Critical Issues in Design and Implementation of Interconnects for Workstation Clusters

José Duato

Dept. of Information Systems and Computer Architecture
Universidad Politécnica de Valencia, Spain
E-mail: `jduato@gap.upv.es` WWW: `http://www.gap.upv.es`

Outline

Introduction

Critical issues

Improving performance

Improving availability/reliability

Improving quality of service (QoS)

Introduction

The performance of most digital systems today is limited by their communication or interconnection, not by their logic or memory

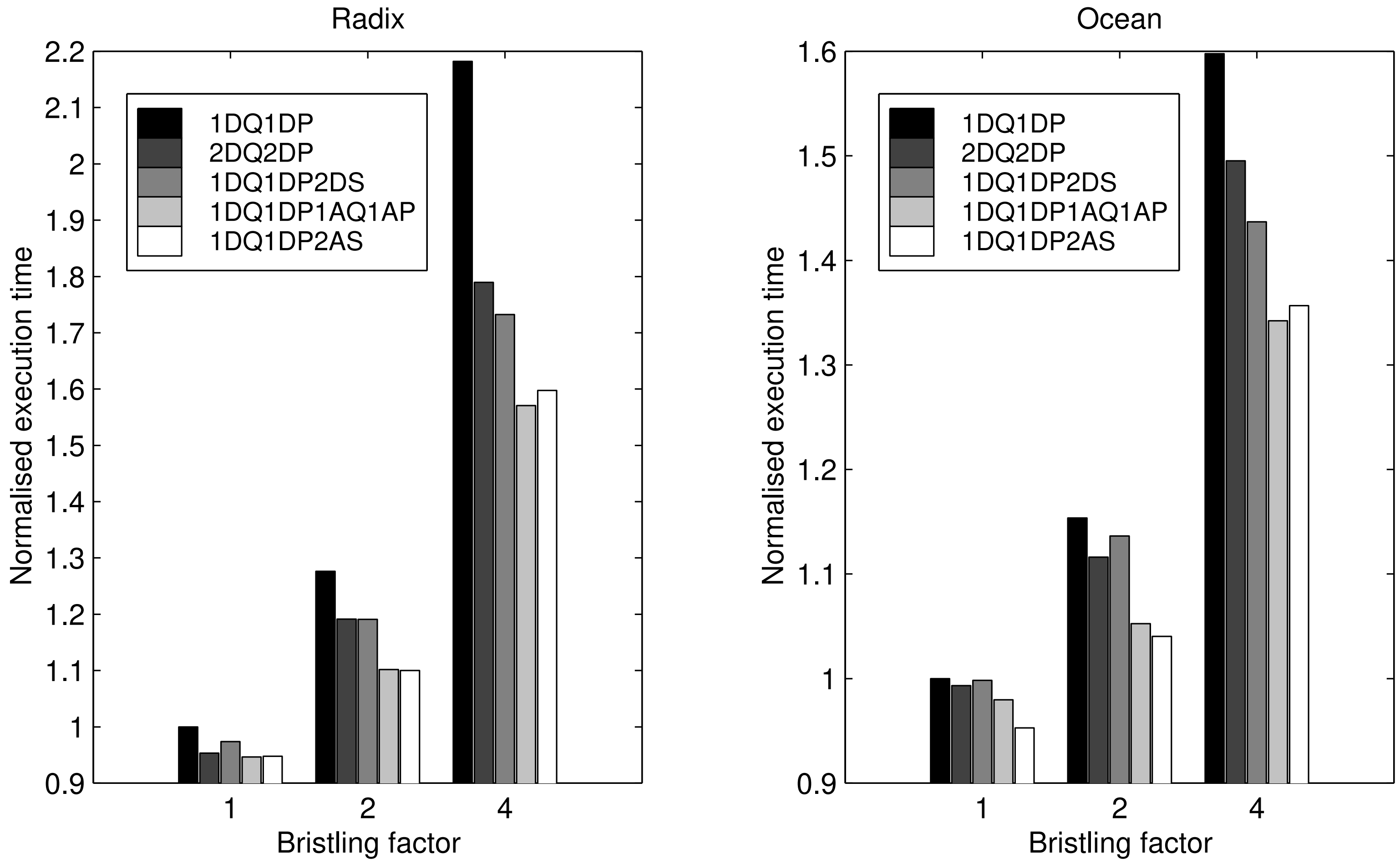
Most of the power is used to drive wires and most of the clock cycle is spent on wire delay, not gate delay

As technology improves, pin density and wiring density are scaling at a slower rate than the components themselves.

The frequency of communication between components is lagging far beyond the clock rates of modern processors

These factors combine to make interconnection the key factor in the success of future digital systems

Example: Effect on execution time of architectural improvements in the network for a DSM with 32 4-way superscalar processors



Applications

Interconnection networks are being and will be used for many different applications, ranging from internal buses in VLSI circuits to wide area computer networks. These applications include:

- On-chip interconnects (between ALUs and register files, SMPs)
- System area networks (SANs) and system I/O (SIO)
- Internal networks for ATM and IP switches and routers
- Physical and logical buses for SMPs
- Interconnects for multicomputers and DSMs
- Networks of workstations (NOWs)
- Local area networks (LANs)
- Metropolitan area networks (MANs)
- Wide area networks (WANs)
- Networks for industrial applications

} Computer networks

Critical issues in the design of NOWs

- Critical issues
  - Performance
  - Availability/reliability
  - Quality of service (QoS)

- Main layers
  - Applications programmer interface (API)
  - Communication libraries
  - Network hardware
    - Network interface controller (NIC)
    - Router

Improving performance

Design issues
- Topology
- Routing algorithm
- Flow control strategy
- Deadlock handling strategy
- Switching technique
- Router architecture

Some definitions

Topology: Defines how the nodes are interconnected by channels

- Direct networks, switch-based networks

Routing algorithm: Determines the path selected by a message to reach its destination

- Deterministic routing, adaptive routing

Flow control: Establishes the handshaking protocol between routers to exchange information and defines the unit of information on which flow control is performed

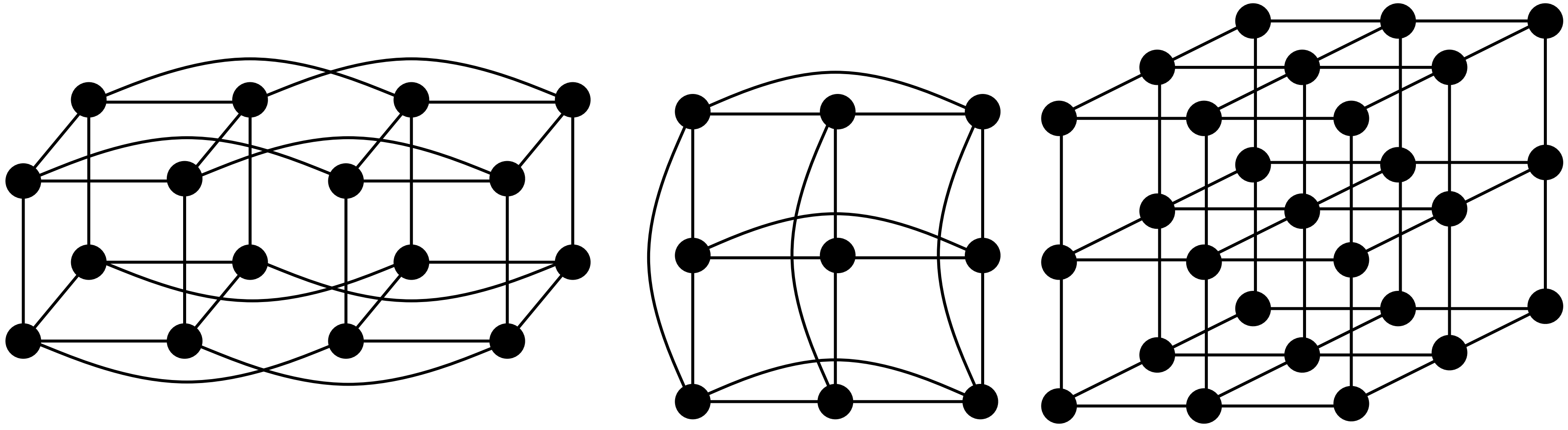
- Credit-based, stop & go

Switching technique: Determines how and when buffers are reserved and switches are configured

- Packet switching, circuit switching, wormhole, virtual cut-through

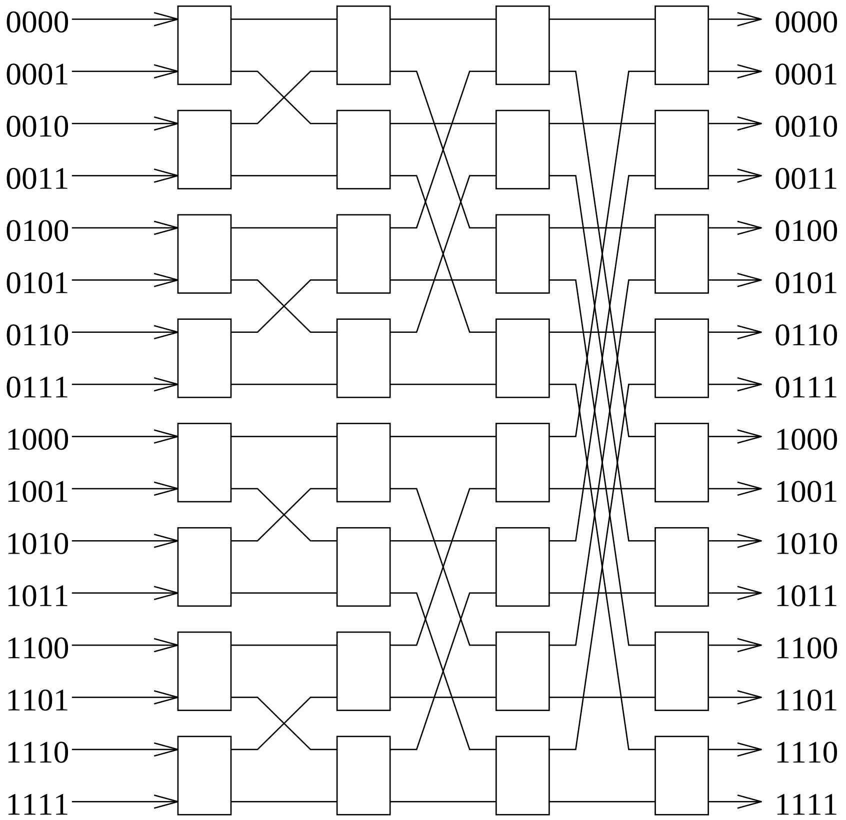
Popular topologies

1. Direct networks

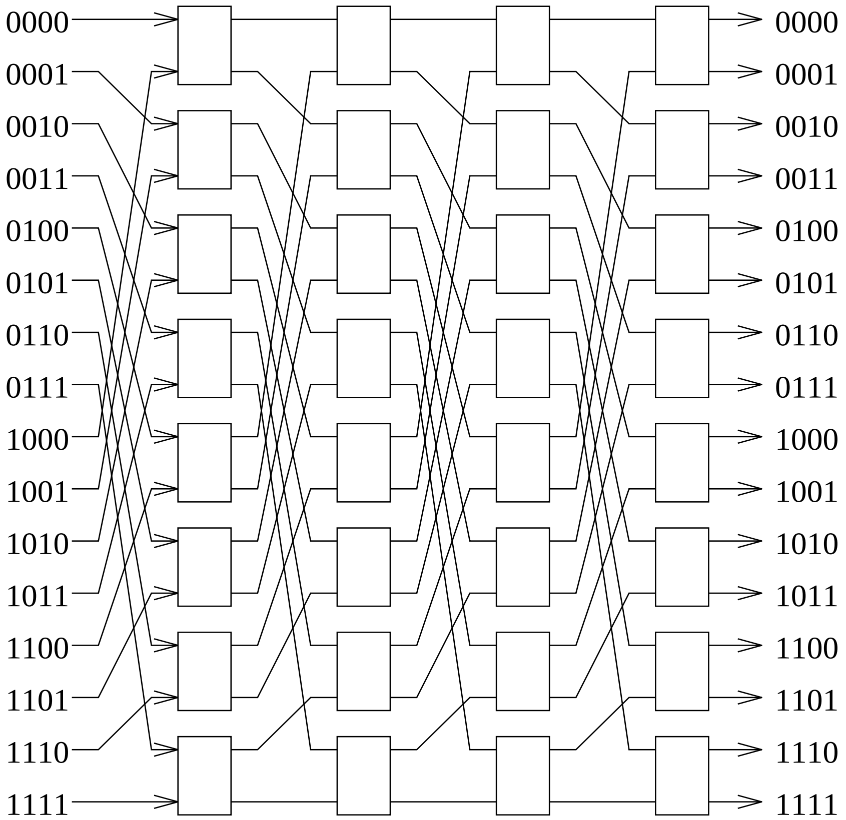


(a) 2-ary 4-cube (hypercube) (b) 3-ary 2-cube (c) 3-ary 3D-mesh

2. Multistage interconnection networks

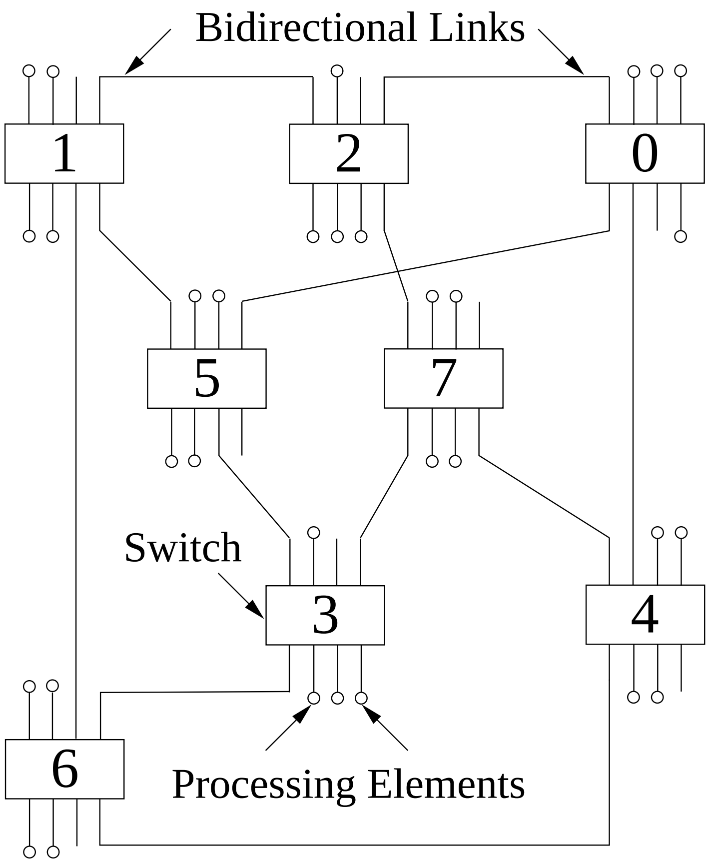


Multistage butterfly network

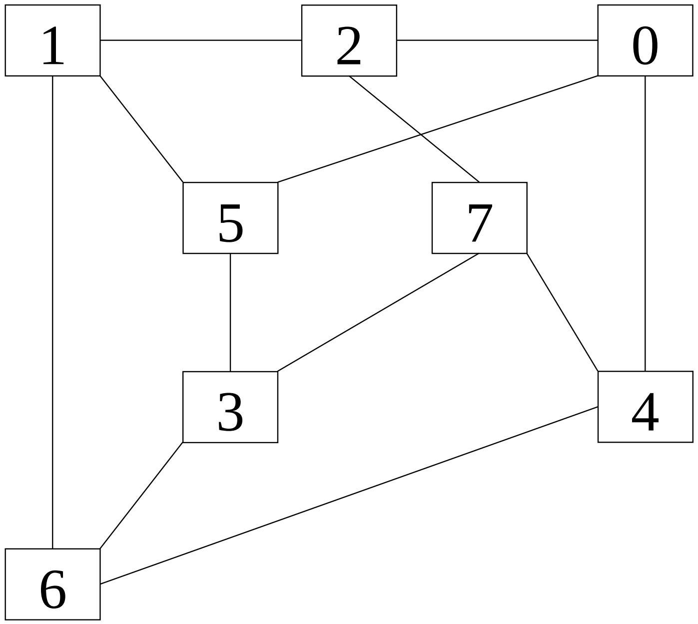


Omega network

3. Switch-based irregular topologies



Switch-Based Network



Graph Representation

Unified View

Some manufacturers developed switches that are suitable to implement either direct or indirect networks (Inmos C104, SGI SPIDER)

We can view networks using point-to-point links as a set of interconnected switches, each one connected to zero, one, or more nodes:

- Direct networks correspond to the case where each switch is connected to a single node
- Crossbar networks correspond to the case where there is a single switch connected to all the nodes
- Multistage interconnection networks correspond to the case where switches are arranged into several stages and the switches in intermediate stages are not connected to any processor
- Switch-based irregular networks correspond to the general case

Application to networks of workstations

Switch-based interconnects like Autonet, Myrinet and ServerNet have been proposed to build networks of workstations with either regular or irregular topology.

The irregularity provides:

- Wiring flexibility.
- Scalability.
- Incremental expansion capability.

Drawback: The irregularity makes deadlock avoidance and routing quite complicated.

Simplest solution: Avoid deadlock by eliminating all the cyclic dependencies between channels

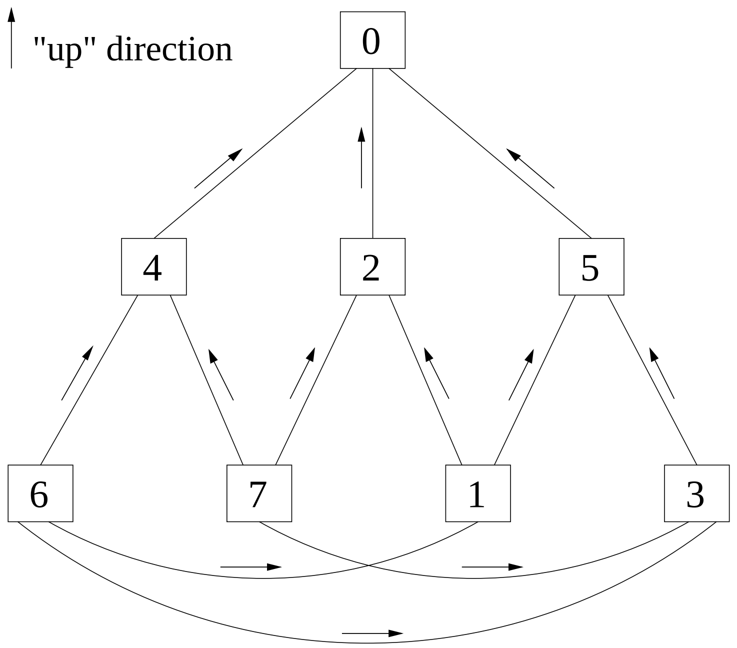
$\Rightarrow$ Many messages are routed following non-minimal paths.

$\rightarrow$ Higher message latency

$\rightarrow$ Waste of resources

$\rightarrow$ Lower throughput

The up*/down* routing algorithm (Autonet)

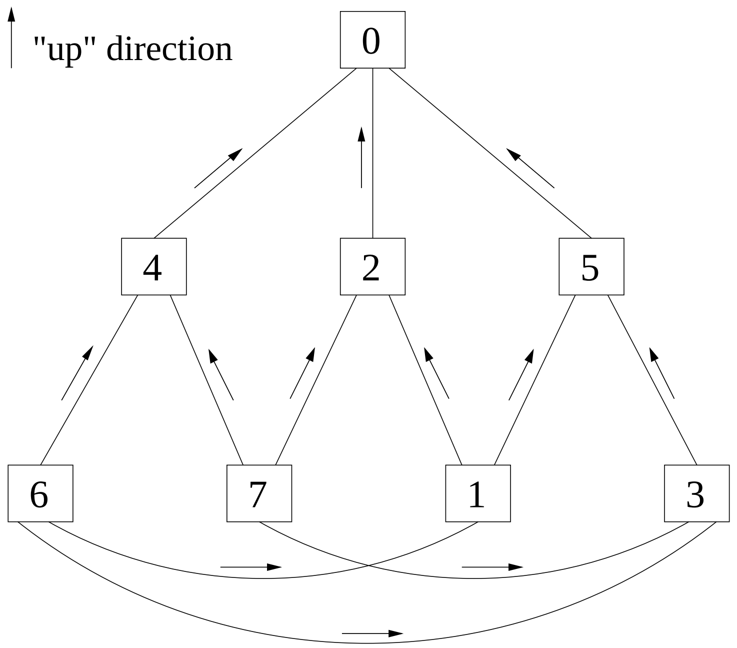


Routing is distributed, implemented using table-lookup, and based on an assignment of direction to the operational links.

Routing rule: a legal route must traverse zero or more links in the "up" direction followed by zero or more links in the "down" direction.

- Each cycle has at least one link in the "up" direction and one link in the "down" direction.
- Cyclic dependencies are avoided: messages cannot cross a link in the "up" direction after one in the "down" direction.

Routing efficiency



- From 7 to 0: OK
- From 2 to 5: lack of adaptivity
- From 4 to 1: non-minimal routing

The basic routing rule prevents from using minimal routing and adaptivity in most cases because of "down" to "up" conflicts.

Probability of non-minimal routing increases with network size.

A design methodology for adaptive routing algorithms

interconnection network + deadlock-free routing function	new methodology $\Longrightarrow$	physical channels duplicated or split into two virtual channels (original and new) + extended routing function

Key for improved performance: Allow cyclic dependencies between channels and implement fully adaptive minimal routing

$\rightarrow$ Reduces contention by increasing routing adaptivity

$\rightarrow$ Allows more messages to follow minimal paths

Extended routing function

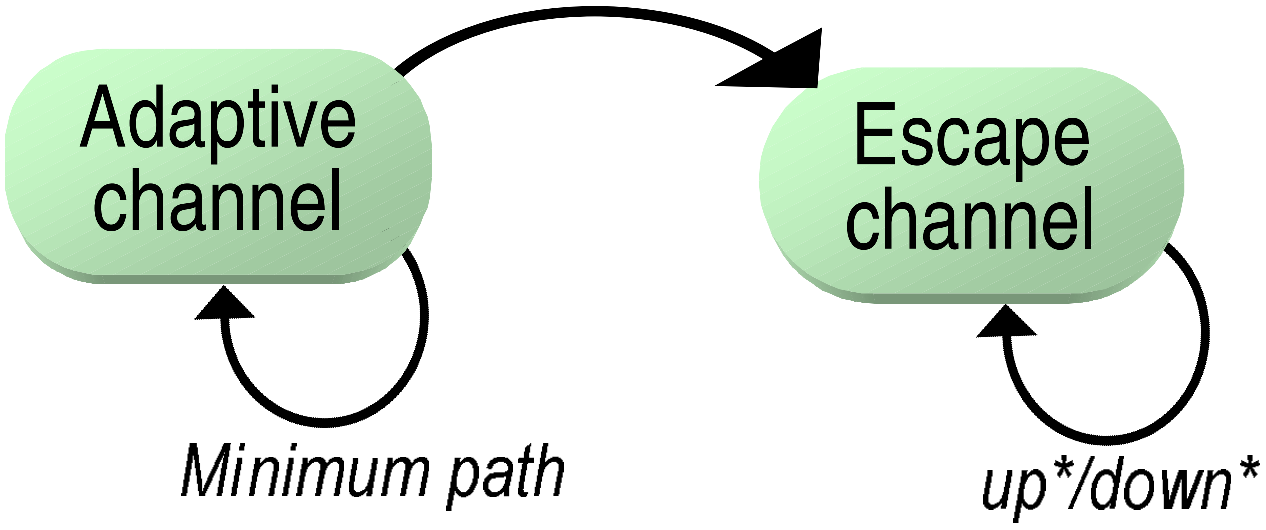
- Newly injected messages can use the new channels without any restriction. For performance reasons, only minimal paths are allowed
- Original channels are used exactly in the same way as in the original routing function
- Once a message reserves one of the original channels, it cannot use any of the new channels again
- When the routing table provides both kinds of channels, give preference to new channels

The extended routing function is deadlock-free

Improving the efficiency of the methodology

- Idea: Focus on minimal routing, even if adaptivity is reduced
- Restrict the transition from new channels to original channels
- Improved adaptive routing function:
  - Newly injected messages can only use new channels
  - At intermediate switches, a higher priority is assigned to the new channels belonging to minimal paths
  - If all the new channels are busy, then an original channel belonging to a minimal path (if any) is selected
  - If none exists, then the one that provides the shortest path is used (this ensures deadlock-freedom)
- Once a message reserves an original channel, it can no longer reserve a new one

Fully adaptive minimal routing algorithm



Performance evaluation

Evaluation of four routing schemes:

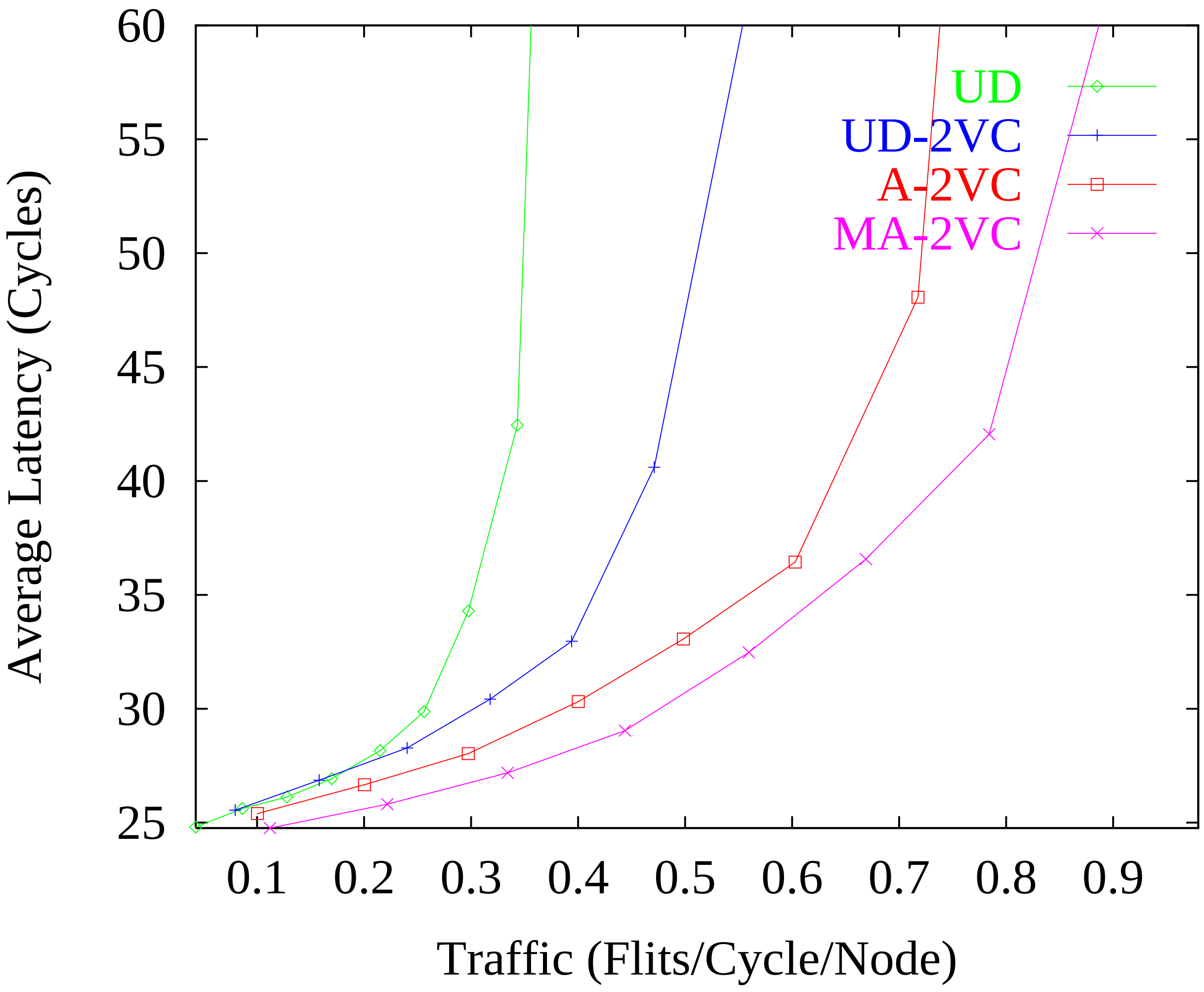
- Basic up*/down* routing scheme (UD).
- Up*/down* routing scheme using two virtual channels per physical channel (UD-2VC).
- Adaptive routing scheme using two virtual channels per physical channel (A-2VC).
- Improved adaptive routing scheme using two virtual channels per physical channel (MA-2VC).

Performance evaluation carried out by simulation.

Network model:

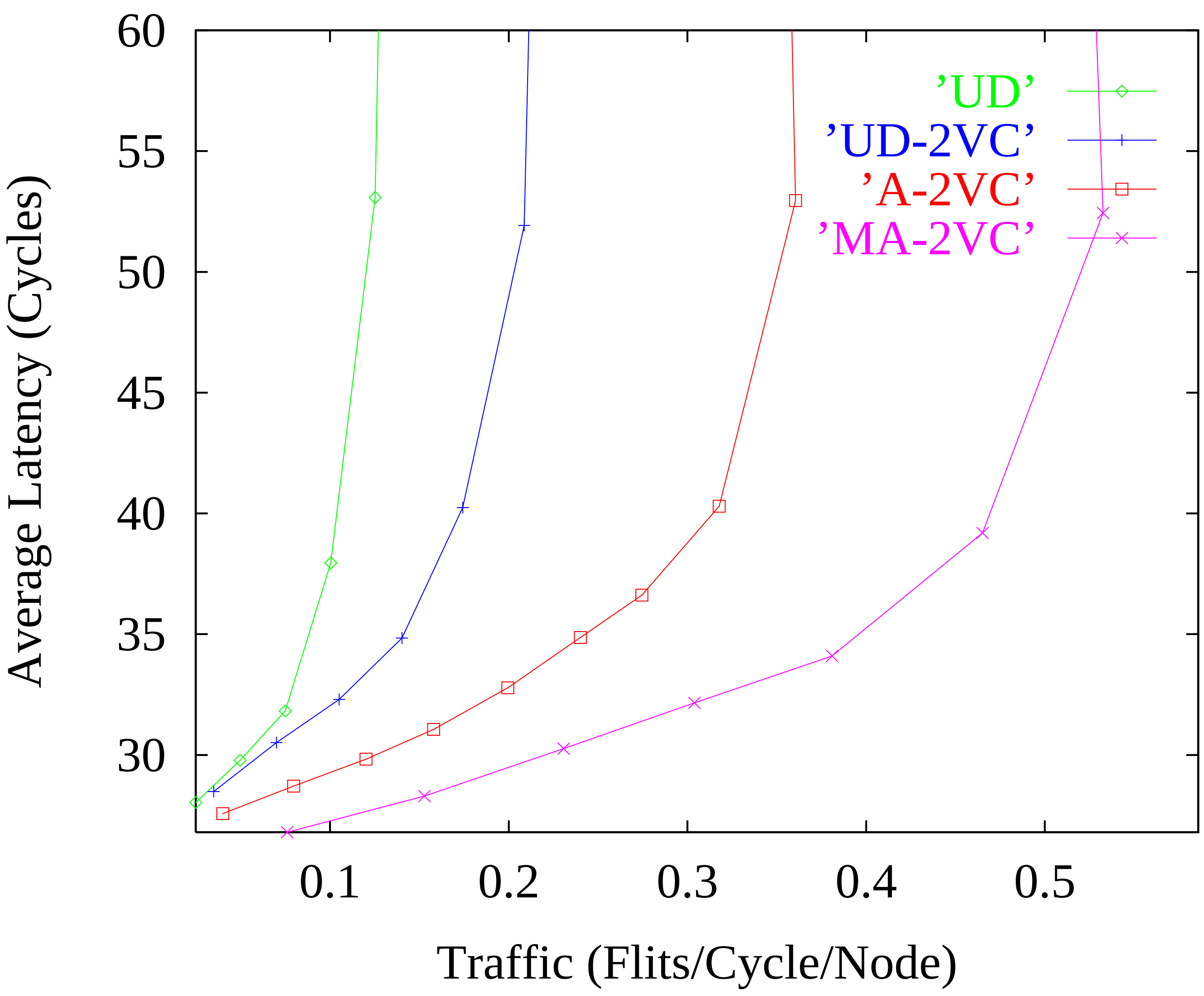
- Topology generated randomly (8-port switches)
- 4 nodes (processors) connected to each switch
- Two adjacent switches are connected by a single link
- One routing control unit per switch (assigned in a round-robin fashion)
- Message destination is randomly chosen among nodes
- It takes one clock cycle to compute the routing algorithm, to transfer one flit from an input buffer to an output buffer, or to transfer one flit across a physical channel

Routing algorithm: simulation results (I)



