Variable Packet Size Buffered Crossbar (CICQ) Switches

Manolis Katevenis, Georgios Passas, Dimitrios Simos,
Ioannis Papaefstathiou, and Nikos Chrysos

FORTH and U. Crete, Greece

Outline

- Background:

  Buffered Crossbars are good

    "Combined Input-Crosspoint Queueing (CICQ)"

- Foreground:

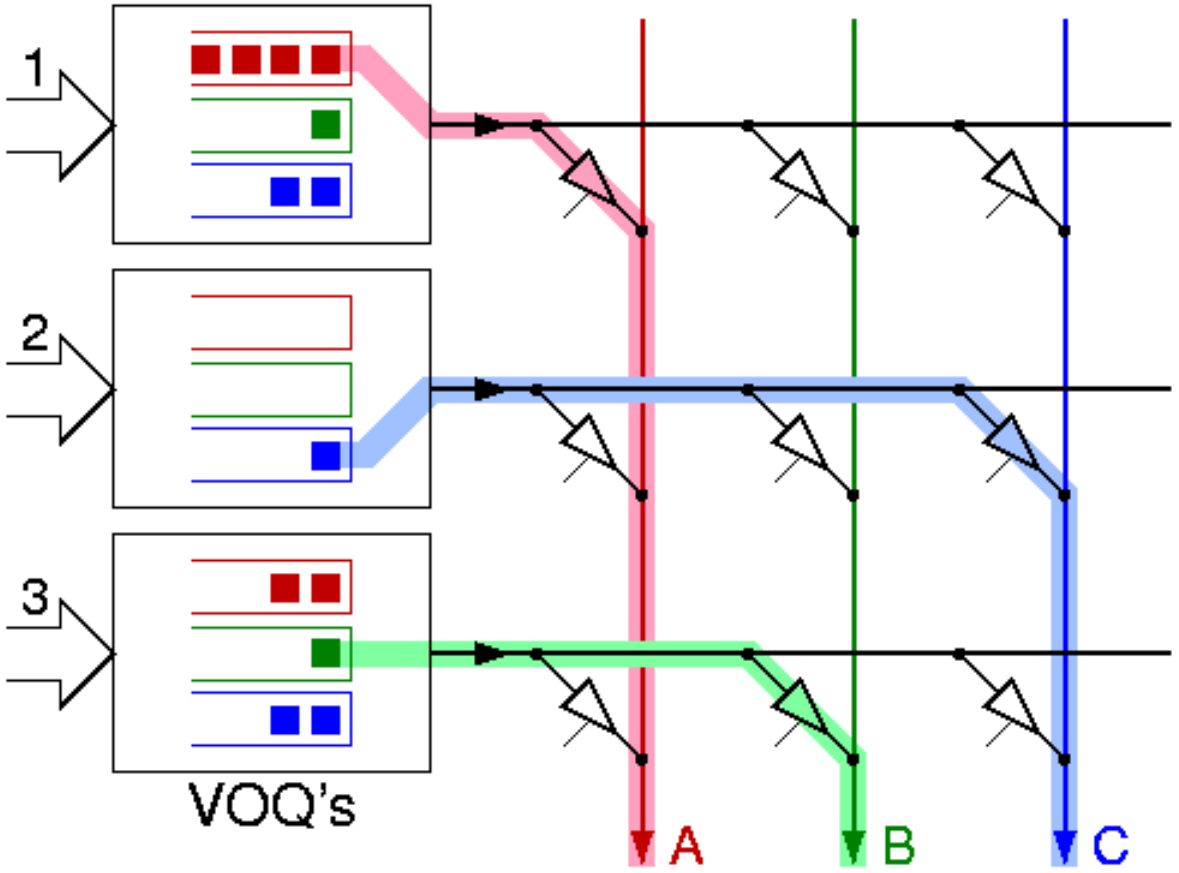
  Variable-Packet-Size BufXbars are even better

  - no SAR ➔ no speedup ➔ higher line rate
  - no output queues ➔ lower cost

- Contributions:

