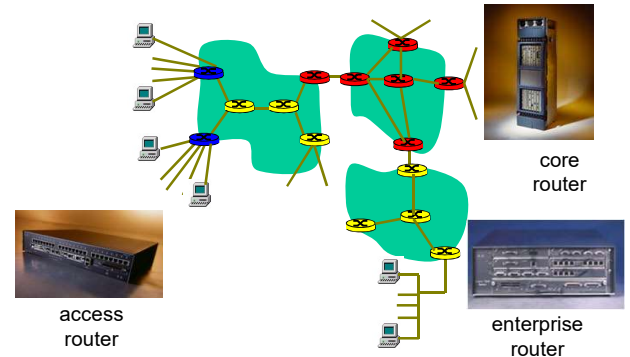


Router/switch architectures

Router/switch architectures

Andrea Bianco
 Telecommunication Network Group
 firstname.lastname@polito.it
<http://www.telematica.polito.it/>

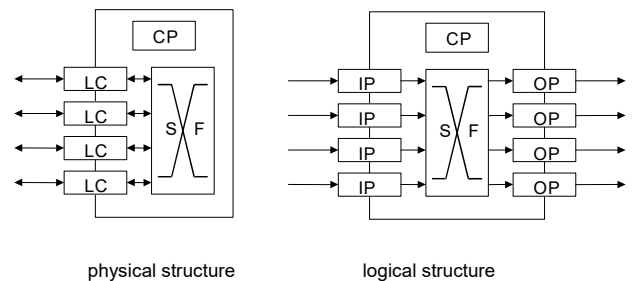
The Internet is a mesh of routers



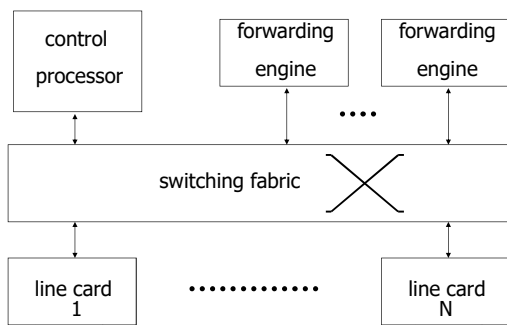
The Internet is a mesh of routers

- Access router:
 - high number of ports at low speed (kbps/Mbps)
 - several access protocols (modem, ADSL, cable)
- Enterprise router:
 - medium number of ports at high speed (Mbps)
 - several services (IP classification, filtering)
- Core router:
 - moderate number of ports at very high speed (Gbps)
 - very high throughput

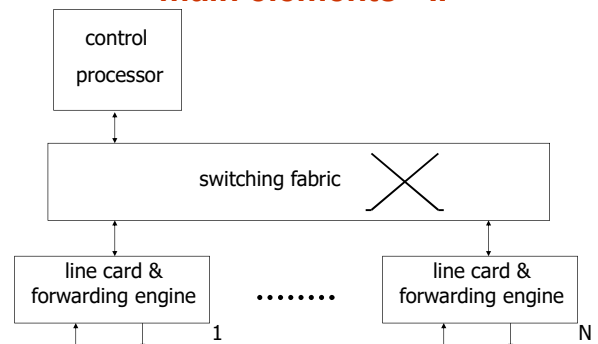
Hardware architecture



Interconnections among main elements - I



Interconnections among main elements - II



Router/switch architectures

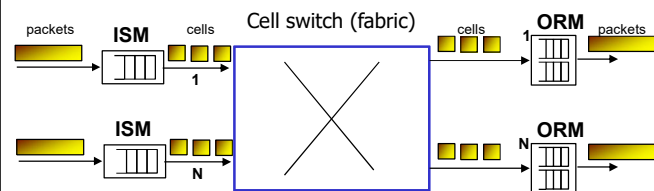
Hardware architecture: main elements

- Line cards
 - support input/output processing and rx/tx
 - store packets
 - adapt packets to the internal format of the switching fabric
 - support data link protocols
 - classify packets
 - schedule packets
 - support security
- Switching fabric
 - transfers packets from input ports to output ports

Hardware architecture: main elements

- Control processor/network processor
 - runs routing protocols
 - computes routing tables
 - manages the overall system
- Forwarding engines
 - compute the packet destination (lookup)
 - inspect packet headers
 - rewrite packet headers

