Algorithms and Architectures for Optical Packet Fabrics

Dimitrios Stiliadis

Bell Labs Research
Lucent Technologies
stiliadi@bell-labs.com

http://www.bell-labs.com/~stiliadi

work done with: M. Zirngibl et.al.

Presentation Outline

- **Background and Motivation**
- **Problems with Optical Packet Fabrics**
- **Achieving Bandwidth and Delay Guarantees**
- **Technologies and architectures**
- **Applications**

Scalability Issues

- **Where is processing power needed ?**
  - Writing and reading packets to the memory
  - Look-ups and packet filtering
    - Performed at lower timescales (per packet and not per-bit)
  - Switch fabric
- **General issues**
  - Memory bandwidth
  - Speed of electronics
- **Other much neglected issues**
  - Power consumption
  - Power consumption
  - Power consumption
  - Methods for distributing port cards over multiple shelves
  - Space requirements

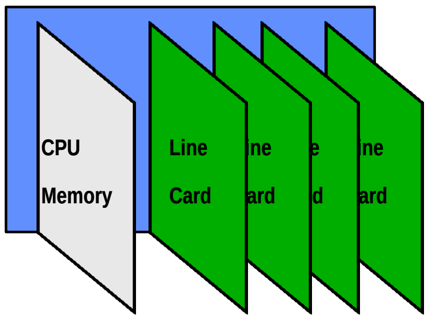
Is there a future for electronic routers and switches at the Core of the network ?

Evolution of Packet Switch Architectures

Bell Laboratories

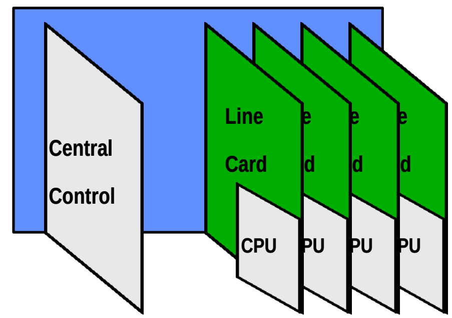
First Generation

Single CPU – multiple line cards
Single electrical backplane



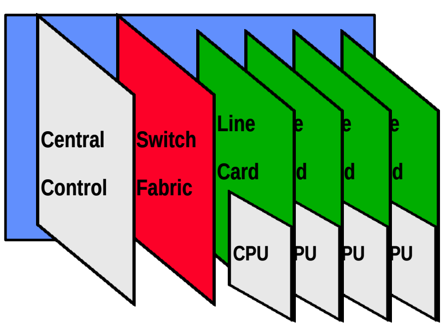
Second Generation

One CPU per Line Card
Central Controller for Routing Protocols



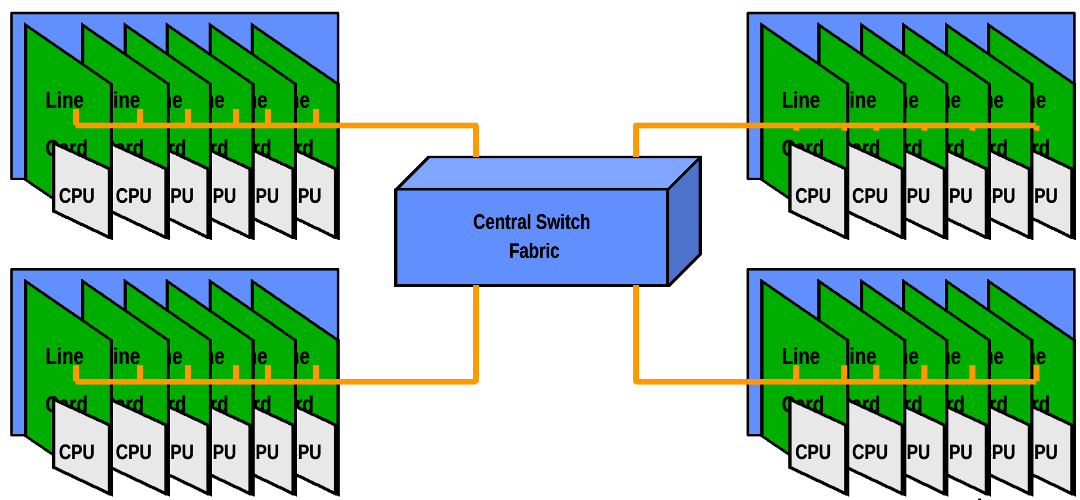
Third Generation

One CPU per Line Card
Central Controller for Routing Protocols
Switch Fabric for inter-connection



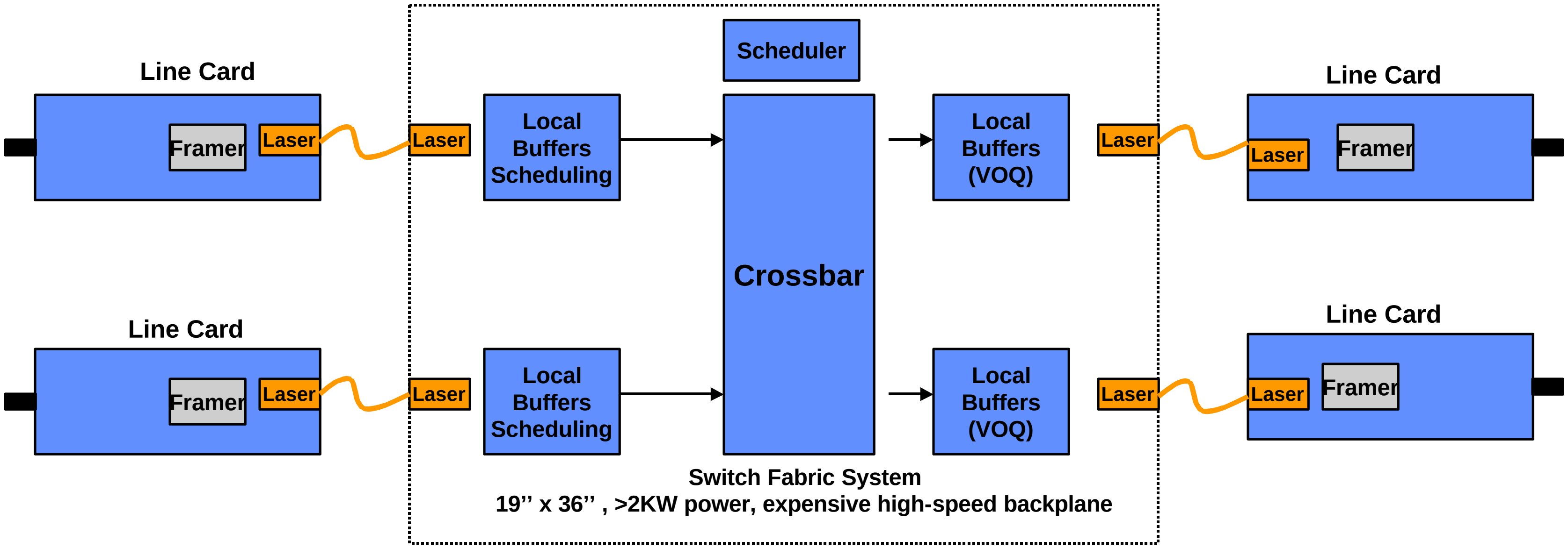
Fourth Generation

Multiple shelves of Line Cards
Centralized Switch Fabric
Optical links interconnecting Line Cards and Fabric