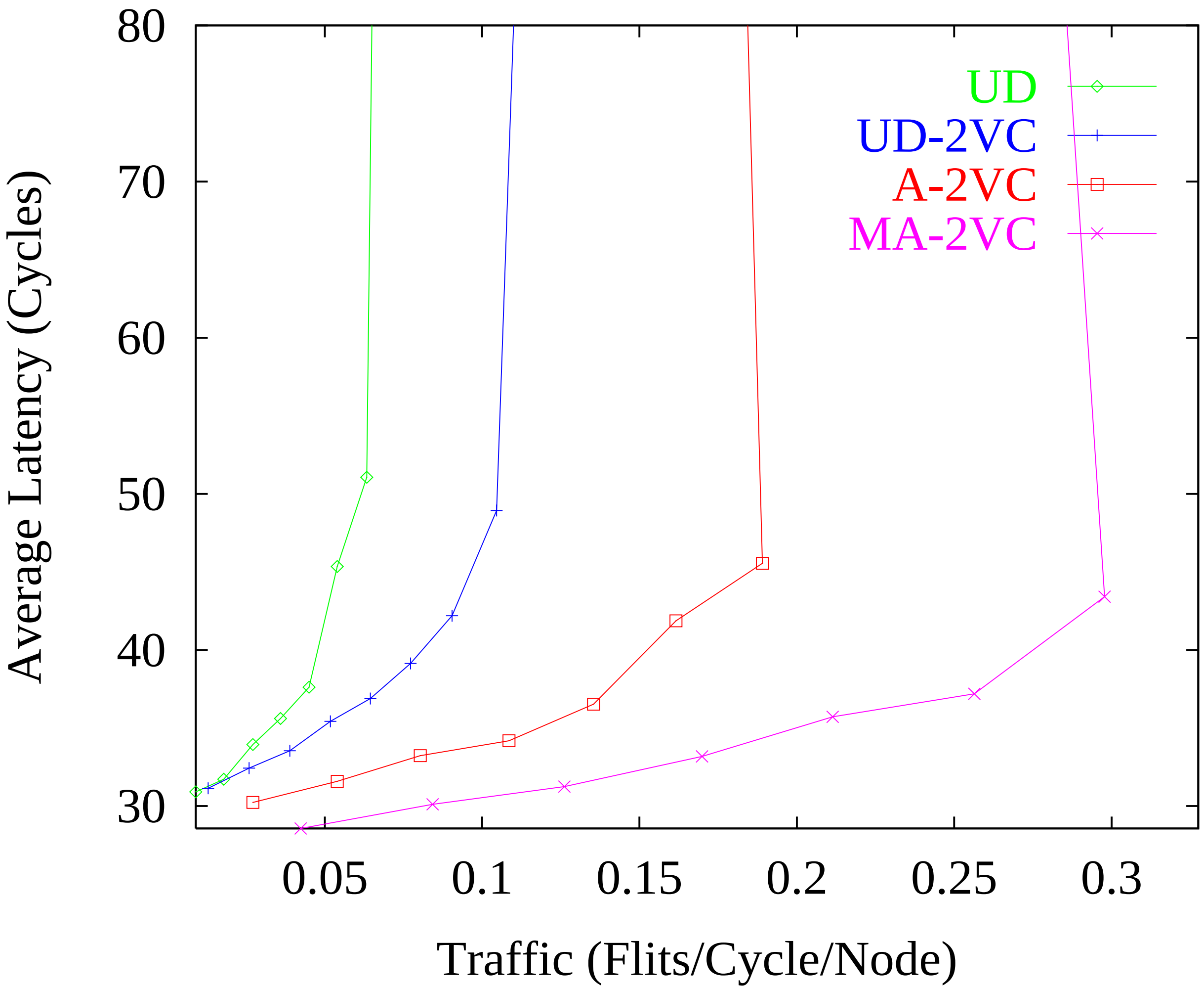
- Network size: 16 switches
- Message length: 16 flits

Routing algorithm: simulation results (II)



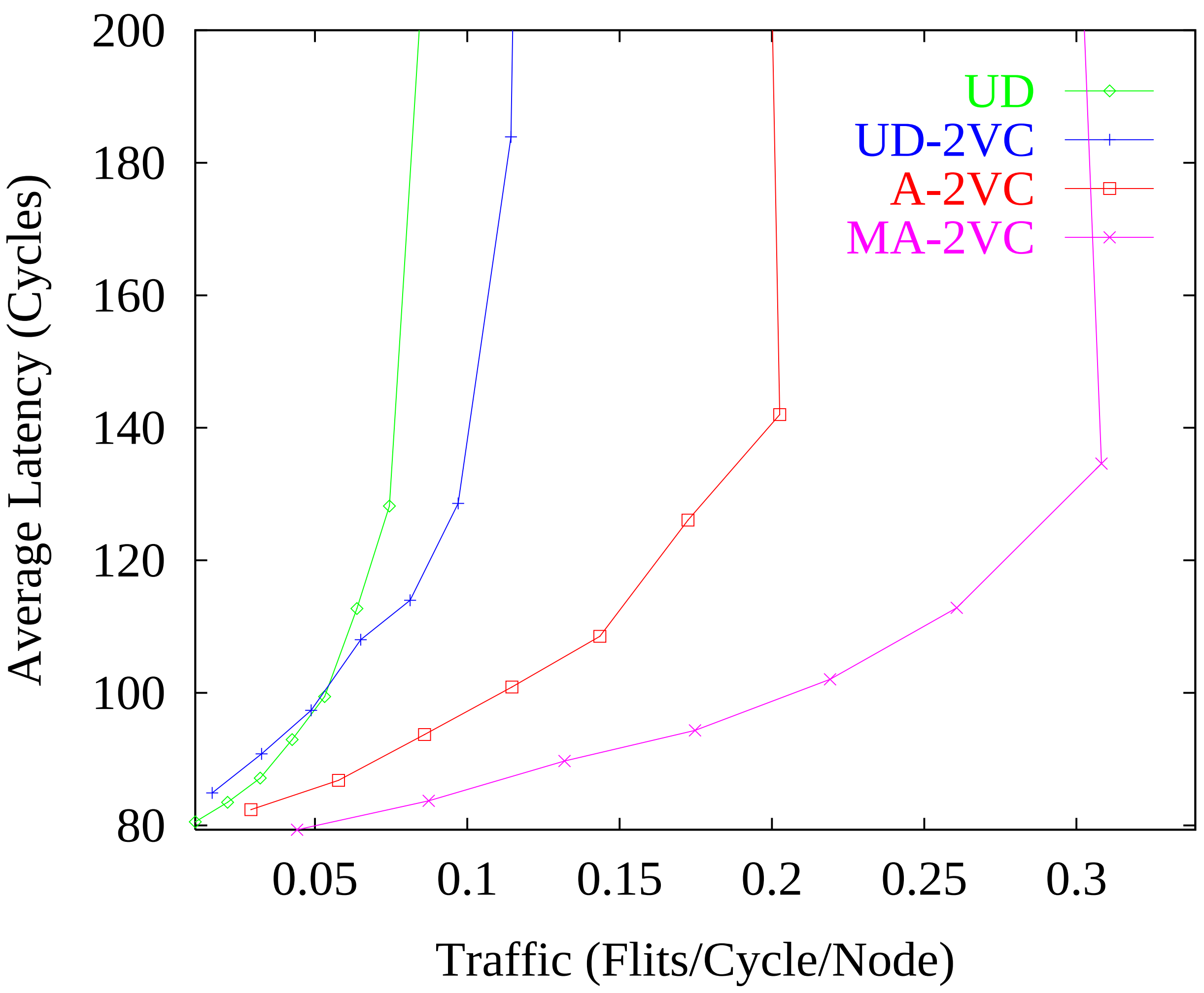
- Network size: 32 switches
- Message length: 16 flits

Routing algorithm: simulation results (III)



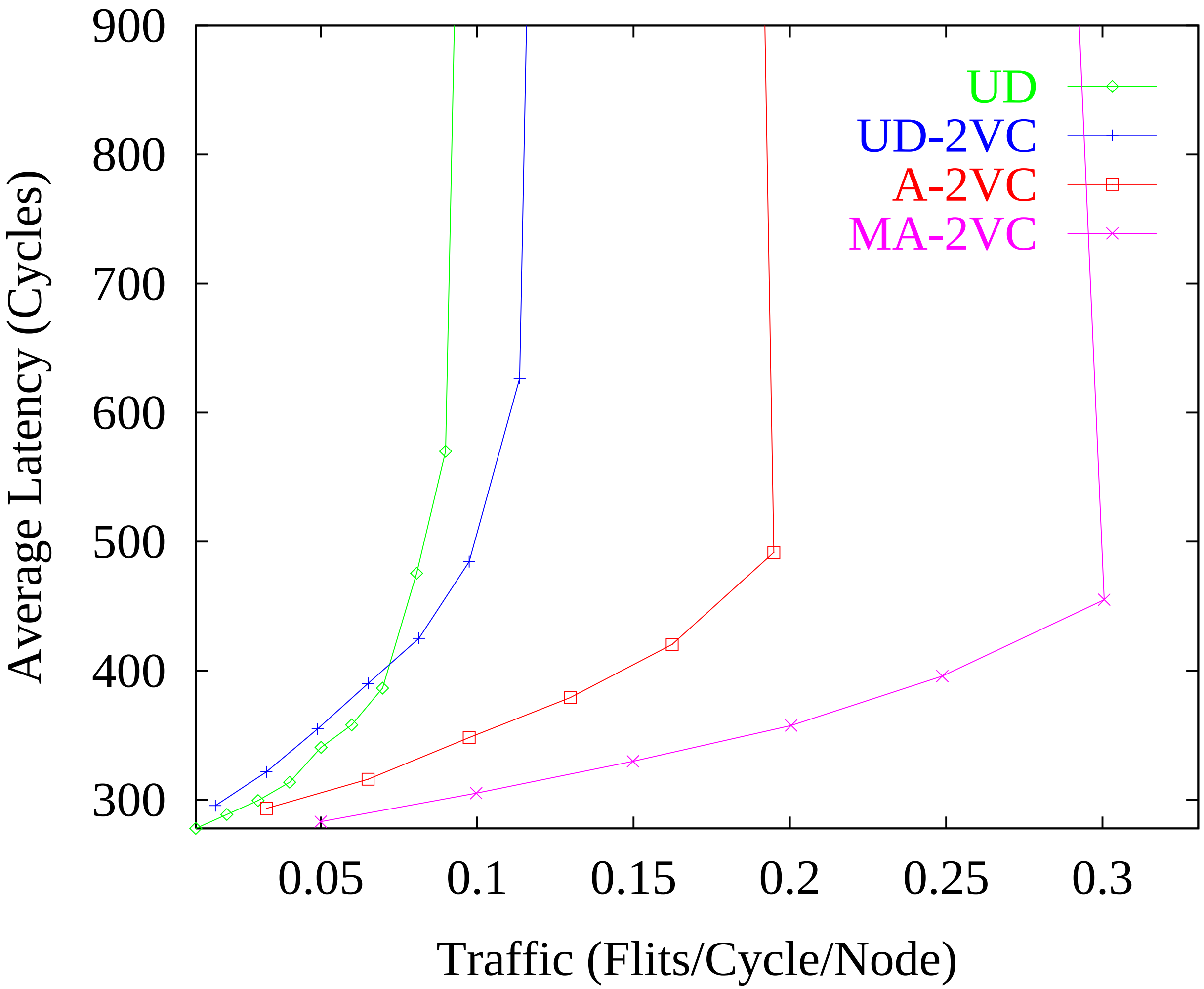
- Network size: 64 switches
- Message length: 16 flits

Routing algorithm: simulation results (IV)



- Network size: 64 switches
- Message length: 64 flits

Routing algorithm: simulation results (V)



- Network size: 64 switches
- Message length: 256 flits

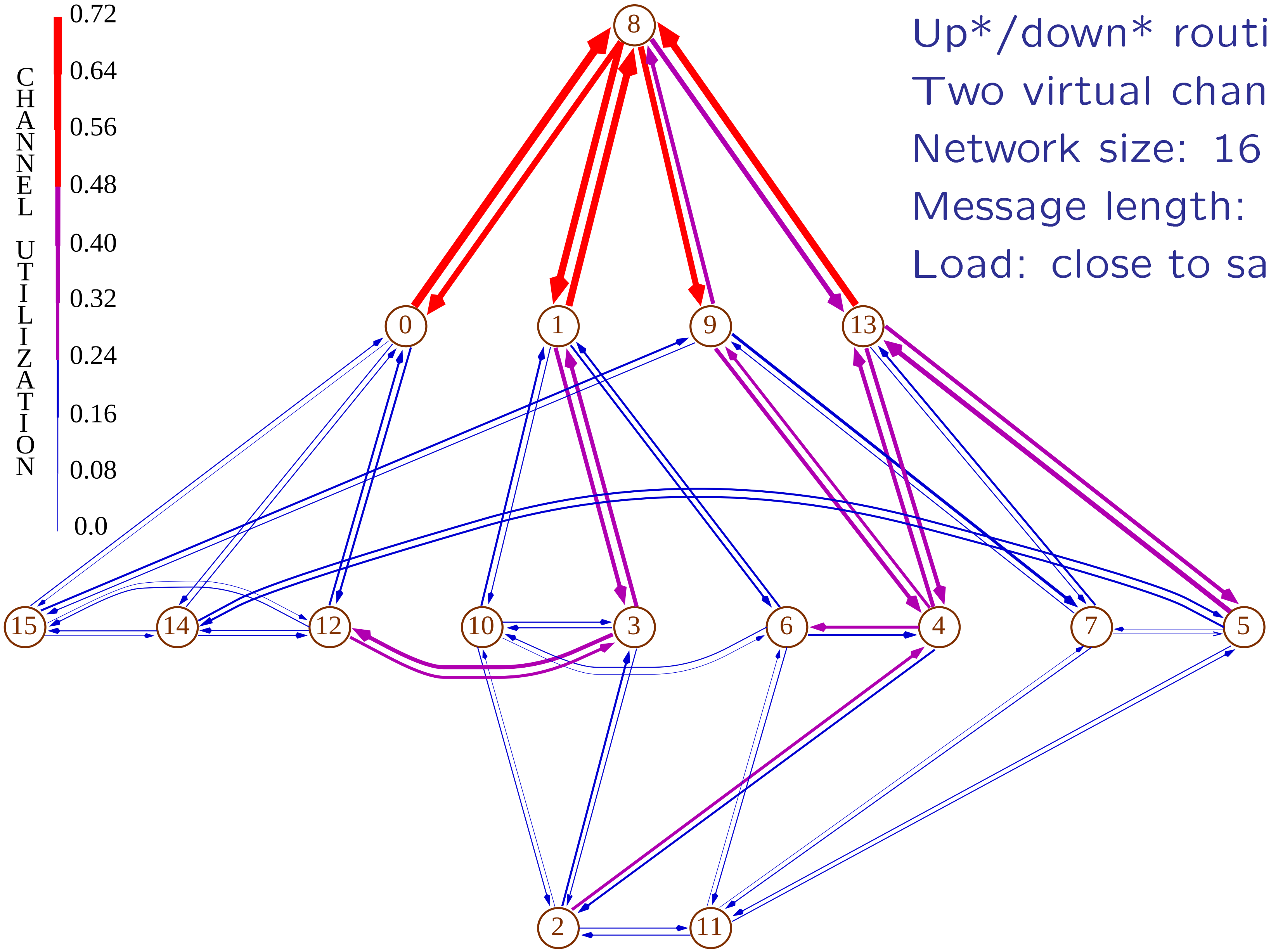
Up*/down* routing.

Two virtual channels.

Network size: 16 switches.

Message length: 16 flits.

Load: close to saturation.



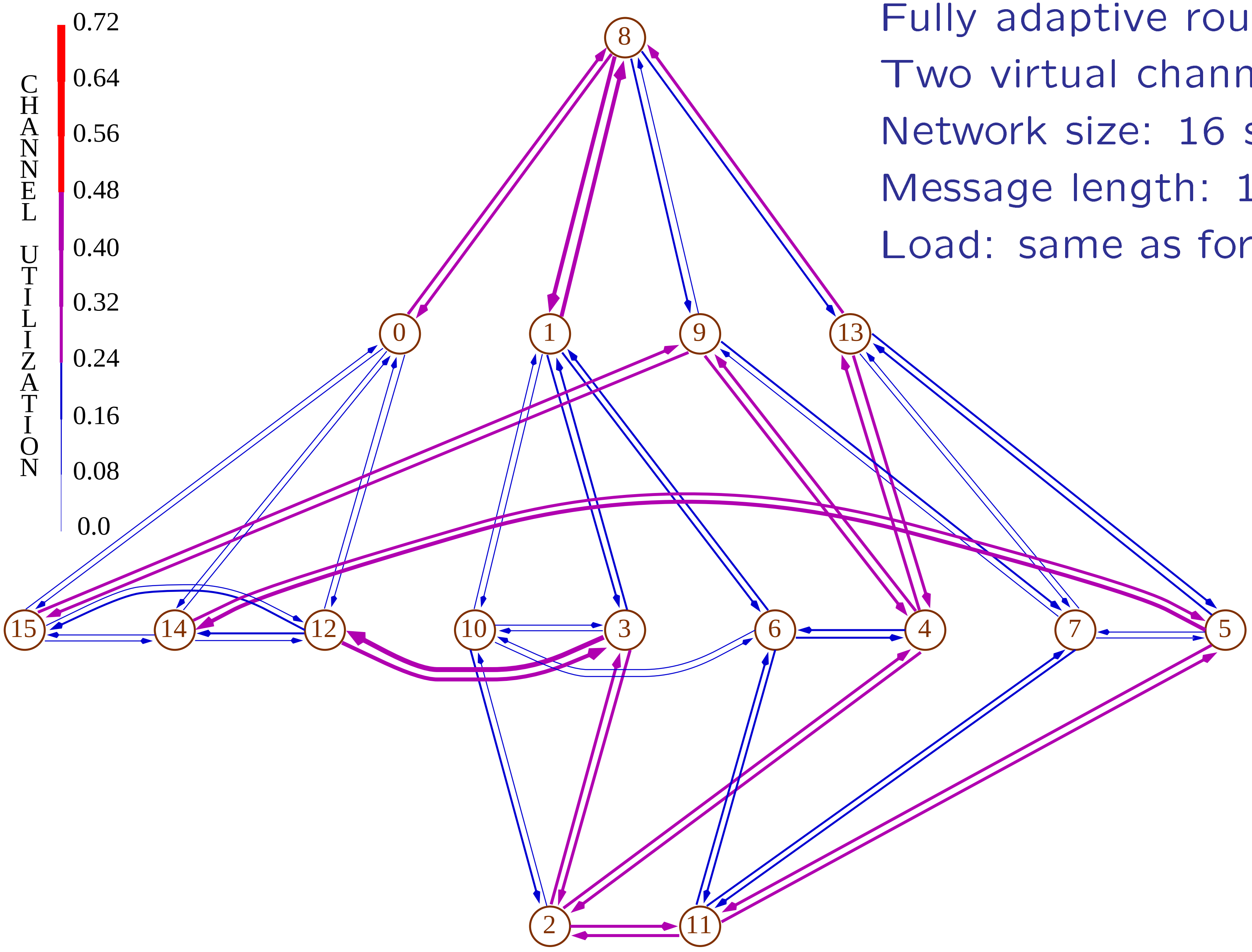
Fully adaptive routing.

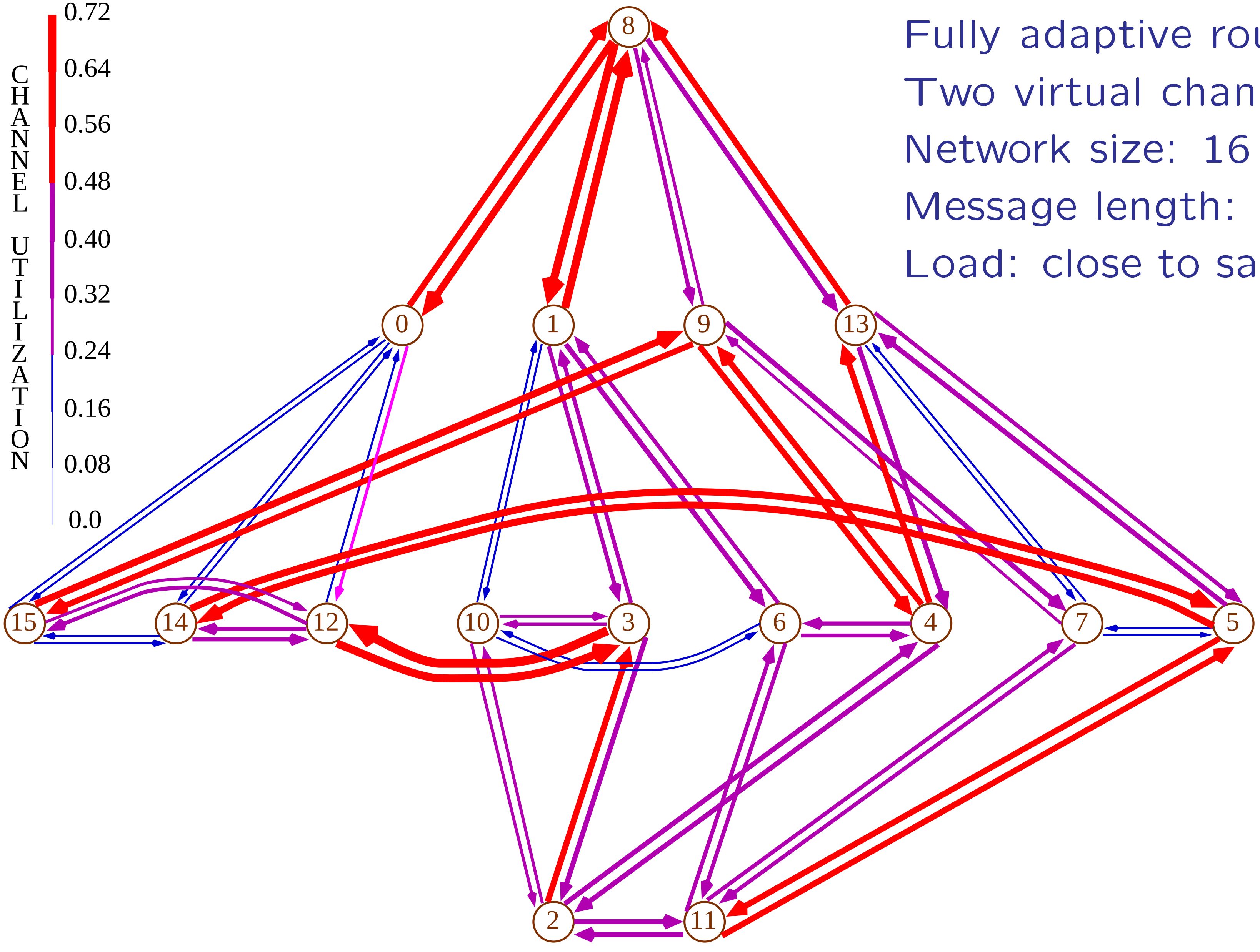
Two virtual channels.

Network size: 16 switches.

Message length: 16 flits.

Load: same as for up*/down*





Fully adaptive routing.

Two virtual channels.

Network size: 16 switches.

Message length: 16 flits.

Load: close to saturation.

Simulation Model Based on Application Message Traces

Is the network in a NOW able to handle the traffic supported by the network in a DSM without saturating?

We gathered message traces at the node controller level in a simulated DSM, then feeding a NOW with those traces.

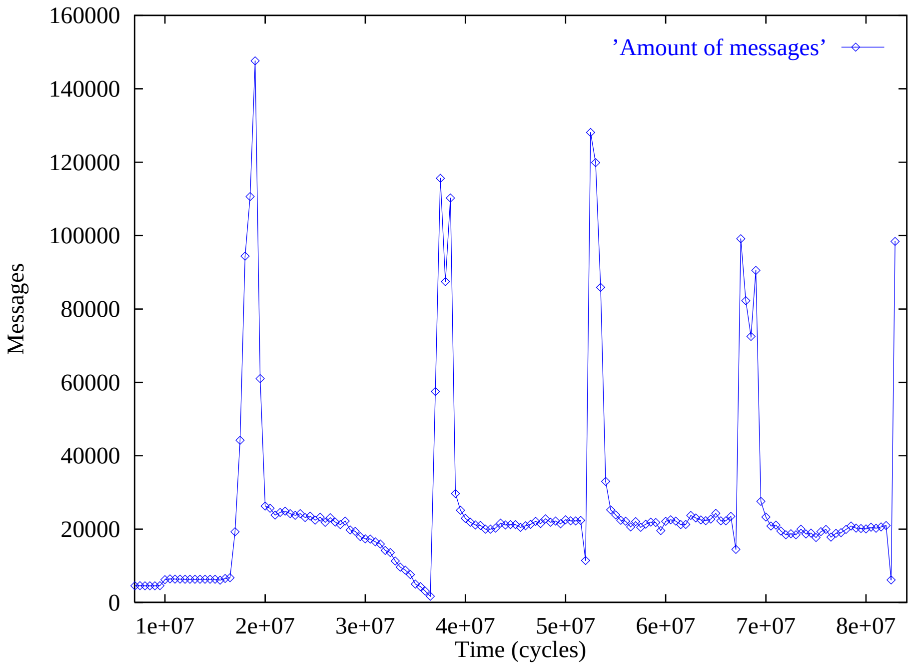
Traces were gathered from SPLASH2 applications:

Application	Problem Size
Barnes-Hut	$8K$ particles, $\theta = 1.0$, 4 time steps
LU	512×512 doubles, 8×8 blocks
Radix	$1M$ keys, $1K$ radix, max $1M$
Water	512 molecules, 4 time steps

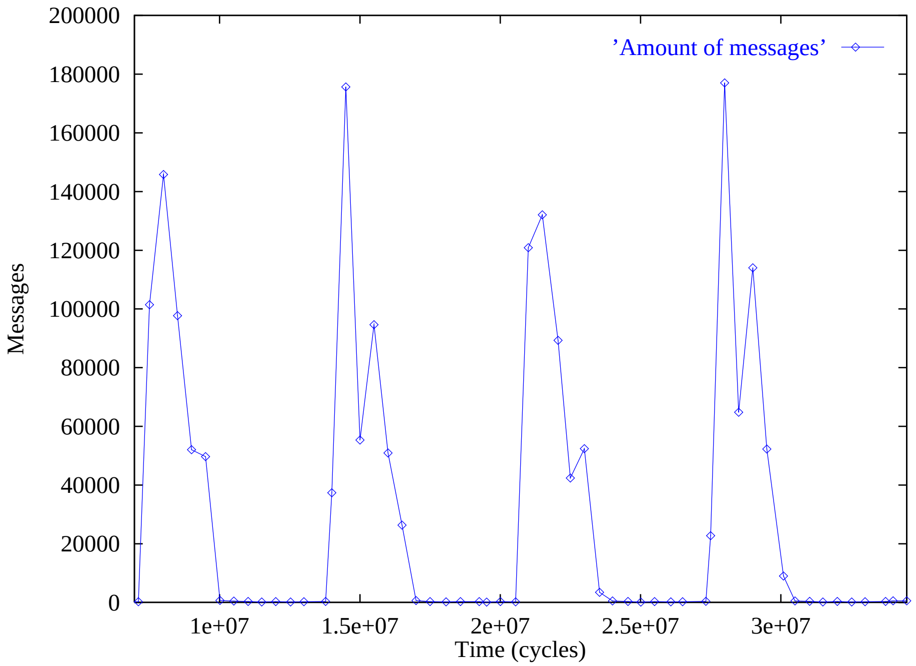
Main characteristics of the simulated systems:

	DSM	NOW
Number of processors	64	64
Topology	8 x 8 mesh	irregular (16 switches)
Routing algorithm	Deterministic	Up/Down, MA-2vc
Link Width	16 bits	8 bits
Link Bandwidth	400 MB/sec	200 MB/sec

Number of messages during execution time

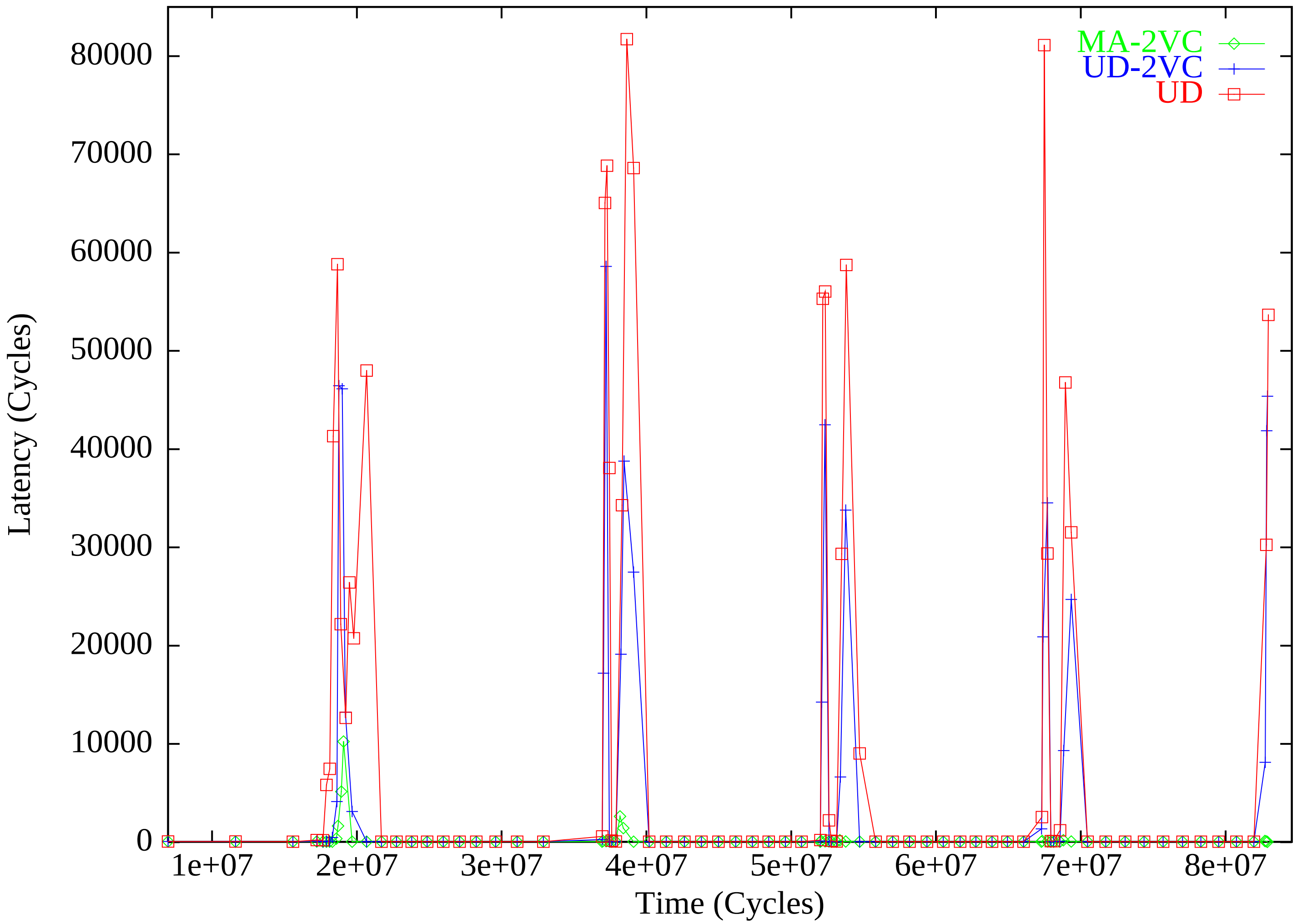


Barnes-Hut

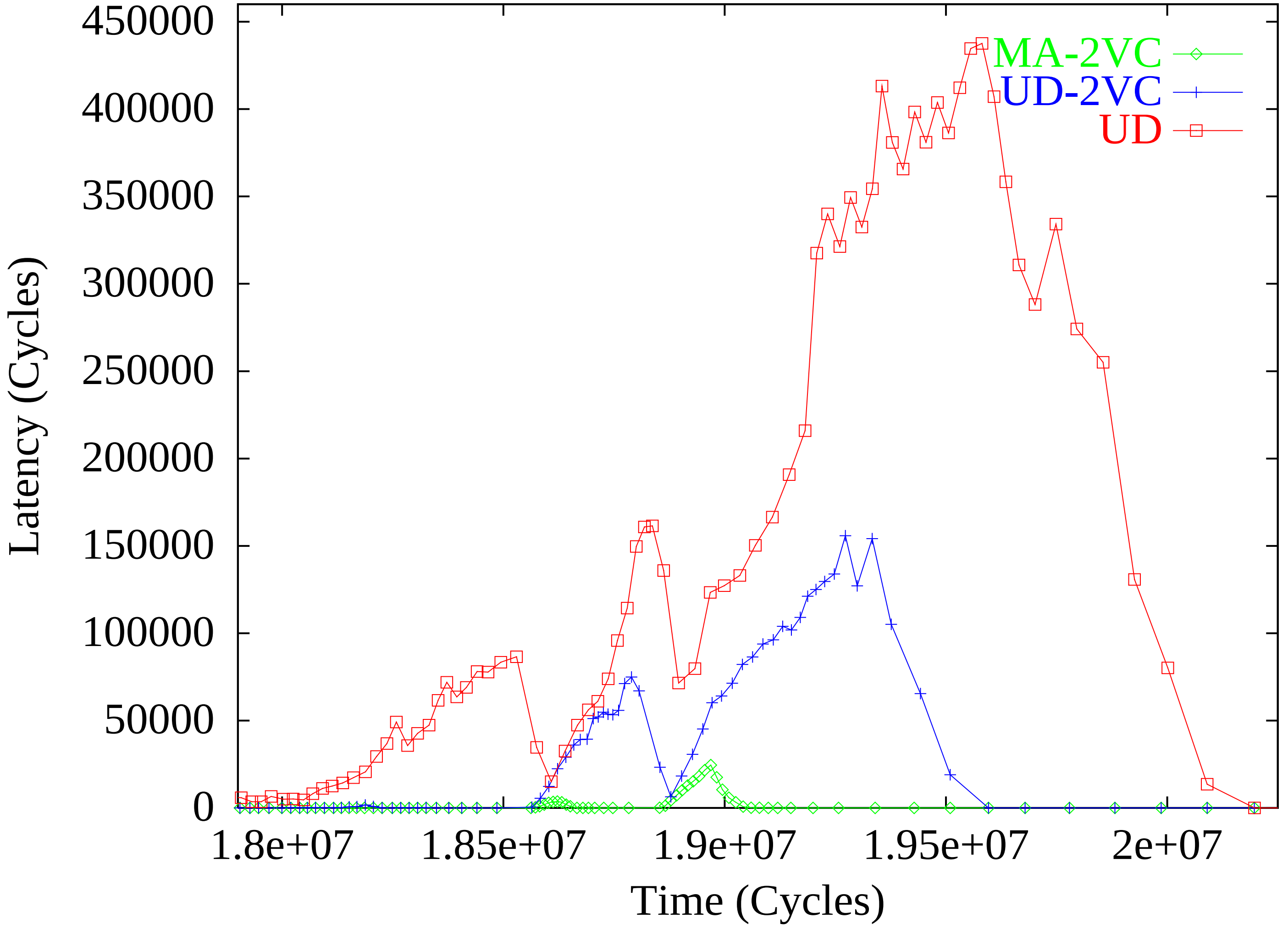


Water

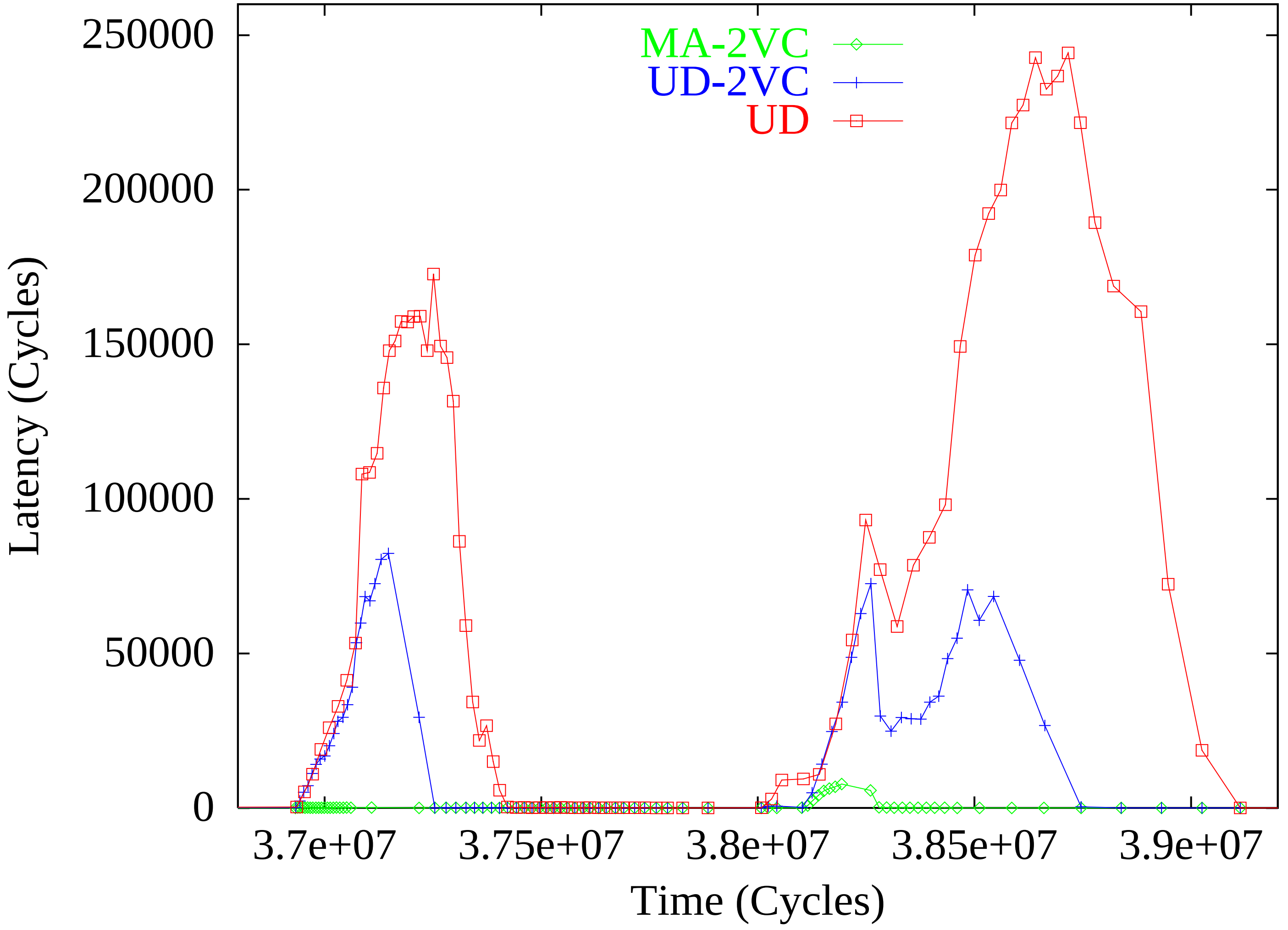
Simulation results for application traces (Barnes-Hut)