Cell-based synchronous routers



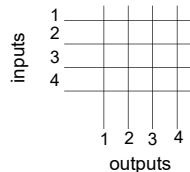
- *Packet*: variable-size data unit
- *Cell*: fixed-size data unit
- ISM: Input-Segmentation Module
- ORM: Output-Reassembly Module

Switching fabric

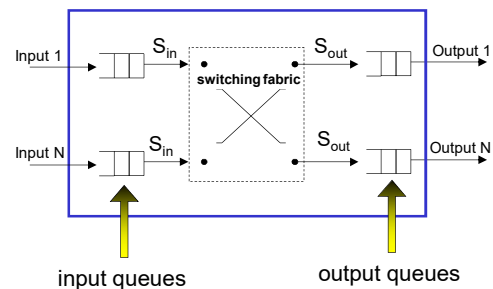
- Our assumptions:
 - Bufferless
 - to reduce internal hardware complexity
 - Non-blocking
 - it is always possible to transfer in parallel from input to output ports any non-conflicting set of cells

Switching fabric

- Examples:
 - Buses
 - Shared memory
 - Crossbar
 - Multi-stage
 - rearrangeable Clos network
 - Benes network
 - Batcher-Banyan network (self-routing)
 - Buffered cross-bars (not considered)



Memory and speedup



Router/switch architectures

Speedup

- The speedup (increase in speed with respect to line speed) determines switch performance:
 - S_{in} = reading speed from input queues
 - S_{out} = writing speed to output queues
- The speedup is also a technological constraint
- Maximum speedup factor:
 - $S = \max(S_{in}, S_{out})$

Faster and faster

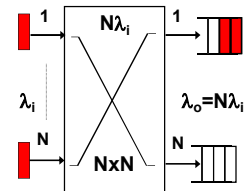
- Need for high performance routers
 - to accommodate the bandwidth demands for new users and new services
 - to support QoS
 - to reduce costs
- Moore's law (electronic packet processing power) is too slow with respect to the increase in link speed
- The bottleneck is memory speed

Single packet processing

- The time to process one packet is becoming shorter and shorter
- Worst case: 40-Byte packets (ACKs)
 - 3.2 μ s at 100 Mbps
 - 320 ns at 1 Gps
 - 32 ns at 10 Gps
 - 3.2 ns at 100 Gbps
 - 320 ps at 1Tbps

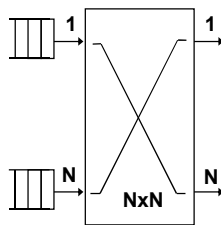
Switches with queues at outputs

- OQ (Output Queued)
- The switching fabric is able to transfer to any output all cells received in one time slot
- 100% throughput
- Optimal average delay
- **Speedup N** with respect to line speed is required in switching fabric speed and in output port memory access



Switches with queues at inputs

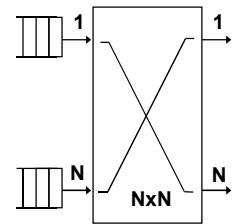
- IQ (Input Queued)
- Switching constraints
 - at most one cell for each input and for each output can be transferred
- Advantages:
 - Switching fabrics and memories less costly
 - No speedup required in the switching fabric
 - Memory access speed equal to line speed
 - Speedup=1
 - Only viable solution for very high speed devices



Pag. 3

IQ switches

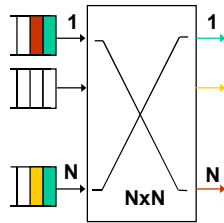
- Two problems:
 - Define **scheduling algorithms** to avoid output contention
 - Select in each time slot data to be transmitted from inputs to outputs
 - Constraint: in each time slot
 - At most 1 data from each input (no speedup in reading)
 - At most 1 data to each output (no speedup in writing because no memory is available)



Router/switch architectures

IQ switches

- Two problems:
 - Memory architecture:
 - If using FIFO queues, HoL (Head of the Line) blocking
 - If choosing cyan packet from queue N, the red packet in queue 1 is blocked by the cyan packet which is at the head of queue 1 even if red output port N is free
 - For uniform unicast traffic throughput limited to 58.6%