Logical Architecture of Multi-Shelf Switches

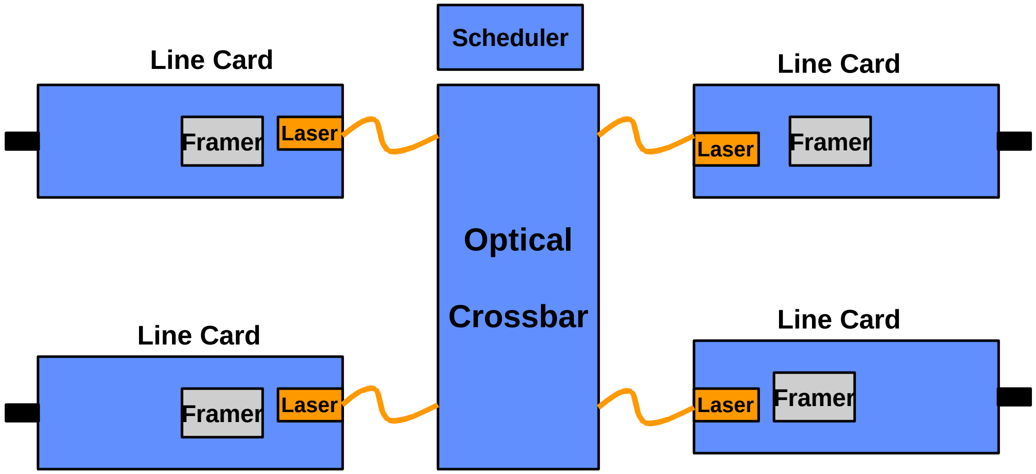
Bell Laboratories



- Data are framed into 64-byte envelopes and transmitted to fabric
  - ➔ Small envelopes can lead to low latency for small packets
- Fabric stores data in Virtual Output Queues and switches through the cross-bar
- Fragmentation effect due to variable size packets
  - ➔ IP packets are not always integral multiples of 64-byte envelopes
  - ➔ Speed up required
- Power consumption is high
  - ➔ Double laser receivers/transmitters
  - ➔ Buffers on the fabric
  - ➔ Electronic crossbar

What if we could design an optical packet fabric ?

What do we need to build an optical cross-bar ?



- Once packets are framed in envelopes, they can be delivered to the egress Line Cards without any optical-to-electrical conversion
  - ➔ Low power requirements
  - ➔ Scalability
  - ➔ Small space requirements
- Technologies that can be used:
  - ➔ Opto-electronic VLSI
  - ➔ Lithium Niobate Switches
  - ➔ Tunable lasers and Dragone routers
- What are the problems ?

Issues in scaling cross-bar architectures using optical crossbars

- **Technology issues:**
  - ➔ **Clock synchronization, clock jitter**
  - ➔ **Physical distribution**
- **Complexity of arbitration algorithms**
  - ➔ **Increases with number of ports**
- **Overhead for signaling information**
  - ➔ **Issuing requests and receiving grants can be an expensive operation**
- **Small versus large envelopes**
  - ➔ **Latency versus bandwidth utilization**
- **Reconfiguration frequency is the key to scalability**

Technology Issues

- **Increasing number of ports requires spatial distribution of line cards in different shelves**
  - ➔ **Long distances between fabric and line cards**
  - ➔ **Using optical signals seems the obvious solution**
- **Clock skew issues**
  - ➔ **All line cards must be synchronized to start transmission at the same time**
  - ➔ **Central frame clock required**
  - ➔ **Long distances of central controller to Line Card can introduce clock-skew**
- **End-to-end signal recovery**
  - ➔ **End-to-end re-locking of clocks is required**
  - ➔ **PLLs require 25-50 bit-times for frequency locking**
  - ➔ **Assume total envelope size is 64-bytes (or 512-bits)**
    - **Up to 10% of the bandwidth wasted only on clock synchronization**

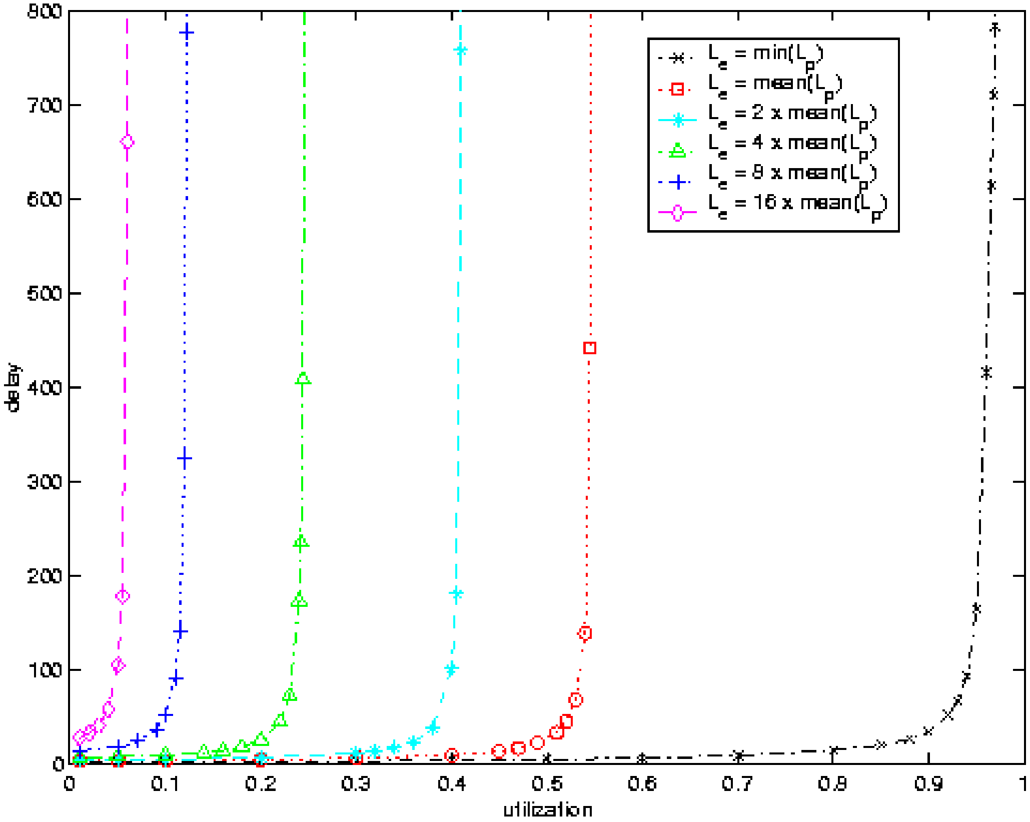
Arbitration Complexity

- **Simple maximal matching algorithms**
  - ➔ **O(log N) complexity for parallel implementations (assume N processors)**
  - ➔ **Performance problems under non-uniform traffic assumptions**
- **Weighted maximal matching**
  - ➔ **O(N^2) complexity for exact implementation**
  - ➔ **Performance independent of traffic patterns**
  - ➔ **Starvation issues**
- **Emulating of output buffered switches with a speed-up of 2**
  - ➔ **Centralized implementation with O(N) complexity**
  - ➔ **However, speed-up of 2 means that half the time is available for the arbitration algorithm**
- **As bandwidth speeds increase, matching algorithms become very difficult to implement**
  - ➔ **A 64-byte envelope at OC768 speeds requires an 8-ns algorithm for arbitration**
  - ➔ **Since arbitration is a function of switch ports it does not scale to large fabrics**

