A Low-Overhead, Fully-Distributed, Guaranteed-Delivery Routing Algorithm for Faulty Network-on-Chips

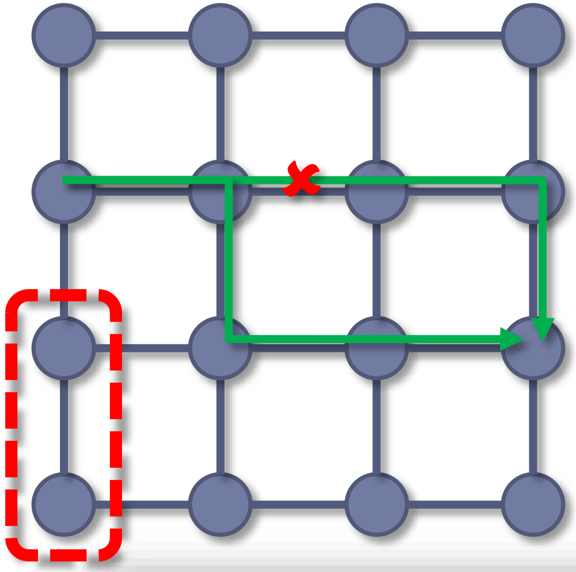
Mohammad Fattah[1], Antti Airola[1], Rachata Ausavarungnirun[2], Nima Mirzaei[3],
Pasi Liljeberg[1], Juha Plosila[1], Siamak Mohammadi[3], Tapio Pahikkala[1],
Onur Mutlu[2] and Hannu Tenhunen[1]

Turun yliopisto
University of Turku

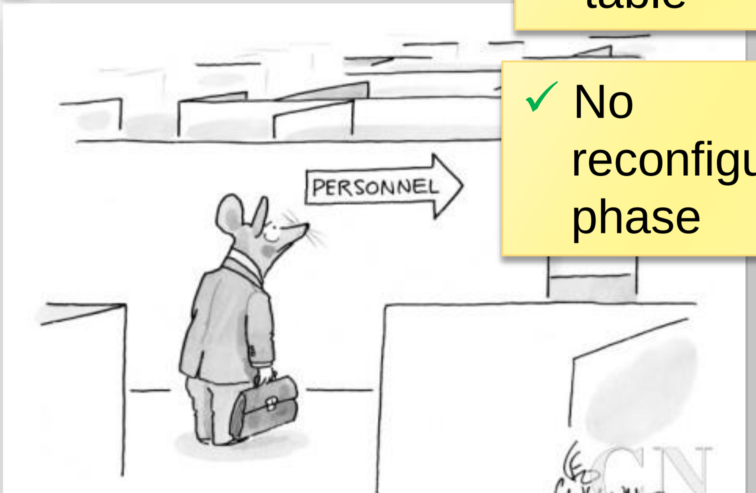
Carnegie Mellon University

What is This Talk About?

- Overtime, routers and links can become faulty.
- Dynamically find alternative paths.
- Previous works have at least one of the following limitations:
  - Cover only few number of faults
  - Use a central controller
  - High area overhead
  - High reconfiguration overhead upon new faults
- Maze-Routing overcomes all the above limitations:
  - **Full-coverage:** formally proven
  - **Fully-distributed:** using autonomous and standalone routers
  - **Low area overhead:** using an algorithmic approach (16X less area compared to routing tables)
  - **Low reconfiguration overhead:** by on the fly path exploration (Instantaneous operation on new failures)
  - **Better performance:** 50% higher saturation throughput and, 28% lower latency on SPEC benchmarks compared to state-of-the-art



- ✓ Any # of faults
- ✓ Detect partitioning
- ✓ No central component
- ✓ No routing table
- ✓ No reconfiguration phase



Source: condenaststore.com

Aggressive Transistor Scaling

Key Benefit

- Integrating many IPs
  - Processors
  - Cache slices
  - Memory controllers
  - Specialized HW
  - Etc.