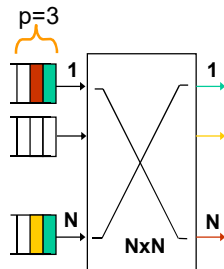


Memory architecture in IQ

- To avoid HoL blocking phenomenon, a more complex memory architecture is needed
- Two possible solutions:
 - p-window queueing
 - VOQ (Virtual Output Queueing)

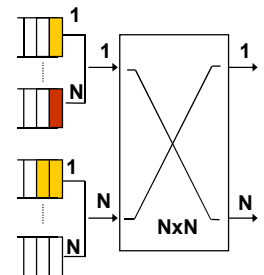
Memory architecture

- p-window queueing:
 - p is the window size
 - The first p cells of each queue are considered for scheduling
 - Higher complexity
 - Scheduler deals with pN cells
 - Non FIFO queues
 - HoL blocking reduced, completely eliminated only for $P \rightarrow \infty$



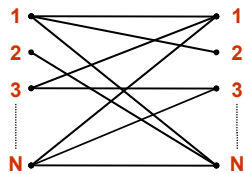
Memory architecture

- VOQ (virtual output queueing)
 - At each input data are stored in separate queues according to data destination (N queues for each input)
 - N^2 queues in total
 - Eliminates HoL blocking and it permits to achieve 100% throughput with a proper scheduling algorithm



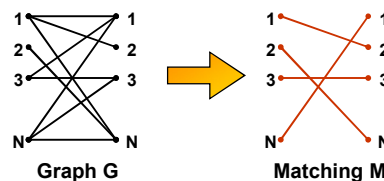
Scheduling problem modelling

- Problem to be solved by the scheduling can be represented using a bipartite graph
- An edge $i \rightarrow j$ exist if there is at least a data at input i willing to reach output j
- Edges may be weighted
- Weight
 - Binary
 - Queue length
 - HoL cell age or waiting time
 - Other metrics



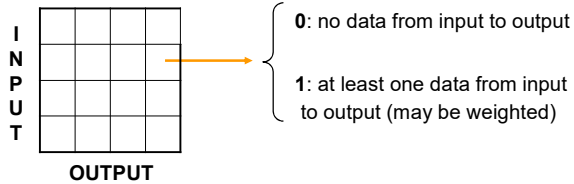
Scheduling problem modeling

- The scheduling algorithm tries to determine, in each time slot, a matching over the bipartite graphs.
- Select at most N edges with constraints
 - At most one edge for each input
 - At most one edge for each output



Scheduling problem modeling

- Another possible representation is based on a (Request Matrix RM), which stores the information related to data transfer request



- Matching \rightarrow it is a permutation matrix i.e., a matrix such that the row and column sum is at most equal to 1

Scheduling in IQ switches

- Request Matrix
- Permutation matrix

$$\begin{bmatrix} 3 & 5 & 0 & 0 \\ 2 & 0 & 0 & 4 \\ 4 & 5 & 0 & 0 \\ 0 & 0 & 8 & 2 \end{bmatrix} \longrightarrow \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Traffic description

- $A_{ij}(n) = 1$ if a packet arrives at time n at input i , with destination reachable through output j
- $\lambda_{ij} = E[A_{ij}(n)]$
- An arrival process is *admissible* if:
 - $-\sum_i \lambda_{ij} < 1$
 - $-\sum_j \lambda_{ij} < 1$
 - no input and no output are overloaded on average
 - OQ switches exhibit finite delays (for admissible traffic)
- Traffic matrix:** $\Lambda = [\lambda_{ij}]$

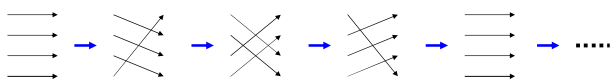
Scheduling policies: objective

- Let us consider a $N \times N$ IQ
- Denote by i the input port index and by j the output port index
- Goal: assuming infinite buffer size, transfer any admissible traffic pattern with no losses
- Solutions are known
 - If traffic pattern is known in advance
 - TDM of Birkhoff von Neumann algorithm
 - For admissible unknown traffic pattern
 - Maximum Weight Matching
 - Maximum Size Matching
- Several heuristics are proposed for unknown traffic pattern
 - iSLIP, iLQF, iOCF, 2DRR (WFA), MUCS, RPA, and many others