Signaling information

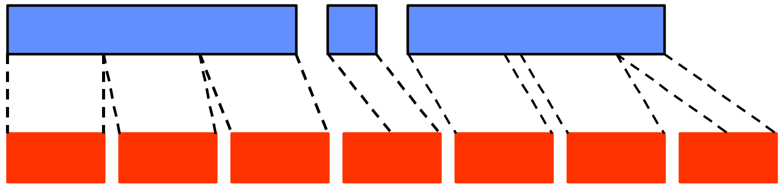
- **Arbitration algorithms require up-to-date information**
  - **In maximal matching, each request can be N bits**
  - **In maximal weighted matching, each request can be N words**
- **Arbiter must distribute schedule to all line cards**
  - **At least one word must be transmitted from the arbiter to every line card during each envelope time**
- **An example:**
  - **Assume a 256x256 40Gbps crossbar switching 64-byte envelopes**
  - **Communication from the arbiter to the fabric requires a 80Mx8bits=640Mbits per second communication path**
    - **Total overhead is 256x640Mbps = 162Gbps**
  - **Communication from the line cards to the arbiter requires at least 640Mbps bandwidth from each line card assuming one update per cell time**
- **Distributed scheduling algorithms might reduce these requirements**
  - **Penalty is the use of old information for scheduling**

Small vs. Large Envelopes

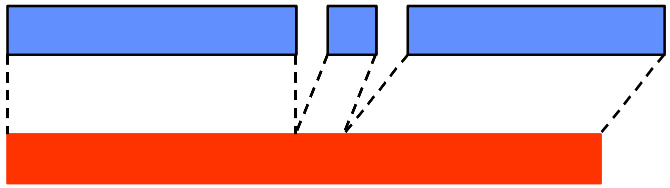


- **As the envelope size becomes larger than the minimum IP packet size the throughput of the crossbar is reduced**
- **Simulations assume 40% of packets are 40 bytes (TCP acks)**

Decreasing the re-configuration frequency



Fragmentation method
Split packets into equal sized envelopes

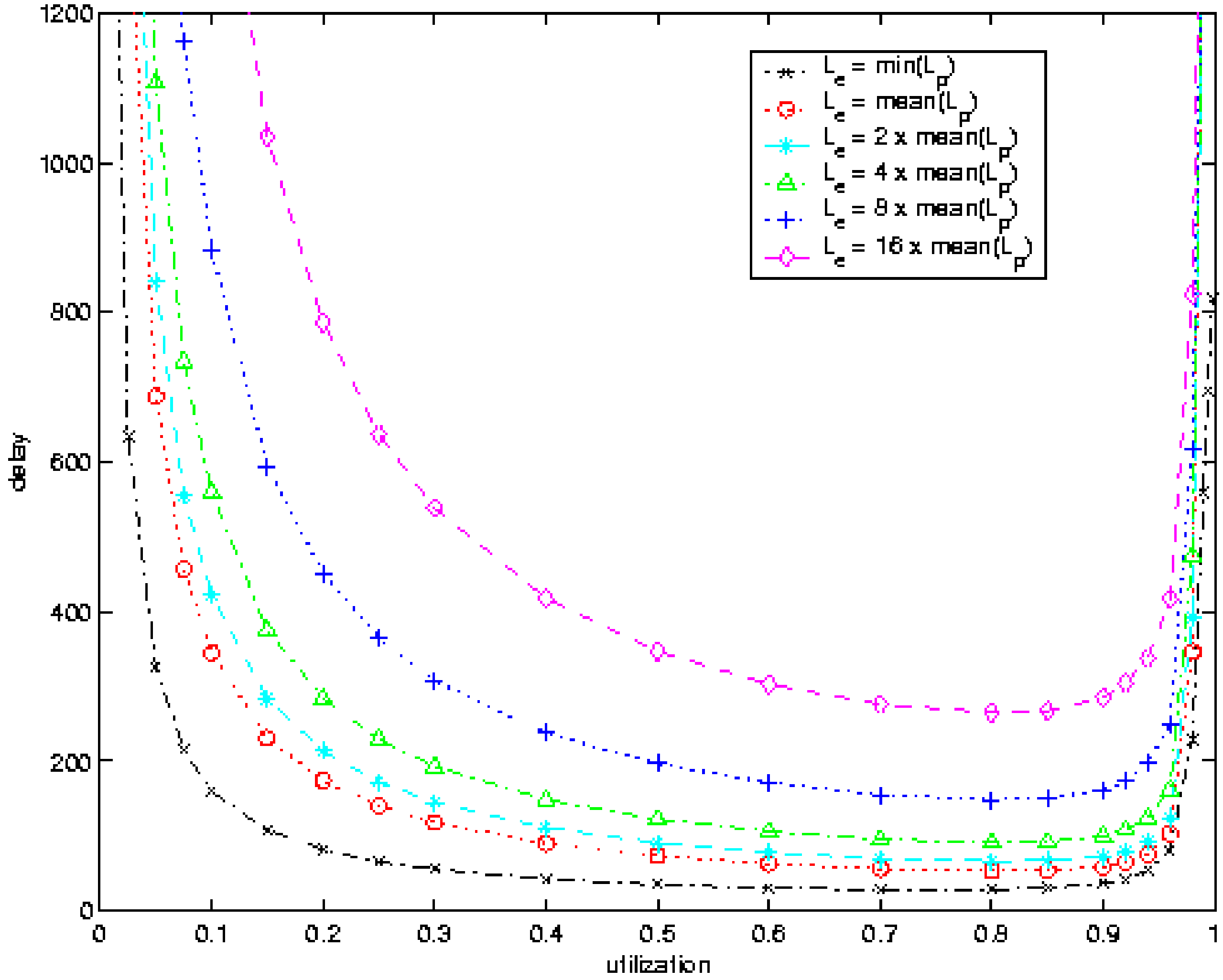


Packetization method
Map multiple packets into large envelopes

- **Cross-bar can switch large envelopes that transfer bursts of packets**
- **Advantages**
  - ➔ **Reconfiguration frequency is a function is reduced to match design parameters**
- **Issues**
  - ➔ **What if there are not enough packets to fill an envelope ?**
  - ➔ **How is latency and end-to-end delay performance affected ?**
  - ➔ **How can we provide delay/bandwidth guarantees with such a scheme ?**