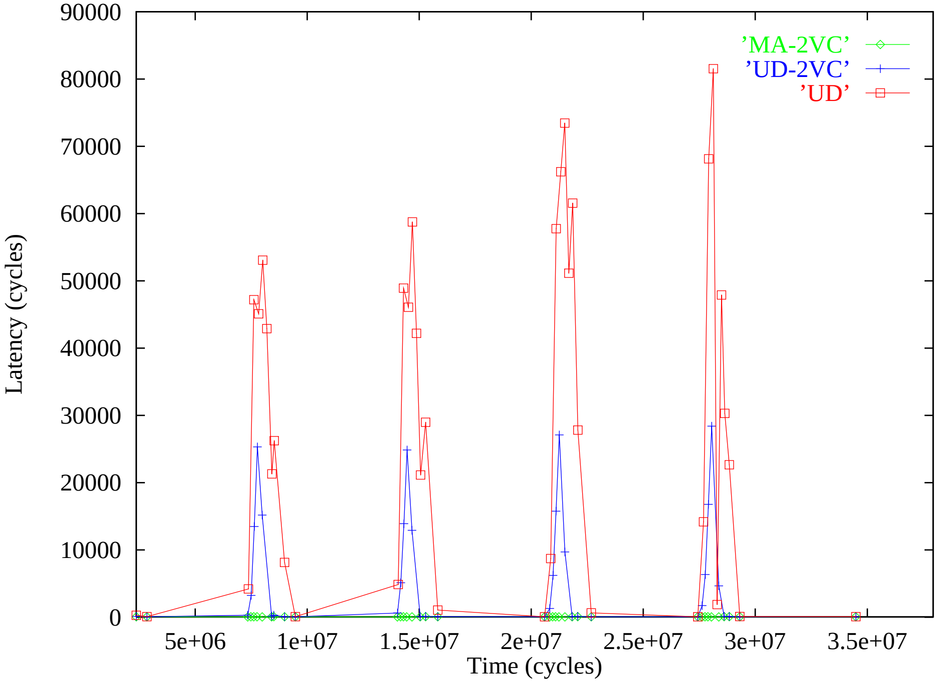
Zoom of the first peak (Barnes-Hut)

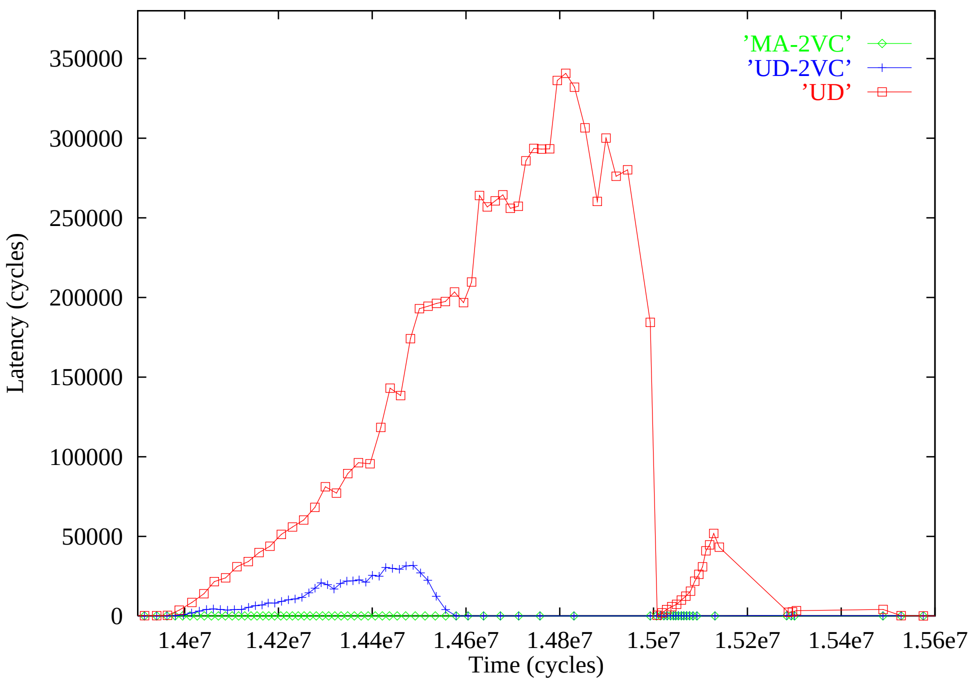


Zoom of the second peak (Barnes-Hut)



Simulation results for Water

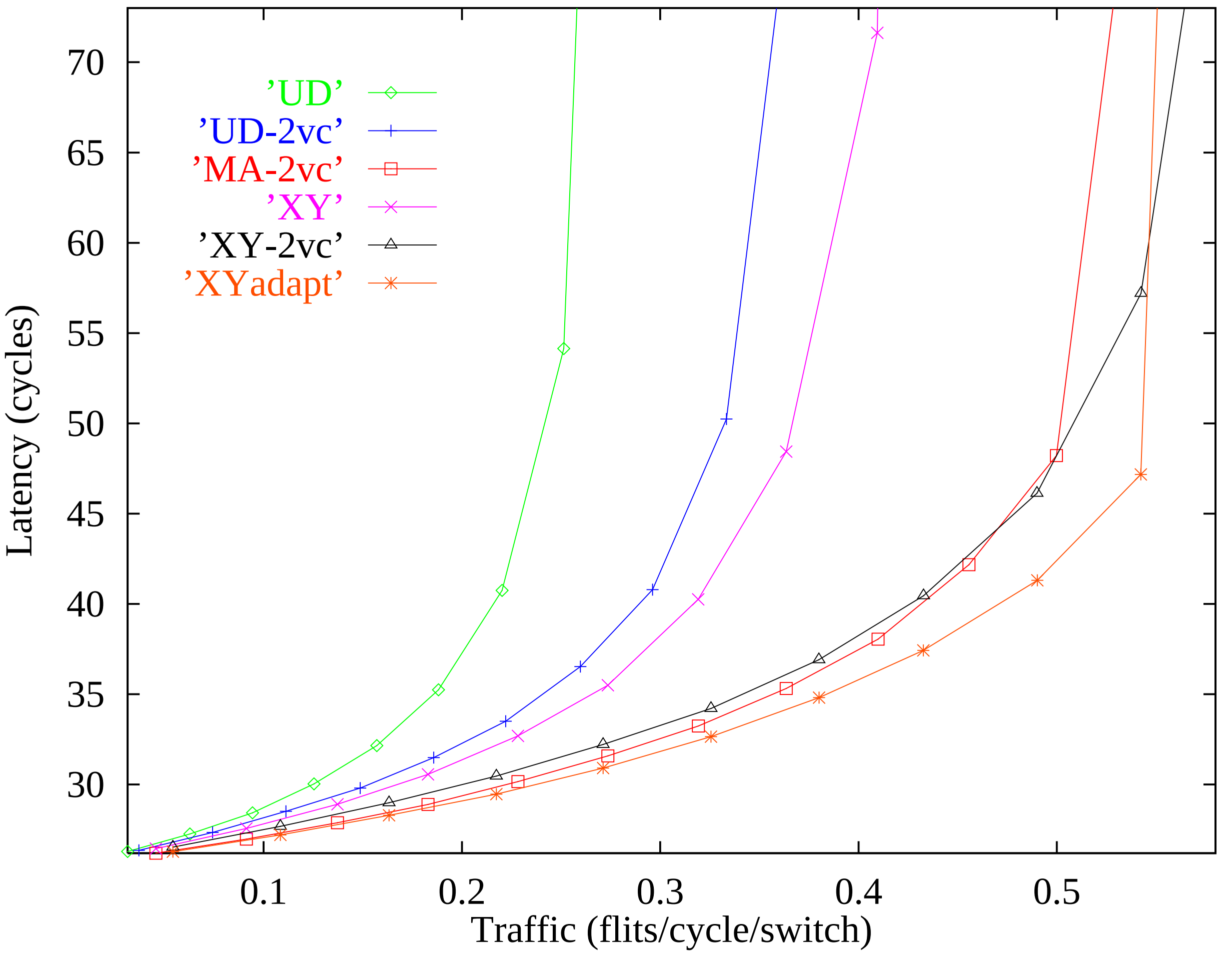




Average message latency during Water execution

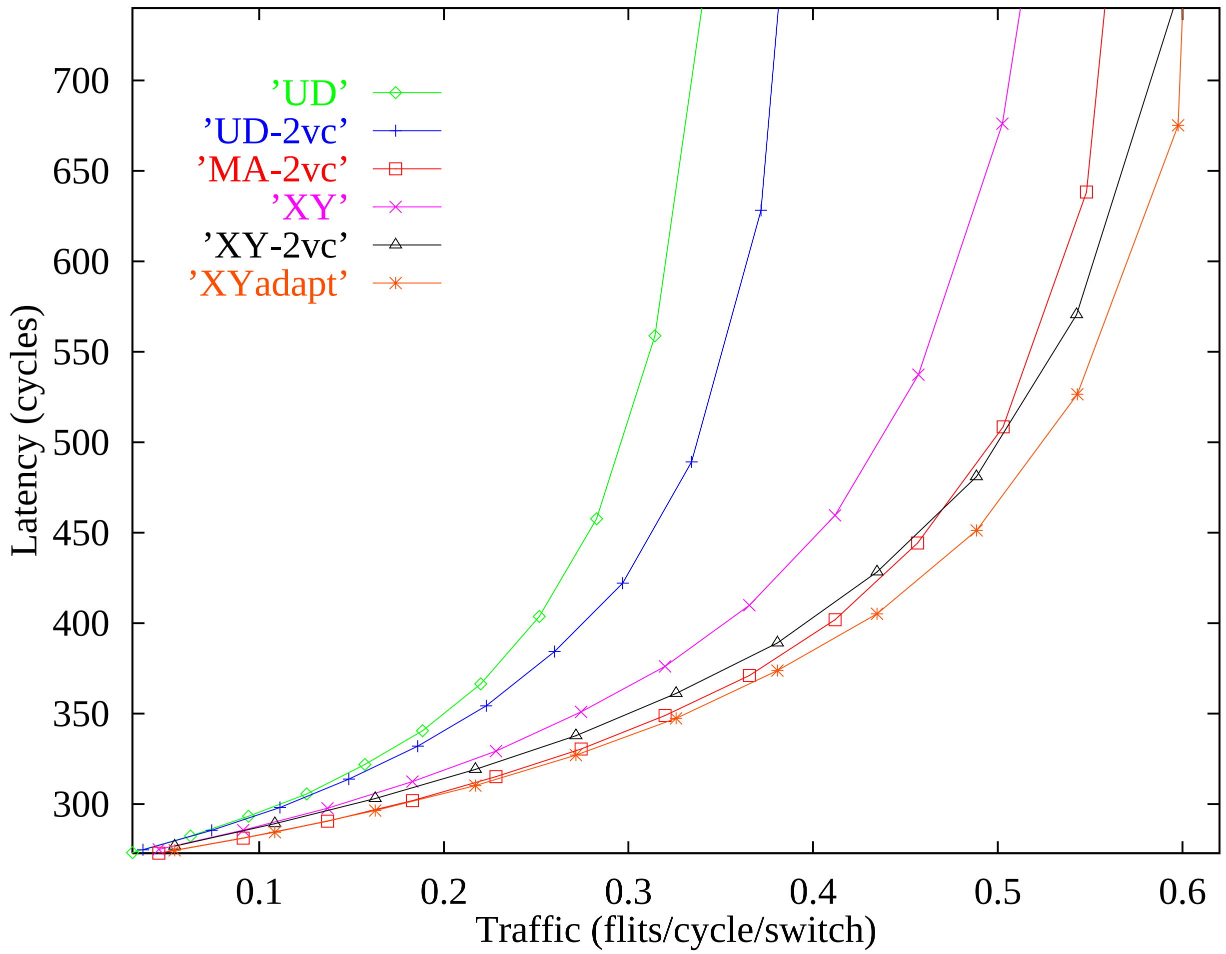
Zoom of the second saturation point

How do irregular networks compare with regular ones (I)?



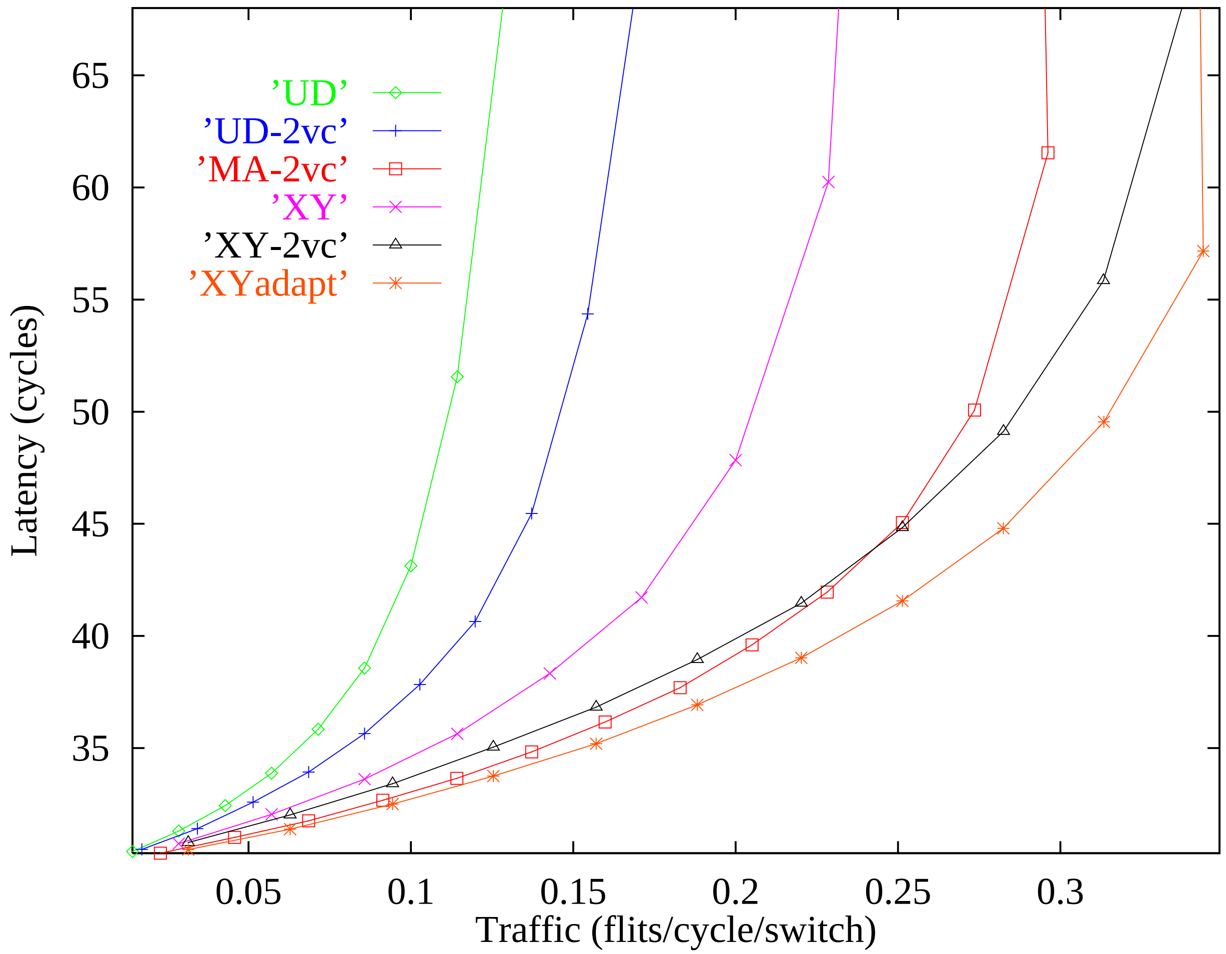
- Network size: 16 switches
- Message length: 16 flits

How do irregular networks compare with regular ones (II)?



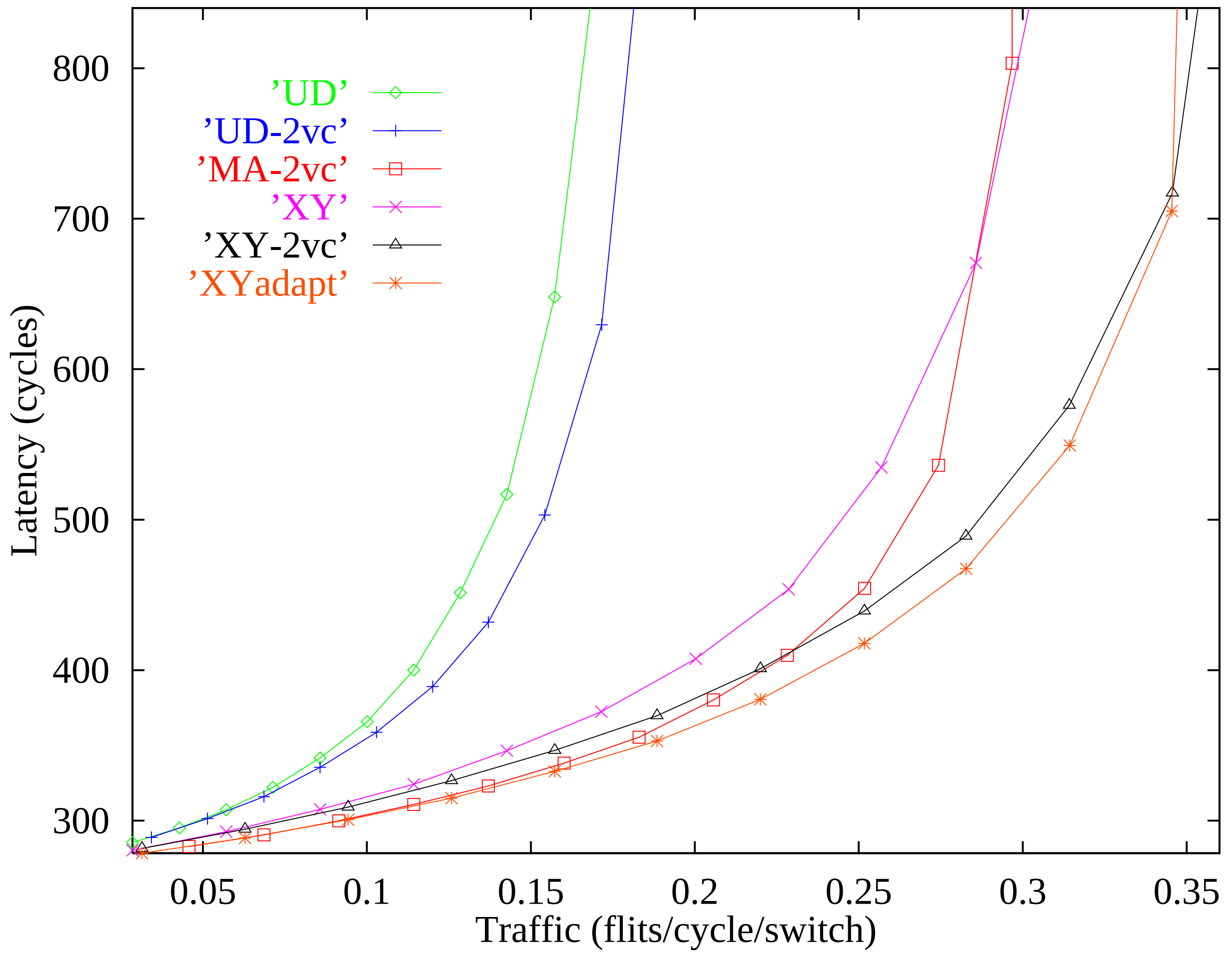
- Network size: 16 switches
- Message length: 256 flits

How do irregular networks compare with regular ones (III)?



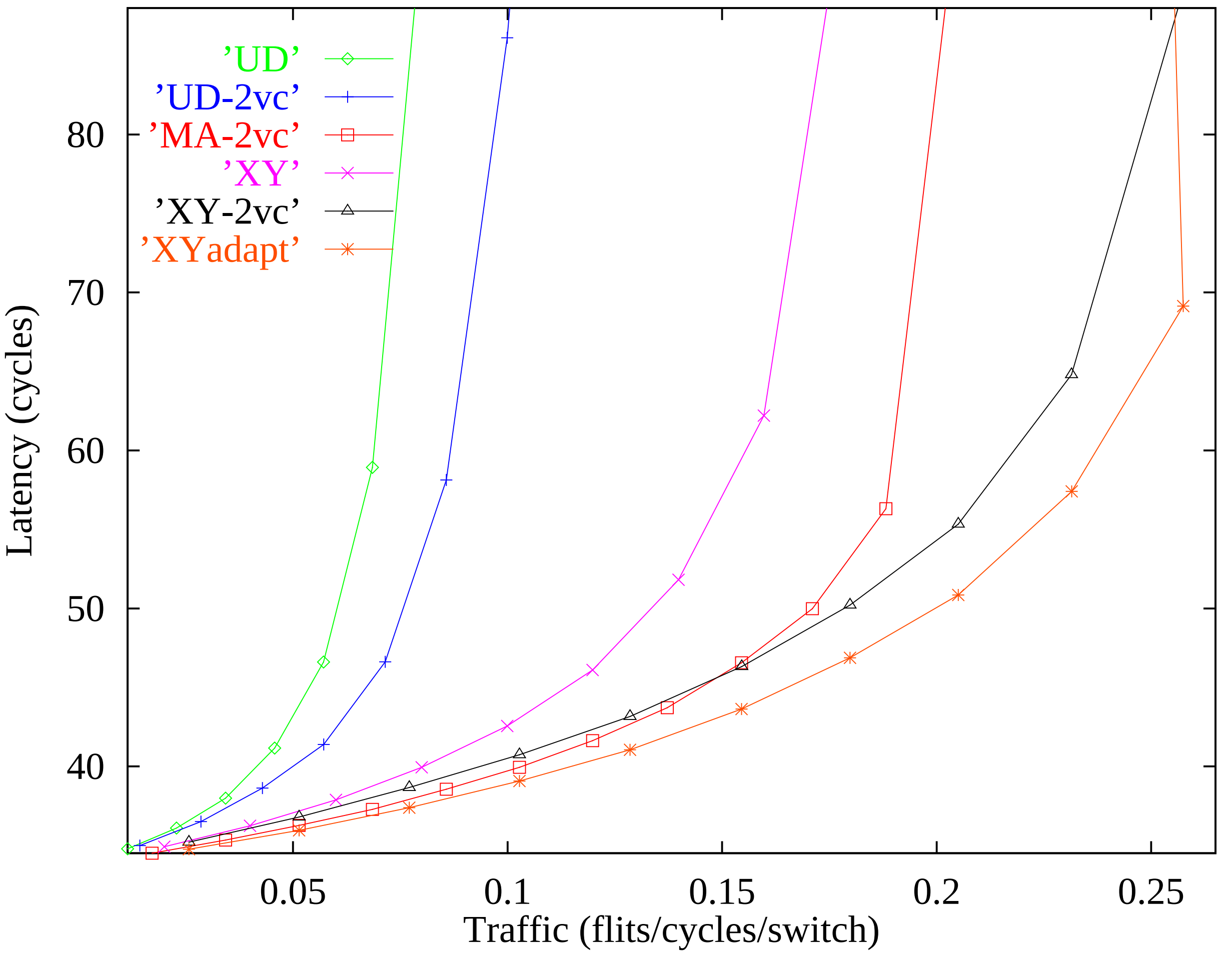
- Network size: 36 switches
- Message length: 16 flits

How do irregular networks compare with regular ones (IV)?



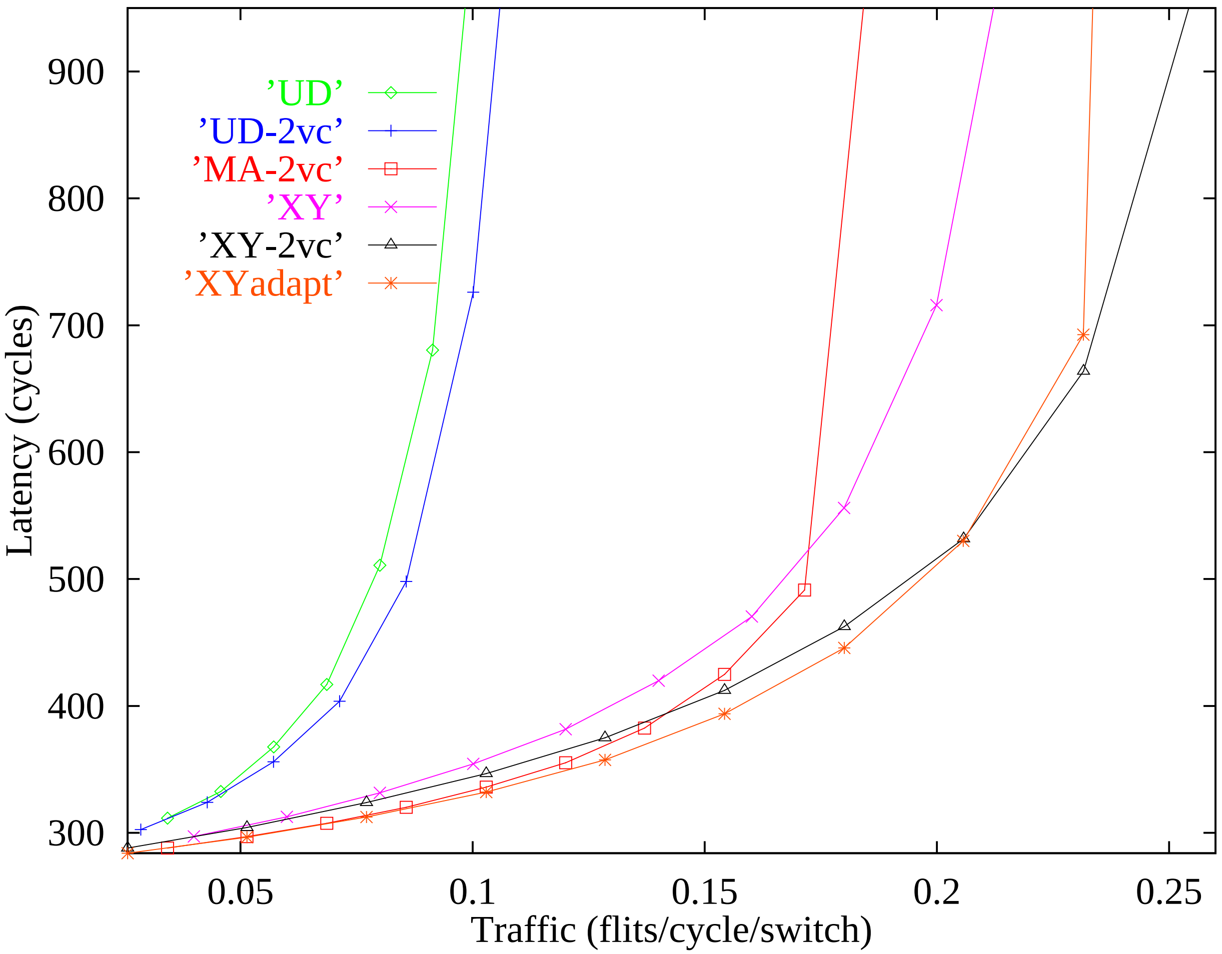
- Network size: 36 switches
- Message length: 256 flits

How do irregular networks compare with regular ones (V)?



- Network size: 64 switches
- Message length: 16 flits

How do irregular networks compare with regular ones (VI)?



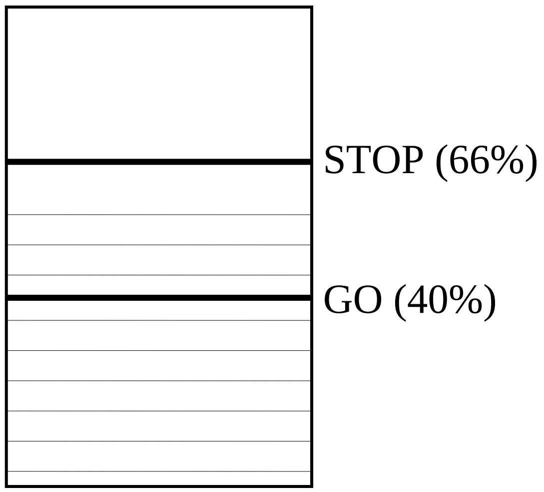
- Network size: 64 switches
- Message length: 256 flits

Flow control in NOWs

Links connecting switches are usually long and may have very different lengths $\Rightarrow$ Channel delay is high and may reduce clock frequency.
Solution: Use channel pipelining

Channel width is small $\Rightarrow$ It is not interesting to use dedicated lines for flow control signals.
Solution: Use control flits and Stop & Go flow control

"Stop" & "Go" protocol



Designed to minimize the traffic of control flits

Input buffers have two watermarks: "Stop" and "Go", that divide the buffer into three parts

Two control flits: "Stop" and "Go"

Virtual Channels

No dedicated flow control lines $\Rightarrow$ control flits are required to manage virtual channels

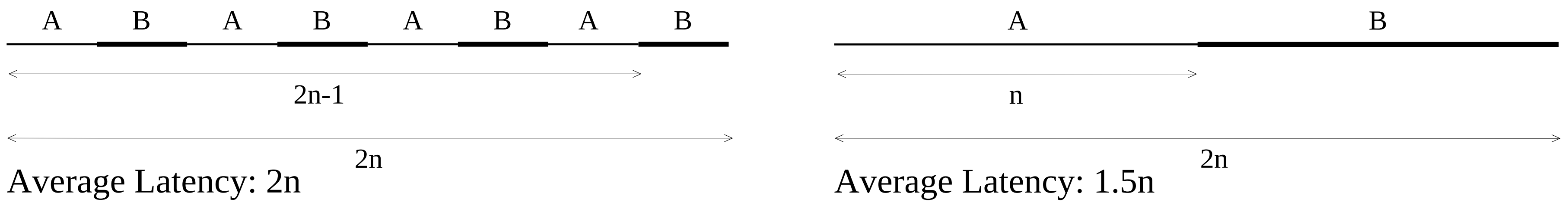
If flit-by-flit multiplexing is used, 50% of the bandwidth is wasted due to the transmission of control flits

How to perform channel multiplexing efficiently?