Scheduling uniform known traffic

- A number of algorithms give 100% throughput when admissible traffic is uniform
 - For example:
 - TDM and a few variants
 - iSLIP (see later)

Example of a TDM schedule for a 4x4 switch



Birkhoff - von Neumann theorem

- Any doubly stochastic matrix Λ can be expressed as convex combination of permutation matrices π_n :

$$\Lambda = \sum_n a_n \pi_n$$

- with
 - $a_n \geq 0$
 - $\sum_n a_n = 1$

Router/switch architectures

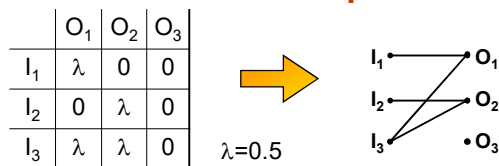
Scheduling non-uniform known traffic

- Thanks to the Birkhoff - von Neumann theorem
- If the traffic is known and admissible, 100% throughput can be achieved by a TDM scheme using:
 - for a fraction of time a_1 matching M_1 (π_1)
 - for a fraction of time a_2 matching M_2 (π_2)
 - for a fraction of time a_k matching M_k (π_k)

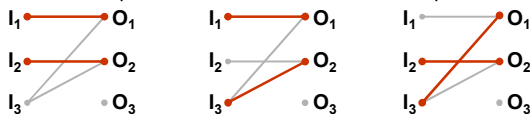
MSM: Maximum Size Matching

- MSM maximizes the number of data transferred in a single time slot, i.e. select the maximum number of edges
 - Instantaneous throughput maximization.
- Asymptotic computational complexity is $O(N^{2.5})$
- Non optimal algorithm
 - Some admissible traffic pattern cannot be scheduled, i.e. it does not always achieve 100% throughput.

MSM: example



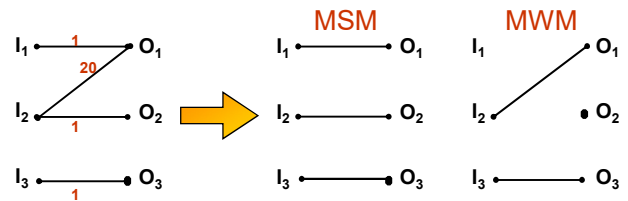
Three MSM are possible in overload, i.e. when all queues are full



When the first matching is chosen, the maximum throughput cannot be achieved

MWM: Maximum Weight Matching

- A weight is associated with each edge
- The MWM, among all possible $N!$ matchings, selects the one with the highest weight (sum of edge metrics)



MWM: Maximum Weight Matching

- MWM does not maximize instantaneous throughput (worse than MSM)
- It was demonstrated that a MWM algorithm
 - in IQ switches with VOQ architecture
 - under admissible traffic
 - with infinite queue size
 - when using as weight either the queue length or the age of the HoL data
 achieves 100% throughput
- Asymptotic computational complexity $O(N^3)$
- With finite queue size, it behaves similarly to MSM
- Problems with delays and possible starvation

Practical solutions

- Need to define heuristics
 - with reasonable complexity
 - that can be implemented in hardware
- Any scheduling algorithm defines three aspects:
 - A method to compute the weights to be associated with each edge (metric):
 - Approximate MSM
 - Binary (queue occupancy)
 - Approximate MWM
 - Queue length (it is an indication of the fact that the queue, which is associated with an input/output pair, is suffering)
 - Age of HoL data
 - Interface load
 - Ad hoc metrics to select critical edges/nodes

Router/switch architectures

Practical solutions

- ...
- A heuristic algorithm to determine a matching
- A contention resolution algorithm among edges with the same metric:
 - round-robin (initial choice state dependent)
 - sequential search (initial choice non state dependent)
 - random

i-SLIP

- Iterative algorithm
- It defines a heuristic algorithm which determines, with a proper number of iterations, a *maximal size* matching (i.e., a matching that cannot be further extended with other edges selection)
- Metric is the queue occupancy
- To solve contentions, it exploits an arbiter for each input and for each output

i-SLIP

- In each iteration, three phases can be identified:
 - Request: each unmatched input sends a request to every output for which it has a cell
 - Grant: each unmatched output that has received requests, sends a grant to one of the requesting inputs.
 - Contentions solved by a round robin mechanism.
 - Accept: if an unmatched input receives grants, it selects an output and becomes matched to it
 - Contentions solved by a round robin mechanism