Performance of packetization scheme

Bell Laboratories

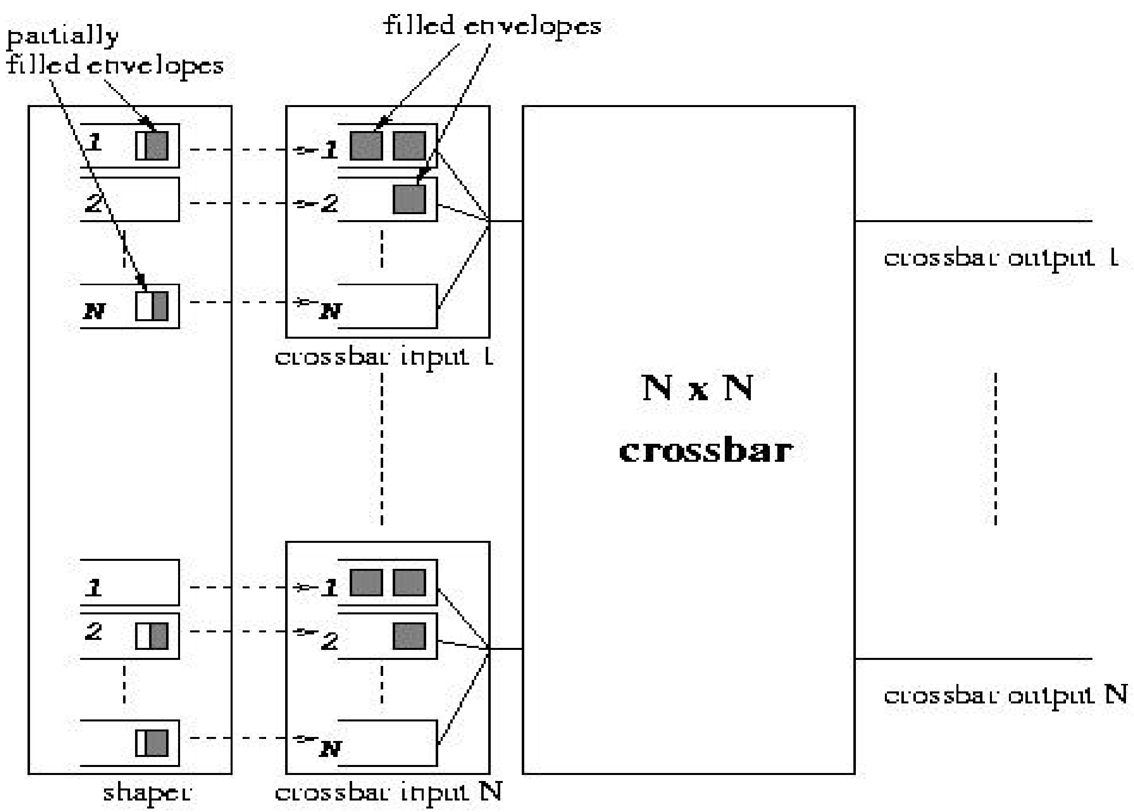


Starvation Issues

- **Releasing half-full envelopes to the crossbar will waste bandwidth**
  - ➔ An input port might have traffic for another output with a full envelope
  - ➔ An output port can receive traffic from another input with a full envelope
- **Envelopes are not allowed to depart unless they are full**
  - ➔ Starvation at low loads
    - A small packet might have to wait a long time until the envelope becomes full
  - ➔ Good performance at high-loads
    - When queues build up there is always enough traffic to fill the envelopes
- **How do we avoid starvation ?**

Avoiding Starvation

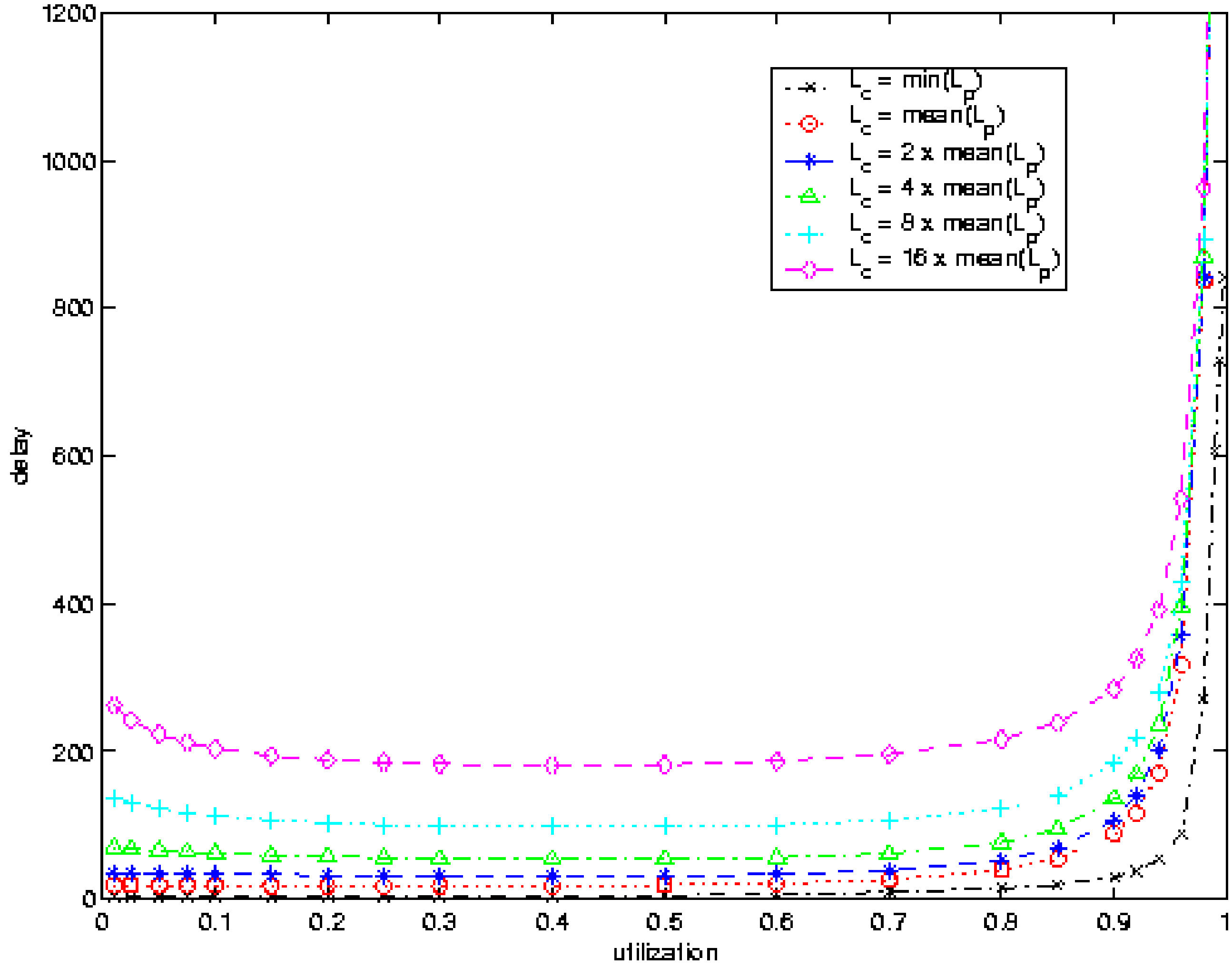
Bell Laboratories



- **Time-out for partially filled envelopes**
  - ➔ Always released after waiting in queues for specific interval of time
- **Half-full envelopes can be forwarded over the fabric to reduce latencies**
- **Configuring time-out interval is possible only when traffic requirements are know a-priori**
  - ➔ Traffic distribution can affect the performance for a given time-out interval