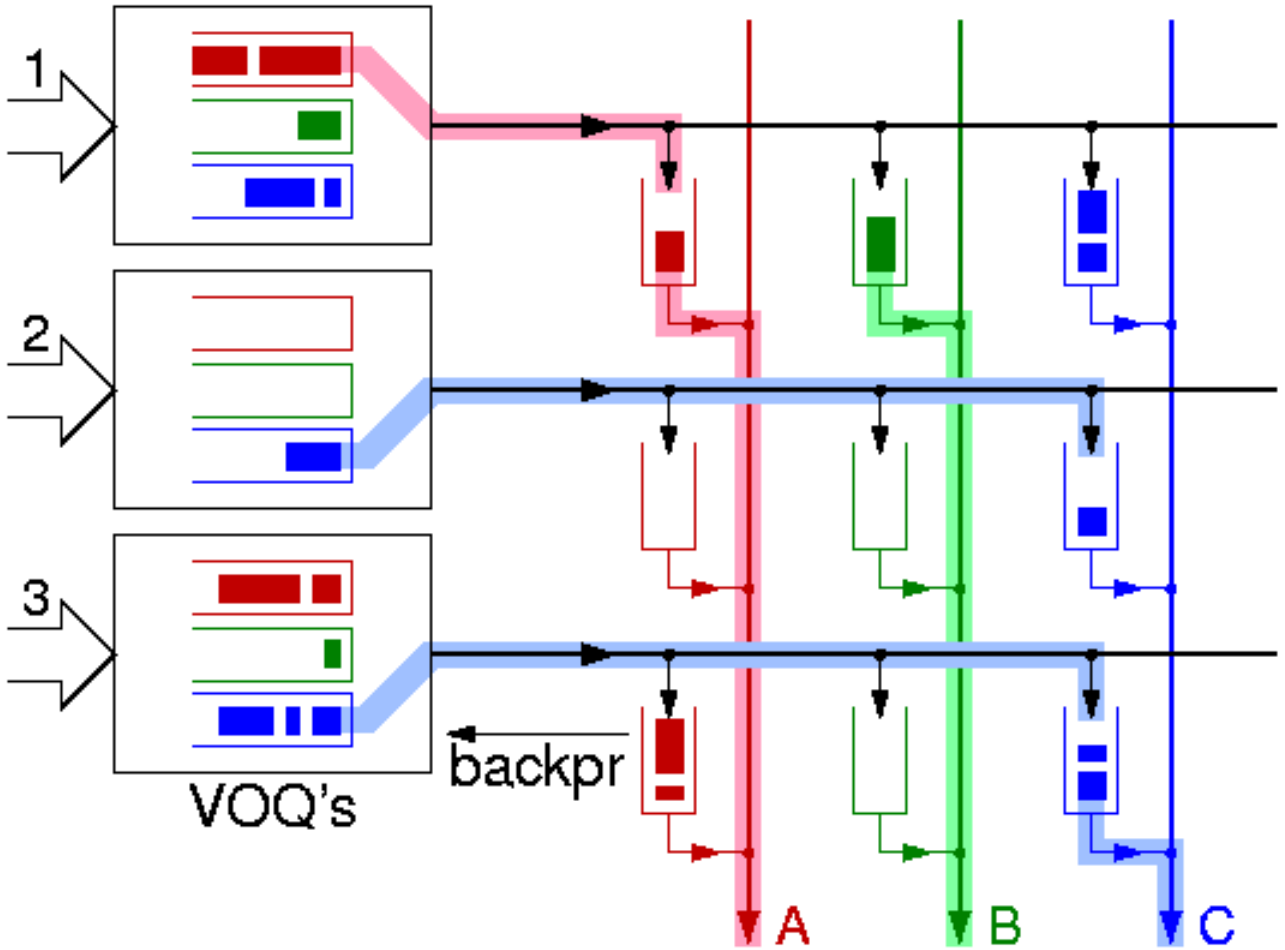
  - performance evaluation – more extensive & accurate
  - chip design ➔ verification, area, power

Background: Unbuffered Crossbar



- No output conflicts allowed: dependent scheduler decisions
  → central scheduling, fixed-size cell operation