A Major Curse

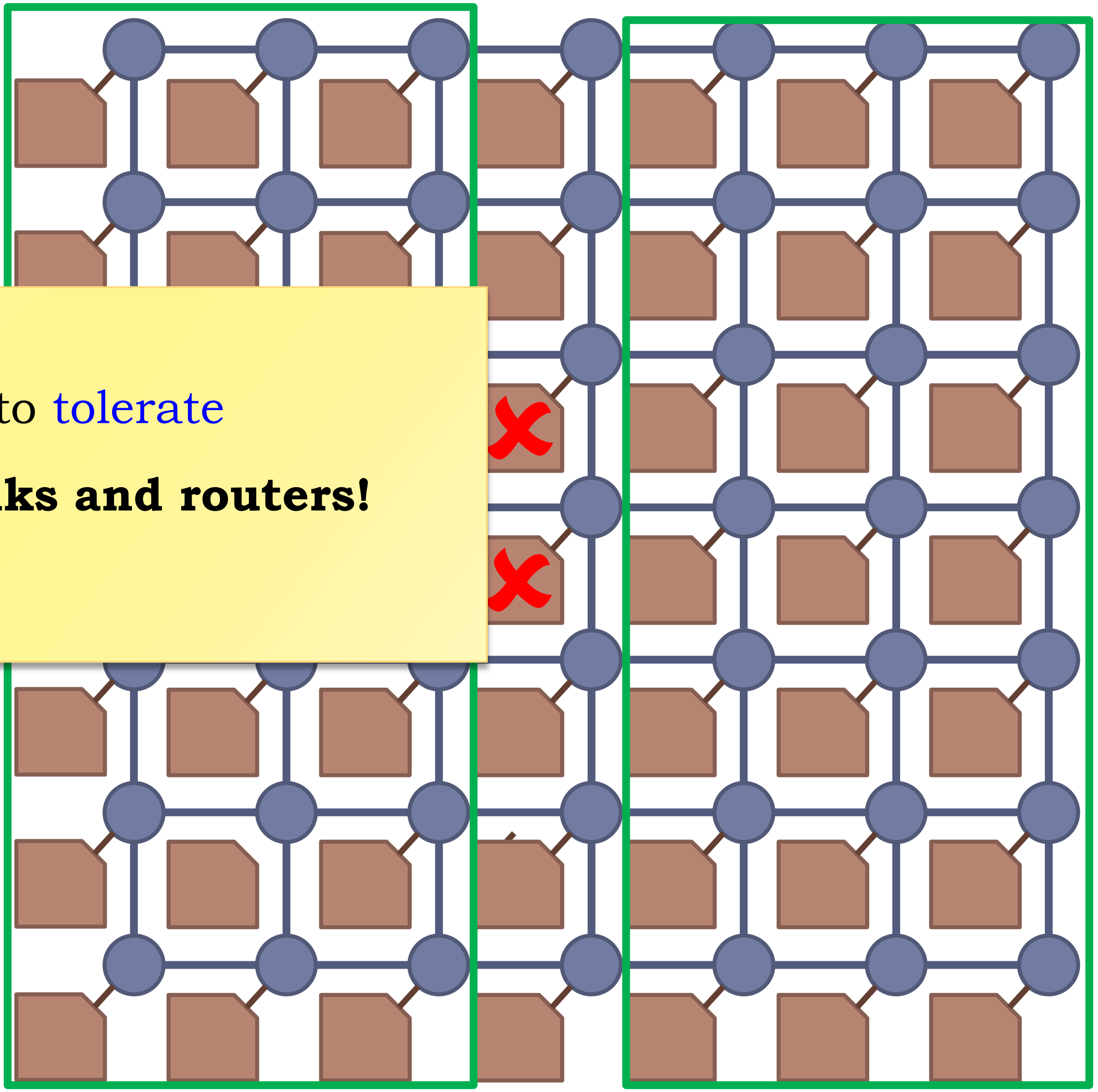
- Reduced reliability
  - Fabrication time:
  - tion
  - temperature instability (NBTI)
  - Hot carrier injection (HCI)
  - Gate oxide breakdown
  - Electro-Migration

Our designs must be:

Fault-tolerant by construction!

IP vs. Network Faults

- IP
  - Degrades the performance
  - Rest of the system can continue
- Network Elements
  - Cripples the perfor
  - Single point of failu



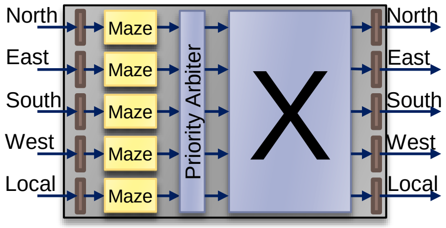
It is crucial to tolerate

Many faults in links and routers!