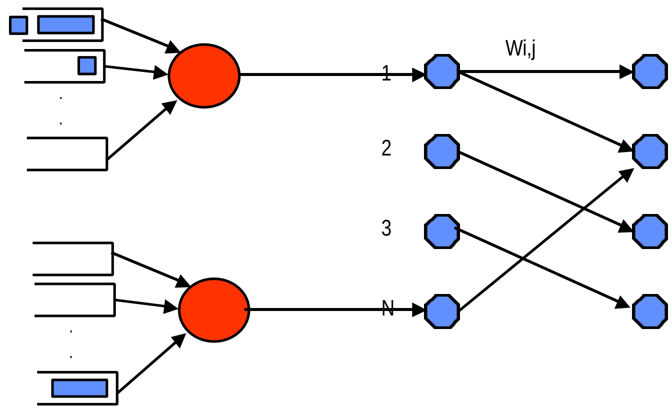
Avoiding Starvation

Bell Laboratories



Delay vs. utilization plots with timeout

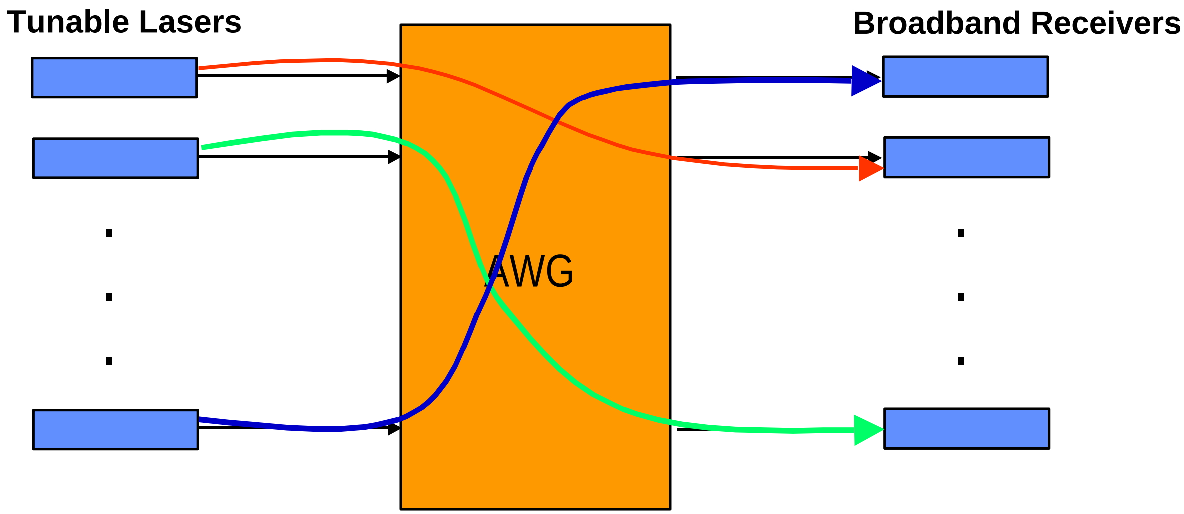
Extending maximal weighted matching



- **Arbitration problem is reduced to a maximal weighted matching in a bi-partite graph**
- **Weights can be a function of the occupancy of each virtual output queue**
  - ➔ **A request is send to the arbiter even for half-full envelopes**
  - ➔ **The weight is equal to the occupancy of the envelope**
- **Algorithm will dynamically adjust weights to maximize the fabric throughput**
  - ➔ **Under-filled envelopes will be transmitted when there is no contention at the inputs or output**
- **Longest queue first approximation**
  - ➔ **In bipartite matching algorithm, serve first the longest queues**

Application of Ideas and Technologies

Array Wave Guides



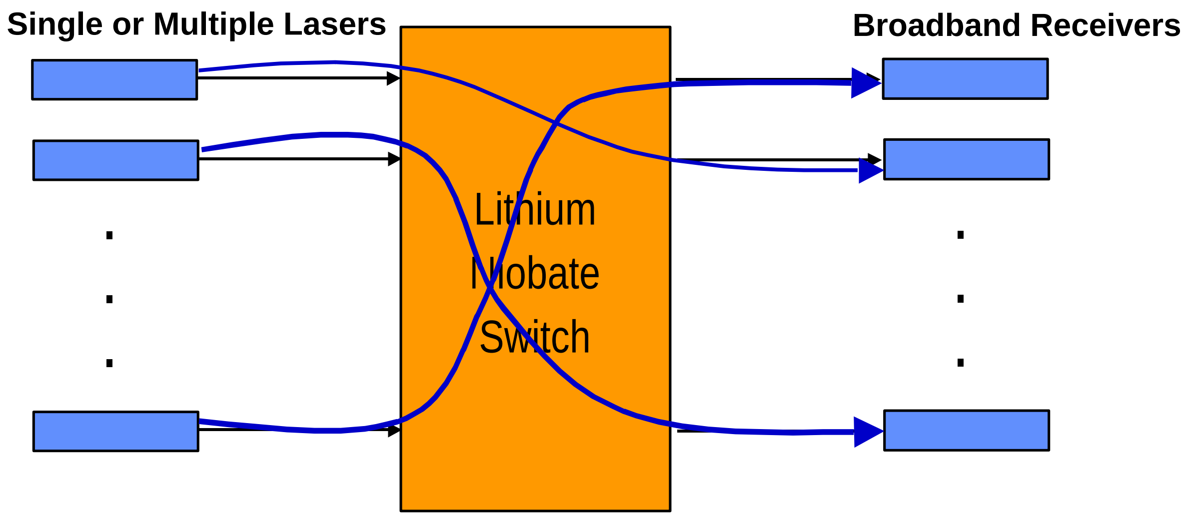
- **Array Wave Guide Router (AWG)**
  - ➔ A passive device made of "prisms"
  - ➔ Optical signals are routed to different directions based on their wavelength
  - ➔ Up to N^2 wavelengths can be "in transit" over an AWG at any time
  - ➔ No electronic or moving parts
- **Switching is achieved by tuning sources to different wavelengths**
- **Broadband receivers can receive at any speed**

Array Wave Guides

- **Issues and limitations**
  - Time required to tune a laser in the order of 50ns
  - Additional time required for clock and data recovery
    - Approximately 16 bit times
- **Scalability**
  - Switch size limited by the number of wavelengths that a given laser can tune into
  - Number of wavelengths decreases with wavelength speed
    - Larger number of wavelengths available for OC192 than for OC768
- **Physical design issues**
  - Temperature stabilization is the largest overhead of the AWG
  - Constant monitoring of signal quality is done in the optical domain
- **Cost**
  - Same devices used in de-multiplexing of DWDM signals
  - Follow the same cost curve as DWDM devices