Solution: Transmit blocks of flits

Block multiplexing

Message Size: n flits



(a) (b)

The virtual channel that owns the link sends flits until its queue empties or it receives a “Stop” control flit

Block multiplexing minimizes the use of control flits and may also reduce average message latency

Flow control protocol

- Before transmitting data flits, a "Select n" control flit is sent, where n is the virtual channel number
- Data flits are sent until the message blocks or it has been completely transferred
- If the input buffer fills over the "Stop" mark, a "Stop n" control flit is sent. When a "Stop n" control flit is received, the transmitter switches to the next ready virtual channel
- If the input buffer empties below the "Go" mark, a "Go n" control flit is sent. When a "Go n" control flit is received, the transmitter simply records the status change

Performance evaluation

Evaluation of four flow control protocols:

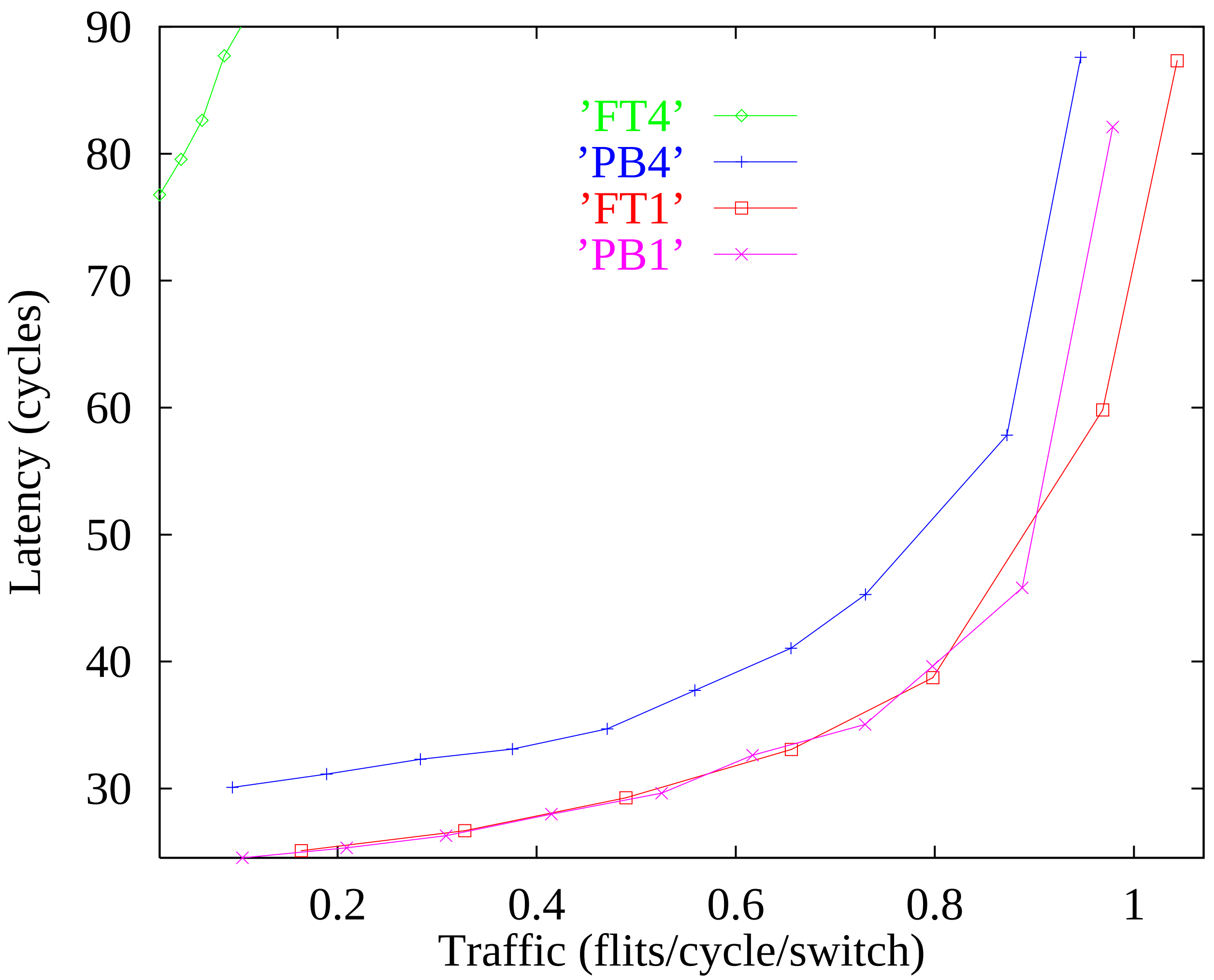
- Flit-by-flit multiplexing with long wires (FT4 or FT8 depending on wire delay)
- Pipelined channels with block multiplexing (PB4 or PB8 depending on wire delay)
- Ideal network (flit-by-flit multiplexing with short wires) (FT1)
- Block multiplexing with short wires (PB1)

Performance evaluation carried out by simulation

Network model:

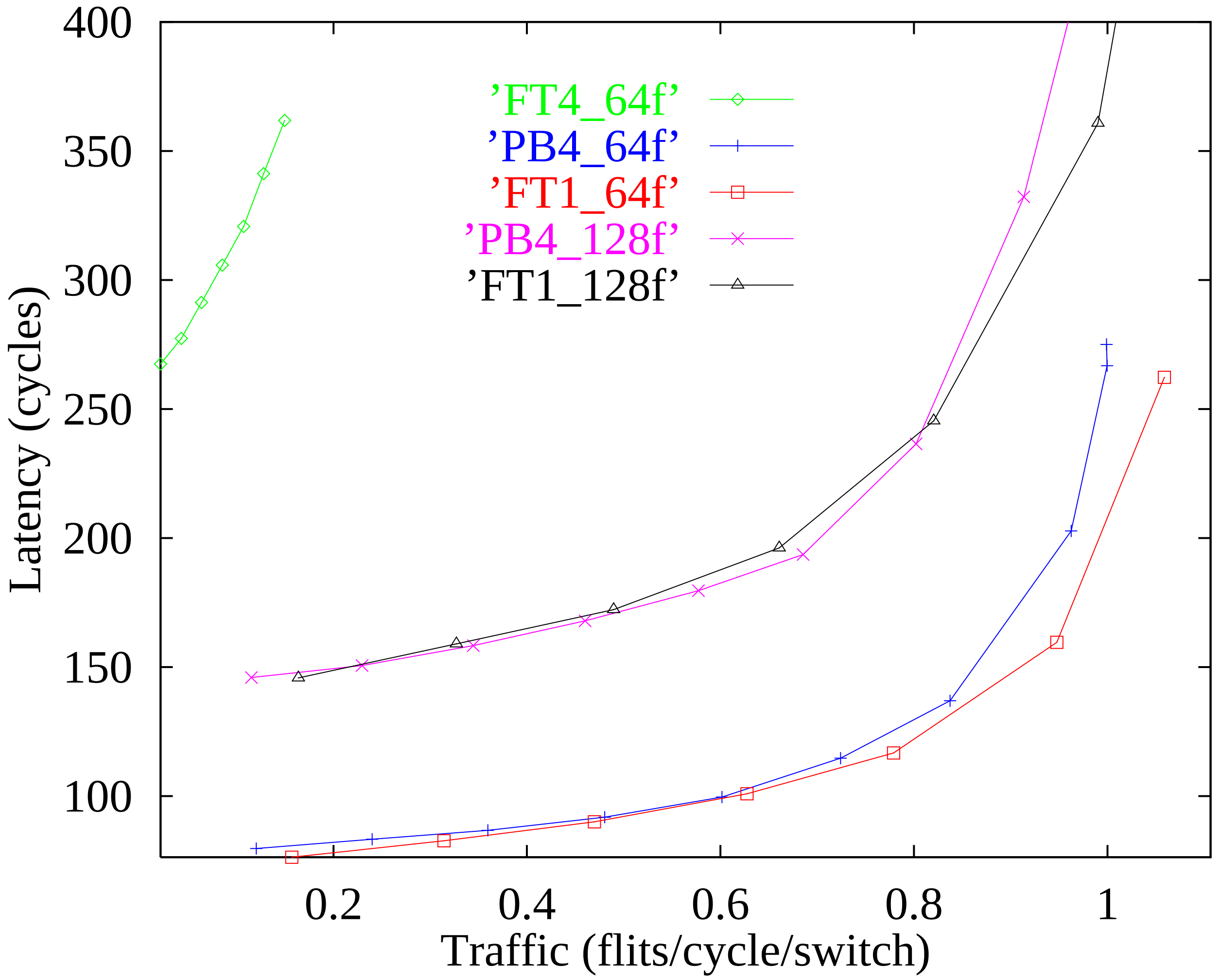
- Topology generated randomly
- 4 nodes (processors) connected to each switch
- Two contiguous nodes connected by a single link
- 8-port switches
- One routing control unit per switch (assigned in a round-robin fashion)
- Fully adaptive routing
- Message destination is randomly chosen among nodes

Flow control: simulation results (I)



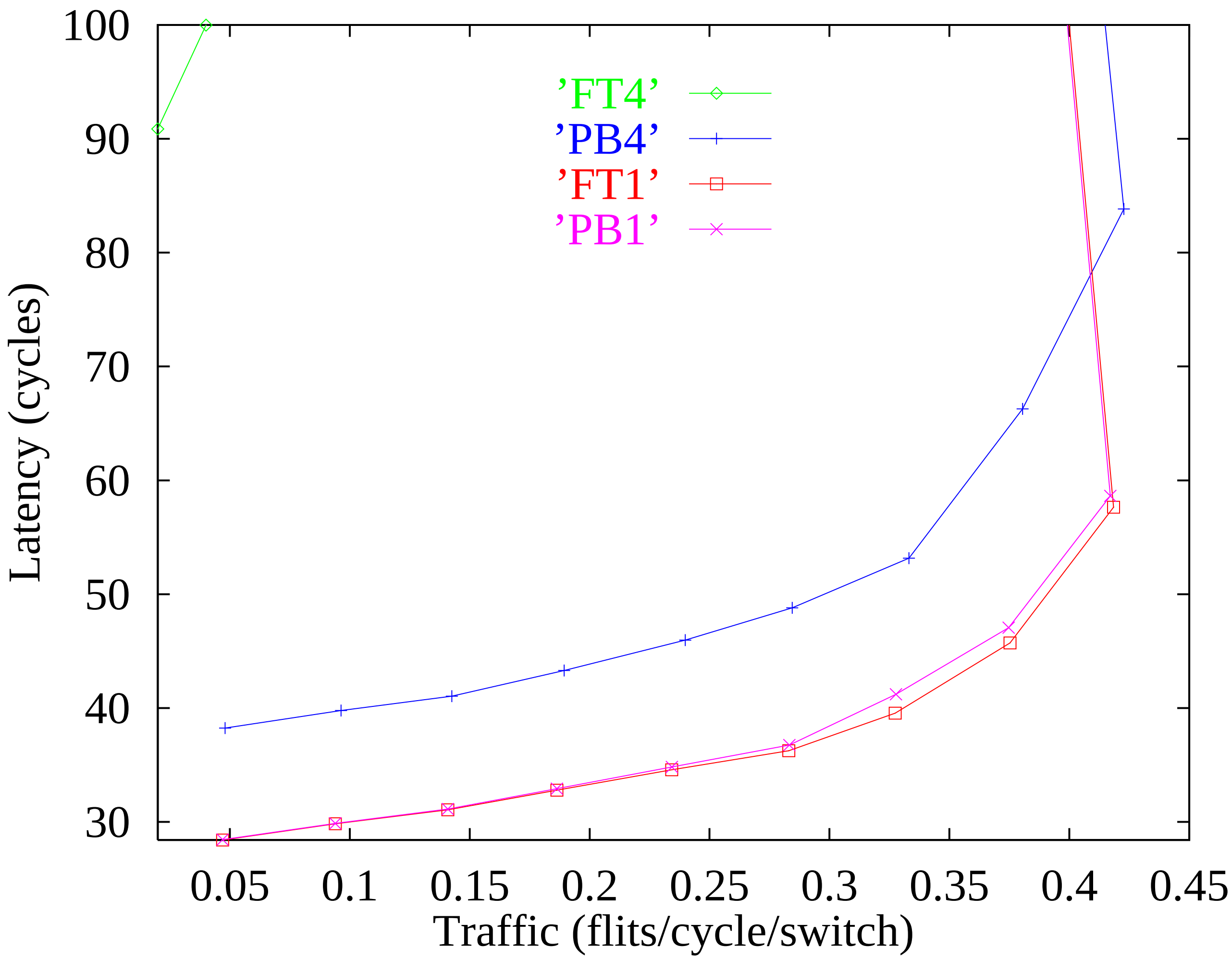
- Network size: 16 switches
- Message length: 16 flits
- Fly time: 4 cycles

Flow control: simulation results (II)



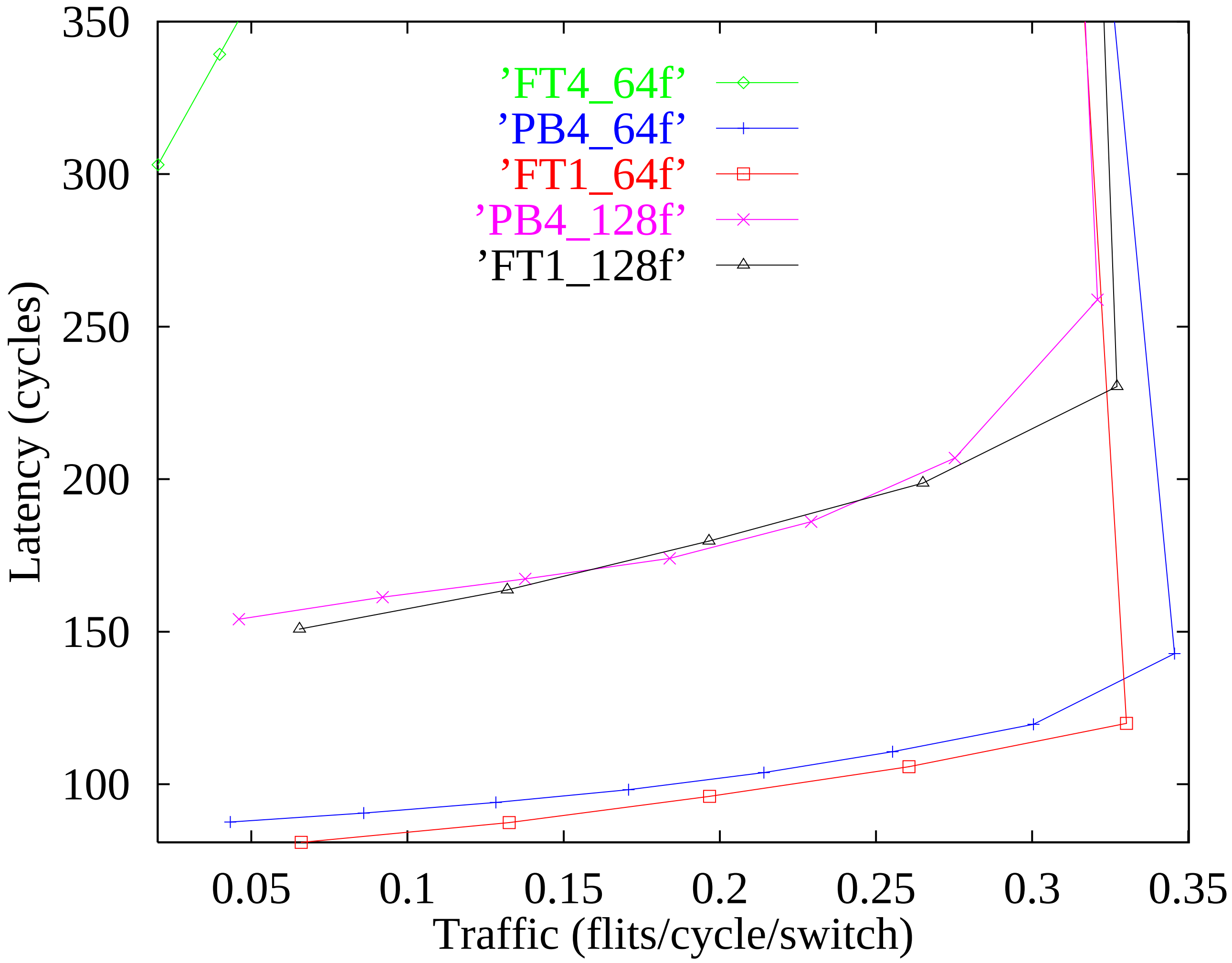
- Network size: 16 switches
- Message length: 64 and 128 flits
- Fly time: 4 cycles

Flow control: simulation results (III)



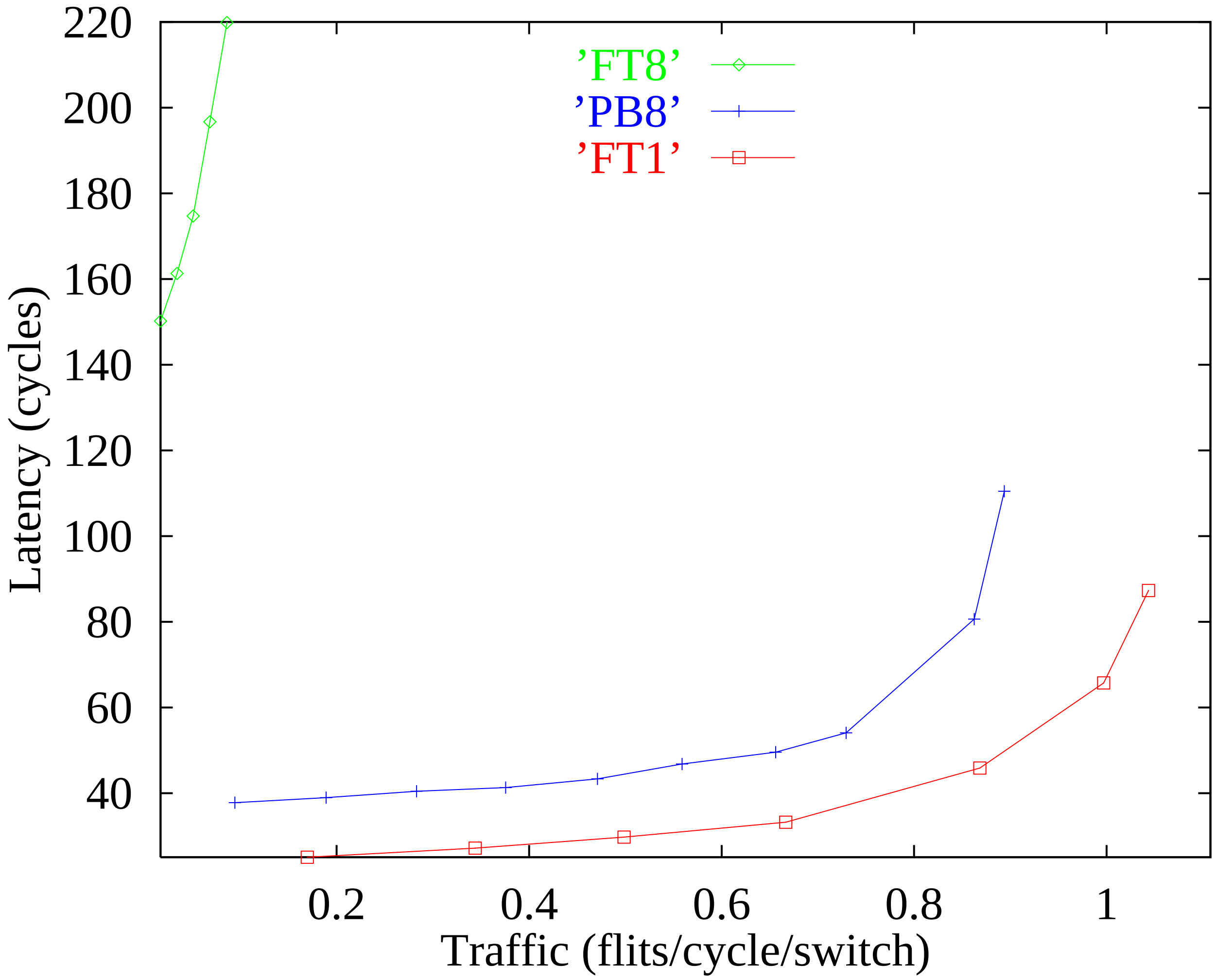
- Network size: 64 switches
- Message length: 16 flits
- Fly time: 4 cycles

Flow control: simulation results (IV)



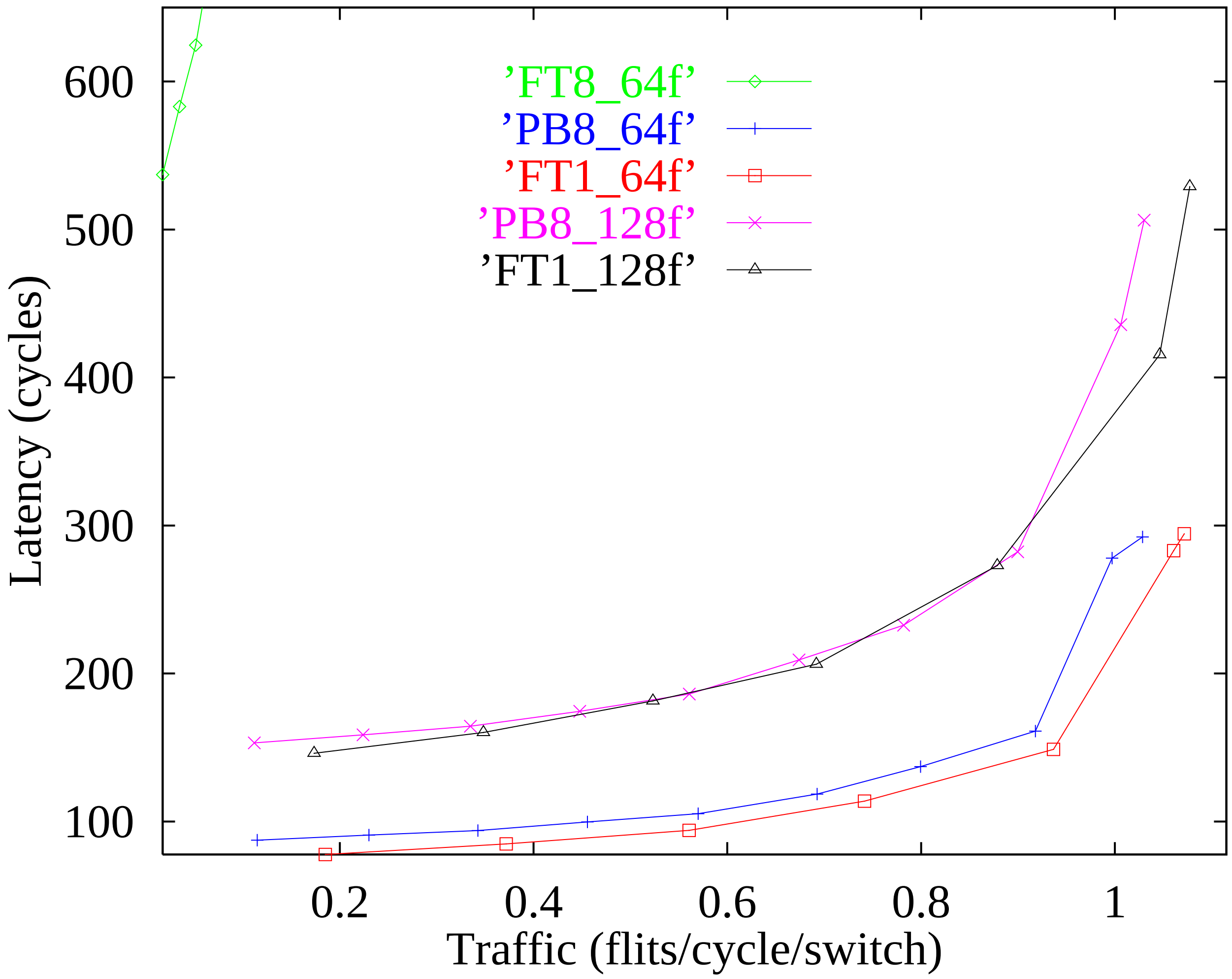
- Network size: 64 switches.
- Message length: 64 and 128 flits.
- Fly time: 4 cycles

Flow control: simulation results (V)



- Network size: 16 switches
- Message length: 16 flits
- Fly time: 8 cycles

Flow control: simulation results (VI)



- Network size: 16 switches
- Message length: 64 and 128 flits
- Fly time: 4 cycles

Deadlock handling strategy

Usually, routing algorithms avoid deadlock by restricting routing

Deadlock recovery schemes allow true fully adaptive minimal routing without devoting virtual channels to deadlock avoidance

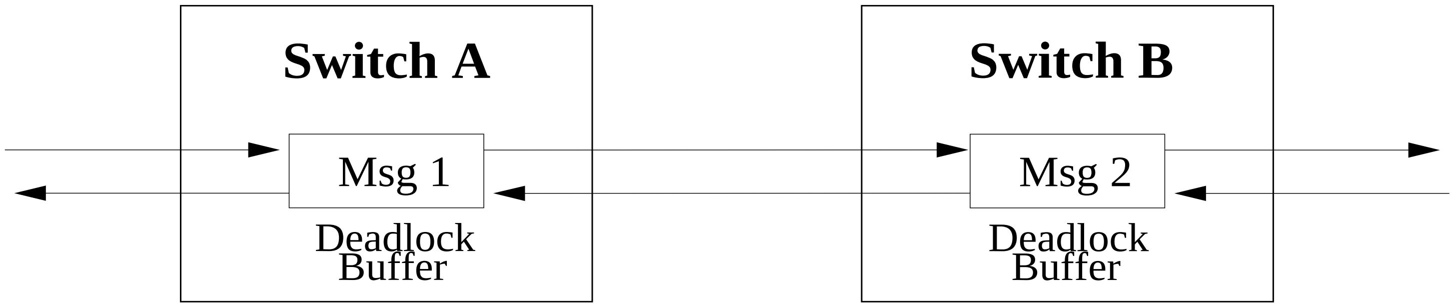
Deadlock may arise $\Rightarrow$ Mechanisms to detect and handle deadlocks are required: timeouts, improved detection mechanisms (Martinez et al, ICPP97; López et al, HPCA 98)

Disha is a deadlock recovery technique that allows fully adaptive routing and requires the minimum amount of dedicated resources to recover from deadlocks

Each switch must be equipped with one or two additional central buffers, called deadlock buffers

Applying Disha to irregular networks

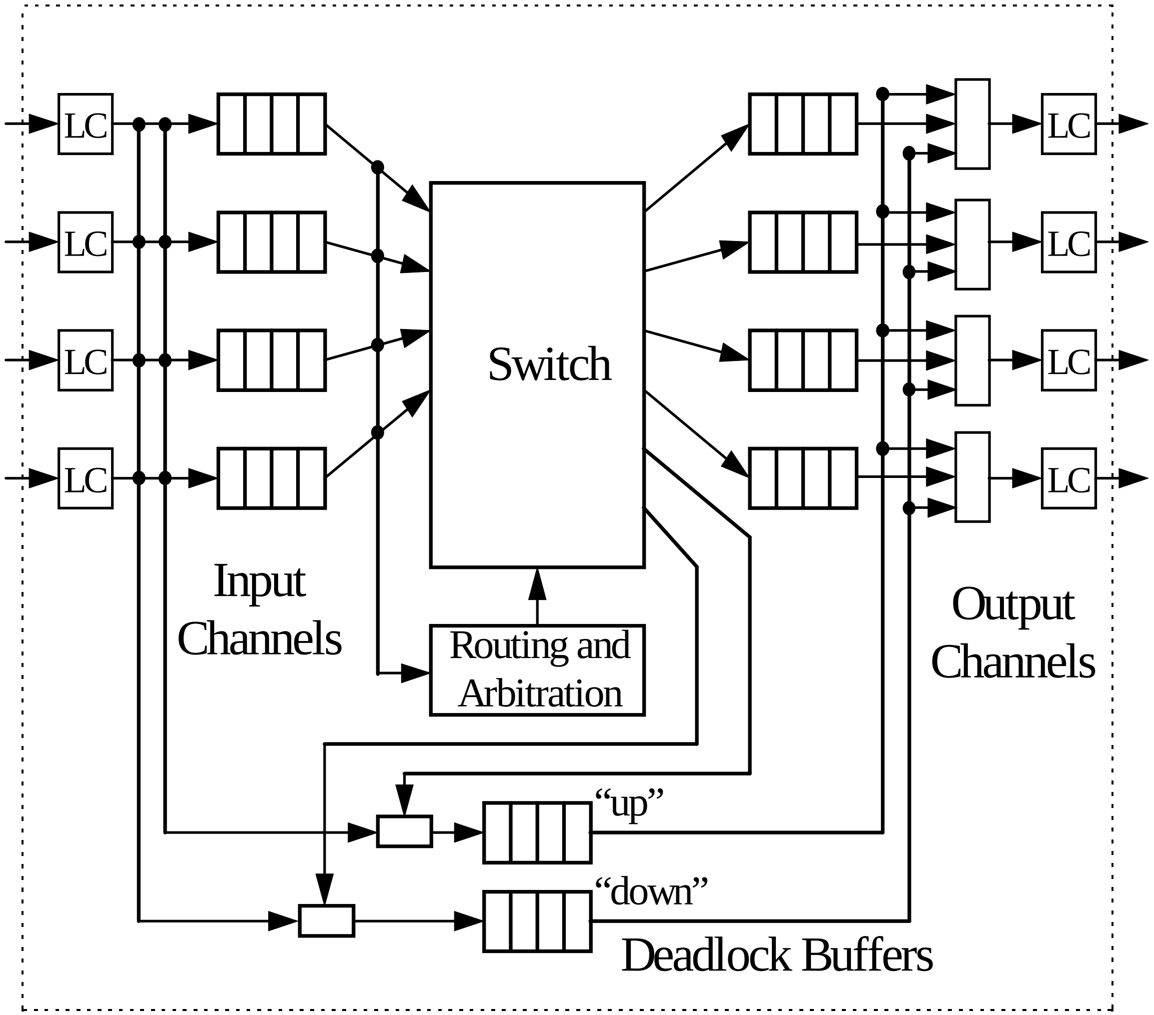
Implementing Disha in an irregular network with a single deadlock buffer per switch may be impossible. Example:



Disha can be implemented with two deadlock buffers

Up*/down* routing can be used to route messages in the deadlock buffers

Switch organization for Disha



Performance evaluation

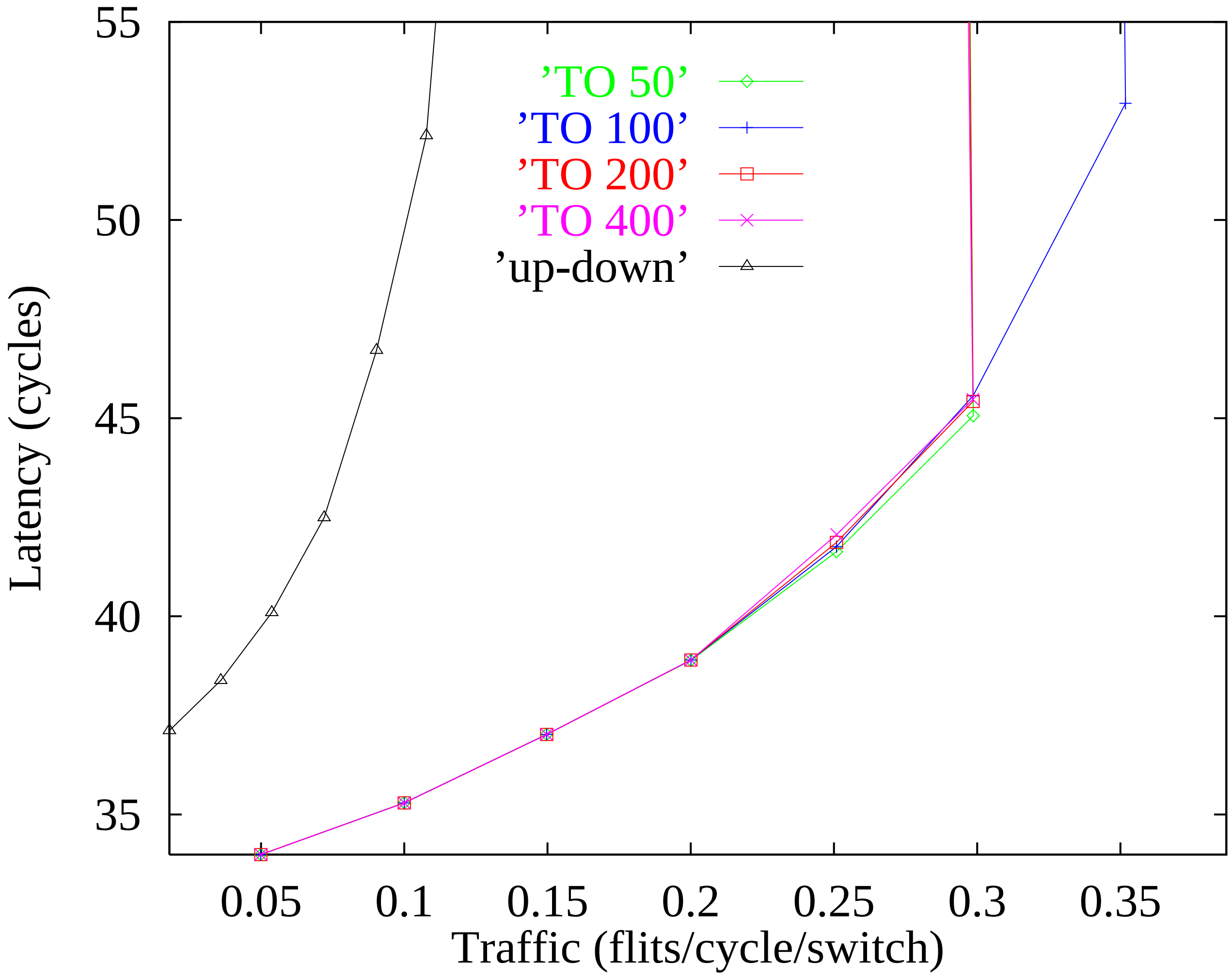
Networks with one and two virtual channels per physical channel have been evaluated

The performance of Disha has been compared with the performance of the avoidance-based fully adaptive routing (MA-2vc algorithm) and the up*/down* algorithm