Maze-Routing

Fault-Tolerant by Construction

- It is not:
  - A router architecture with fault tolerance
- Rather, it is
  - Essentially a routin… which
  - Is inherently fault-tolerant



Maze-Routing is

The first to provide all!

Four Critical Goals

- Full coverage (guaranteed delivery)
- …d operation
- Low area footprint
- No reconfiguration component/phase

Our 4 Goals

- Full coverage
- Full distribution
- Low area cost
- Fast adaptation

Maze-Routing

- Finding the path
- Detecting disconnected nodes

Results

- Area
- Throughput
- Reconfiguration overhead

Our 4 Goals

- Full coverage
- Full distribution
- Low area cost
- Fast adaptation

Goal 1: Full (Fault) Coverage

Literature

- Limited number of faults

> We propose an ultra-low-cost reconfigurable routing algorithm supporting any one-faulty-router topology. [DAC'08]

> [DATE'15] That is, we achieve 100% 1-link failure coverage. For 2-link failures the coverage is 98.8% for an 8x8 mesh, which grows to 99.3 for a 10x10 mesh. The

> The net result achieved by d^2-*LBDR* is 100% coverage support for 1-link and 2-link failures [DATE'15]

- Limited fault pattern

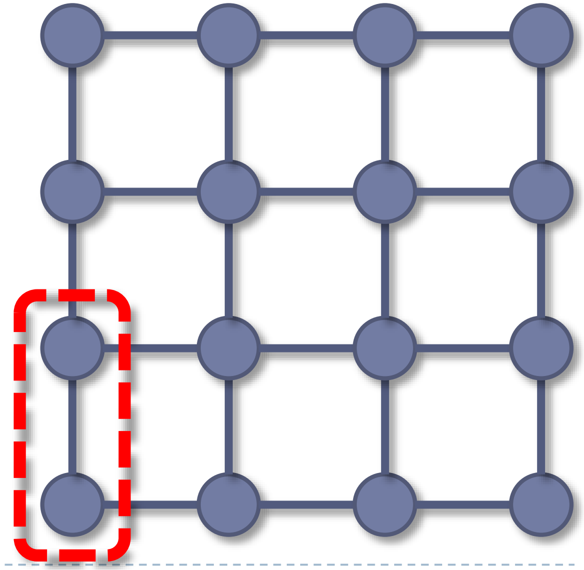
> BLINC reconfiguration is capable of tolerating a single link failure per segment. [DATE'14]

- Limited when disconnected

> [IEEE TVLSI'13] The faulty region can be any shape as long as it does not disconnect the network.

Maze-Routing

- No restriction on
  - Fault count
  - Fault pattern
- Detect disconnected nodes
  - At router level