Lithium Niobate Switches

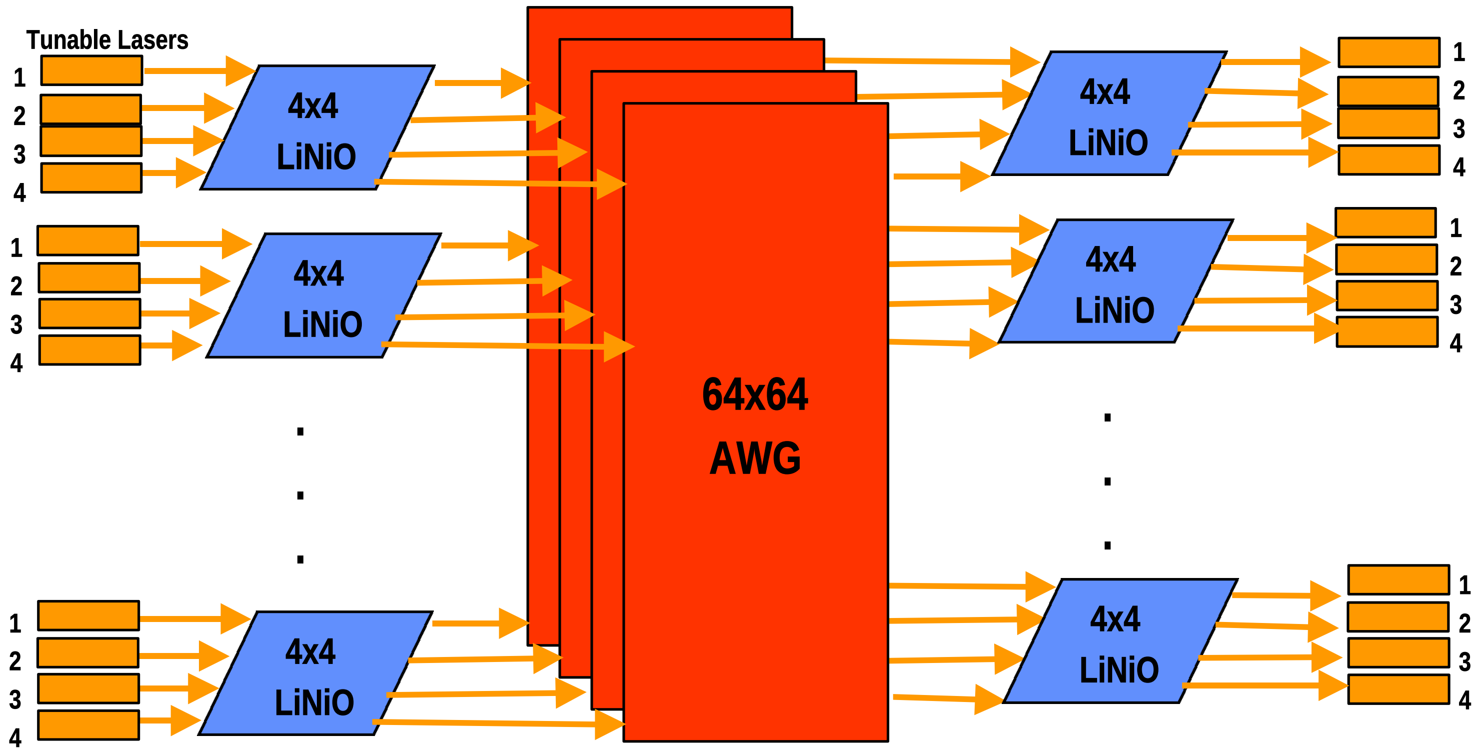
Bell Laboratories



- **Provide a full space cross-bar inter-connect**
- **Any optical signal that arrives in a given input can be routed to any output**
- **Reconfiguration times are very low (less than 5-10ns)**
- **Essentially consist of 2x2 switching devices**
- **Scalability issues:**
  - ➔ High optical losses
  - ➔ Losses increase with the size of the switch
  - ➔ Can not scale to sizes more than 8x8 or 16x16

Combining Technologies

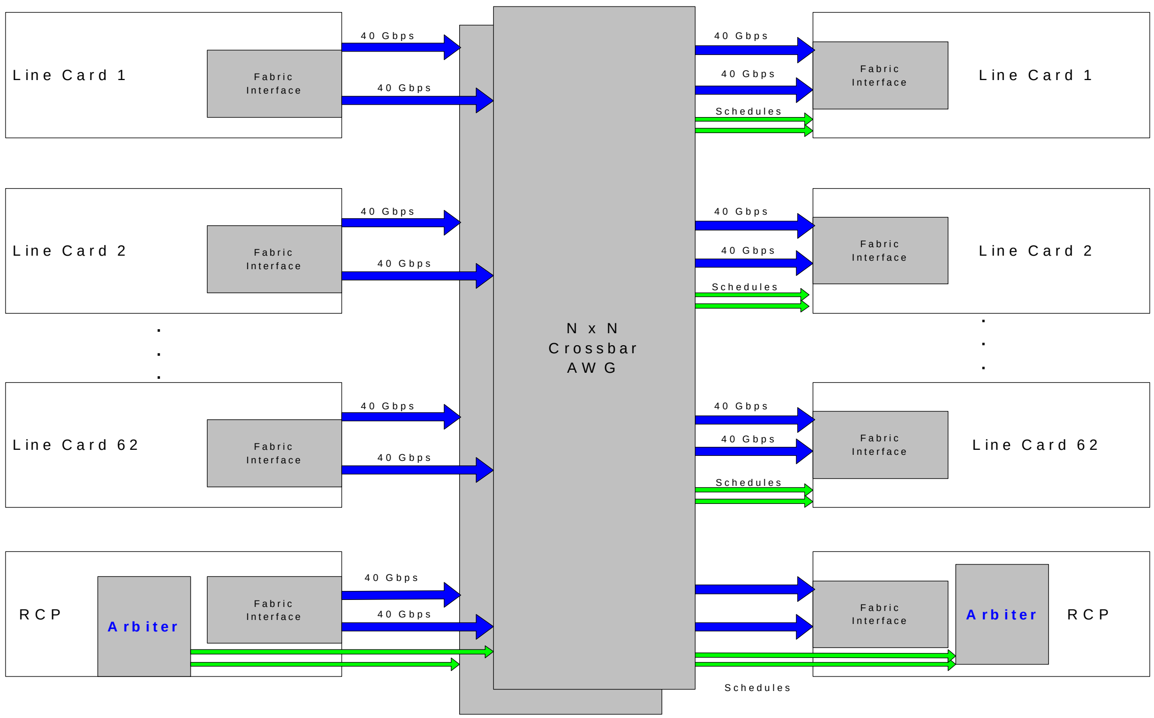
Bell Laboratories



- **Non-blocking Clos network**
  - ➔ Although no speed-up, it is re-configured on every envelope time
  - ➔ Multi-stage buffer-less architecture
- **Tunable lasers configured to appropriate wavelengths**
- **4x4 crossbars distribute load over 4 AWGs**

Solving the Arbiter Communication Problem

Bell Laboratories



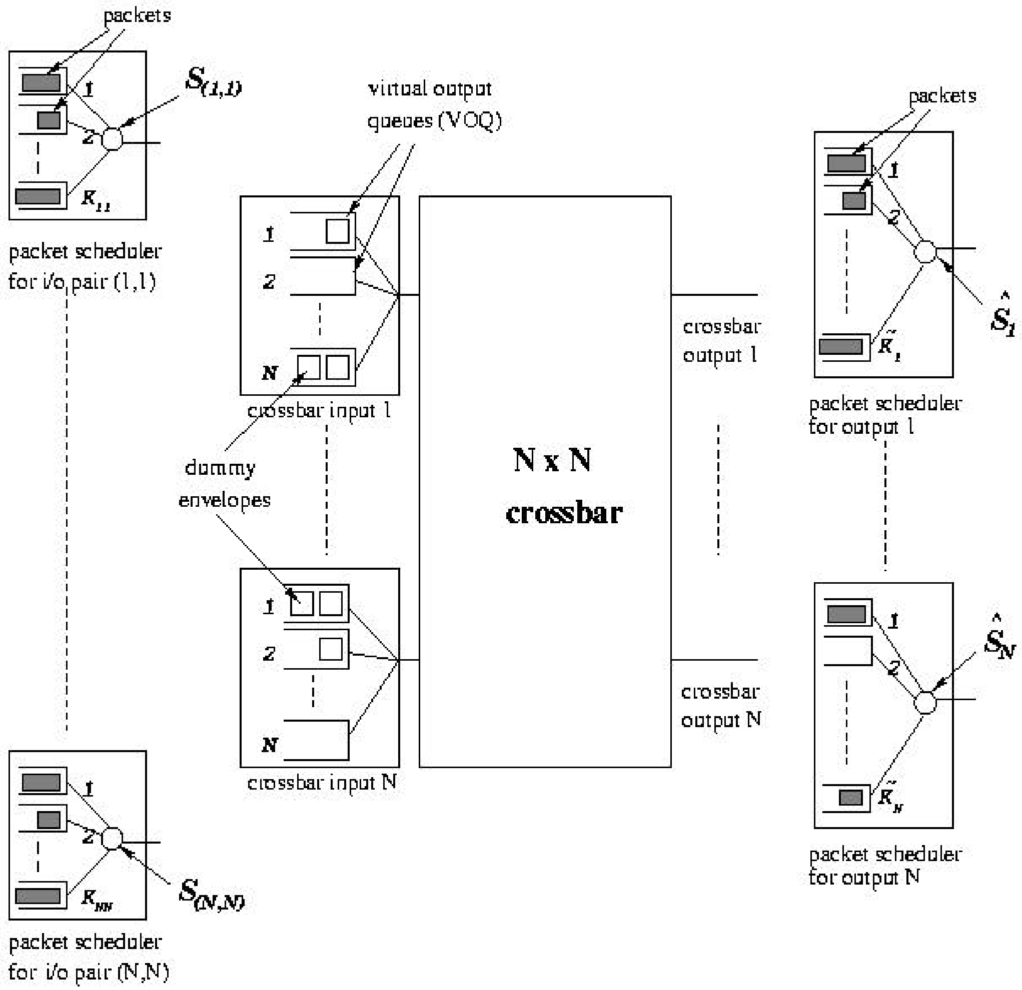
- **Combination of in-band and out-of-band signaling**
  - ➔ Requests from line cards to arbiter are in-band
  - ➔ Round robin access of Line Cards to arbiter
  - ➔ 1/ N of the bandwidth is wasted
  - ➔ Arbiter responses out-of-band
- **Implementation**
  - ➔ Parallel 1Gbps broadcast optical path at 1300 nms distributes schedules