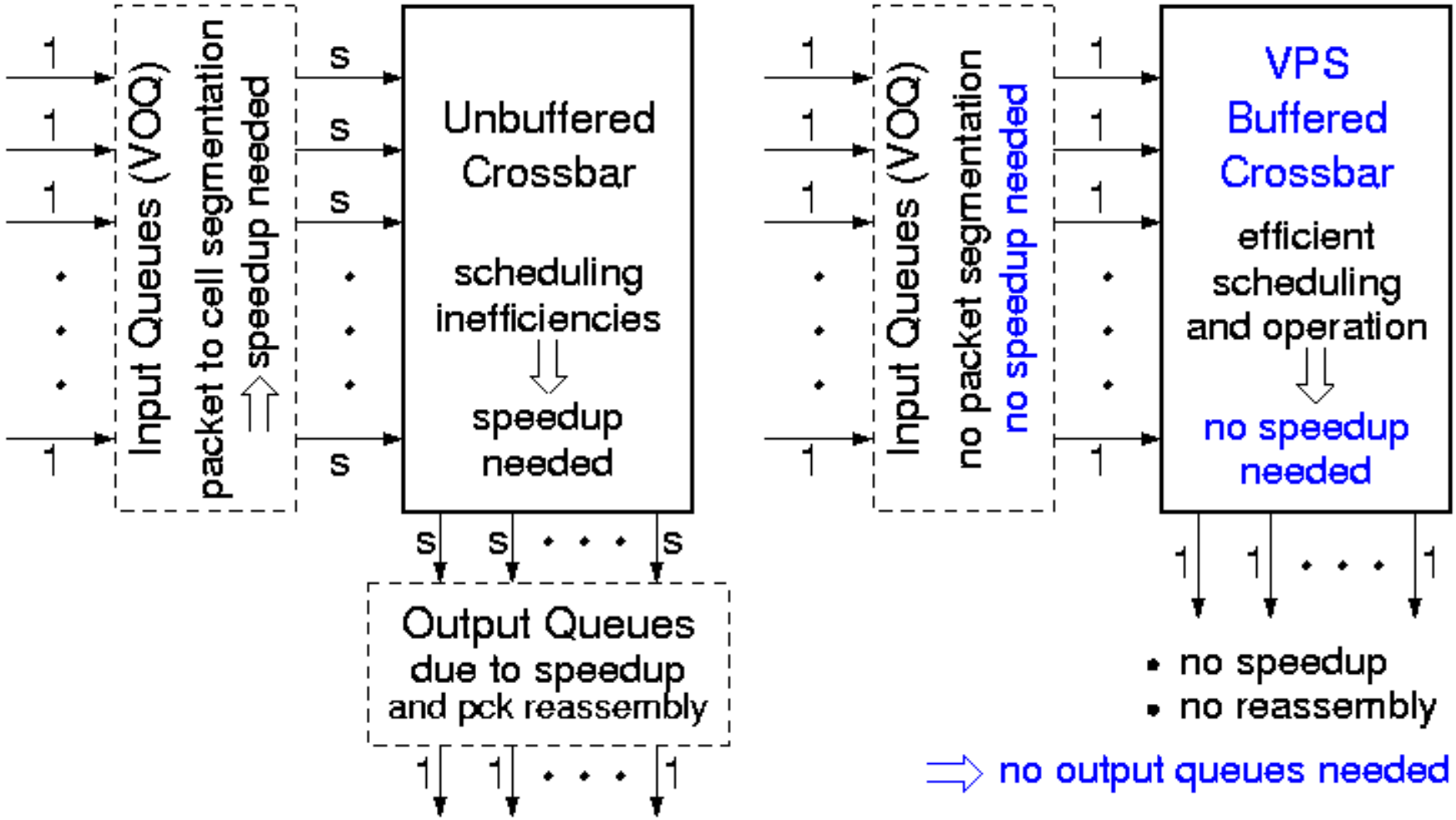
Buffered Crossbar (CICQ):



- Independent decisions: distributed scheduling

  → can operate directly on variable-size packets

Variable Packet Size (VPS) Buffered Crossbar



- With same-speed crossbar:
  ➔ s times faster line rate with VPS buffered crossbar (s = 2 to 3)

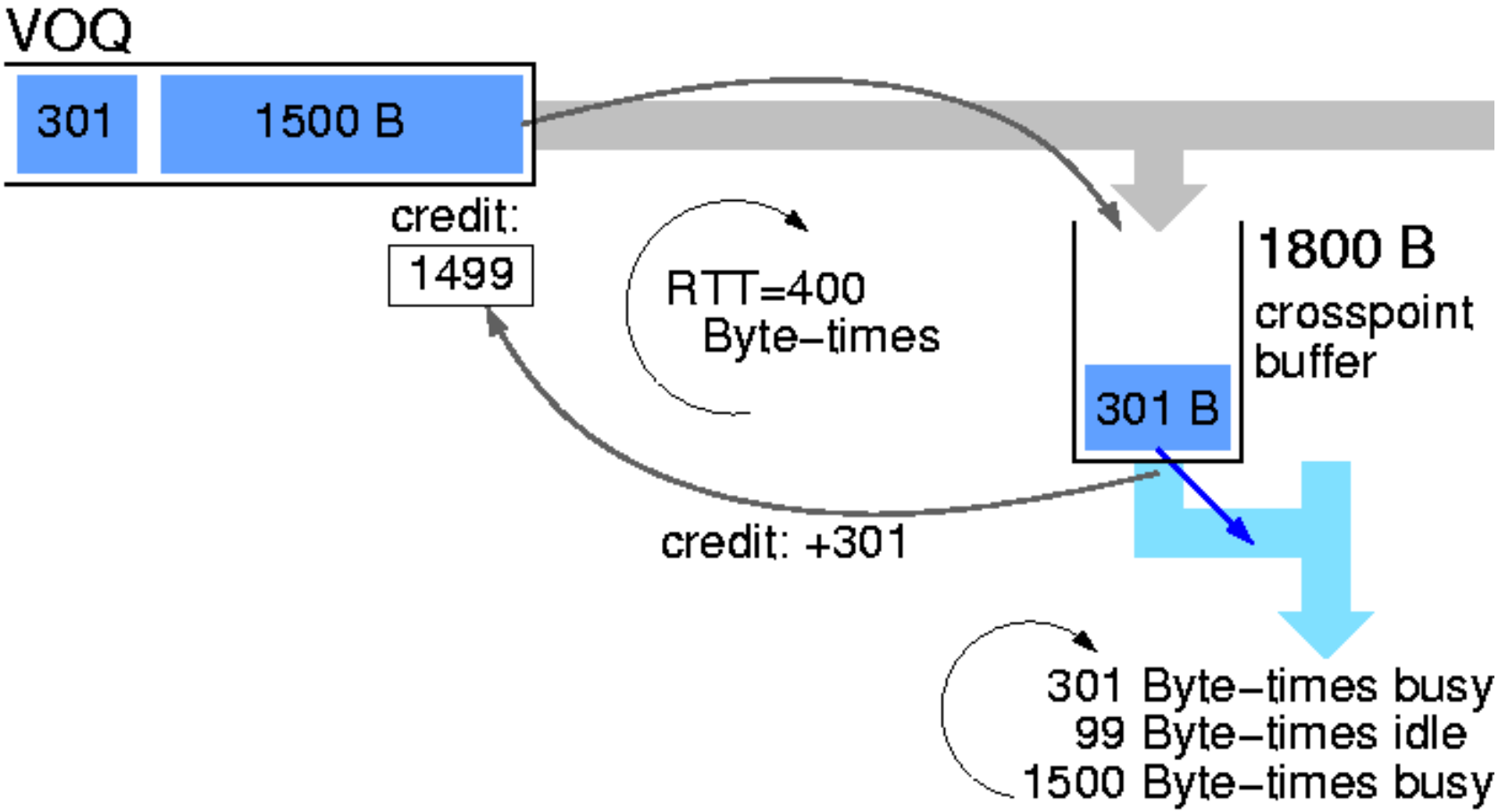
Contributions:

