smaller** the variation
- Rule-of-thumb does not hold.

If flows are not synchronized

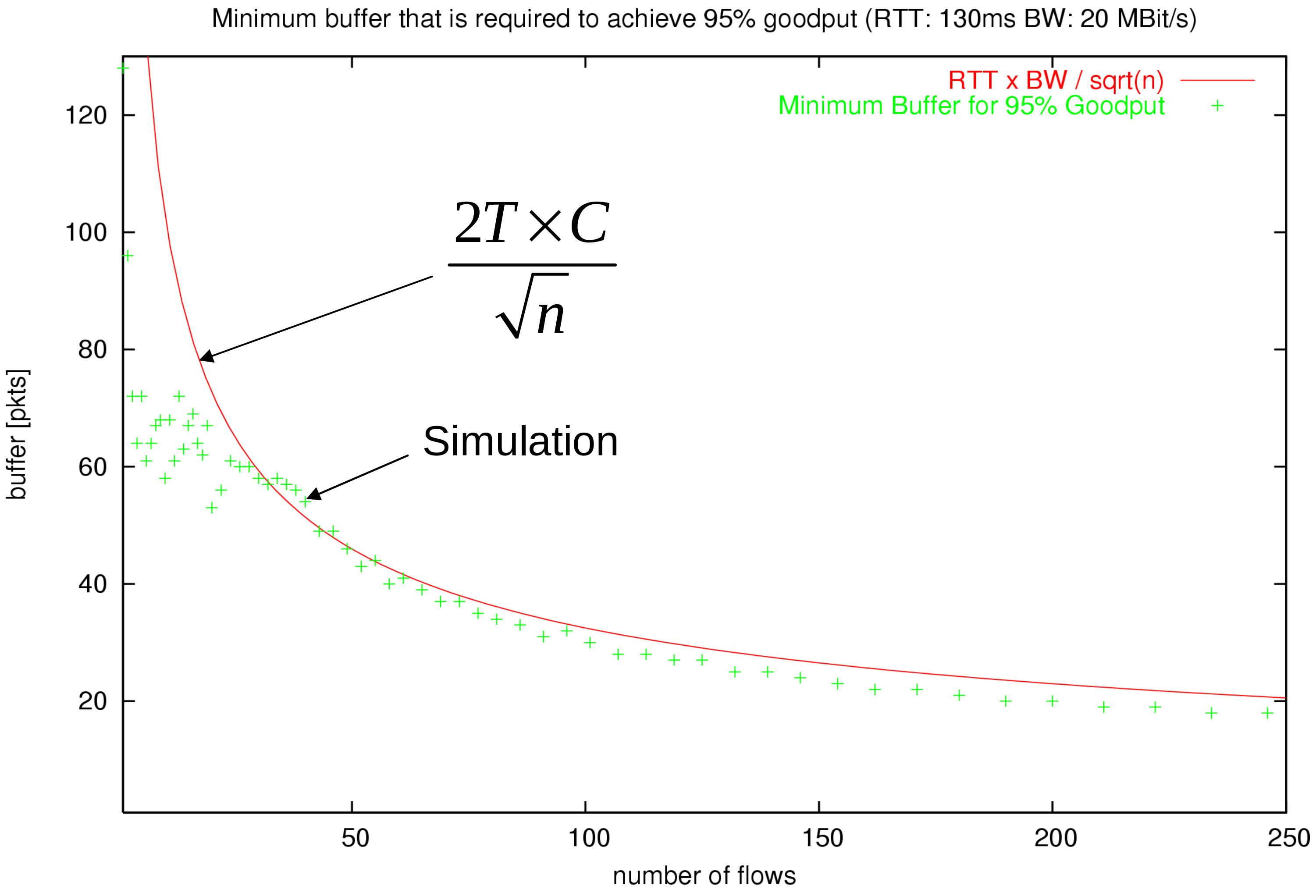
- With a large number of flows (>500) central limit theorem applies

$$\sum W_{\max} \xrightarrow{d} N(\mu, \sigma)$$



- Therefore, we can pick the utilization we want, and determine the buffer size.