Applications

- **Very high capacity routers**
  - ➔ Hundreds of 40Gbps ports
- **High-speed interconnect for super-computers**
  - ➔ Inter-connecting large number of SMP nodes
  - ➔ SMP like performance required for the overall system
  - ➔ Optical fabric used for memory-to-memory copies
  - ➔ Fabric interface becomes and interface card for super-computers

Some Results on Delay Bounds

Achieving Bandwidth and Delay Guarantees



Achieving Bandwidth and Delay Guarantees

- **If we assume that the crossbar switch can provide a guaranteed bandwidth between any input/output port**
- **The total delay offered to a flow *n* that is shaped by a leaky bucket with parameters (s_n, r_n) is bounded by**

$$\frac{s_n + C_n + L_{n,\max}}{r_n} + \frac{a(i, j)}{r_{(i,j)}}$$

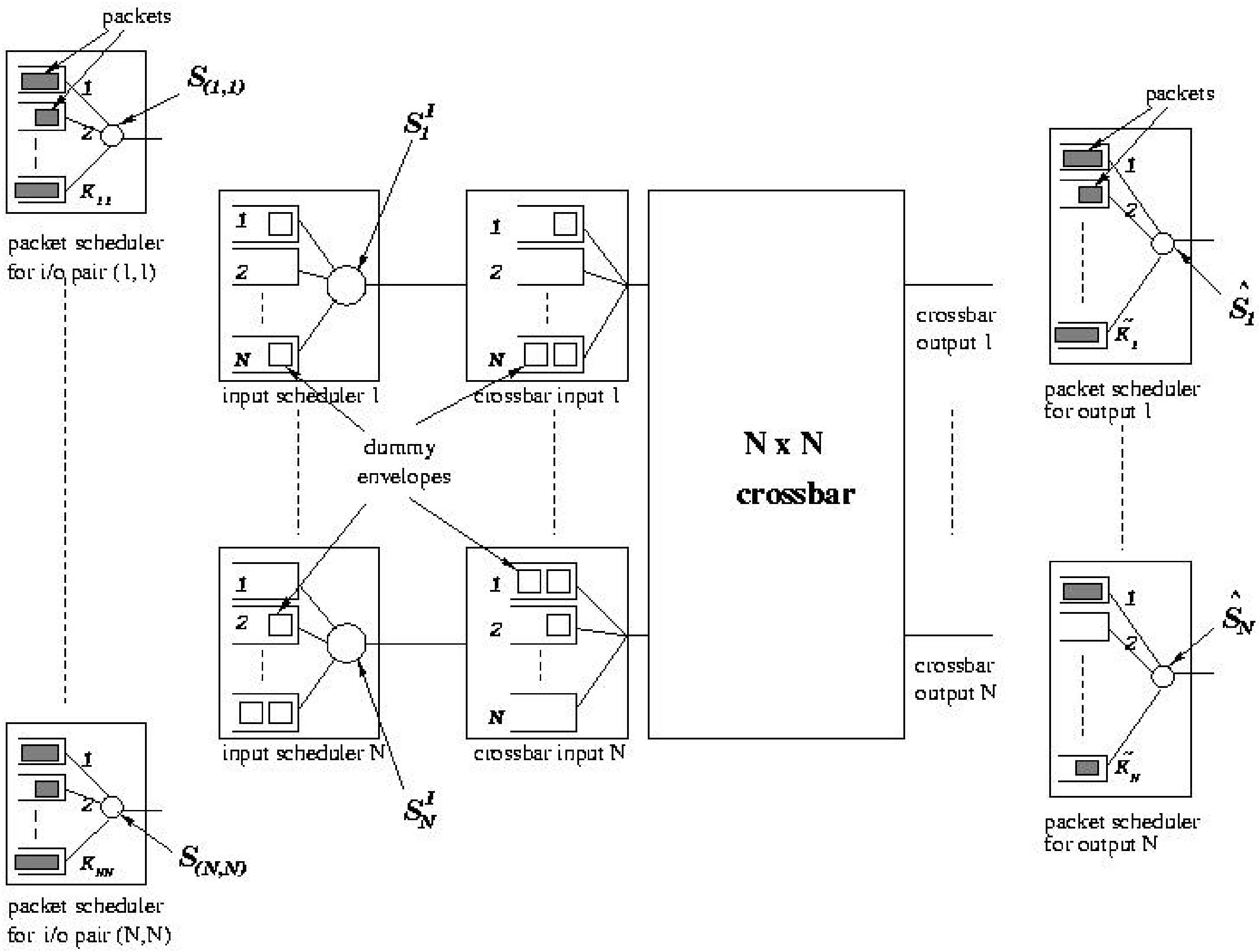
$r_{(i,j)}$ is the guaranteed rate between input i and output j

$L_{p,\max}$ is the maximum packet size of flow n

C_n is a constant that depends on the latency of the scheduler

$a_{(i,j)}$ Is the worst-case fairness index of the crossbar scheduler

Output Queueing Emulation Algorithms



Achieving bandwidth/delay guarantees for output queueing emulation algorithms

An Example Delay Bound

- **Schedulers**
  - input envelope schedulers follow *Shaped Virtual Clock*
  - crossbar emulates *Weighted Fair Queueing*
  - input and output packet schedulers follow *Weighted Fair Queueing*