Deadlock detection mechanism: crude timeouts

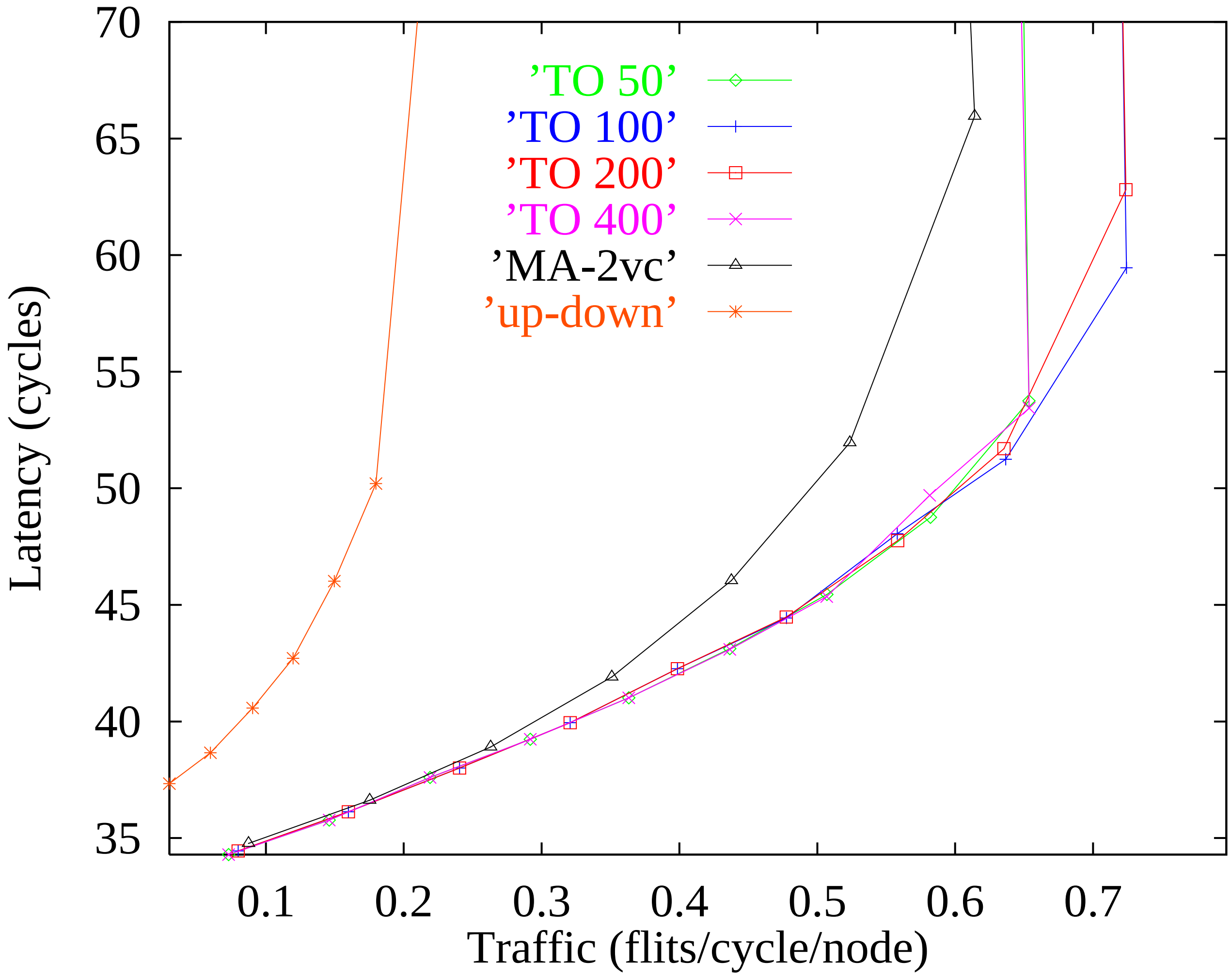
- Evaluated network sizes: 16, 32, and 64 switches
- Message lengths considered: 16, 64, 128, and 256-flit messages
- Fly time equal to 4 cycles $\Rightarrow$ Input buffer size is 27 flits

Deadlock handling: simulation results (I)



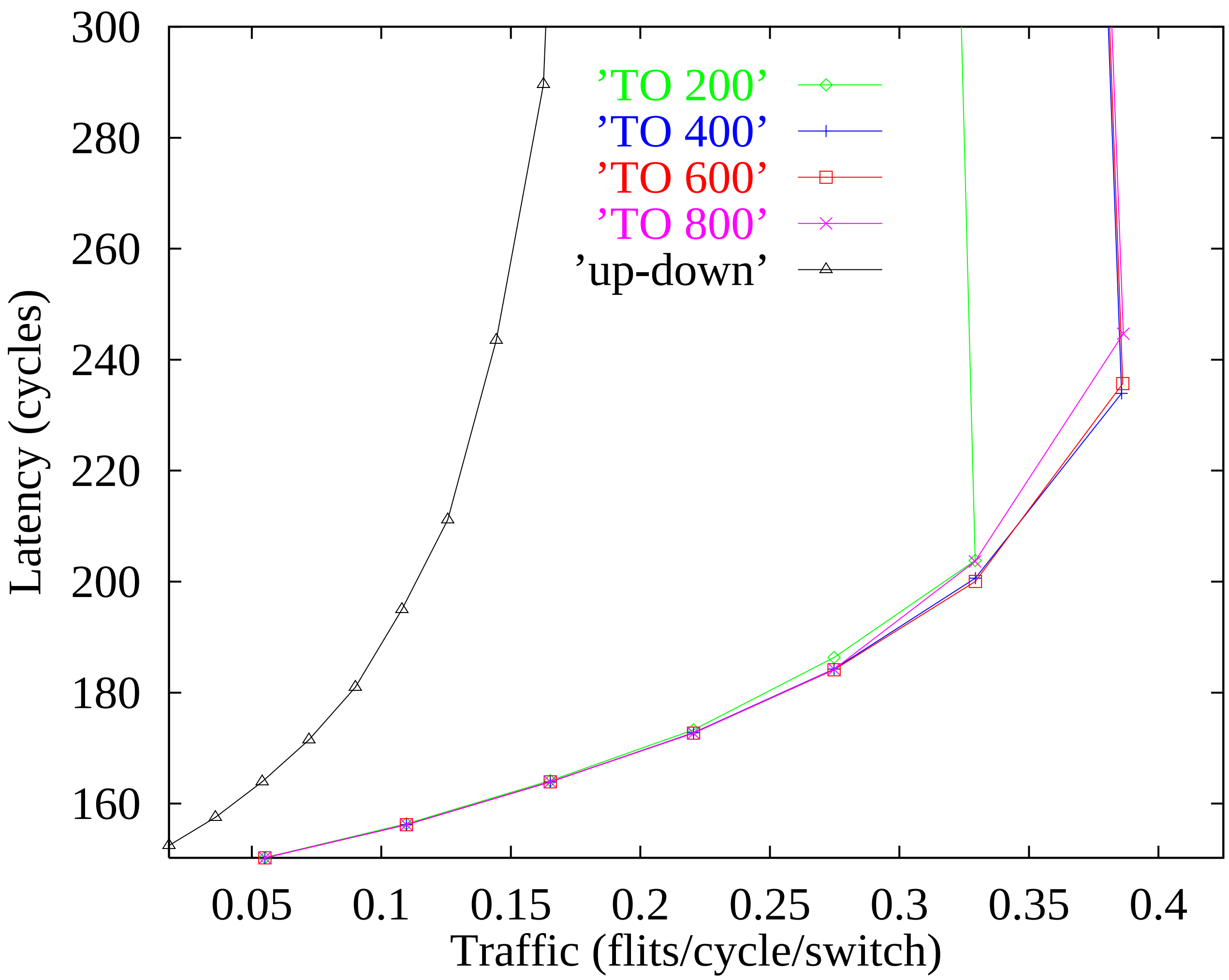
- Network size: 32 switches.
- Message length: 16 flits.
- Virtual channels: 1

Deadlock handling: simulation results (II)



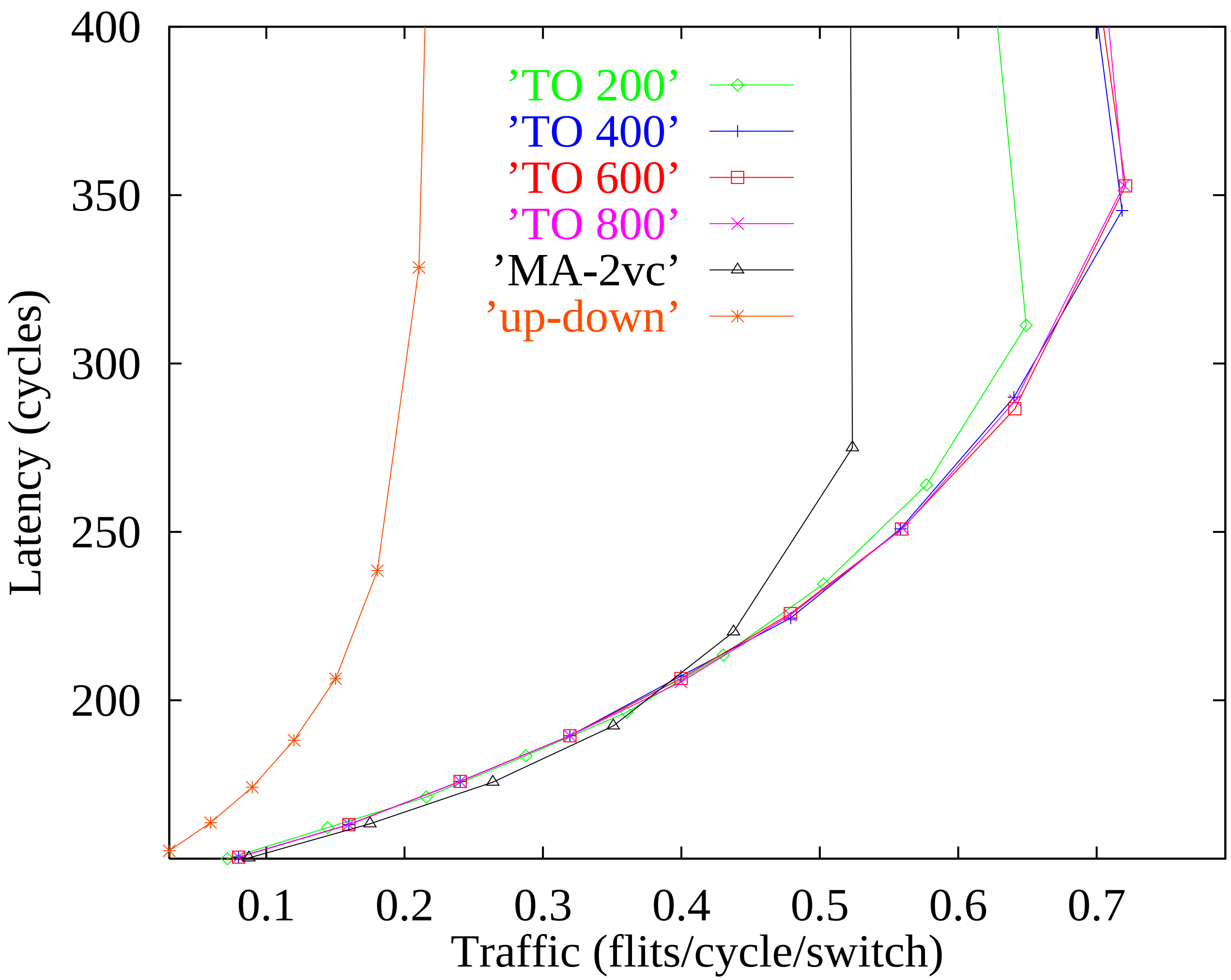
- Network size: 32 switches.
- Message length: 16 flits.
- Virtual channels: 2

Deadlock handling: simulation results (III)



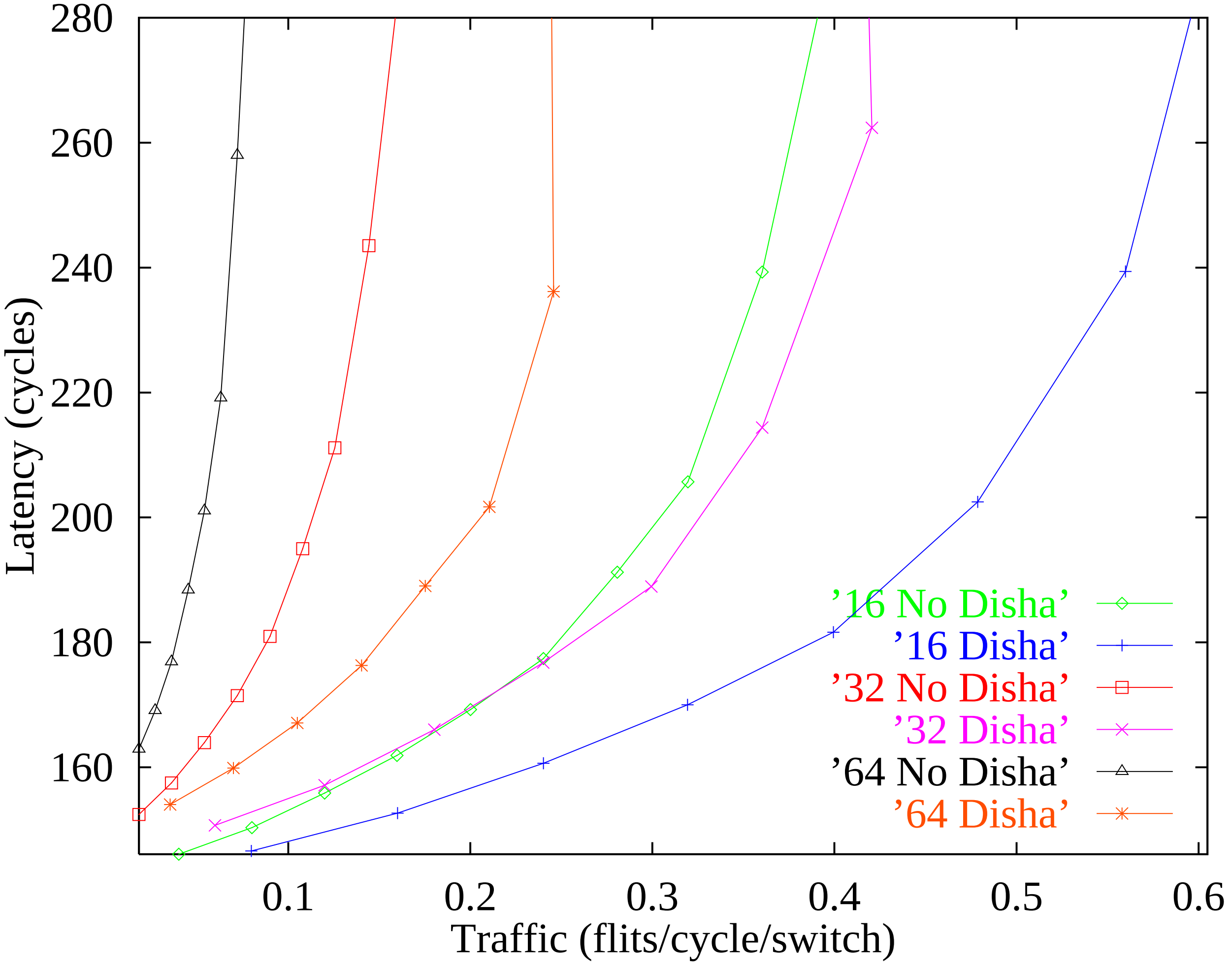
- Network size: 32 switches.
- Message length: 128 flits.
- Virtual channels: 1

Deadlock handling: simulation results (IV)



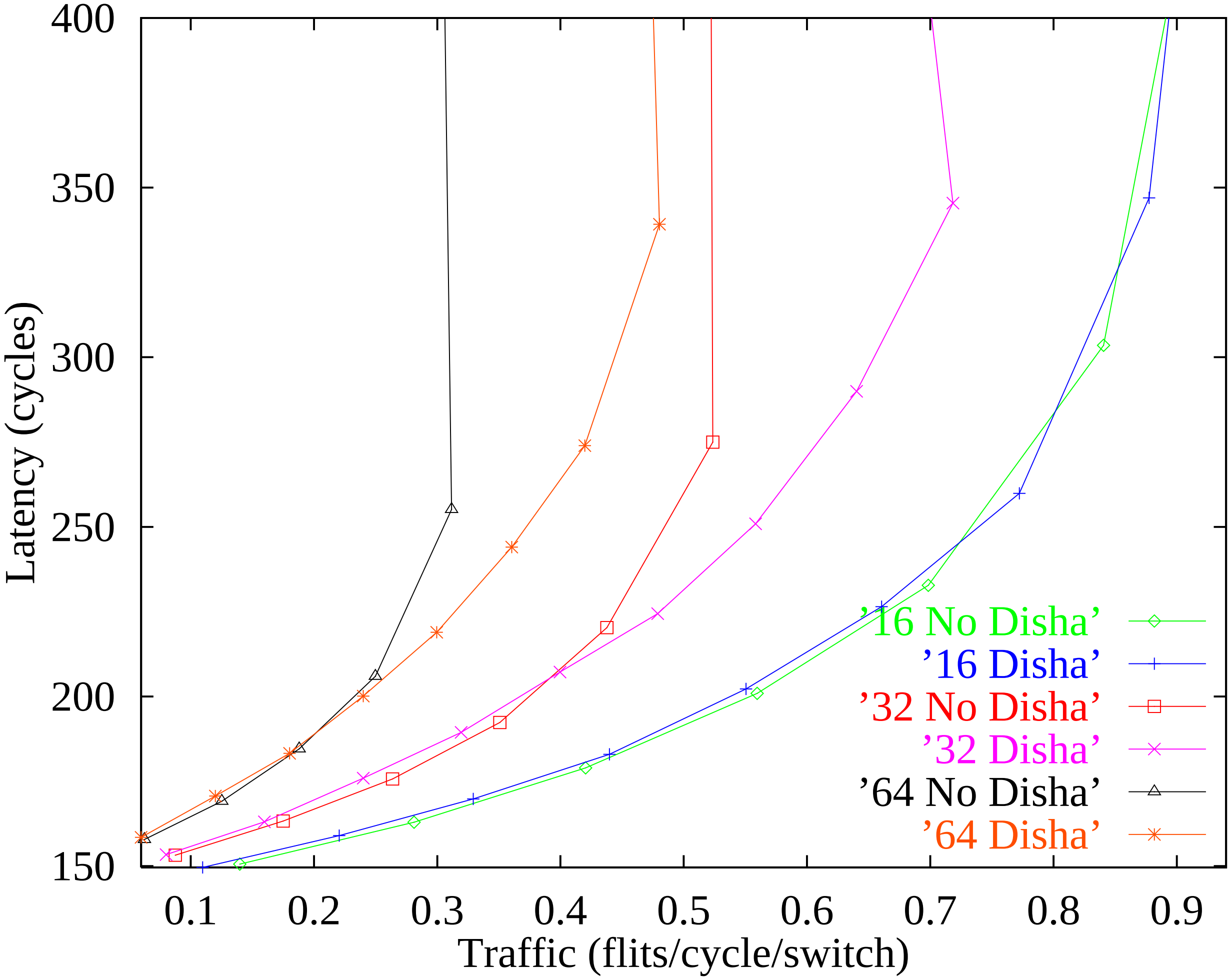
- Network size: 32 switches.
- Message length: 128 flits.
- Virtual channels: 2

Deadlock handling: simulation results (V)



- Timeout: 400 cycles.
- Message length: 128 flits.
- Virtual channels: 1

Deadlock handling: simulation results (VI)



- Timeout: 400 cycles.
- Message length: 128 flits.
- Virtual channels: 2

Switching technique

Disadvantages of wormhole switching in a NOW:

- Buffers must be very large to support channel pipelining
- Wire length is limited by buffer capacity

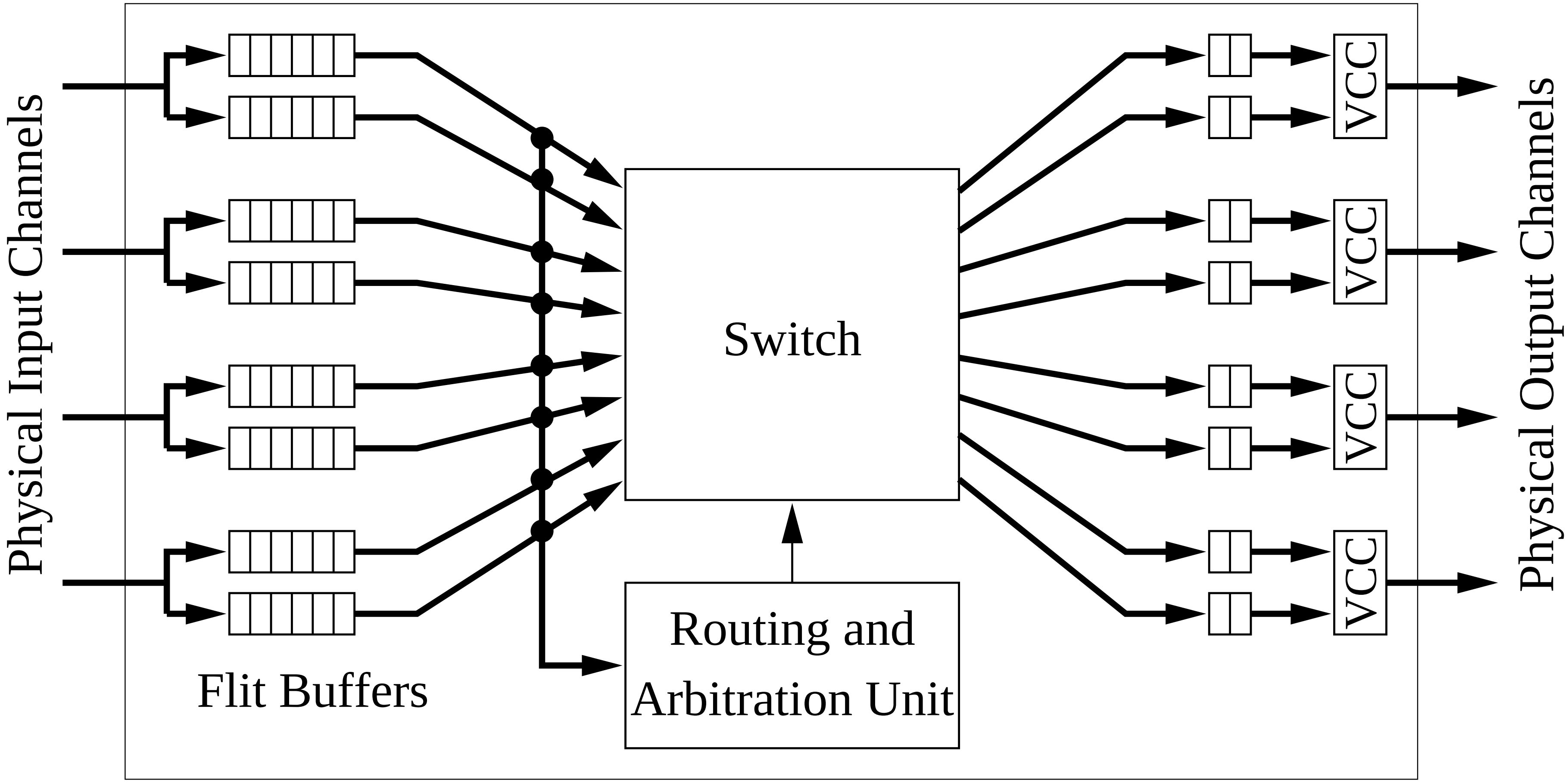
Advantages of virtual cut-through switching in a NOW:

- Simpler flow control (performed at packet level, not at flit level)
- Buffer capacity determined by packet size, independent of wire length

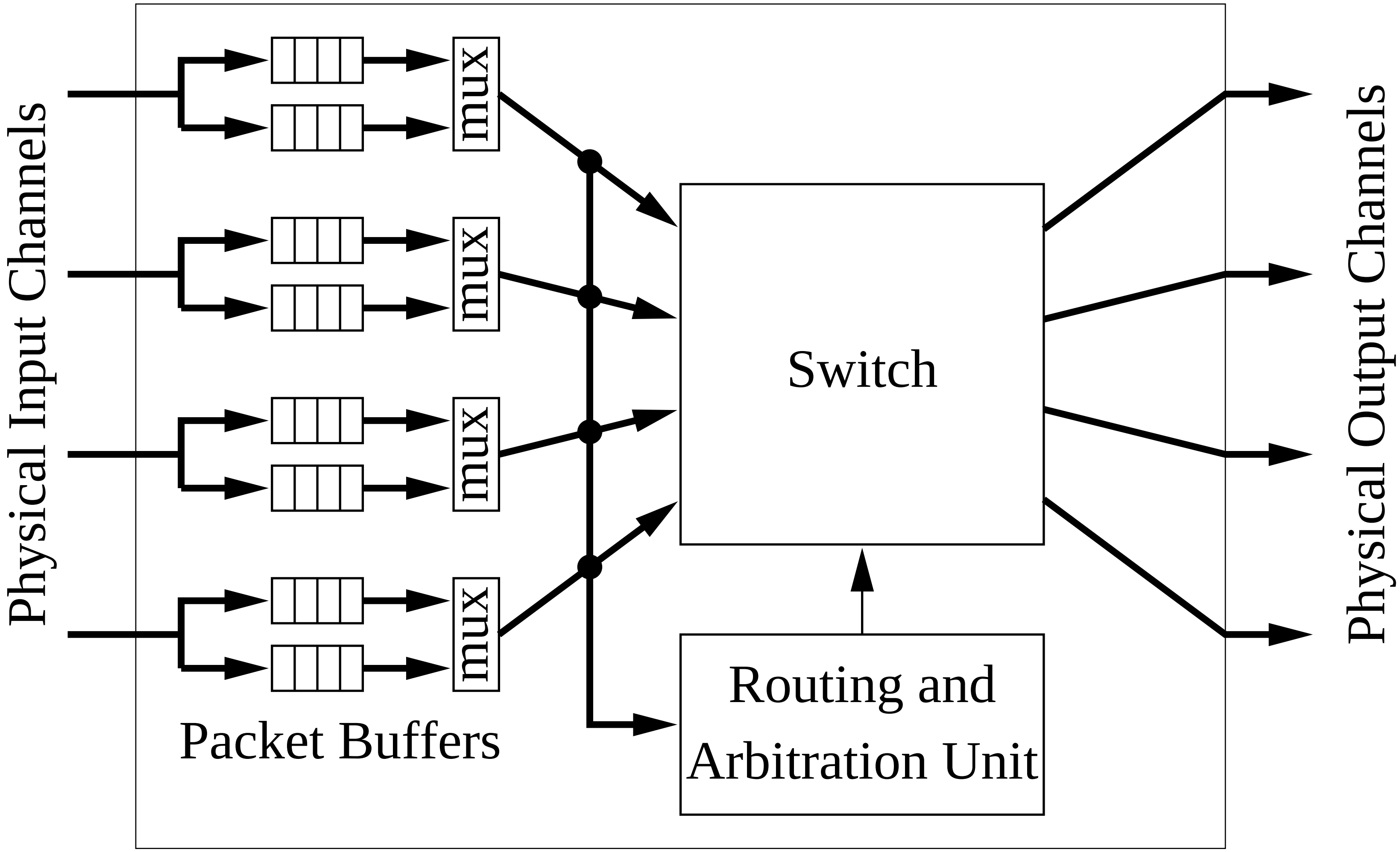
Traditional disadvantages of virtual cut-through switching:

- Large buffer size $\Rightarrow$ Similar to that of wormhole in NOWs
- Packetizing overhead $\Rightarrow$ Some messaging layers (FM) for NOWs increase performance by splitting messages into fixed-size packet

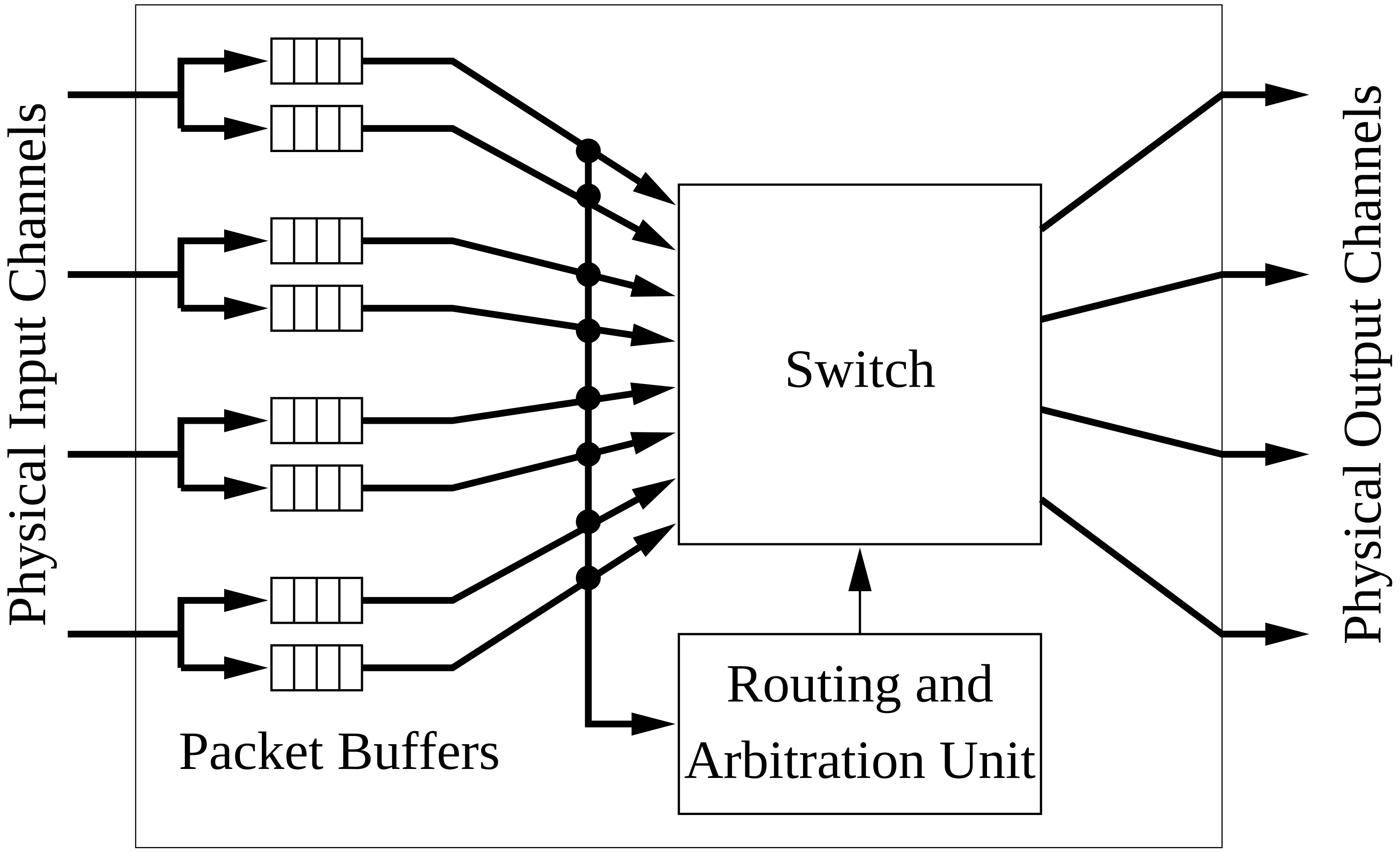
Organization of a wormhole router with virtual channels



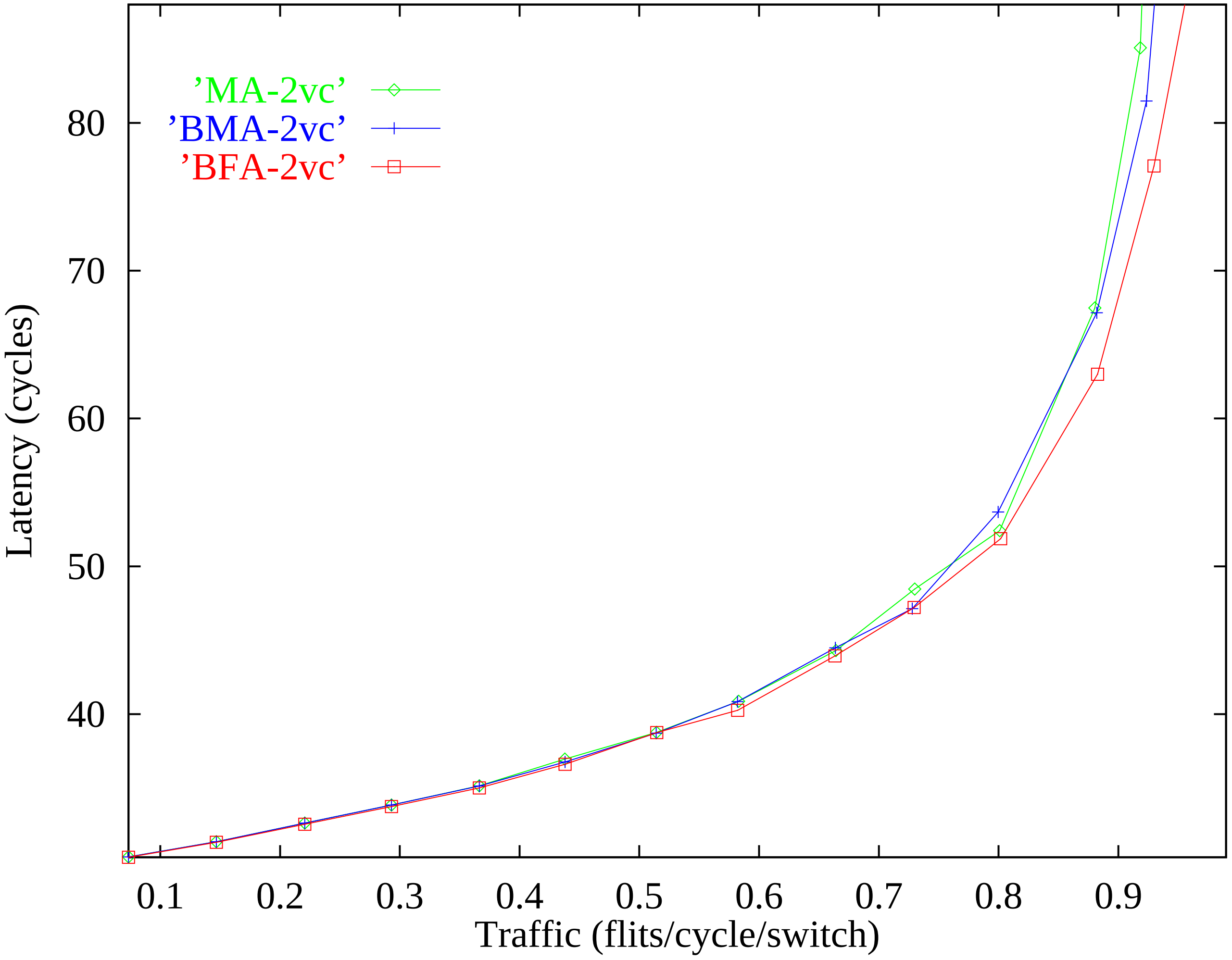
Virtual cut-through router with multiplexed crossbar