i-SLIP: counters

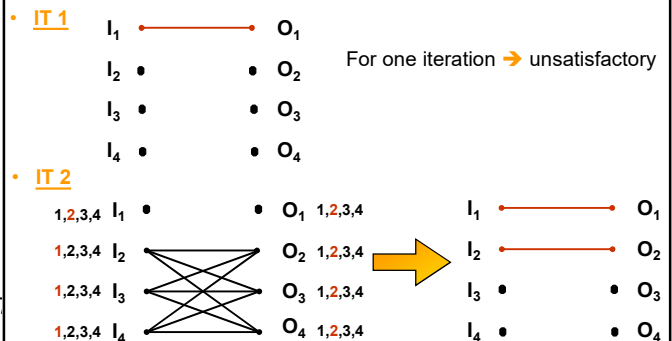
- Each input (output) has a pointer associated with to solve contentions
- The output pointer is incremented, modulo N, by one unit beyond the index of the input to which the grant was issued
- The input pointer is incremented, modulo N, by one unit beyond the index of the output from which an accept was received

i-SLIP: properties

- For uniform overload → the bipartite graph is a full mesh → the higher the number of alternatives, the easier is to determine a good matching. iSLIP degenerates to a TDM scheme ES:



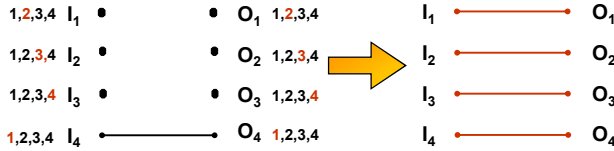
i-SLIP: properties



i-SLIP: properties

- At the end:

IT N

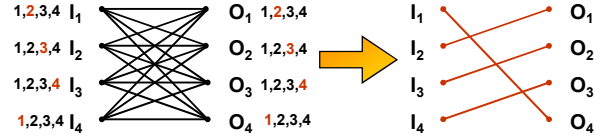


- A maximum (implies maximal) matching is obtained after N iterations

i-SLIP: properties

In the following steps, pointers are staggered → one iteration is enough to obtain a maximum matching

Step 2



Step 3



i-SLIP: properties

- Each iteration has a computational complexity of $O(N^2)$, but it can be easily made parallel
- Worst case in one iteration: 1 edge is selected
- When executing N iterations, the matching is maximal (depends on the choice made but cannot be extended) → however, the computational complexity is $O(N^3)$
- Experimental results show that $\log_2 N$ iterations are in general enough to obtain good performance
- Performance drops if pointers are badly synchronized
- iSLIP was implemented on a single chip in the Cisco 12000 router

iSLIP: extensions

- Use the same heuristic algorithm (3-phase) but with different metrics
 - Queue length $iLQF$
 - HoL cell age $iOCF$
- Input send requests containing the weight
- Contentions are solved using the weight first, only for equal weights the choice is random
- Does not exploit pointer synchronization to obtain good performance, rather the edge weight
- OCF has better delay properties (never starves data), but the increase in complexity is significant and makes the algorithm practically unfeasible

2DRR: Two Dimensional Round Robin

- Operates on the request matrix
- Extension of the WFA (Wave Front Arbiter), very easily implementable in hardware
- Definitions
 - Generalized diagonal is a set of N elements of a matrix $N \times N$ such that two elements do not belong to the same row or column
- A set of N diagonal is said to be covering if each element of the matrix belongs to one and only one diagonal
- In each time slot, the algorithm goes through N iterations

2DRR: Two Dimensional Round Robin

- At the beginning, all links (input-output connections) are enabled
- At each iteration, a given generalized diagonal is chosen
 - Only enabled links may be selected if they are covered by the elements belonging to the chosen diagonal
- If a link from input i to output j is selected, all requests issued by i or sent to j are disabled for the current time slot (cannot be chosen in the matching)
- In N iterations, all N generalized diagonal are considered and the request matrix is fully covered