- **Delay bound** (for flow n going between input i and output j)

$$\frac{s_n + 3L_{p,\max}}{r_n} + \frac{4L_e}{r_{(i,j)}}$$

s_n : burstiness of flow n

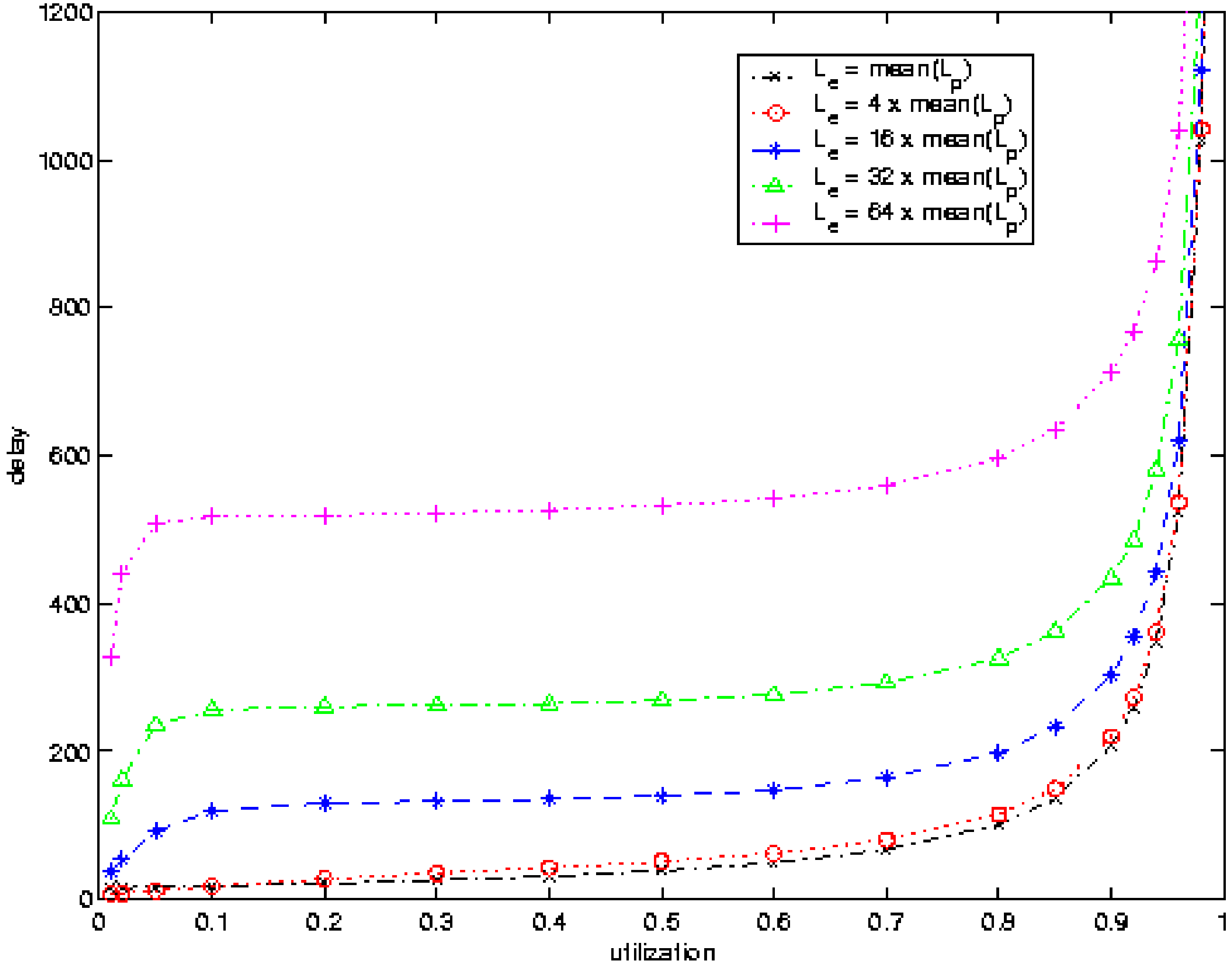
r_n : rate off traffic of flow n

$r_{(i,j)}$: total rate of traffic between input i and output j

$L_{p,\max}$: maximum packet length

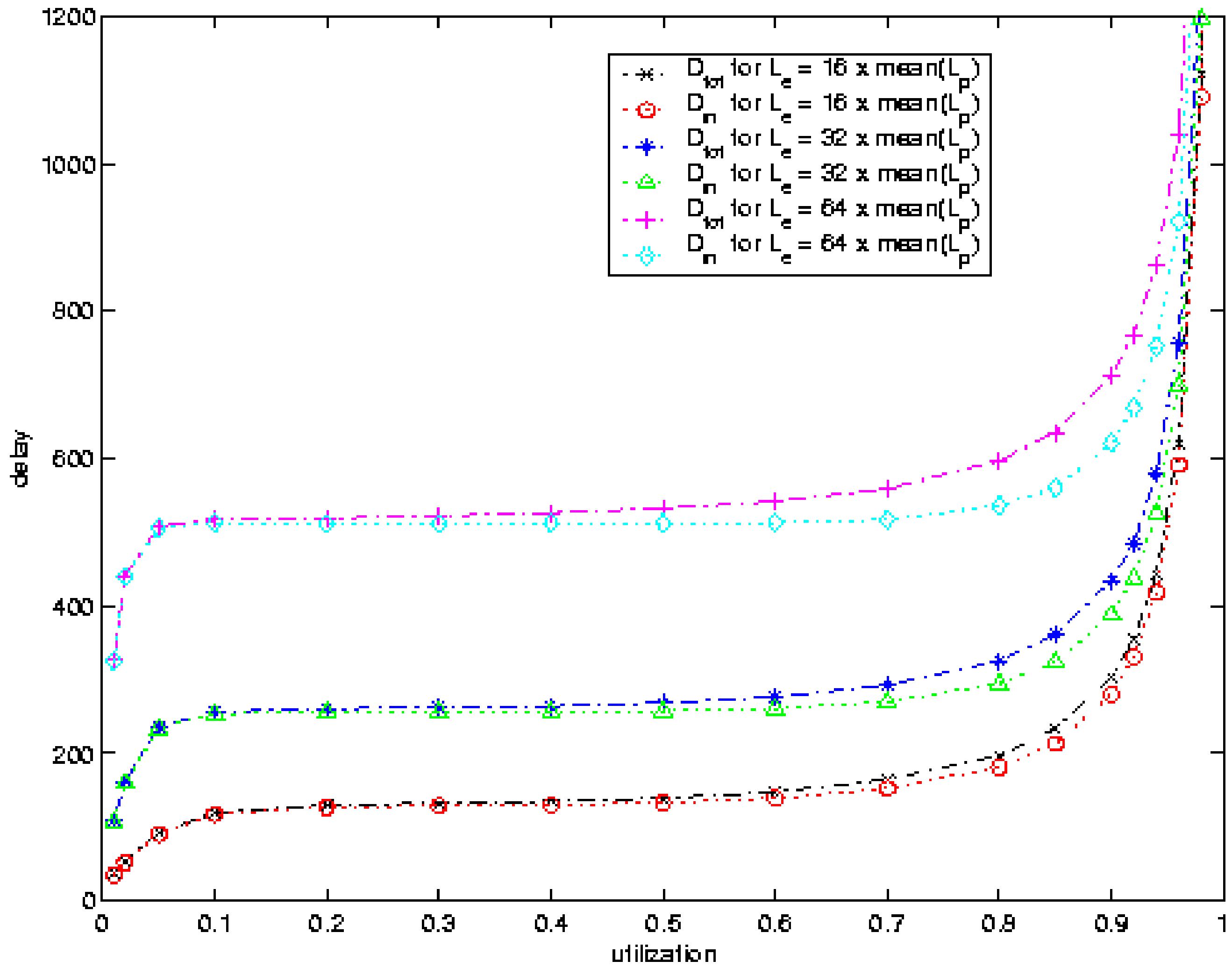
Simulation Results

Bell Laboratories



Delay vs. utilization plots for various envelope sizes

Simulation Results (contd.)



Total delay and delay at input vs. utilization plots for various envelope sizes

Summary

- **We observe**
  - Fast scheduling/reconfiguration is a key constraint in scalability of crossbars
  - Slow reconfiguration is also an absolute requirement for optical crossbars
  - Variable size packets need to be handled efficiently

- **We propose schemes where**
  - frequency of scheduling decisions can be slowed down considerably
  - bandwidth loss due to variable size packets is avoided
  - delay guarantees comparable to that of output queued switches can be obtained