Virtual cut-through router with partially multiplexed crossbar

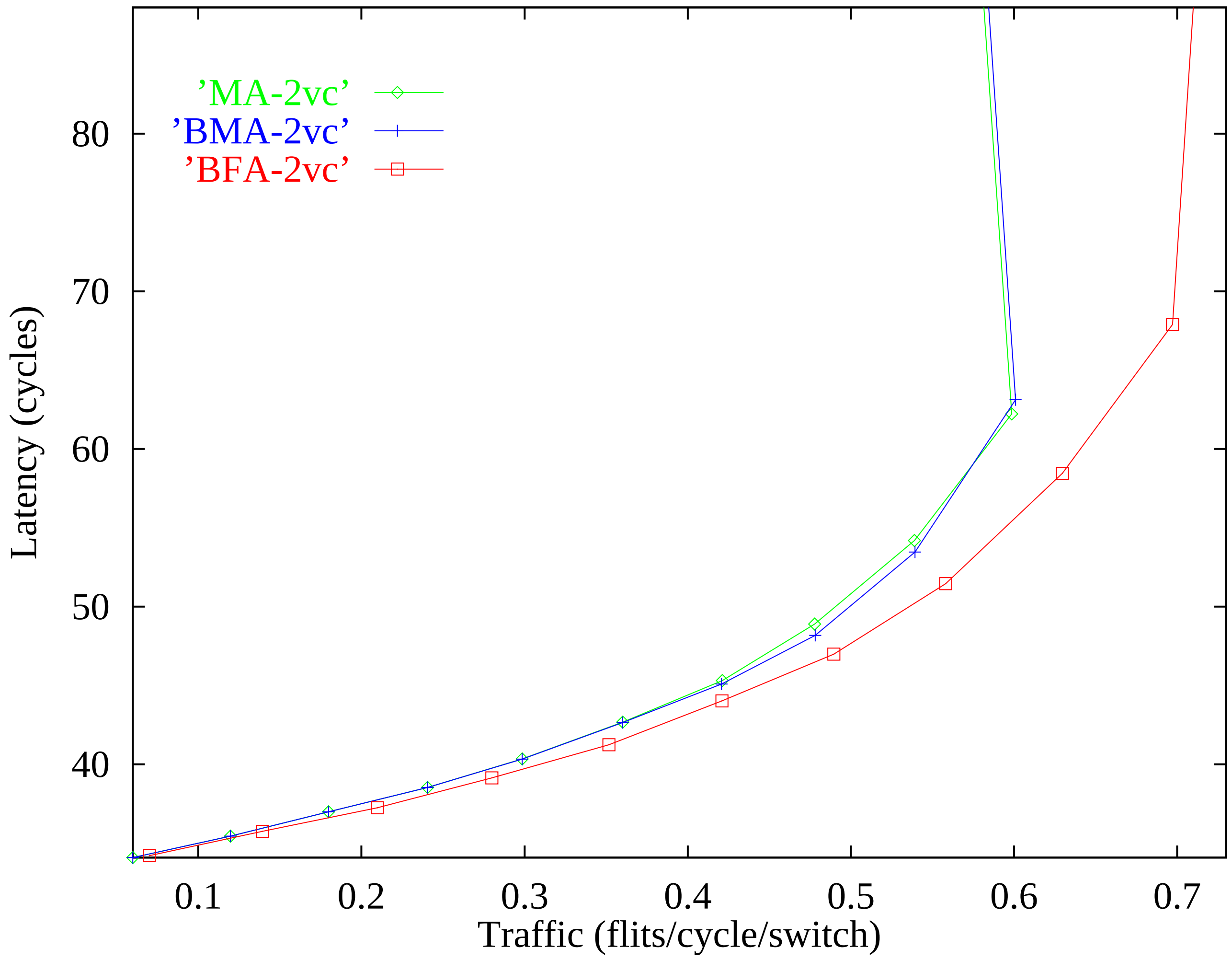


Effect of switching technique (I)



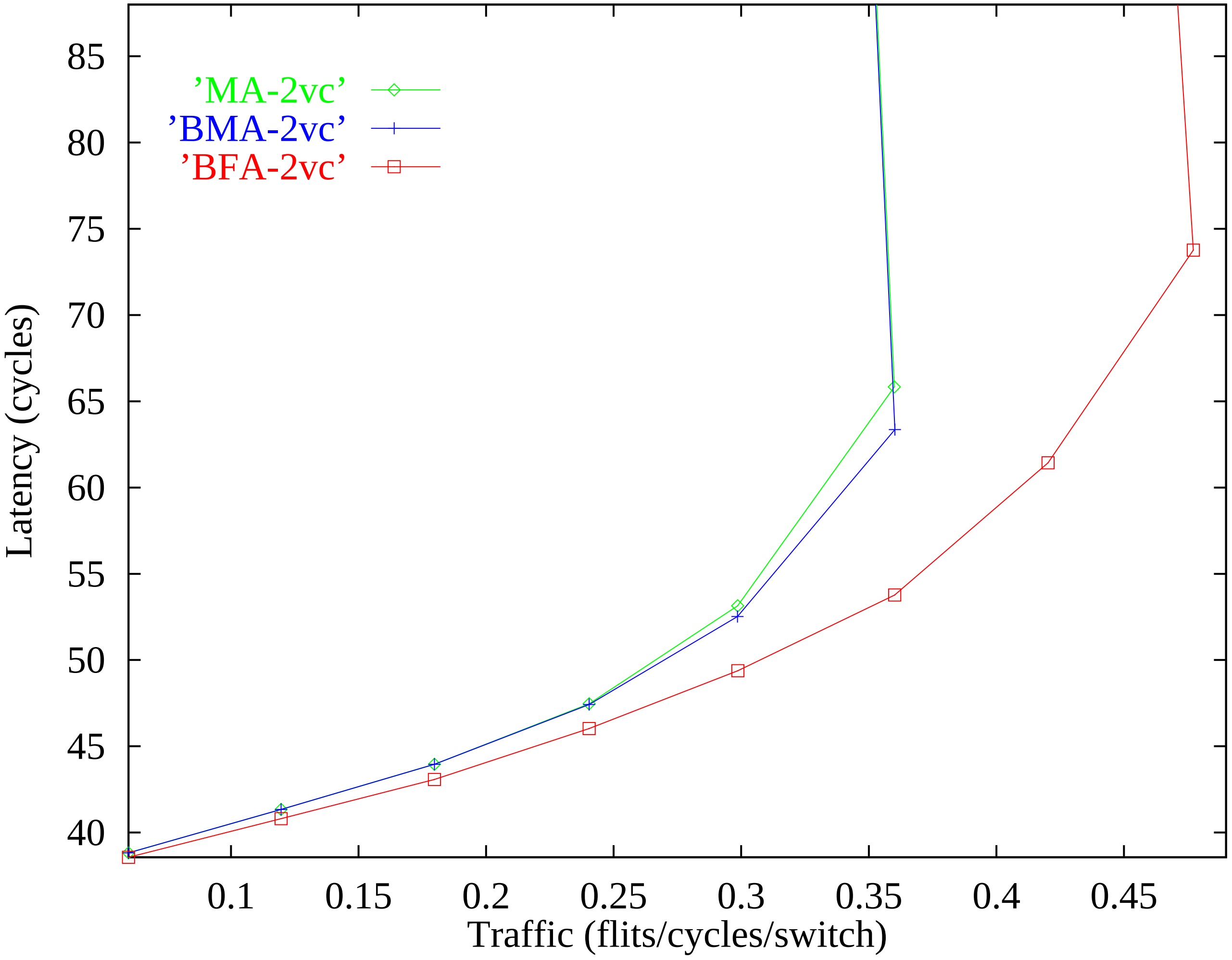
- Network size: 16 switches
- Message length: 16 flits
- Fly time: 4 cycles

Effect of switching technique (II)



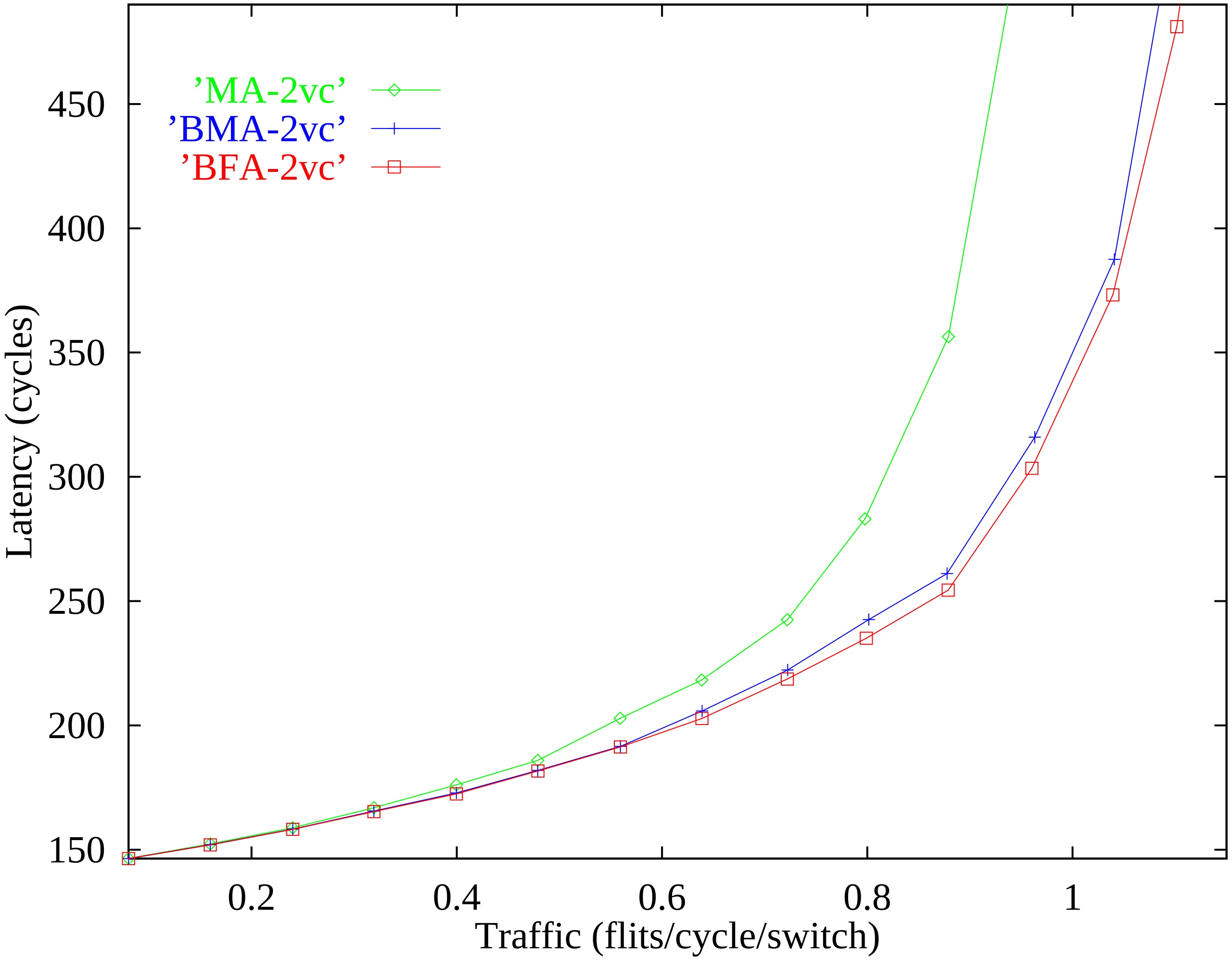
- Network size: 32 switches
- Message length: 16 flits
- Fly time: 4 cycles

Effect of switching technique (III)



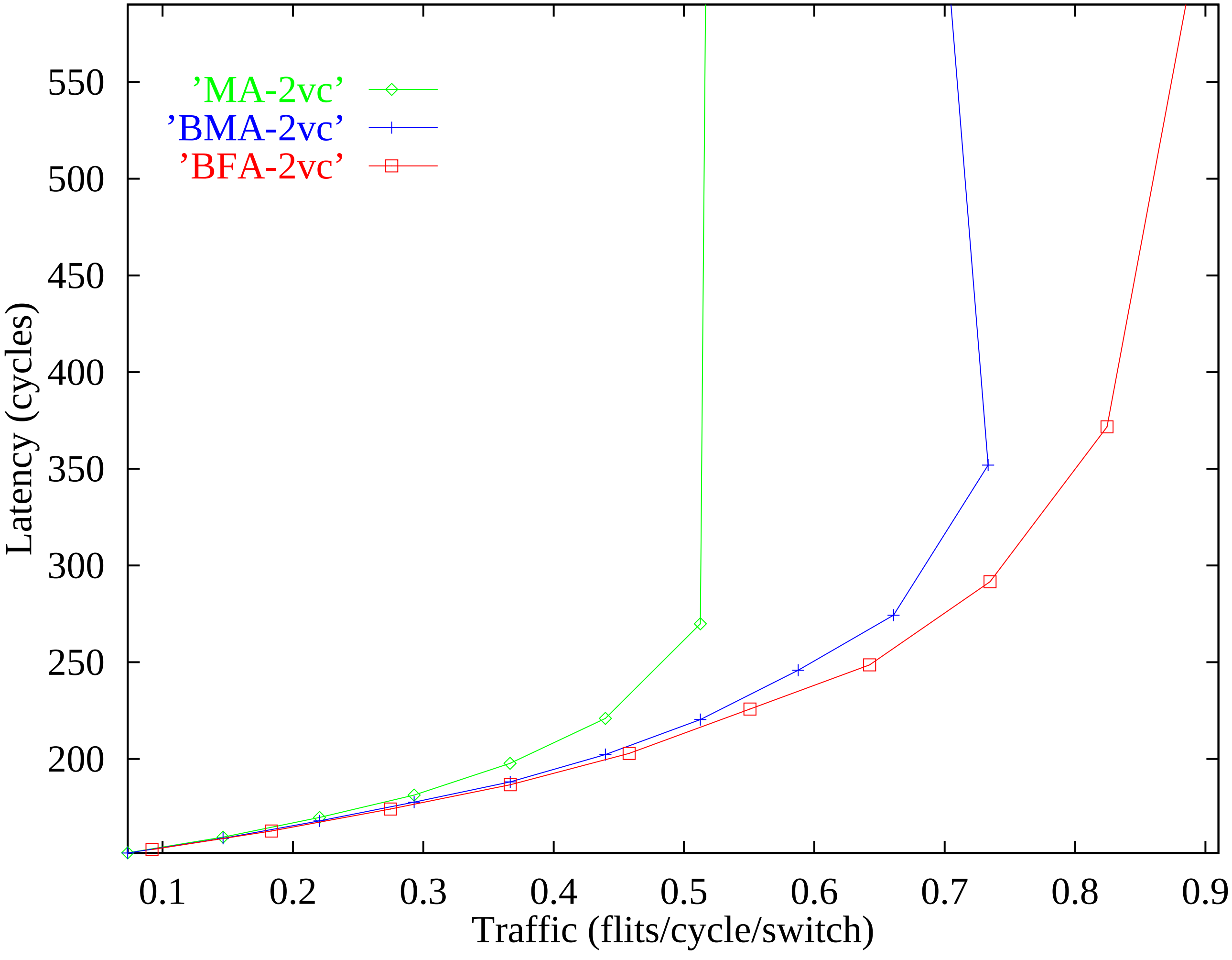
- Network size: 64 switches
- Message length: 16 flits
- Fly time: 4 cycles

Effect of switching technique (IV)



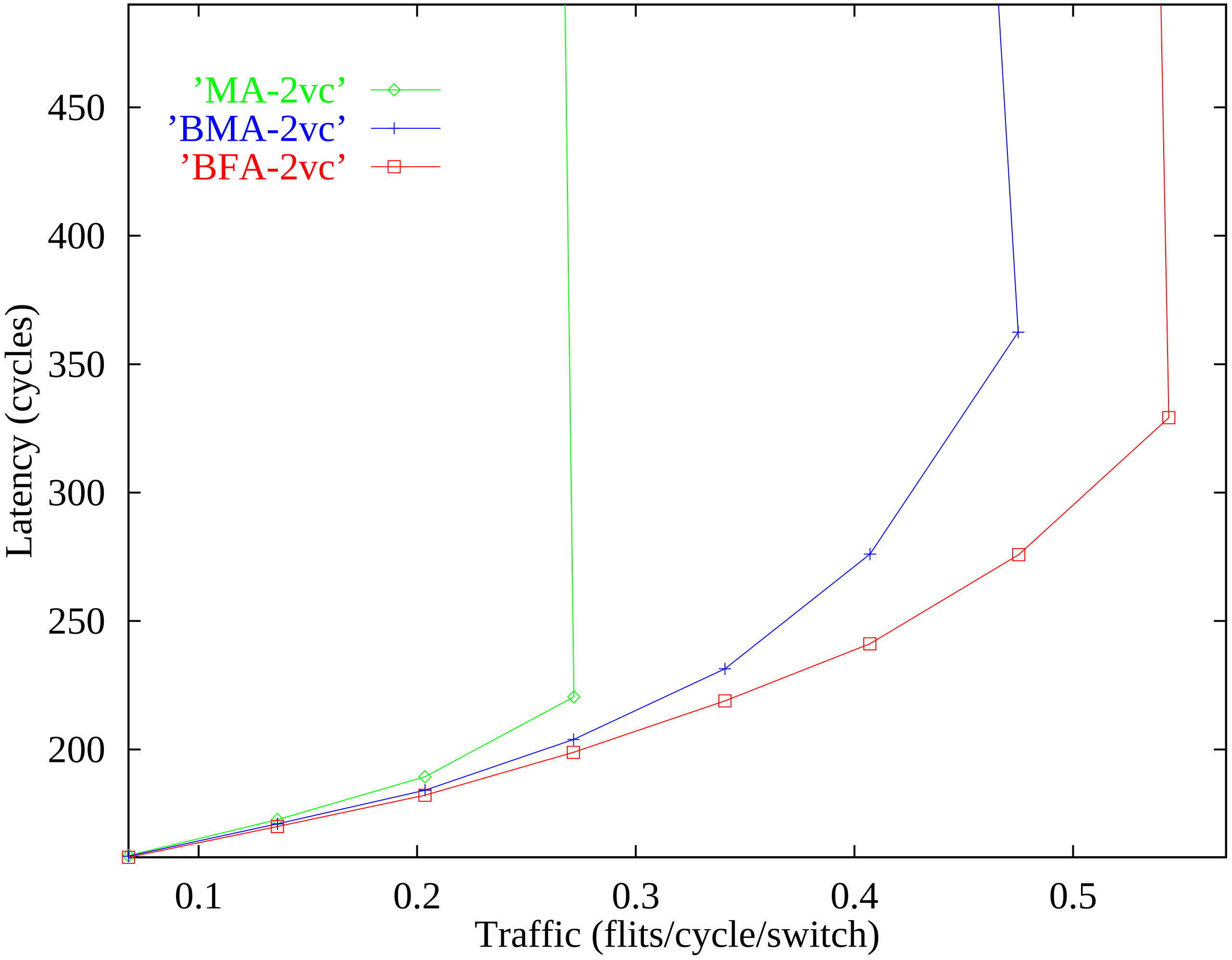
- Network size: 16 switches
- Message length: 128 flits
- Fly time: 4 cycles

Effect of switching technique (V)



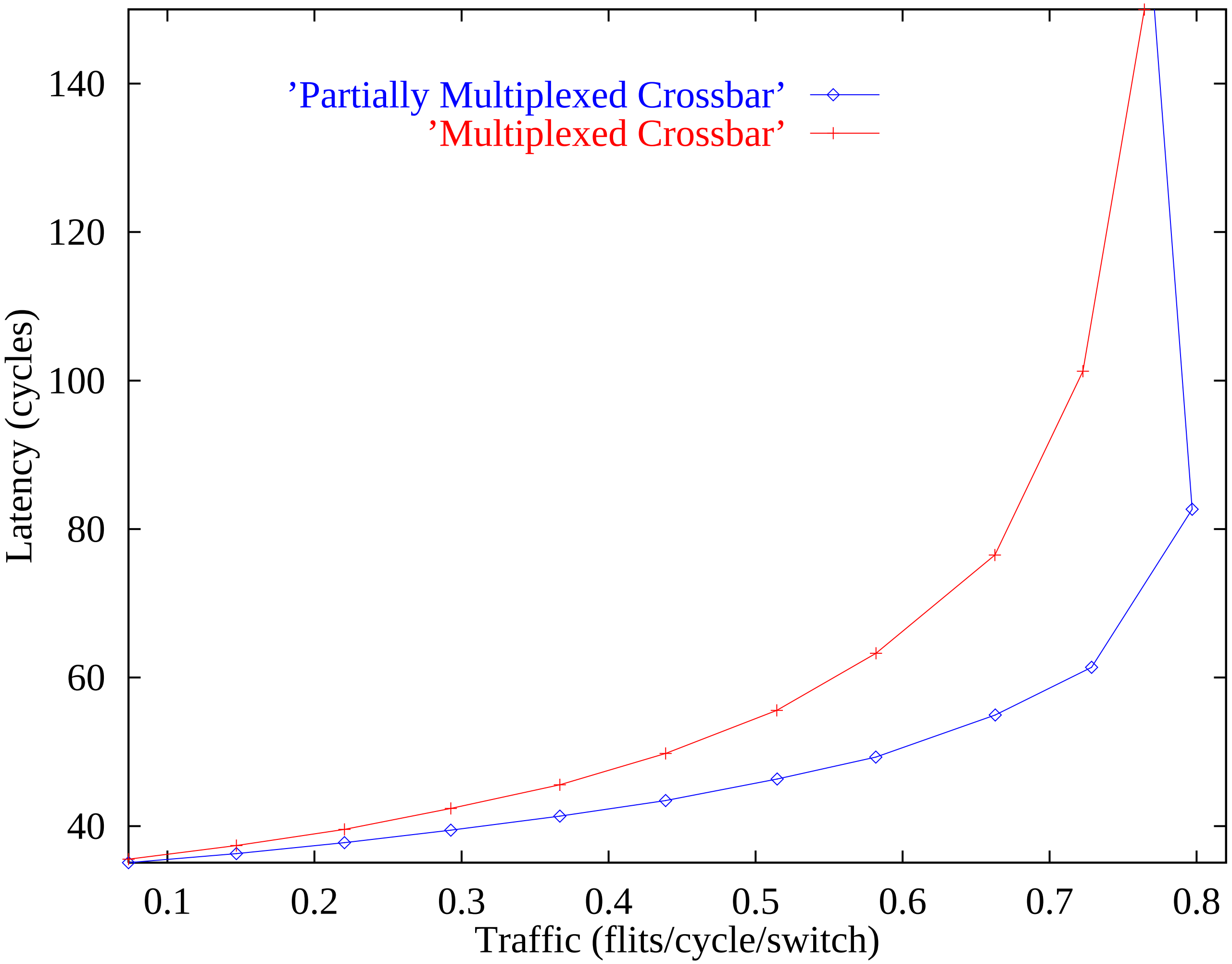
- Network size: 32 switches
- Message length: 128 flits
- Fly time: 4 cycles

Effect of switching technique (VI)



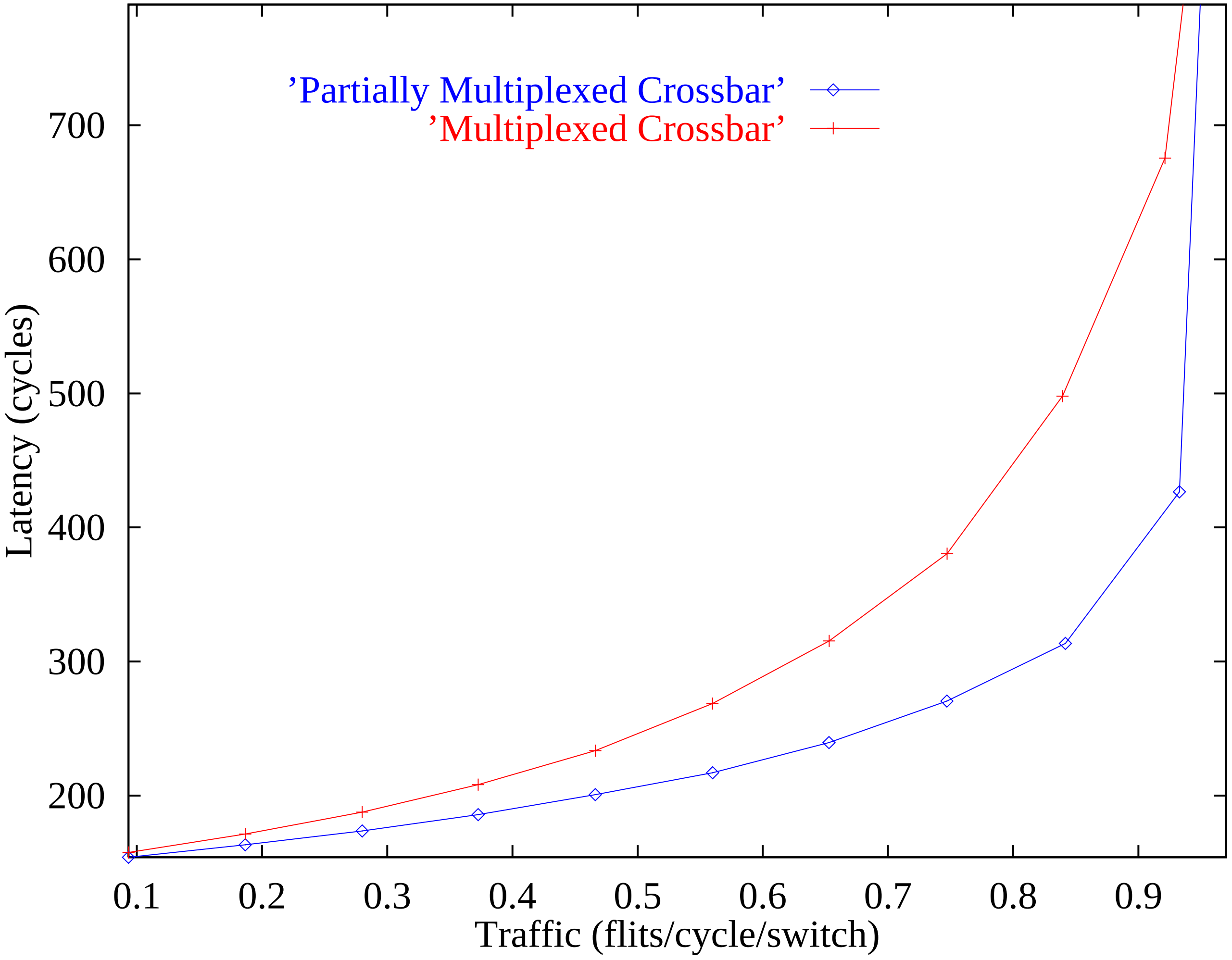
- Network size: 64 switches
- Message length: 128 flits
- Fly time: 4 cycles

Effect of crossbar size (I)



- Network size: 32 switches
- Message length: 16 flits
- Fly time: 4 cycles

Effect of crossbar size (II)



- Network size: 32 switches
- Message length: 128 flits
- Fly time: 4 cycles

Remarks

Irregular topologies provide wiring flexibility, scalability, and incremental expansion at the cost of higher latency and lower throughput

Fully adaptive routing considerably outperforms deterministic and partially adaptive routing on irregular topologies, approaching the performance of regular topologies except for large networks

Pipelined links drastically improve performance over non-pipelined links. Efficient flow control is crucial to minimize control overhead

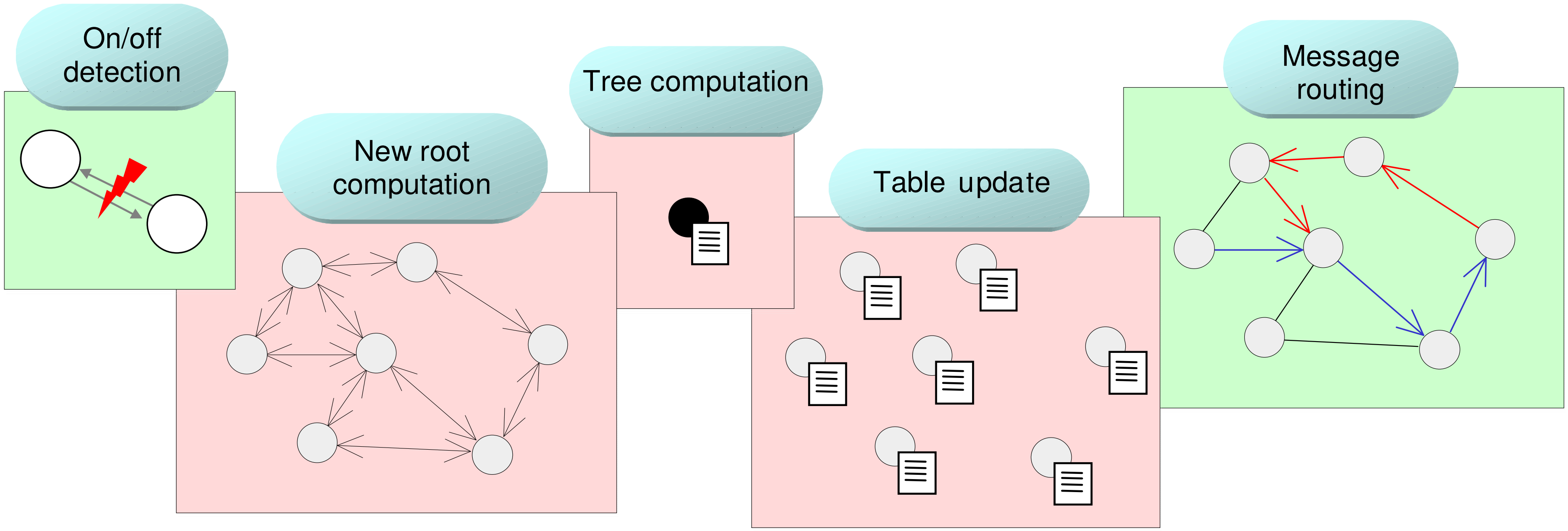
Deadlock recovery significantly improves performance over deadlock avoidance, especially when virtual channels are not implemented. However, it requires efficient deadlock detection techniques

Virtual cut-through outperforms wormhole switching in a NOW environment, also featuring simpler flow control, simpler router architecture, and buffer size independent of wire length

Improving availability/reliability

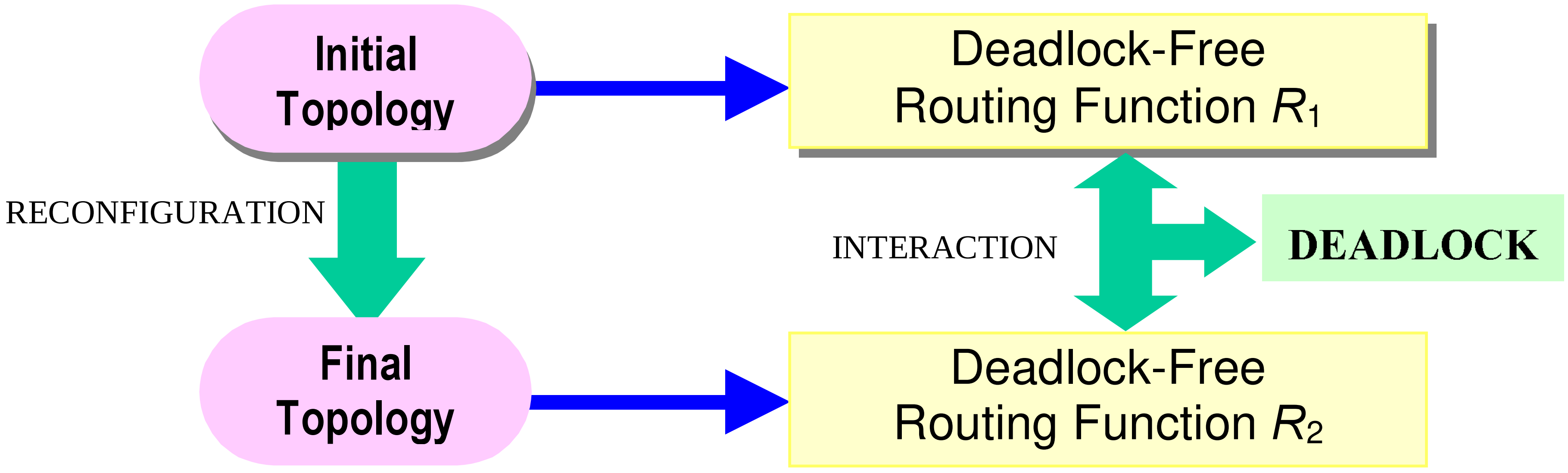
- Design issues
  - Topology change/failure detection
  - Change/fault resolution
    - Fault-tolerant routing
    - Static reconfiguration
    - Dynamic reconfiguration

Network reconfiguration (irregular topologies)