Router/switch architectures

2DRR: example

RM =

1	0	1	1
0	1	0	1
1	1	0	1
0	0	1	0

→

1	0	1	1
0	1	0	1
1	1	0	1
0	0	1	0

1	0	1	1
0	1	0	1
1	1	0	1
0	0	1	0

→

1	0	1	1
0	1	0	1
1	1	0	1
0	0	1	0

→

1	0	1	1
0	1	0	1
1	1	0	1
0	0	1	0

→

1	0	1	1
0	1	0	1
1	1	0	1
0	0	1	0

→

1	0	0	0
0	1	0	0
0	0	0	1
0	0	1	0

Andrea Bianco – TNG group - Politecnico di Torino Computer Networks Design and Management - 49

2DRR: Two Dimensional Round Robin

- At each time slot, a different covering set of generalized diagonal is chosen, to improve fairness
 - Indeed, edges covered to the first diagonal chosen are more likely selected
 - Round robin over different sets of covering diagonal and round robin on each element in the set
- Emulates a MSM
- Not easy to extend to other metrics
- Asymptotic computational complexity $O(N^2)$

Andrea Bianco – TNG group - Politecnico di Torino Computer Networks Design and Management - 50

Traffic scenarios

- Uniform traffic
 - Bernoulli i.i.d. arrivals
 - usual testbed in the literature
 - "easy to schedule"
- Diagonal traffic
 - Bernoulli i.i.d arrivals
 - critical to schedule, since only two matchings are good

$$\Lambda = \frac{\rho}{N} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$$\Lambda = \frac{\rho}{3} \begin{bmatrix} 2 & 1 & 0 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 0 & 2 & 1 \\ 1 & 0 & 0 & 2 \end{bmatrix}$$

Andrea Bianco – TNG group - Politecnico di Torino Computer Networks Design and Management - 51

Uniform traffic

- Comparison between MWM, iSLIP, iLQF, RPA

Andrea Bianco – TNG group - Politecnico di Torino Computer Networks Design and Management - 52

Diagonal traffic

- RPA achieves 98% throughput, iLQF 87%, iSLIP 83%

Andrea Bianco – TNG group - Politecnico di Torino Computer Networks Design and Management - 53

Issues in IQ switches

- Signalling:
 - Signalling bandwidth required to transfer weights from inputs to the controller may be significant with respect to the available bandwidth in the switching fabric
 - The more complex the adopted metric, the larger the signalling bandwidth required
 - Differential signalling may be adopted
- Multiple classes:
 - Given K classes, first the VOQ architecture must be extended, by using KN queues at each input
 - Scheduling algorithms must be extended to support priorities

Andrea Bianco – TNG group - Politecnico di Torino Computer Networks Design and Management - 54

Router/switch architectures

Issues in IQ switches

- QoS (fair queueing)
 - Scheduling for QoS (need to serve the most urgent packet) has a difficult interaction with the scheduling to transfer data from inputs to outputs
 - Need to balance performance and fairness
 - No ideal optimal solution known
- Frame scheduling
 - Operate on a frame of length F slot, and compute a schedule on the frame and not on a slot by slot basis
 - Scheduling algorithm executes only at frame boundaries
 - Relatively easy to provide QoS guarantees for each input-output pair
 - Delay increases at low loads

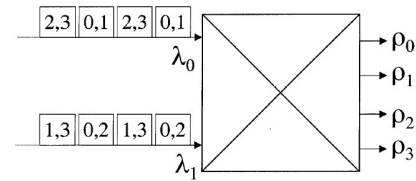
Issues in IQ switches

- Variable packet length support
 - May introduce packet scheduling instead of cell scheduling
 - Packets transferred as trains of cells
 - An edge is selected when the first cell of a packet arrives and is kept in all the following matchings until the last cell of the packet is transferred
 - It avoids reassembly machines at outputs
 - Same throughput guarantees
 - Packet delay may be larger or shorter
 - Depends on packet length distribution

Issues in IQ switches

- Multicast:
 - 2^N possible different multicast flows
 - May be treated as unicast through input port replication (often named multicopy)
 - At each input a number of copies equal to the packet fanout are created, for the proper outputs, and inserted in the proper VOQ
 - Speedup required
 - Increases the instantaneous input load
 - May lead to low throughput (unable to sustain a single broadcast flow)
 - Scheduling for multicast must be defined to exploit switching fabric multicast capabilities
 - Balance fanout splitting and no-fanout splitting
 - Often a single FIFO for multicast is proposed (HoL blocking, less critical with respect to unicast)
 - Critical traffic patterns when few inputs are active