Required buffer size



$$\frac{2T \times C}{\sqrt{n}}$$

Simulation

Experiments with backbone router

GSR 12000

TCP Flows	Router Buffer			Link Utilization		
	$\frac{2T \times C}{\sqrt{n}}$	Pkts	RAM	Model	Sim	Exp
100	0.5 x	64	1Mb	96.9%	94.7%	94.9%
	1 x	129	2Mb	99.9%	99.3%	98.1%
	2 x	258	4Mb	100%	99.9%	99.8%
	3 x	387	8Mb	100%	99.8%	99.7%
400	0.5 x	32	512kb	99.7%	99.2%	99.5%
	1 x	64	1Mb	100%	99.8%	100%
	2 x	128	2Mb	100%	100%	100%
	3 x	192	4Mb	100%	100%	99.9%

Thanks: Experiments conducted by Paul Barford and Joel Sommers, U of Wisconsin

In Summary

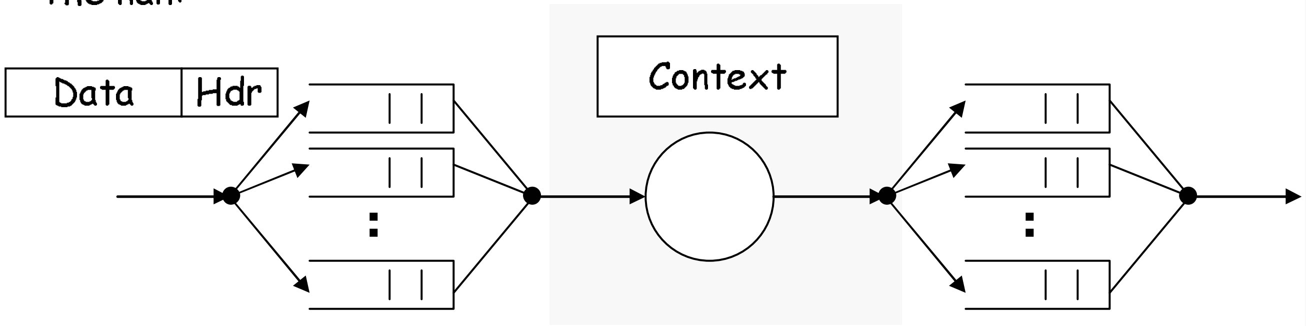
- Buffer size dictated by long TCP flows.
- 10Gb/s linecard with 200,000 x 56kb/s flows
  - Rule-of-thumb: Buffer = 2.5Gbits
    - Requires external, slow DRAM
  - Becomes: Buffer = 6Mbits
    - Can use on-chip, fast SRAM
    - Completion time halved for short-flows
- 40Gb/s linecard with 40,000 x 1Mb/s flows
  - Rule-of-thumb: Buffer = 10Gbits
  - Becomes: Buffer = 50Mbits

Outline

- What I <u>was</u> going to say
- Network processors and their memory
  - Packet processing is all about getting packets into and out of a chip and memory.
  - Computation is a side-issue.
  - Memory speed is everything: Speed matters more than size
- Remarks

My 2c on network processors

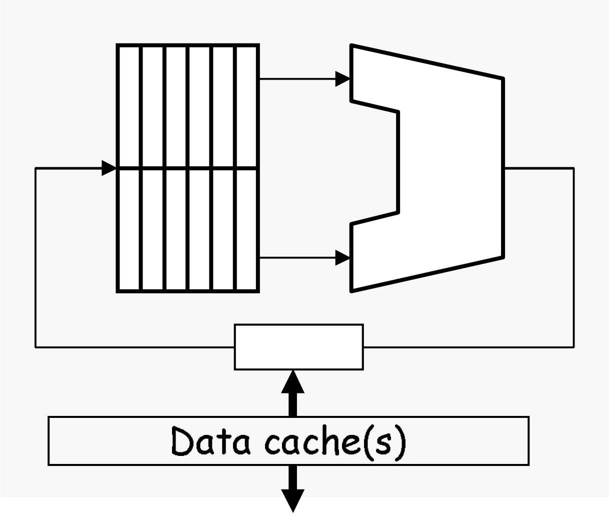
The nail:



Characteristics:

1. Stream processing.
2. Multiple flows.
3. Most processing on header, not data.
4. Two sets of data: packets, context.
5. Packets have no temporal locality, and special spatial locality.
6. Context has temporal and spatial locality.