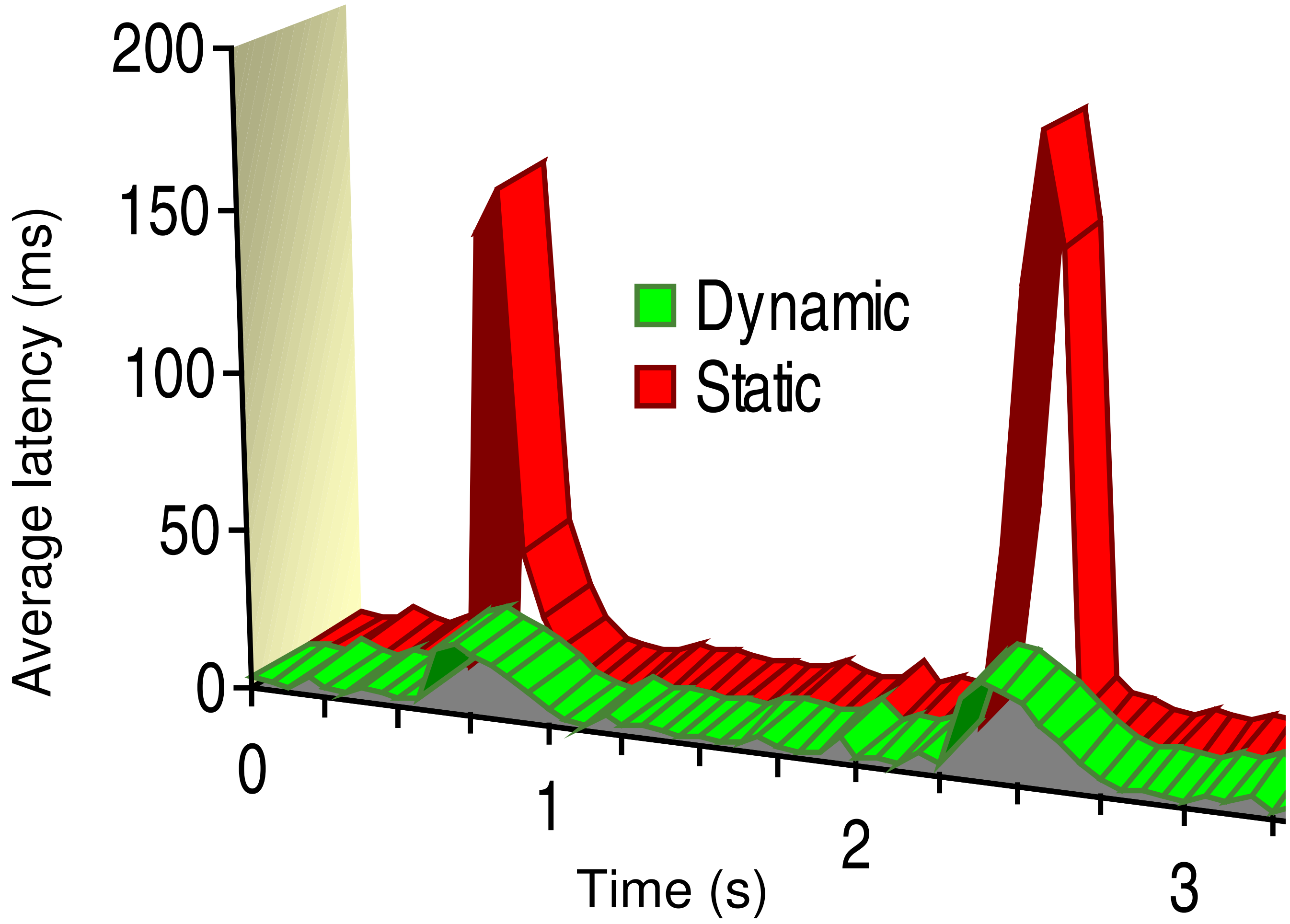
Switches are independent from each other and routing tables cannot be synchronously updated

Problems associated with network reconfiguration



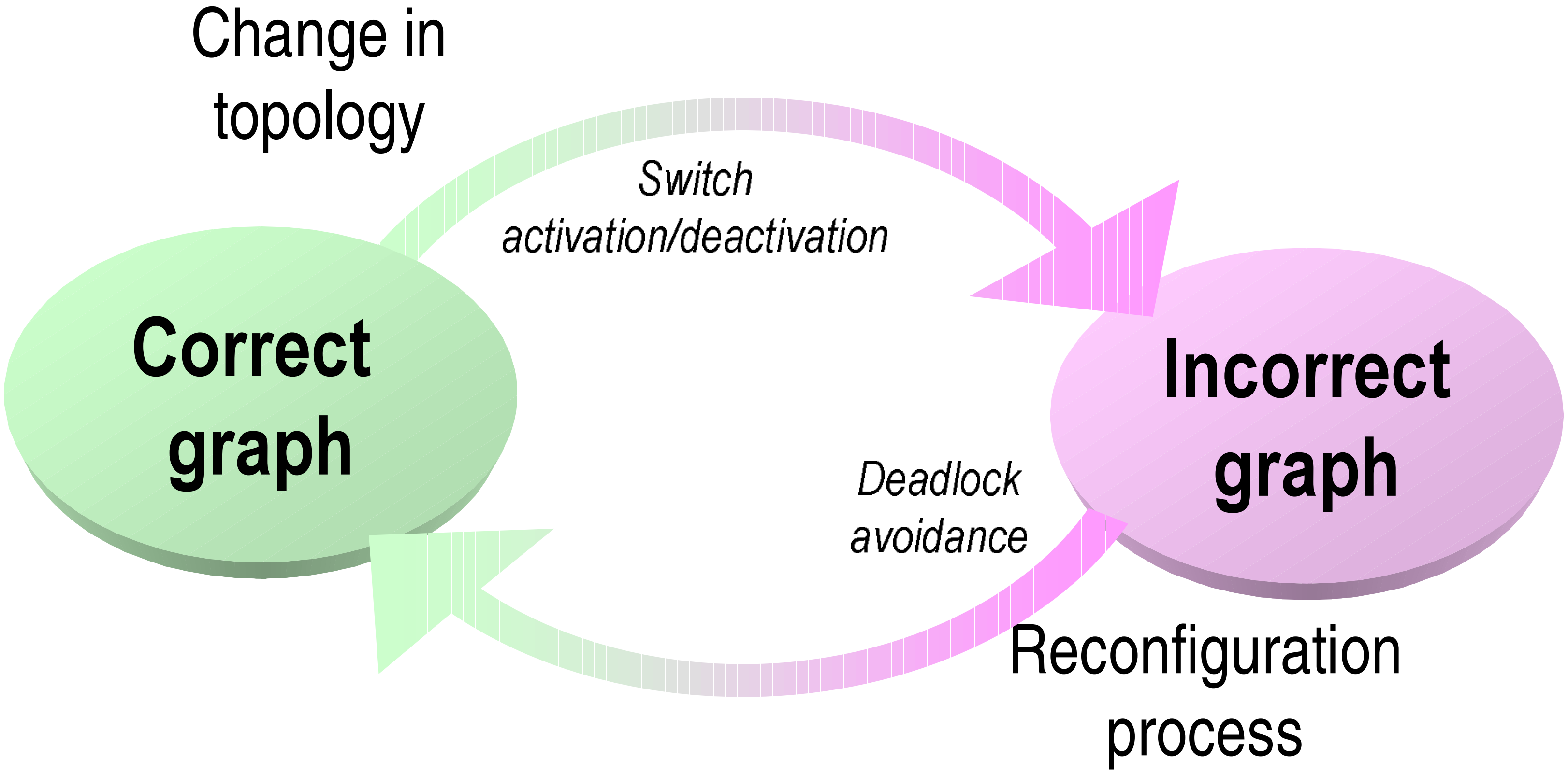
Traditional solution: stop network traffic in the region being reconfigured (static reconfiguration)

Cost of static reconfiguration



Goal

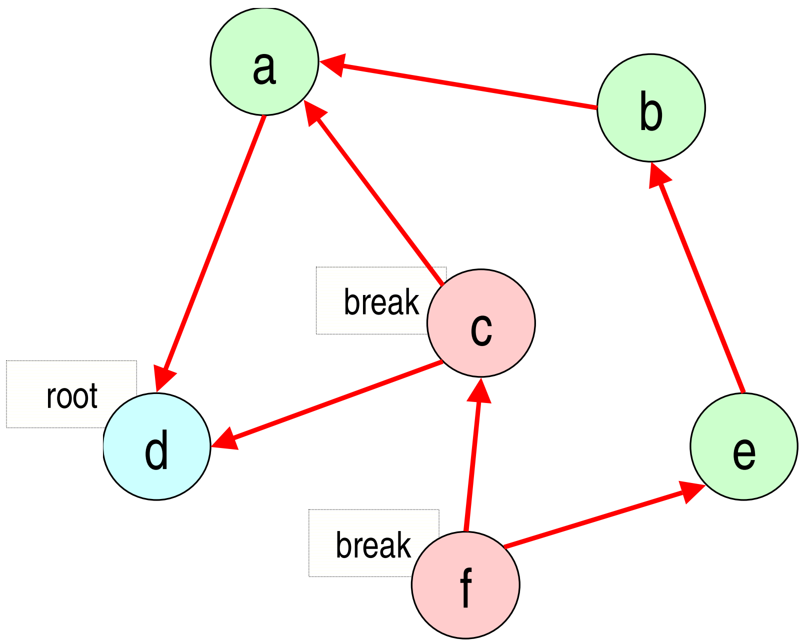
Provide support for dynamic reconfiguration while avoiding deadlock



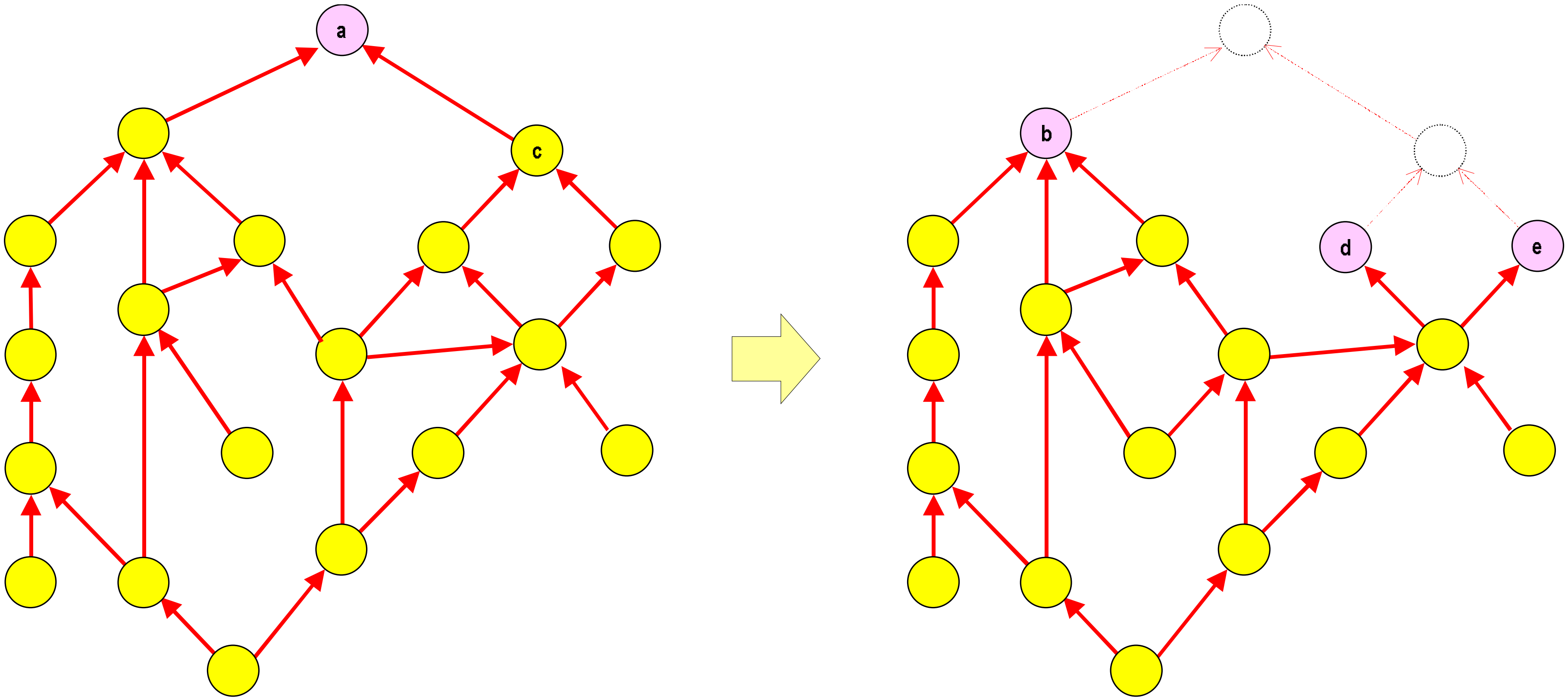
Difficulties arising in dynamic reconfiguration

There is no deadlock-free sequence of routing table updates
$\Rightarrow$ Perform a sequence of partial routing table updates