- Performance Evaluation:
  - Crosspoint buffer sizing
  - under Internet-style, uniformly-destined traffic
  - Hot-spots: no degradation to others –see paper
  - under Unbalanced traffic –see paper
- Full Chip Design:
  - Cut-through
  - Verification
  - Area & power, per function

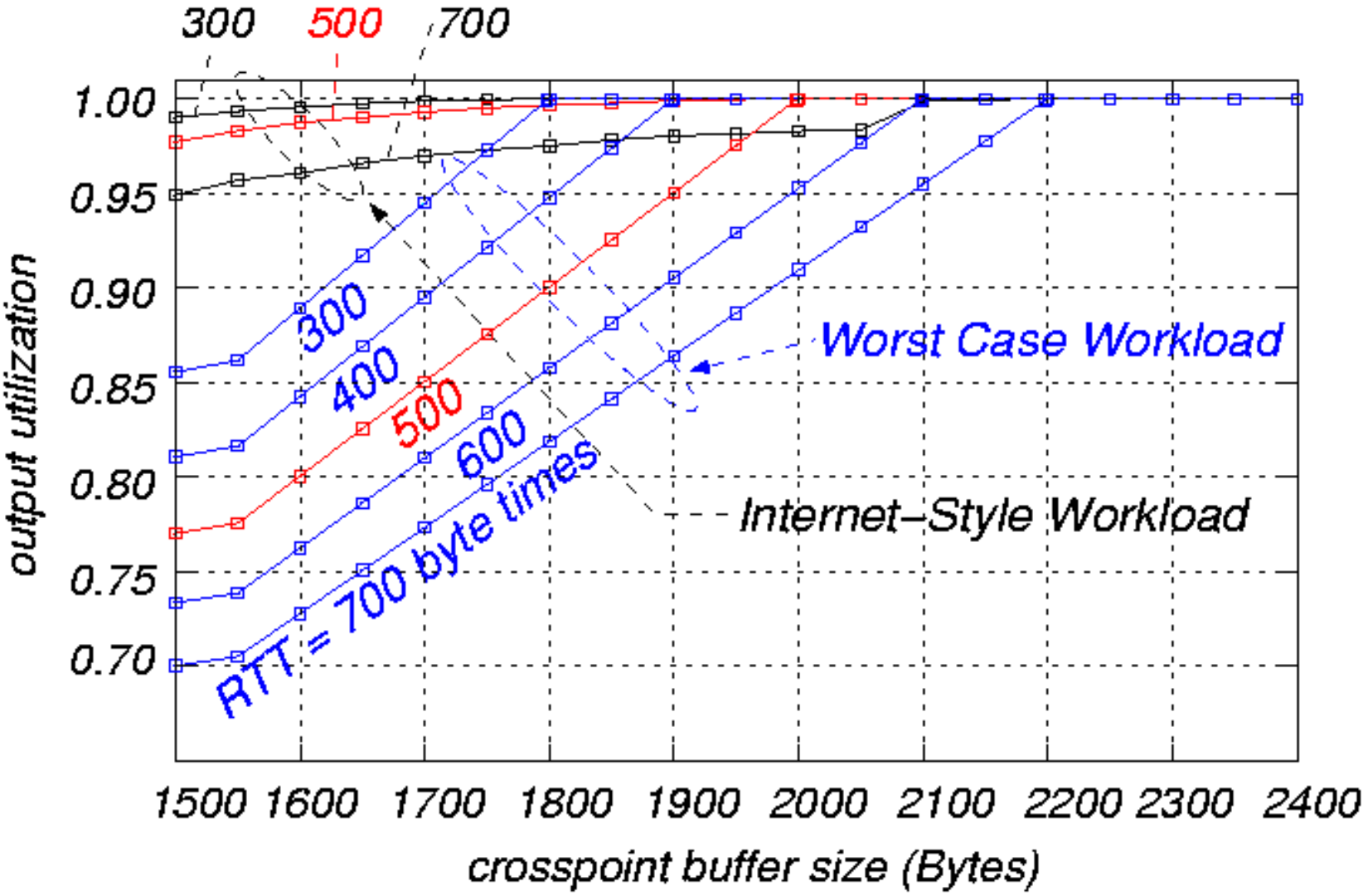
Crosspoint Buffer Sizing

- For full throughput under worst-case single active flow:

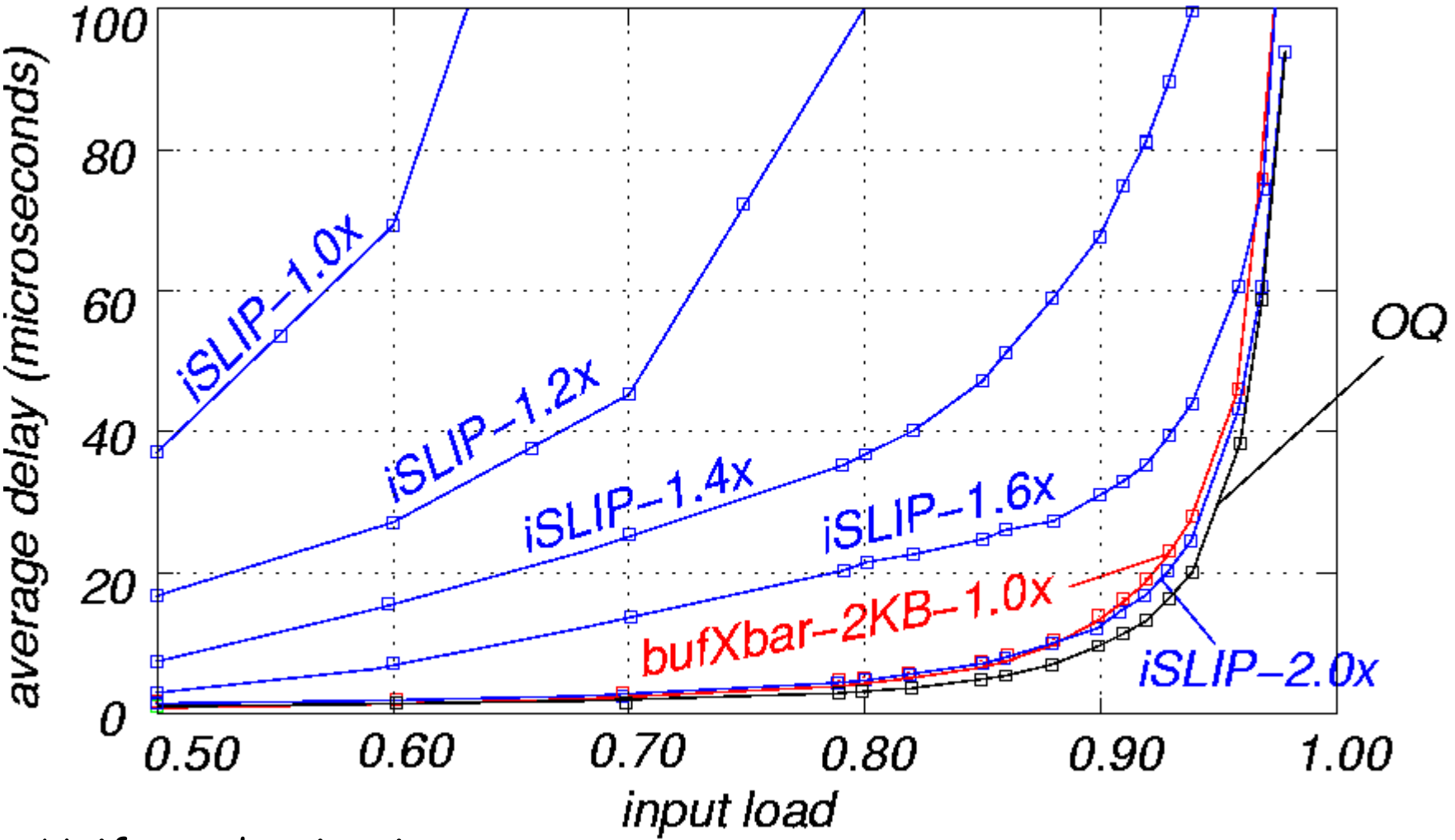
  CrosspBufSize ≥ MaxPacketSize + RTTwindow