Goal 2: Fully Distributed Operation

Literature

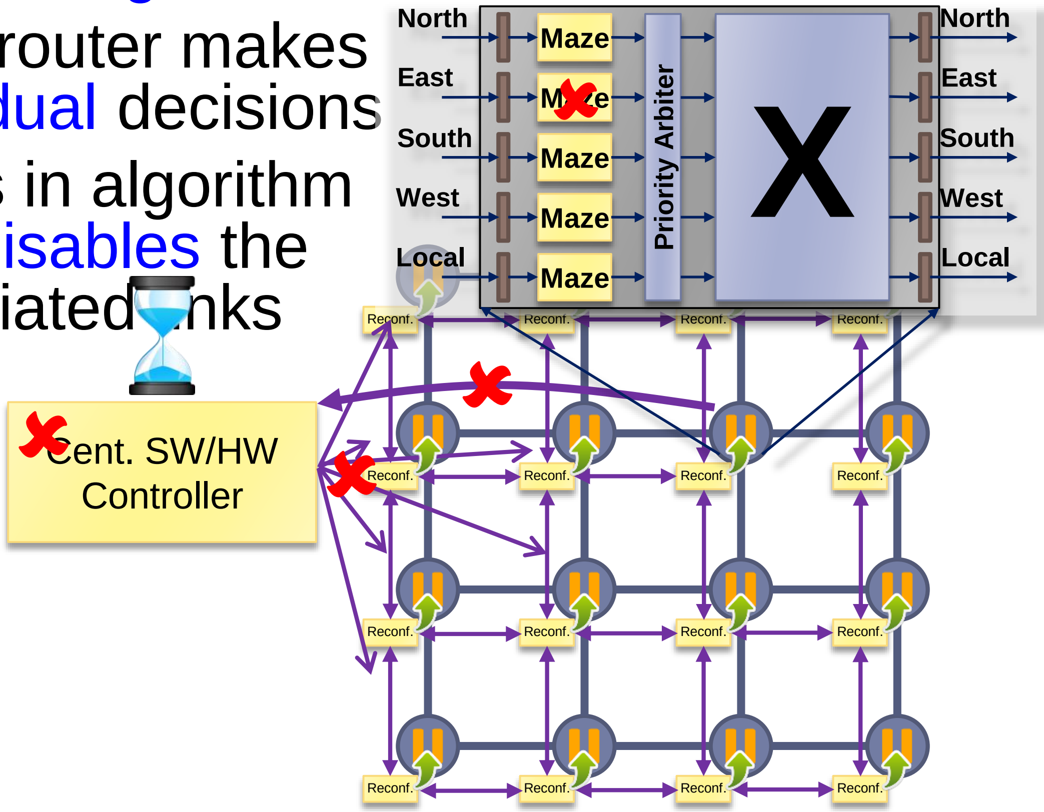
[DATE'09] In addition, it is assumed that the routers know when they need to invoke the algorithm and how to resume operation after reconfiguration finishes.

[PACT'11] The most effective way to further protect small hardware structures, such as Ariadne, from failures is triple modular redundancy (TMR).

- Distributed methods
  - Synchronization points.
  - Fault in Reconf. unit.

Maze-Routing

- No central component
- No reconfiguration unit
- Each router makes individual decisions
- Faults in algorithm only disables the associated links



Goal 3: Low Area Overhead

Literature

- Routing tables
  - High area overhead
    - 5 read ports
- Implementation cost
- Power dissipation
- Vulnerability to run-time faults
  - One failed bit: affects the whole router
  - Area ~ fault probability of router

Maze-Routing

- An algorithmic approach
- No routing table

Goal 4: Low Reconfiguration Overhead

Literature

- New failure detected?
  1) Pause the network
  2) Reconfigure to an alternative solution
  3) Resume normal operation
- Issues?
  - Severe degradation of performance
  - aggressive online testing
- Few works with fast reconfiguration

Maze-Routing

- No reconfiguration phase
- Path to destination is dynamically calculated per packet
- Called on the fly reconfiguration

Maze-Routing: The First to Provide All

	Coverage	Reconfiguration	O(Area)	O(Reconf.)
Zhang et al. [43]	few	fully distributed	low	on the fly
LBDR [35]	moderate	central	low	N/A
d2-LBDR [7]	moderate	central	low	N/A
OSR-Lite [38]	moderate	central	low	moderate
TOSR [5]	moderate	distributed	high	fast
BLINC [25]	moderate	distributed	high	fast
uLBDR [36]	high	central	high	N/A
Wachter et al. [39]	high	distributed	high	slow
Fick et al. [19]	high	distributed	high	slow
Face routing [11]	high	fully distributed	excessive	on the fly
FTDR-H [18]	high	fully distributed	high	fast
uDIREC [32]	full	central	high	excessive
ARIADNE [3]	full	distributed	high	slow
Maze-routing	full	fully distributed	low	on the fly