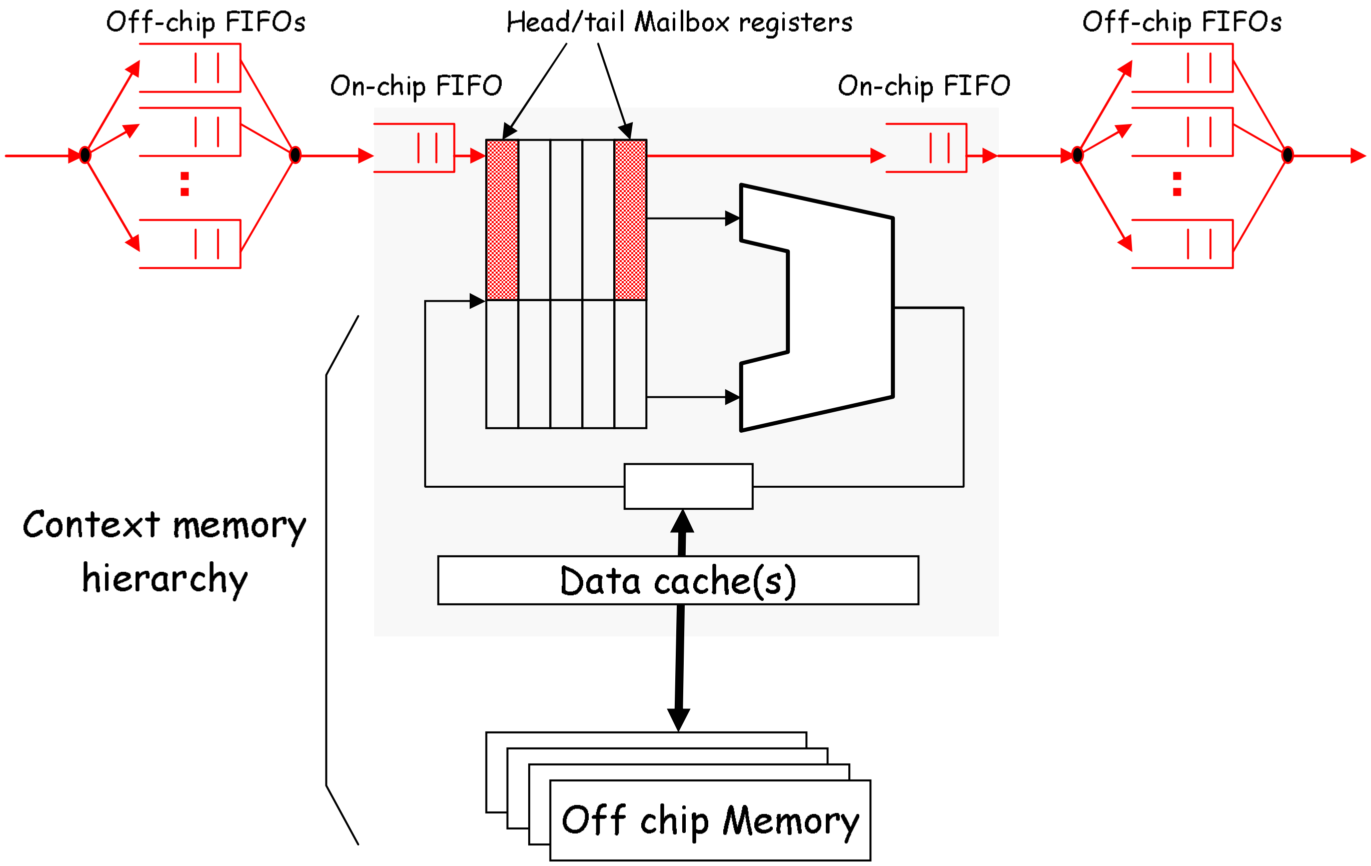
The hammer:



Characteristics:

1. Shared in/out bus.
2. Optimized for data with spatial and temporal locality.
3. Optimized for register accesses.

A network uniprocessor



Add hardware support for multiple threads/contexts.

Recommendations

- Follow the CPU lead…
- Develop quality public benchmarks
- Encourage comparison and debate
- Develop a “DLX” NPU to
  - Compare against
  - Encourage innovation in:
    - Instruction sets
    - (Parallel) programming languages and development tools
- Ride the coat-tails of CPU development
- Watch for the CPU with networking extensions
- NPUs are about memory not computation.
- Memory speed matters more than size.