Crosspoint Buffer ≥ MaxPckSize + RTTwindow



No Speedup needed to approach Output Queuing



- Uniform destinations
- Internet-style synthetic workload; 40-1500 byte packet sizes
- Unbuffered crossbar w. SAR: one-iteration iSLIP, 64-byte segments

A VPS Buffered Crossbar Chip Design

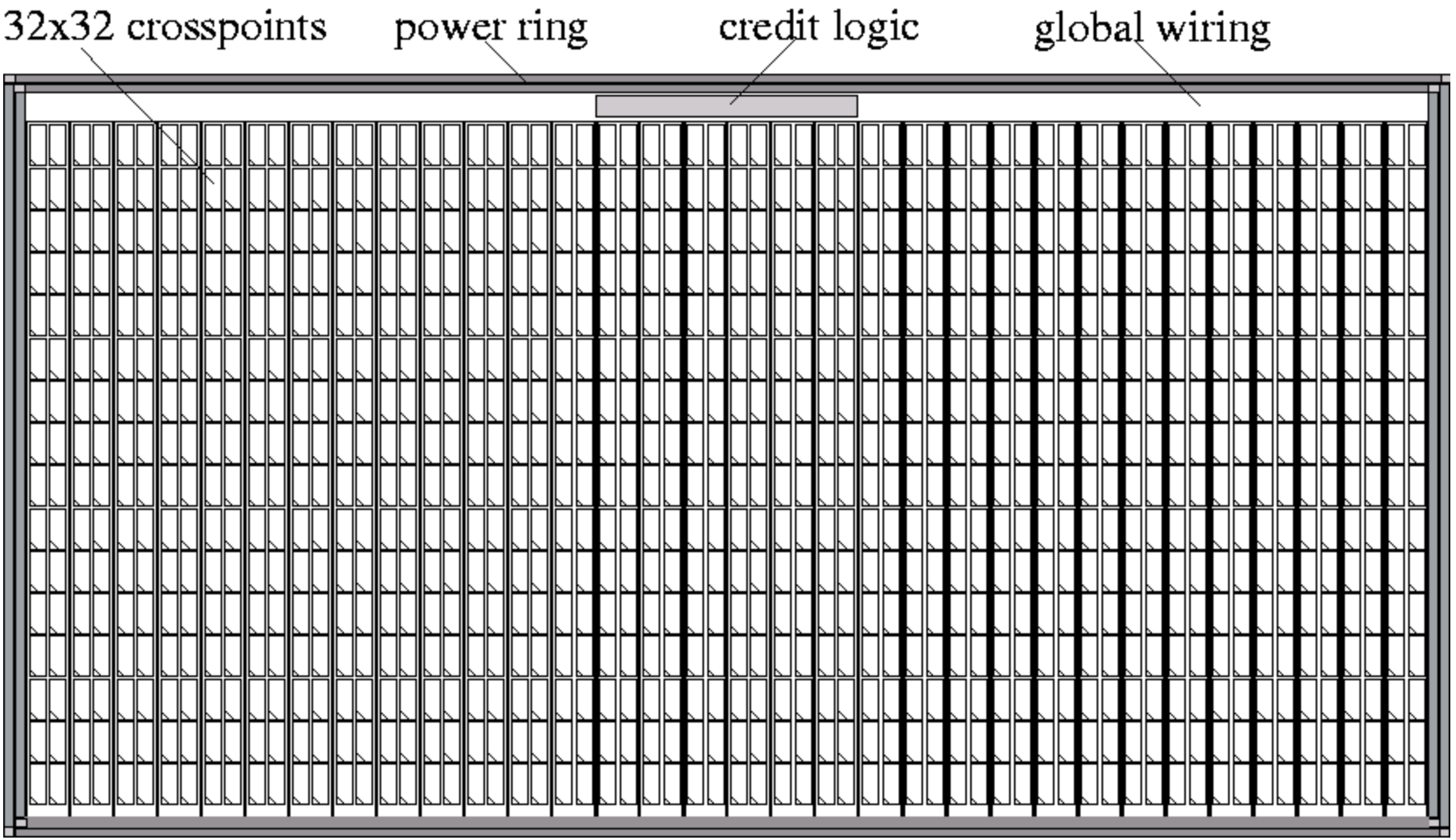
- 32x32 ports, 300 Gbps aggregate throughput
- 2 KBytes / crosspoint buffer x 1024 crosspoints
- Variable-size packets (multiples of 4 Bytes)
- 32-bit datapaths
- Cut-through at the crosspoints
- Fully designed, in Verilog
  - Core only, no pads & transceivers
- Fully verified: Verilog versus C++ performance simulator
- Crosspoint logic = 100 FF + 25 gates (simplicity!)