Our 4 Goals

- Full coverage
- Full distribution
- Low area cost
- Fast adaptation

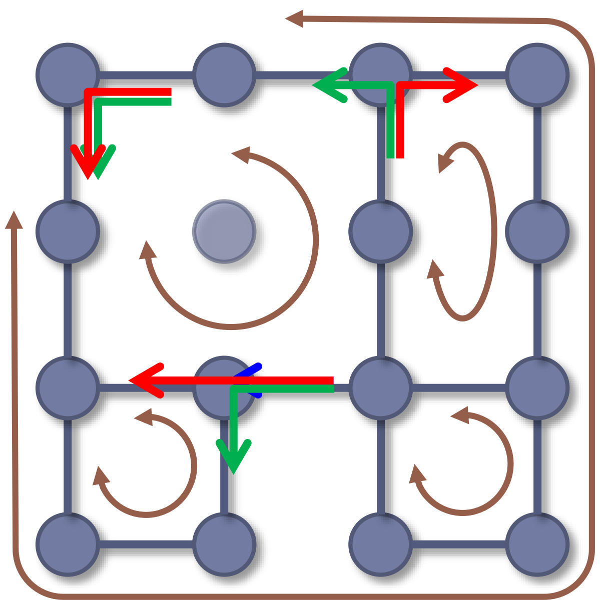
Maze-Routing

- Finding the path
- Detecting disconnected nodes

Preliminaries

- Face: regions bounded by links and routers
  - 4 inner faces
  - 1 outer face

- Right/Left hand rule: exit from first output in right/left side.
  - ↱: clockwise around inner faces
  - ↰: counterclockwise around inner faces
  - Opposite direction around outer faces



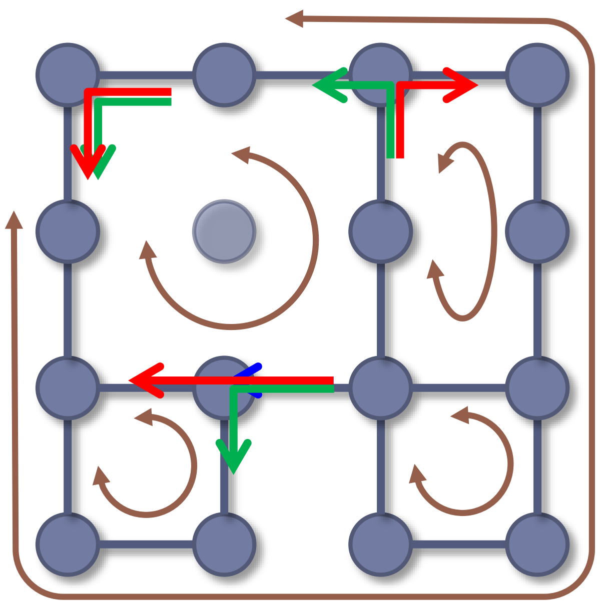
Preliminaries (II)

- Few additional fields in the header

1. MD_{best} : closest distance (MD) to *dst* that the packet has reached so far
   - Initial: $MD_{src,\, dst}$
   - Only decrements