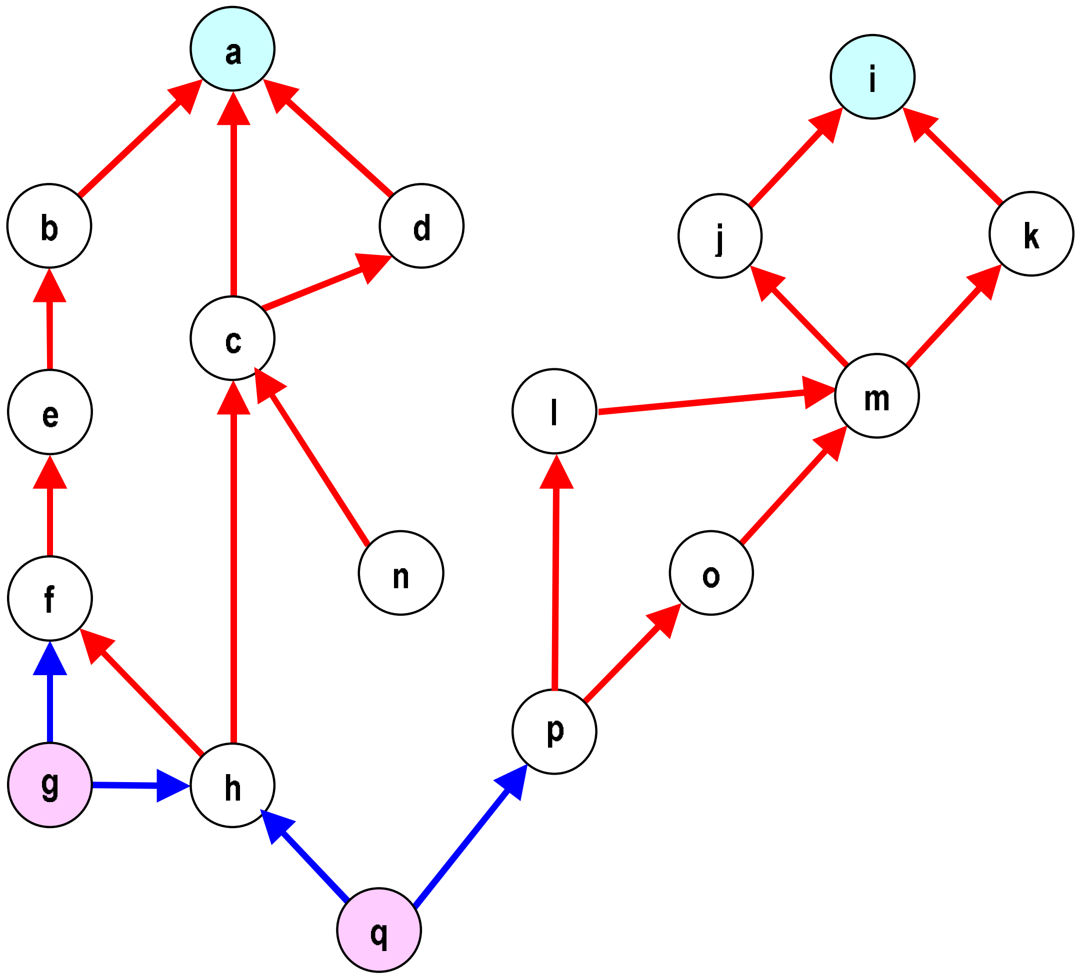
Basic concepts for up*/down* routing



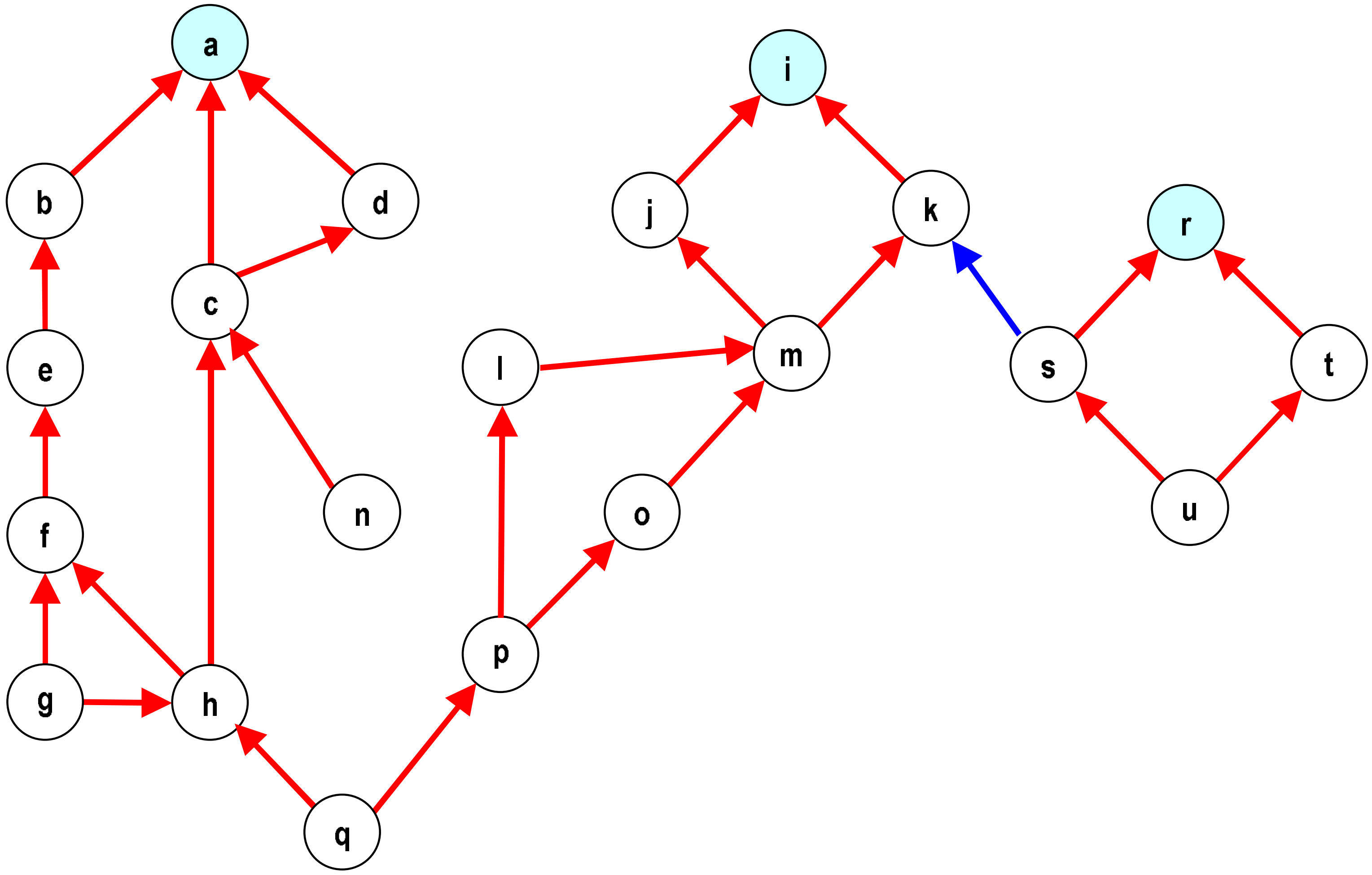
Removing nodes from the network



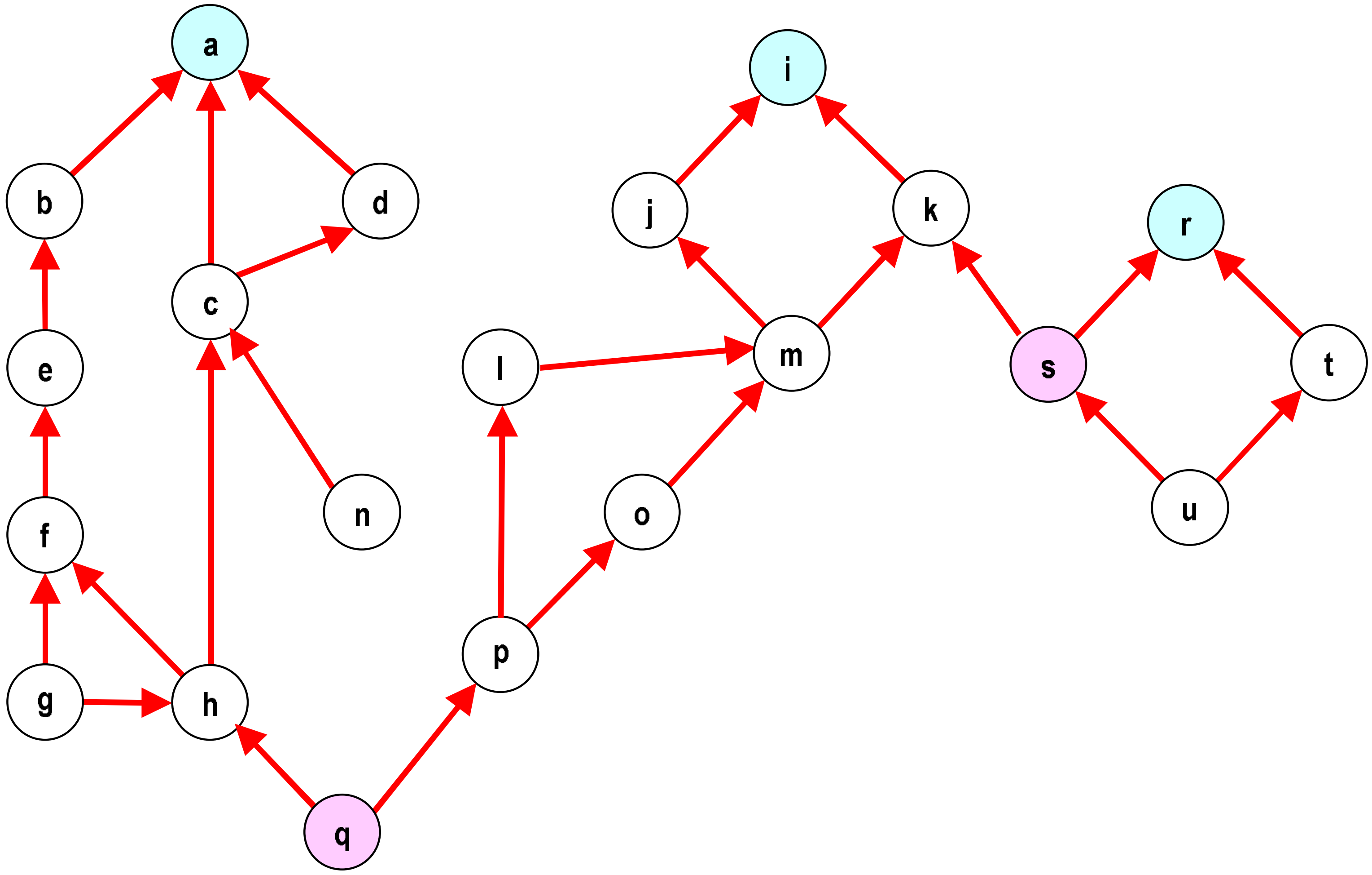
Adding nodes to the network



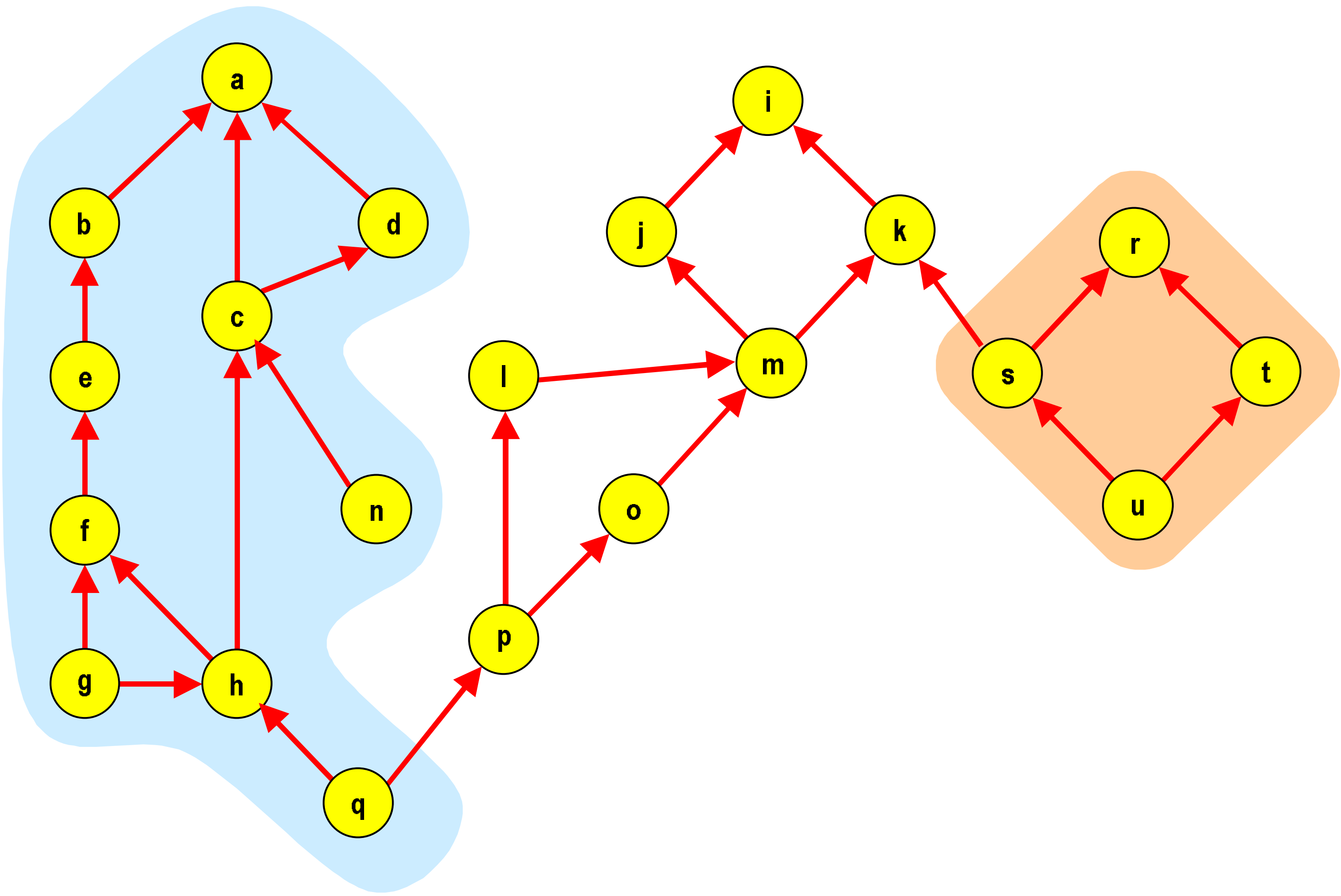
Adding links to the network



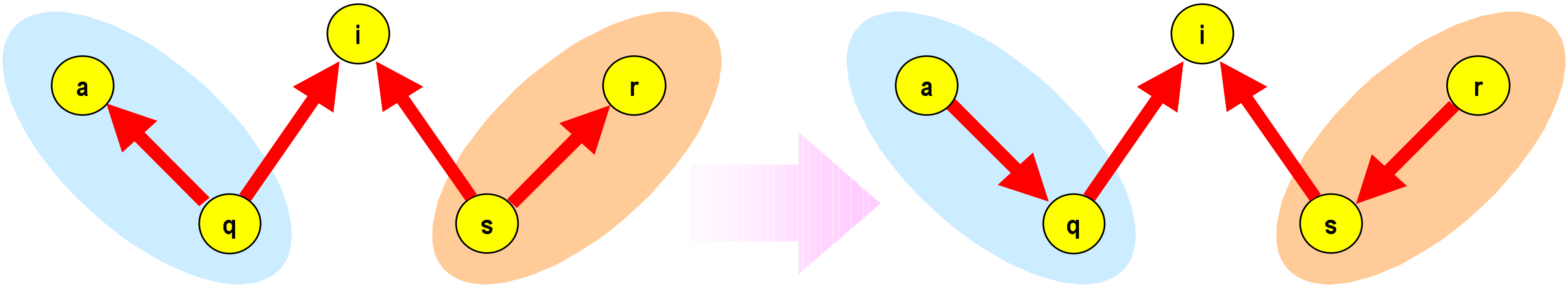
The resulting acyclic network graph



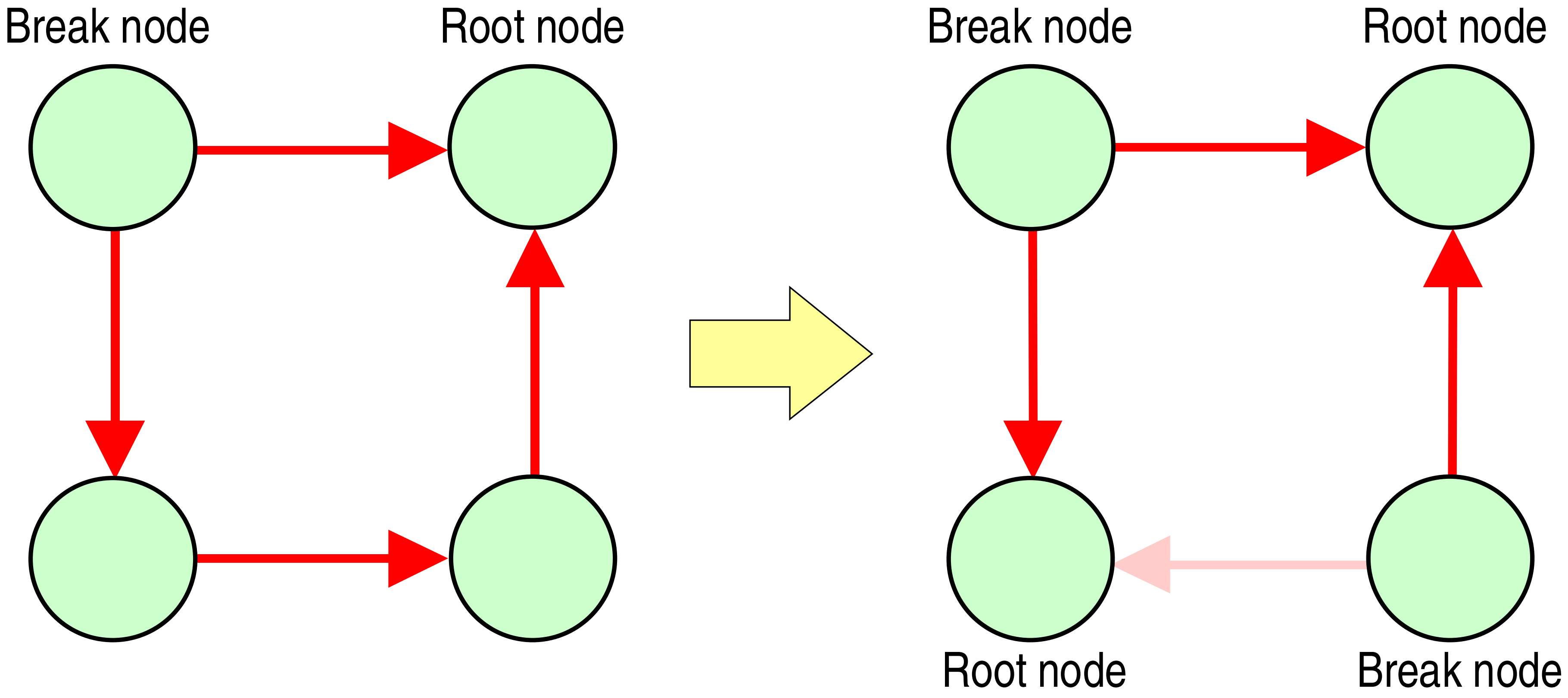
Identifying regions in the network



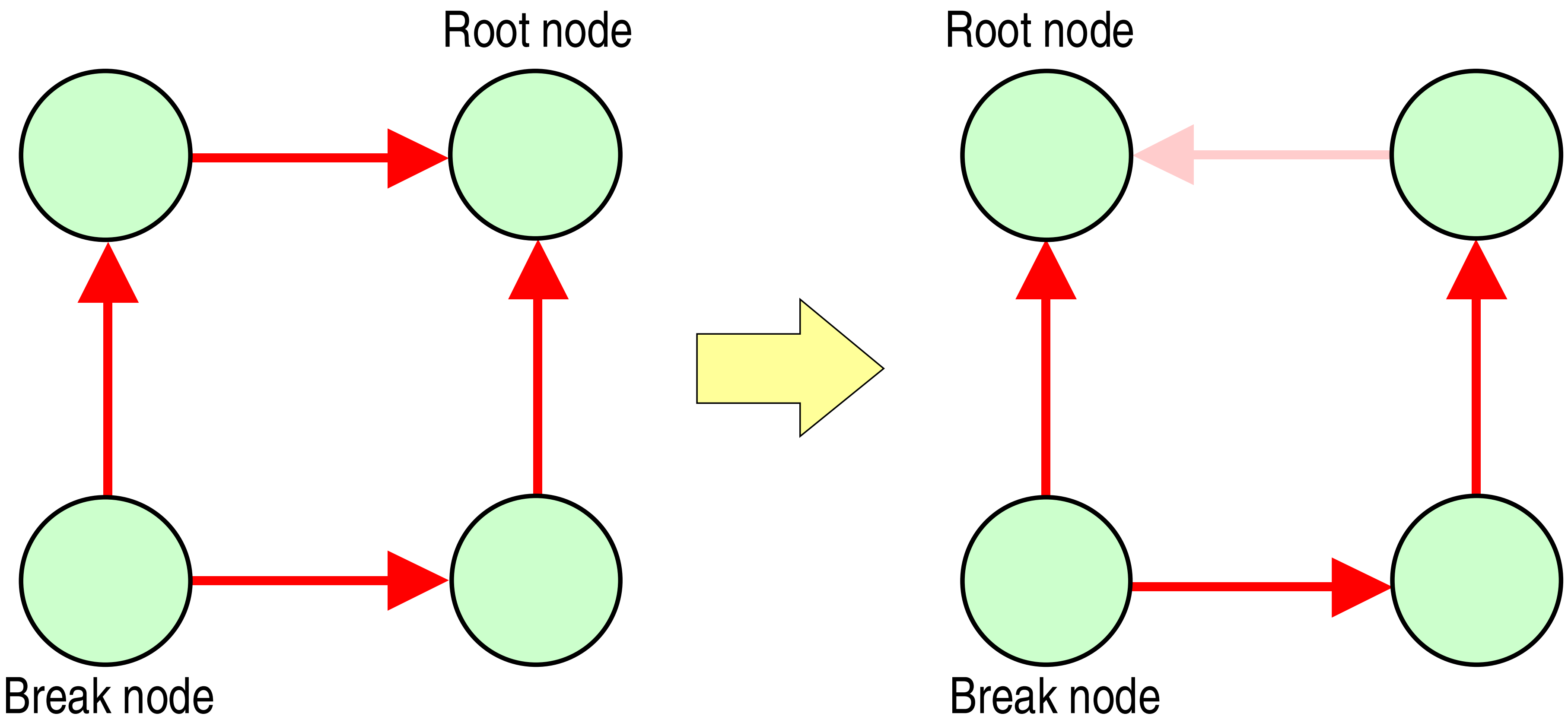
Correcting the inter-region graph



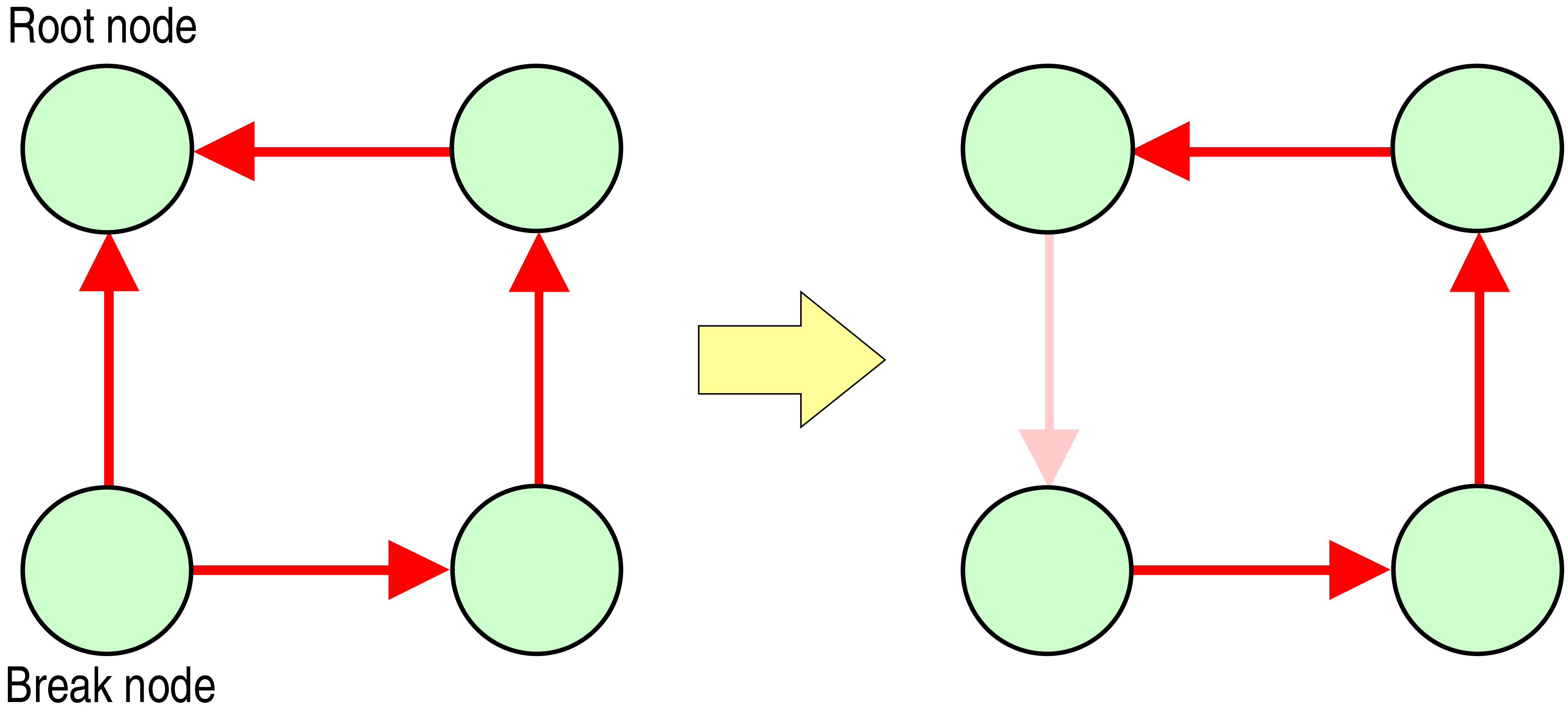
Updating link direction assignments



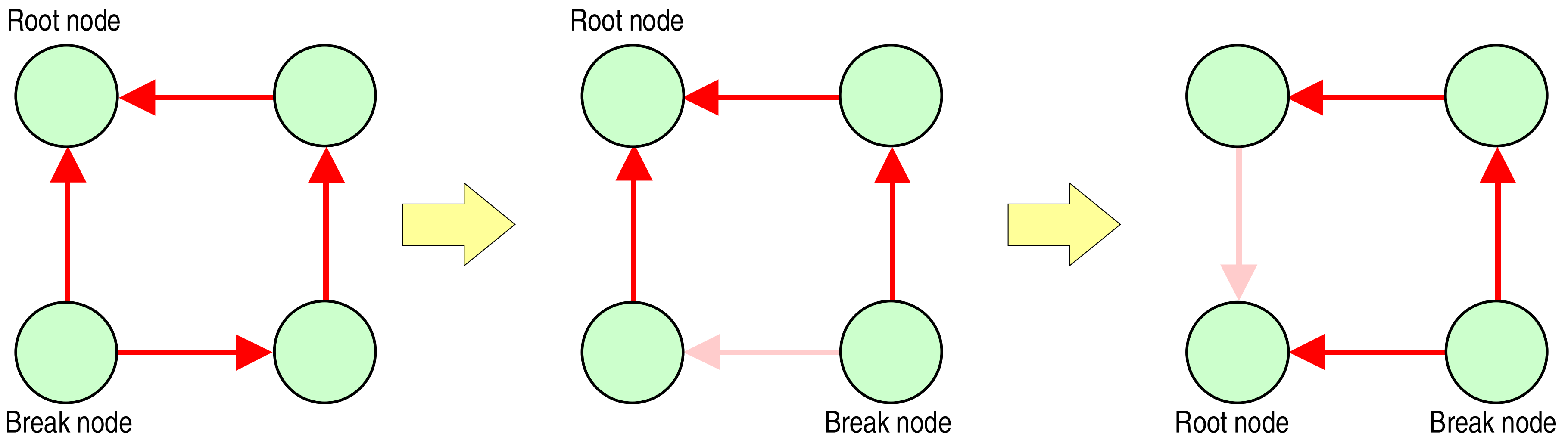
Moving a root node



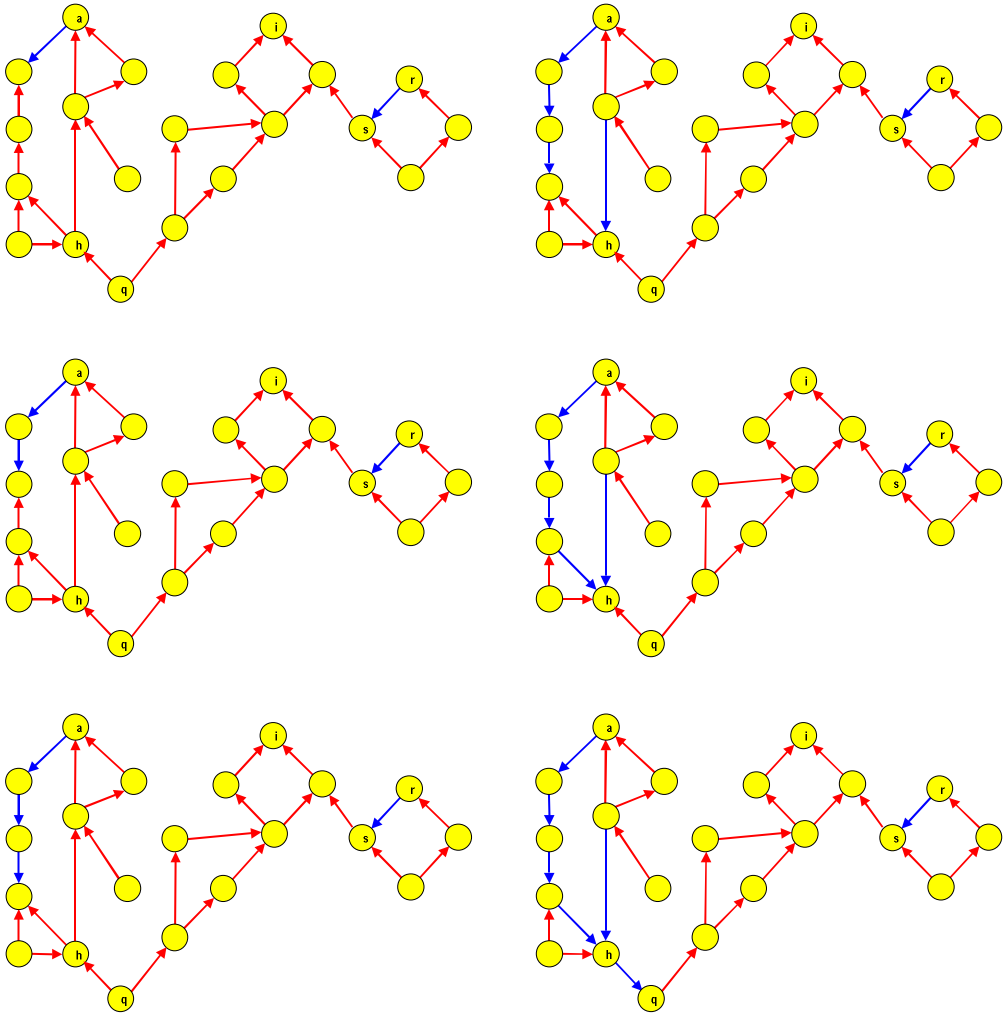
Moving a root node over a break node



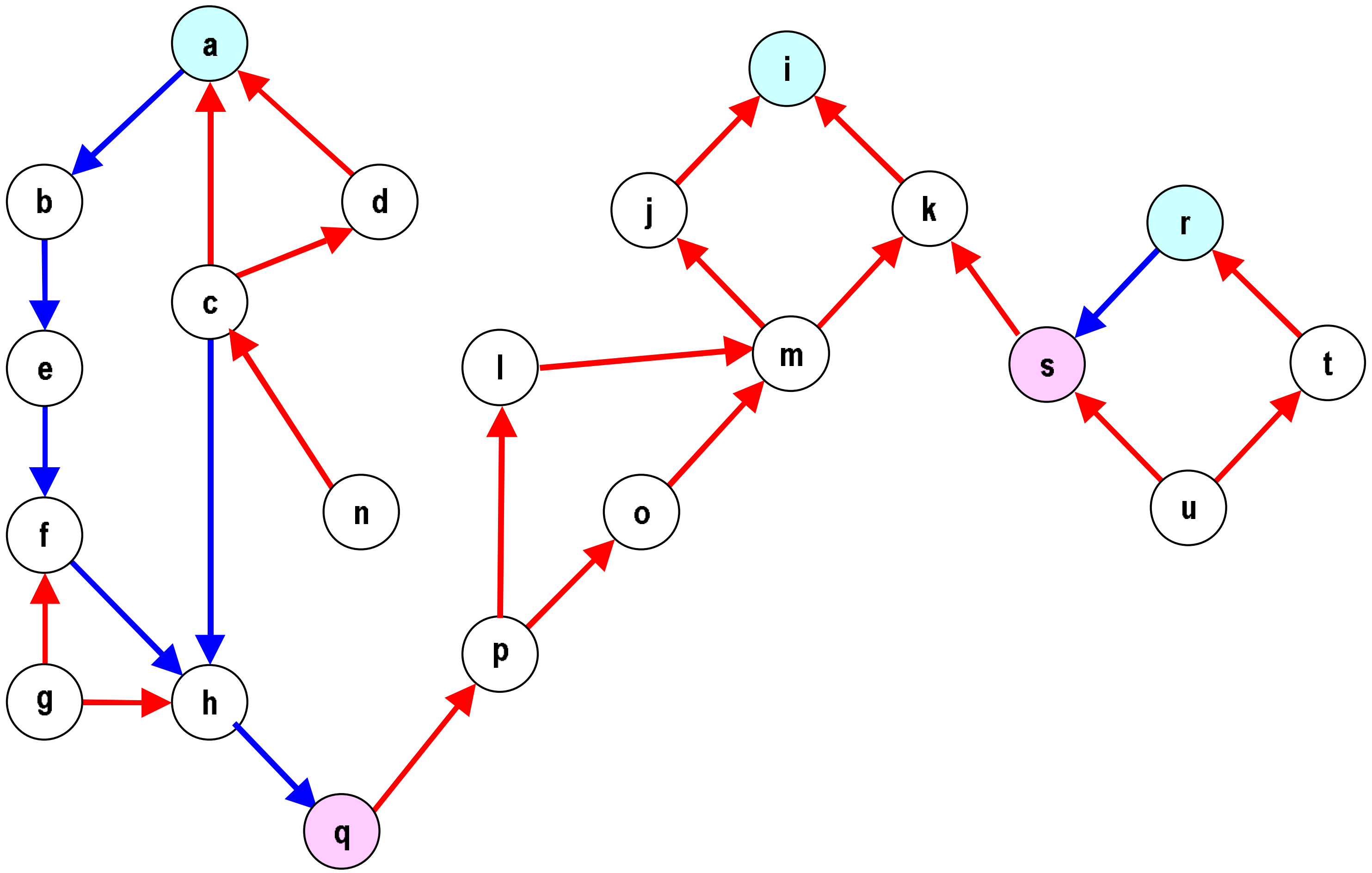
Moving a break node



Changing link direction assignment



The resulting link direction assignment



Remarks

Network reconfiguration is required when components are switched on/off, links are remapped, and components fail

Dynamic network reconfiguration allows routing table update without stopping network traffic

Dynamic reconfiguration is crucial if the network is designed to provide quality of service (QoS) guarantees

In most cases, reconfiguration only affects a few routing tables. Two major exceptions arise when the root node fails, and when two isolated networks are linked together

Improving QoS

Design issues:

- Switching and flow control
- Admision control/bandwidth allocation
- Link/switch scheduling
- Starvation of best-effort traffic
- Router architecture

Goals of the MMR project

Design a single-chip Multimedia Router (MMR) for high-speed LANs that supports a large number of multimedia connections while allocating the remaining bandwidth to best-effort traffic

The MMR should handle this hybrid traffic efficiently, satisfying the QoS requirements of multimedia traffic, minimizing the average latency of best-effort traffic, and maximizing link utilization when the network reaches saturation

Application requirements

- Very long data streams
- Large number of connections
- Jitter sensitive
- Latency tolerant, especially during connection setup
- Short control messages

Switching and flow control

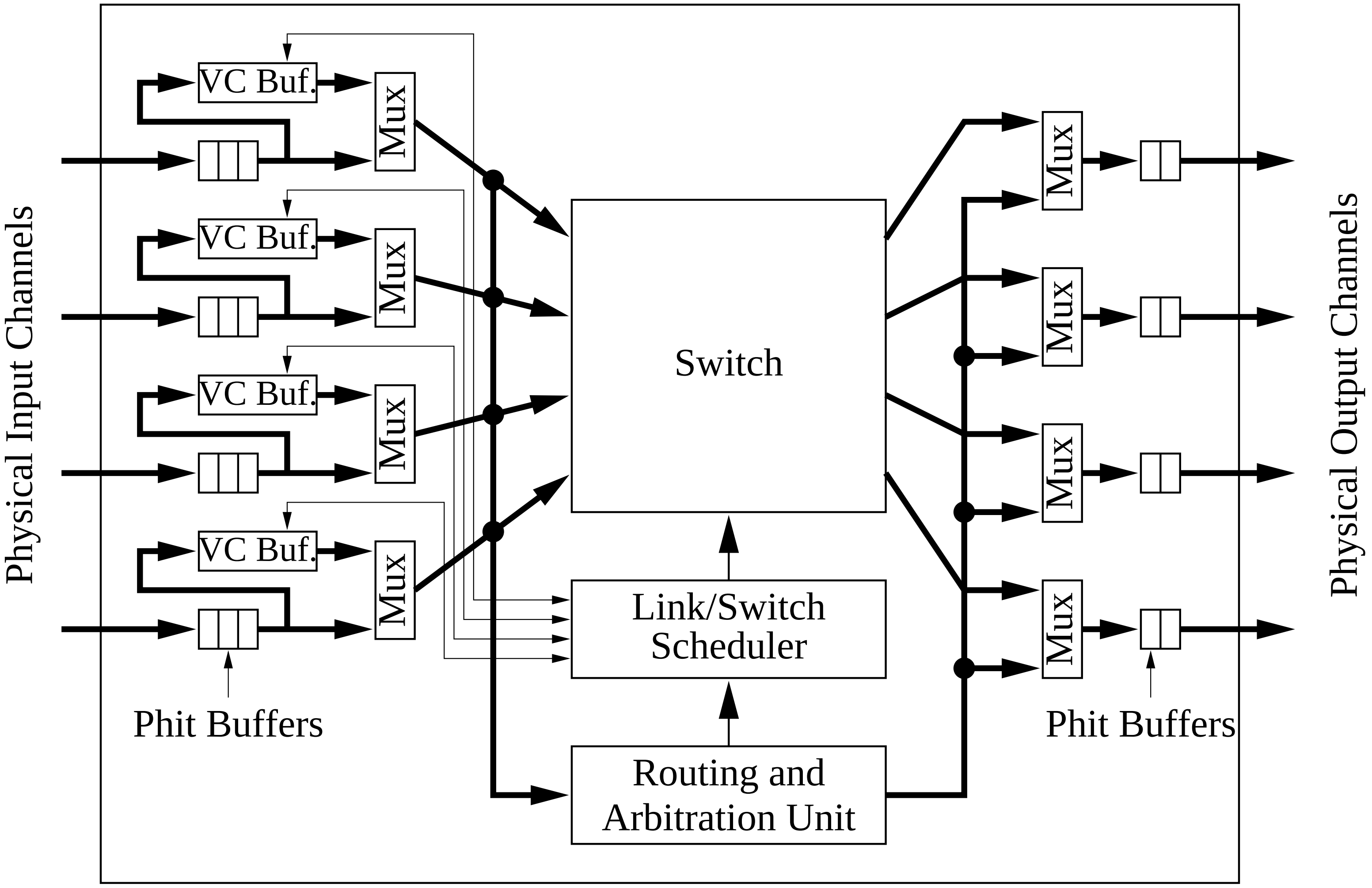
Switching techniques that reserve resources on the fly, like wormhole switching, are not well suited because the time a message is blocked waiting for a busy resource is not bounded

Control messages and best-effort traffic will benefit from low-latency switching techniques like wormhole or VCT switching

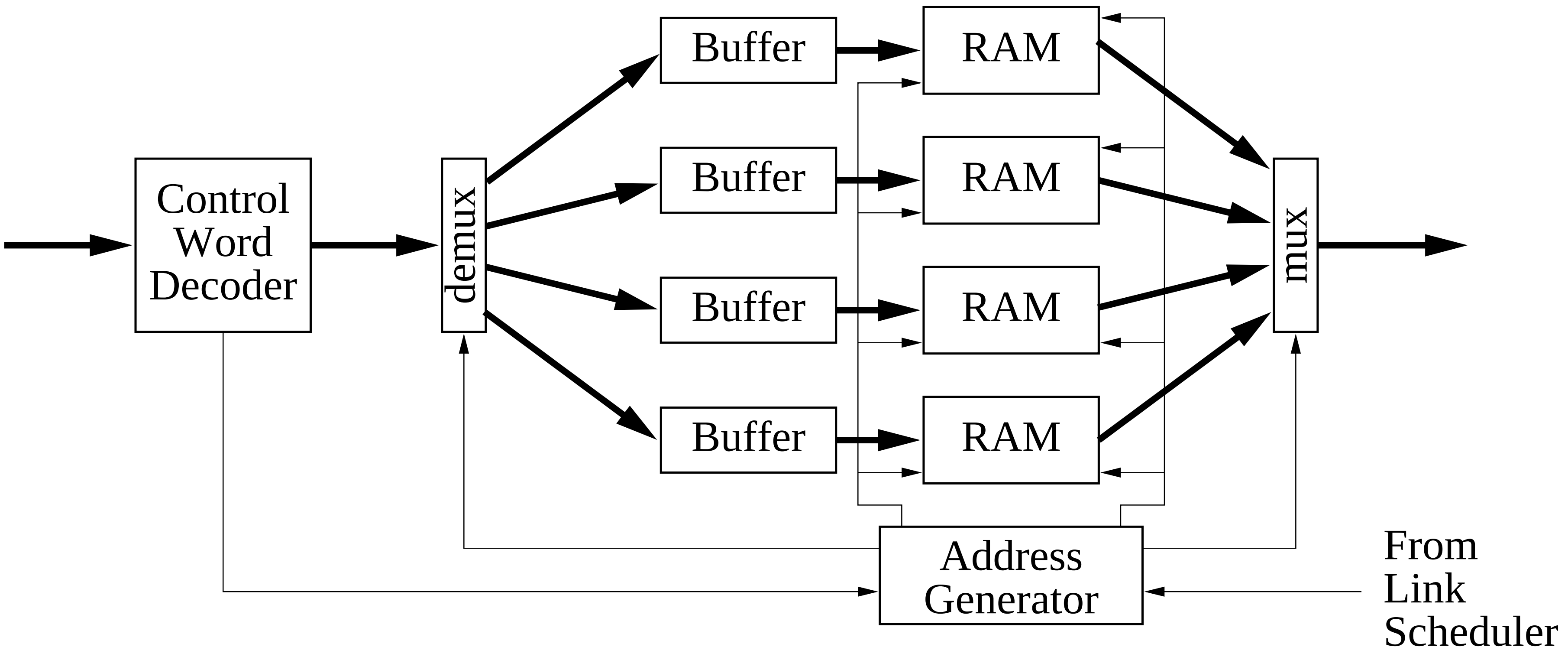
- Long data streams are transmitted by using pipelined circuit switching (PCS) by first establishing a connection from source to destination and then forwarding the data
- Control messages and best-effort traffic are transmitted by using VCT switching

Flow control prevents data losses while requiring only small buffers

Router architecture



Virtual channel buffer organization



Link/switch scheduling

Flits/packets are synchronously transmitted (flit cycle)

Links and switch ports are synchronously scheduled for the next flit cycle during the current flit cycle

Flits are large enough to hide scheduling time. Fast hardware schedulers based on status bit vectors

Flits/packets are scheduled according to their priority (e.g., EDF). For each priority level, each link scheduler supplies several candidates among those virtual channels with ready flits

The switch scheduler assigns switch ports to links, starting with those ports with a smaller number of requests. This strategy maximizes link and switch utilization

Selection matrix and conflict vector

Candidates	Output Ports	Input Ports 0	Input Ports 1	Input Ports 2	Input Ports 3	Conflict Vector
Level 1	0		55Mbps		24Mbps	2
Level 1	1					0
Level 1	2	55Mbps		12Mbps		2
Level 1	3					0
Level 2	0	64Kbps				1
Level 2	1			5Mbps		1
Level 2	2					0
Level 2	3		20Mbps		10Mbps	2

Selection matrix after arbitration steps

Input Ports

Output Ports	0	1	2	3	
0		X			0
1					0
2	55Mbps		12Mbps		2
3					0
0					0
1			5Mbps		1
2					0
3				10Mbps	1

Input Ports

Output Ports	0	1	2	3	
0		X			0
1					0
2	X				0
3					0
0					0
1			5Mbps		1
2					0
3				10Mbps	1

Output Ports	0	1	2	3	
0		X			0
1					0
2	X				0
3					0
0					0
1			X		0
2					0
3				10Mbps	1

Output Ports	0	1	2	3	
0		X			0
1					0
2	X				0
3					0
0					0
1			X		0
2					0
3				X	0

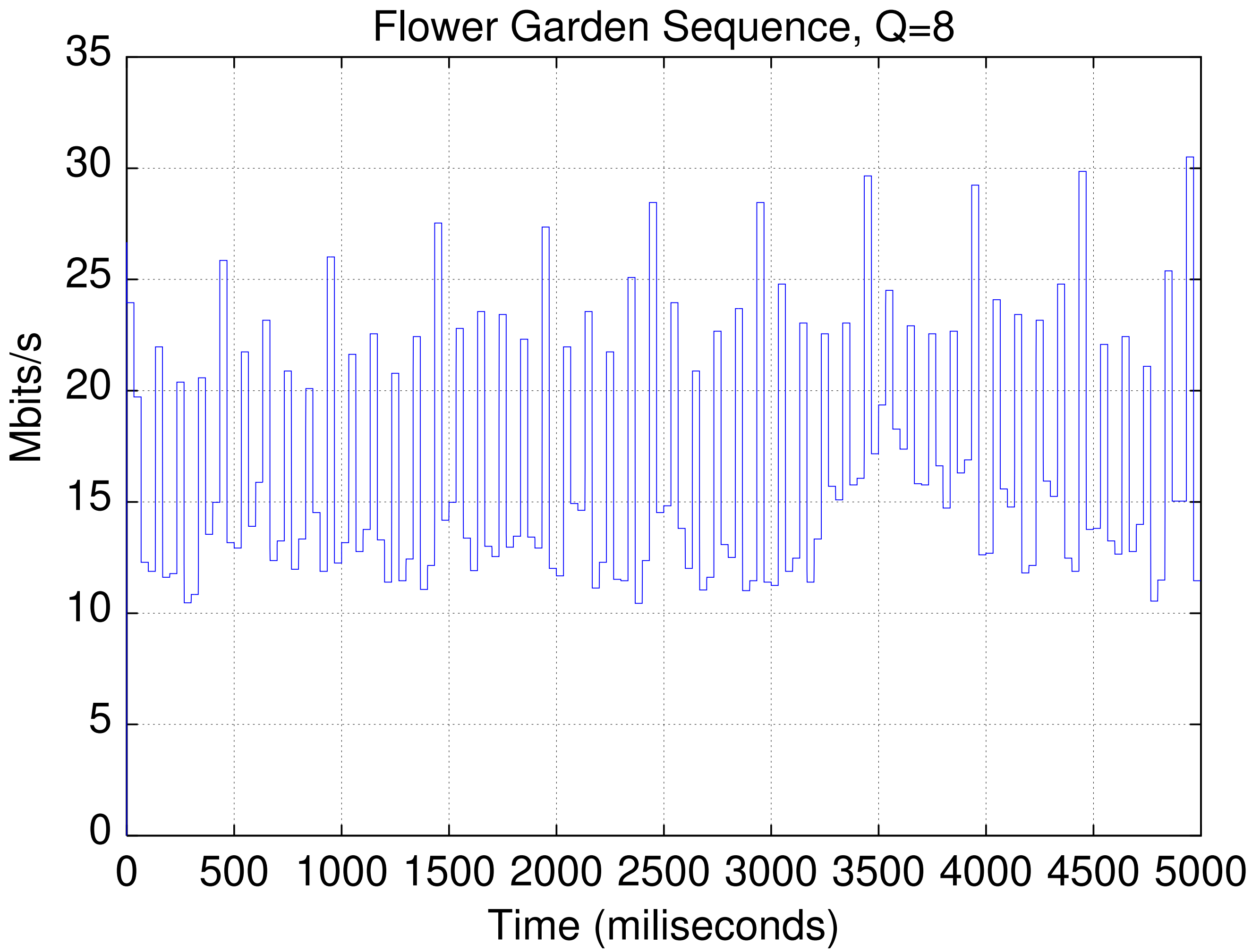
Performance evaluation

Evaluation with a mix of CBR and VBR traffic

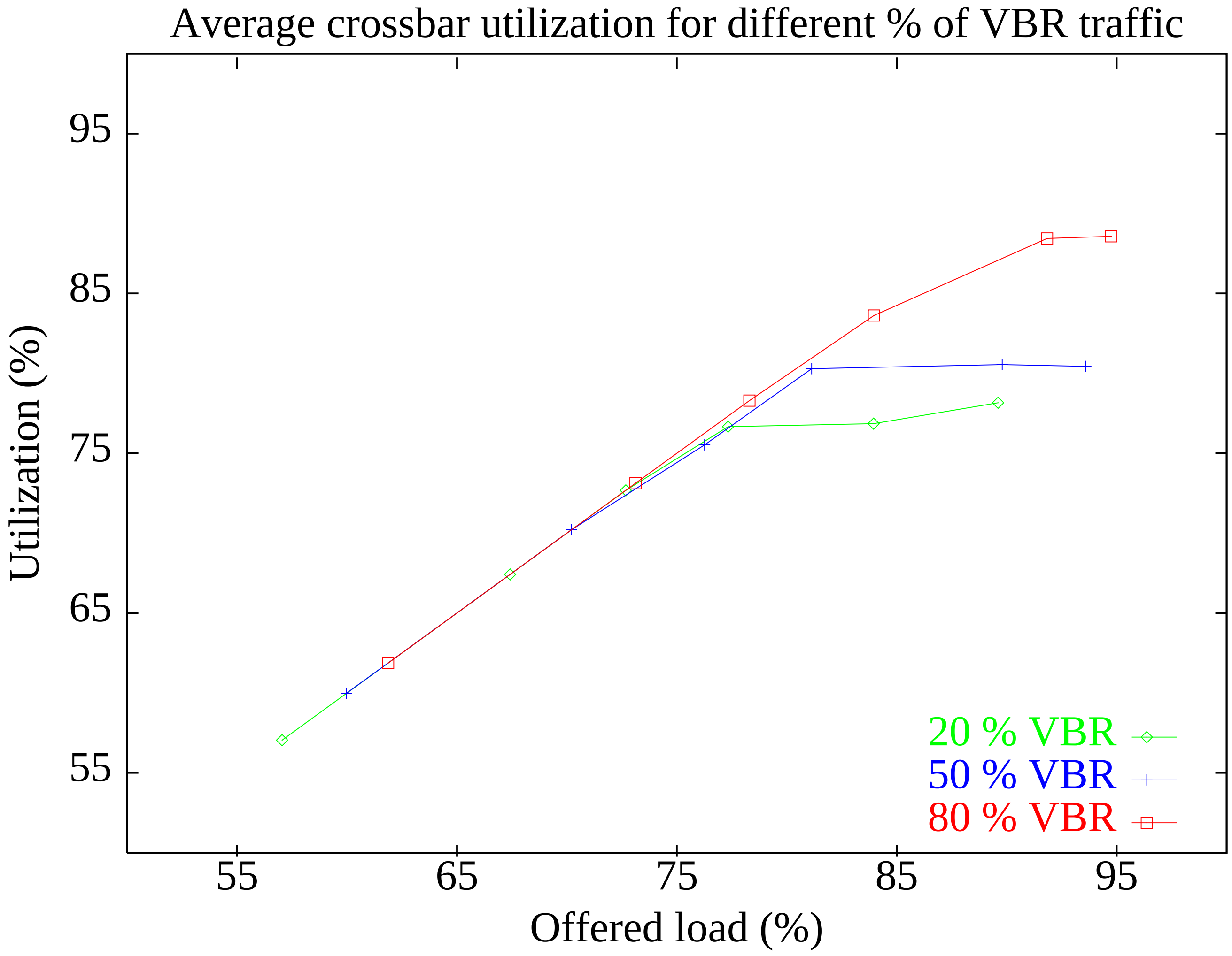
VBR traffic composed of real MPEG-2 video sequences

	Frame Size (bits)		
Video Sequences	Max.	Min.	Average
Ayersroc	535030	148755	232976
Hook	454560	159622	272738
Martin	444588	116094	199880
Flower Garden	900139	308411	497126
Mobile Calendar	970205	412845	600742
Table Tennis	933043	260002	440547
Football	590532	340246	441459

Example of VBR MPEG-2 video sequence



Average crossbar utilization



Flit delay statistics

	Load (%)	Max. Delay	Ave. Delay	Std. Dev.
Total	59.98	29358	196.38	288.00
	70.20	200073	341.28	288.00
	81.13	26915769	283940	1074500784
CBR	59.98	3299	127.50	288.00
	70.20	1913	135.26	288.00
	81.13	2969	148.99	289.00
VBR	59.98	29358	253.76	288.00
	70.20	200073	516.50	288.00
	81.13	26915769	521314	1074500784

Flit delay distribution for CBR traffic

Load (%)	TH1	TH2	TH3	TH4	TH5	TH6
59.98	0.034	0.00056	0.00011	0	0	0
70.20	0.063	0.00075	0	0	0	0

TH1 = 838.5 nanosec, TH2 = 2 $\times$ TH1, ..., TH6 = 268 microsec

Flit delay distribution for VBR traffic

Load (%)	TH1	TH2	TH3	TH4	TH5	TH6
59.98	6.35	1.17	0.29	0.11	0.045	0.004
70.20	19.50	6.81	2.09	0.64	0.20	0.075

50% of the workload composed of VBR traffic

Remarks

Single-chip routers supporting hundreds of connections are feasible. They can be used as building blocks for commercial switches

Small fixed-size buffers can be efficiently implemented using interleaved RAM modules. Flow control prevents buffer overflow. If necessary, information will be discarded at the source node

Simple link/switch scheduling strategies amenable for single-chip implementation can be very efficient. Synchronous scheduling, status bit vectors, and multiple candidates per link are the key design issues

Small phit buffers allow fast processing of control information

Admission control, dynamic bandwidth allocation, and starvation of best-effort traffic are currently under study