Chip Design: Synthesis, Placement & Routing

32x32 ports, 300 Gbps

- Synthesized: Synopsys
- Placed & routed: Cadence Encounter, 0.18 µm UMC
  - → Clock frequency: 300 MHz @ 0.18 µm
    (operates at maximum SRAM clock frequency)
  - → Core Power: 6 Watt typical @ 0.18 µm
  - → Core Area: 420 mm^2 @ 0.18 µm, or 200 mm^2 @ 0.13 µm
- **Conclusion:**
  - 0.18 µm: 24x24 ports (or ~ 10x10 ports w. Jumbo frames)
  - 0.13 µm: 32x32 ports @ 10 Gbps/port
  - 0.09 µm: higher port counts and line rates achievable

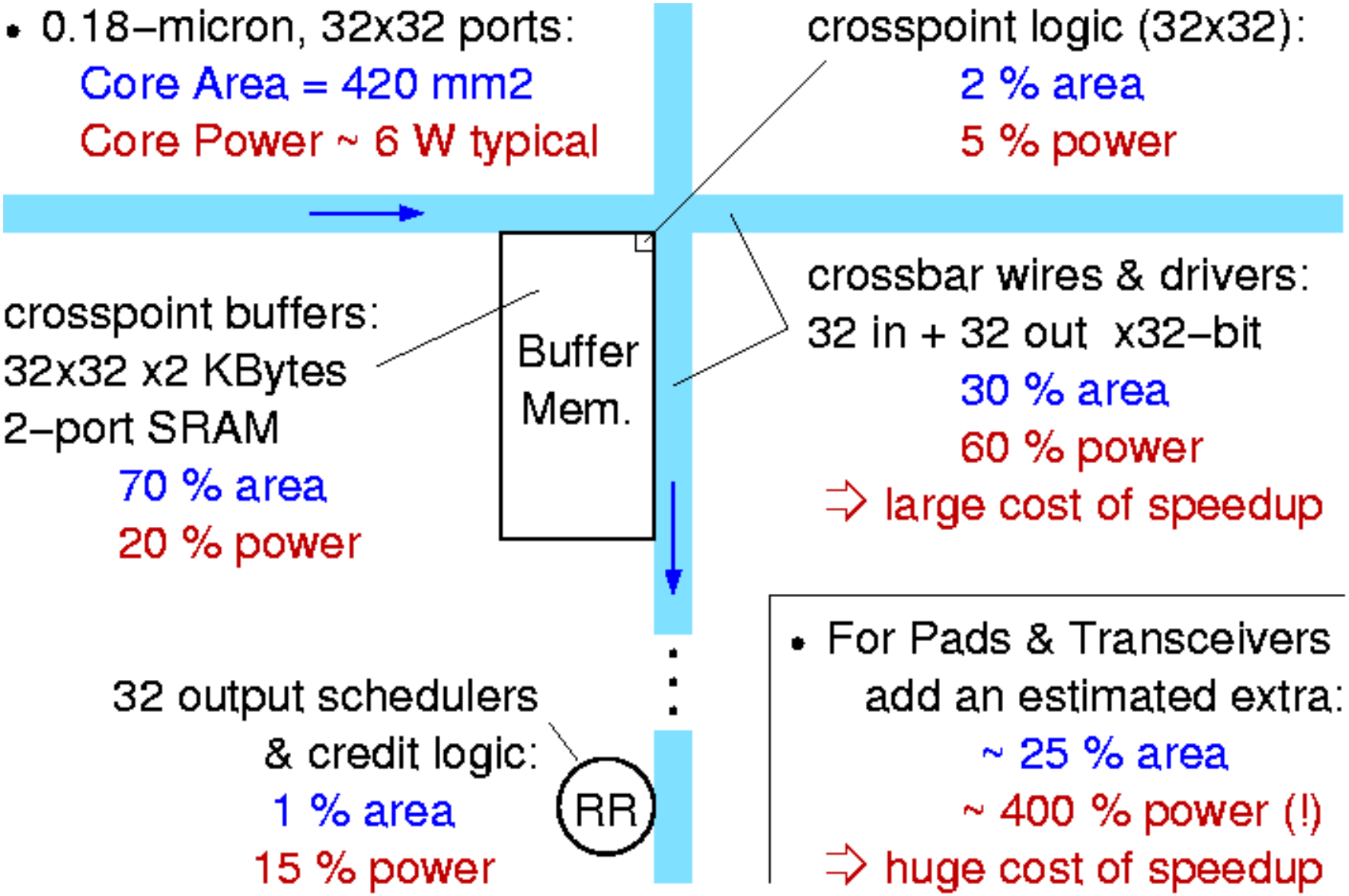
Chip Core Layout



Core Area, Power Allocation:

- 0.18–micron, 32x32 ports:
  - Core Area = 420 mm2
  - Core Power ~ 6 W typical

crosspoint logic (32x32):
- 2 % area
- 5 % power



crosspoint buffers:
32x32 x2 KBytes
2–port SRAM
- 70 % area
- 20 % power

crossbar wires & drivers:
32 in + 32 out x32–bit
- 30 % area
- 60 % power

⇨ large cost of speedup

32 output schedulers
& credit logic:
- 1 % area
- 15 % power

- For Pads & Transceivers add an estimated extra:
  - ~ 25 % area
  - ~ 400 % power (!)