2. Mode: routing mode used for the packet
   - Values: normal, traversal (↱ or ↰), unreachable
   - Initial: normal

- 2 more fields to detect disconnected nodes

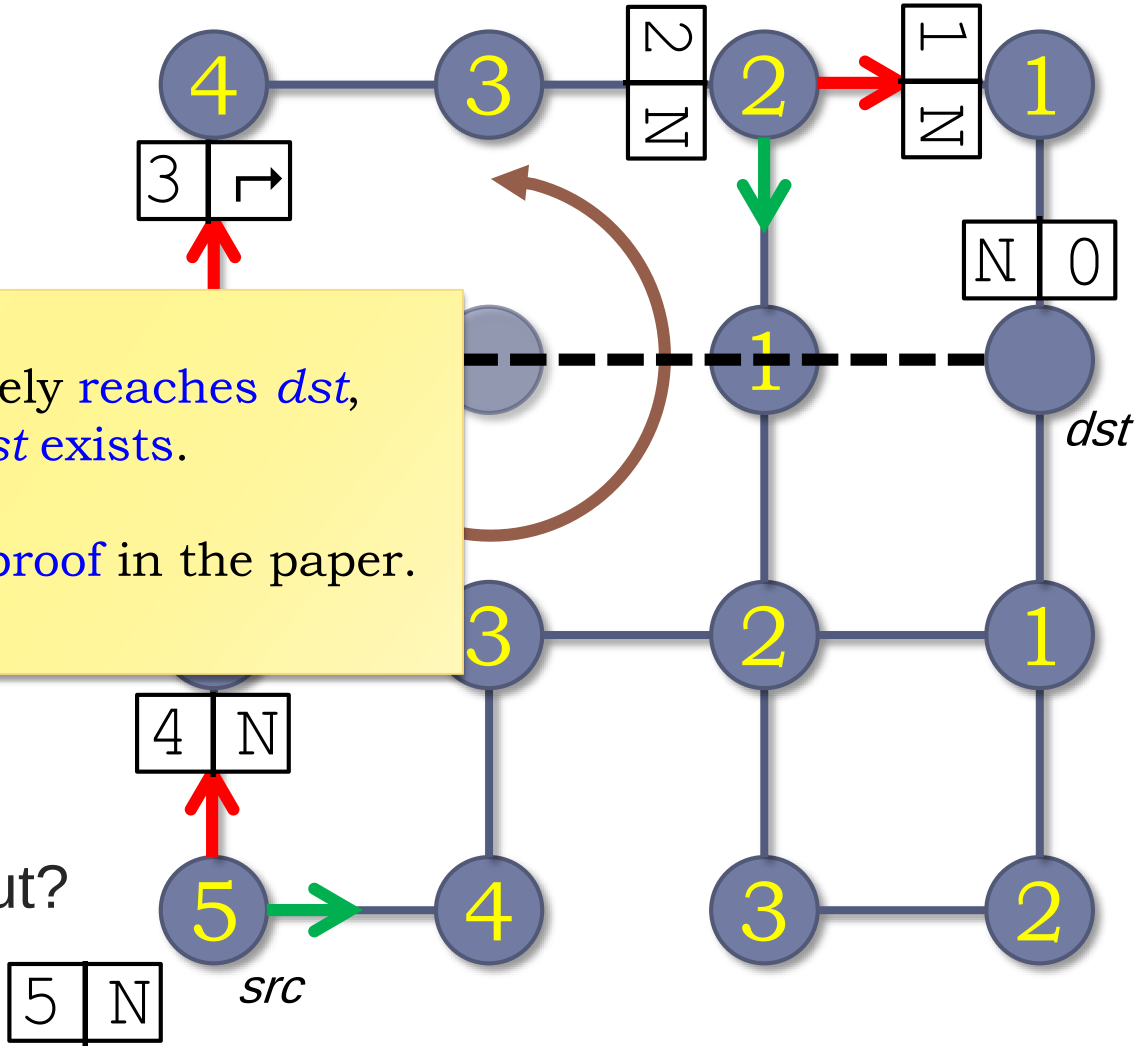


Maze-Routing

- Normal mode:
  - Is there any productive output?
    - Take it and dec(MD_{best})
  - No? we should enter traversal mode:
    - Draw $line_{(cur,\ dst)}$ bet
    - ↱? Take the first ou
    - ↰? Take the first ou
    - Set the mode (eithe

Maze-Routing definitely reaches *dst*,
if a path to *dst* exists.