Example of a critical traffic pattern



Issues in IQ switches

- MC-VOQ architecture
 - 2^N separate queues at each input
 - Best possible solution (no HoL blocking)
 - An optimal scheduling was defined (only theoretically)
 - Implies re-queueing and out-of-sequence
 - However, admissible traffic pattern exist that cannot be scheduled in an IQ switch regardless of the queue architecture and of the scheduling algorithm
 - Scalability problem
 - Number of queues
 - Scheduling algorithm
- Manage a finite number of queues
- CIOQ switches (a moderate speedup helps a lot)

References

- Karol M., Huchyj M., Morgan S., "Input versus output queueing on a space division switch", IEEE Transactions on Communications, vol.35, n.12, Dec.1987
- McKeown N., Anantharam V., Walrand J., "Achieving 100% throughput in an input-queued switch", IEEE INFOCOM96, vol.1, San Francisco, CA, Mar. 1996, pp.296-302
- McKeown N., "ISLIP: a scheduling algorithm for input-queued switches", IEEE Transactions on Networking, vol.7, n.2, Apr. 1999, pp.189-201
- Tamir Y., Chi H.-C., "Symmetric crossbar arbiters for VLSI communication switches", IEEE Transaction on Parallel and Distributed Systems, vol.4, no.1, Jan.1993, pp.13-27
- Anderson T., Owens S., Saxe J., Thacker C., "High speed switch scheduling for local area networks", ACM Transactions on Computer Systems, vol.11, n.4, Nov.1993
- LaMaire R.D., Serpanos D.N., "Two dimensional round-robin schedulers for packet switches with multiple input queues", IEEE/ACM Transaction on Networking, vol.2, n.5, Oct.1994, p.471-482
- Duan H., Lockwood J.W., Kang S.M., Will J.D., "A high performance OC12/OC48 queue design prototype for input buffered ATM switches", IEEE INFOCOM97, vol.1, 1997, pp.20-8, Los Alamitos, CA
- Ajmone Marsan M., Bianco A., Leonardi E., Milla L., "RPA: a flexible scheduling algorithm for input buffered switches", IEEE Transactions on Communications, vol.47, n.12, Dec. 1999, pp.1921-1933
- Ajmone Marsan M., Bianco A., Giaccone P., Leonardi E., Neri F., "Packet scheduling in input-queued cell-based switches", IEEE Trans. on Networking, Oct.2002
- Nong G., Hamdi M., "On the provision of integrated QoS guarantees of unicast and multicast traffic in input-queued switches", IEEE GLOBECOM99, vol.2, 1999
- Hayes J.F., Breaudt R., Mehmet-Ali M.K., "Performance analysis of a multicast switch", IEEE Transactions on Communications, vol.39, n.4, Apr. 1991, pp.581-587
- Kim C.K., Lee T.T., "Call scheduling algorithm in multicast switching systems", IEEE Transactions on Communications, vol.40, n.3, Mar. 1992, pp.625-635
- Prabhakar B., McKeown N., Ahuja R., "Multicast scheduling for input-queued switches", IEEE Journal on Selected Areas in Communications, vol.15, n.5, Jun. 1997, pp.856-866
- Andrews M., Khanna S., Kumaran K., "Integrated scheduling of unicast and multicast traffic in an input-queued switch", IEEE INFOCOM99, vol.3, New York, NY, 1999, pp.1144-1151
- Liu Z., Righter R., "Scheduling multicast input-queued switches", Journal of Scheduling, John Wiley & Sons, May 1999
- Ajmone Marsan M., Bianco A., Giaccone P., Leonardi E., Neri F., "On the throughput of input-queued cell-based switches with multicast traffic", IEEE INFOCOM01, Anchorage Alaska, Apr.2001
- Smljanic A., "Flexible bandwidth allocation in high-capacity packet switches", IEEE/ACM Transactions on Networking, vol.10, n.2, pp.287-293, Apr.2002
- Chang C.S., Lee D.S., Jou Y.S., "Load balanced Birkhoff-von Neumann switches", 2001 IEEE HPSR, 2001, pp.276-280.