⇨ huge cost of speedup

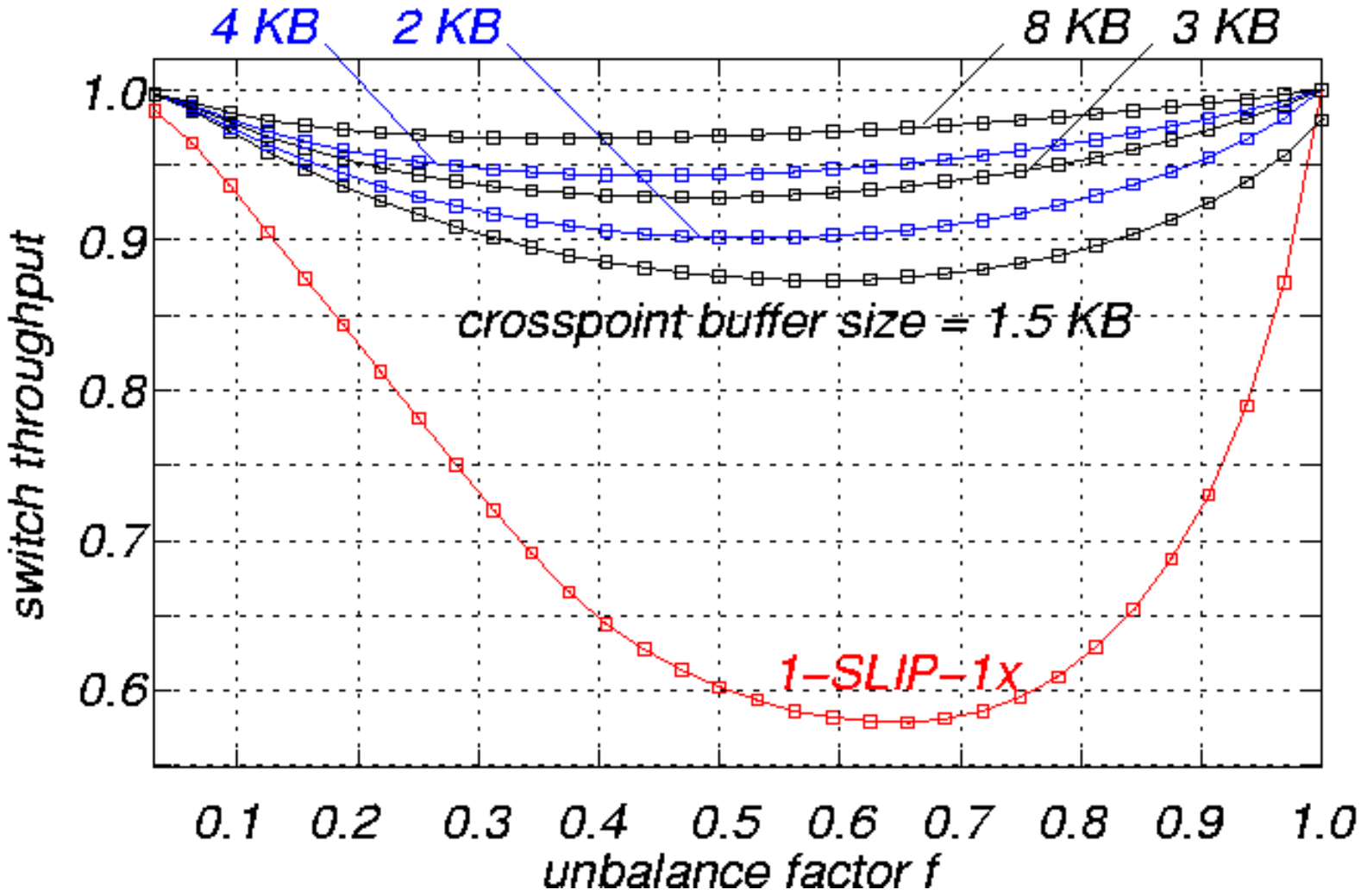
Conclusions

Buffered Crossbars are good

Variable-Packet-Size BufXbars are even better

- no SAR ➔ no speedup ➔ higher line rate
- no output queues ➔ lower cost

Saturation Throughput under Unbalanced Traffic



- Poisson arrivals, Pareto sizes (40-1500)
- For iSLIP, packet sizes are multiples of 64 B (→ no SAR overhead)