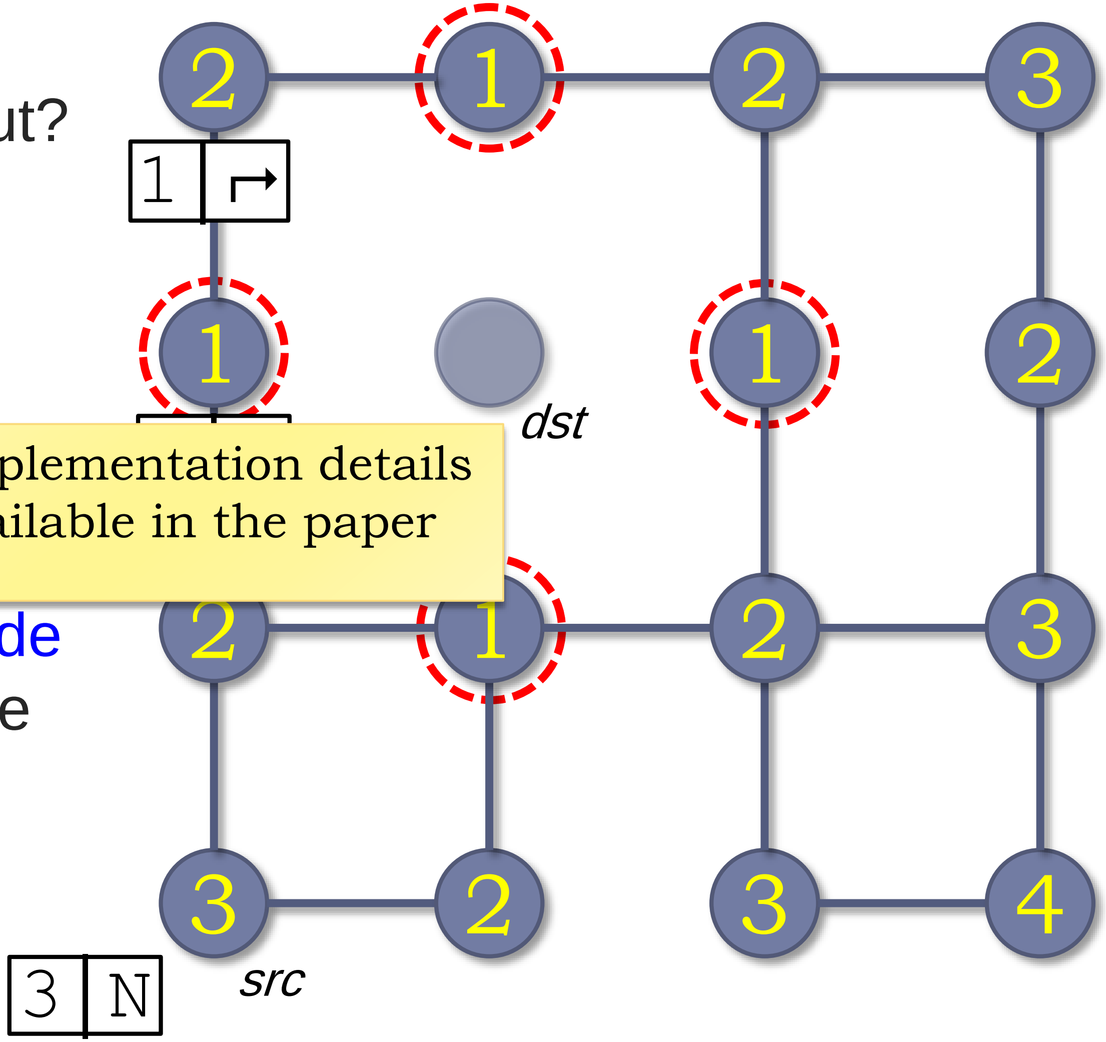
We provide the formal proof in the paper.



- Traversal mode:
  - If $MD_{cur,\ dst} = MD_{best}$ with productive output?
    - Return to (and act as in) normal mode
  - Otherwise, follow the hand rule

Detecting Disconnected Nodes

- Traversal mode:
  - If $MD_{cur,\ dst} = MD_{best}$ with productive output?
    - Return to normal mode
  - No?
    - Follow the hand rule
- The destination is unreachable
  - In traversal mode, we meet the same node as the one we entered the traversal mode
  - The hand rule picks the same output as when we entered the traversal mode



More implementation details are available in the paper

Our 4 Goals

- Full coverage
- Full distribution
- Low area cost
- Fast adaptation

Maze-Routing

- Finding the path
- Detecting disconnected nodes

Results

- Area
- Throughput
- Reconfiguration overhead

Simulation Methodology

- NOCulator[1]
  - 8x8 mesh for performance analysis
  - Synthetic traffic for performance evaluation
  - SPEC CPU2006 benchmarks are also evaluated

- Maze-Routing[2] implanted in minBD[3] routers
  - Deflection-based: deadlock freedom
  - Golden and sliver flits: router-level livelock freedom
  - Retransmit-once: protocol-level deadlock freedom

[1] NOCulator: https://github.com/CMU-SAFARI/NOCulator

[2] Maze-Routing: https://github.com/CMU-SAFARI/NOCulator/tree/Maze-routing

[3] MinBD: Fallin, Chris, et al. "MinBD: Minimally-buffered deflection routing for energy-efficient interconnect." NoCS 2012.

Configurations

- Maze-Routing
  - 16 buffer spaces per (minBD) router

- Base-line router
  - Wormhole buffered routers
  - 1 VC per port
  - 40 buffer spaces per router

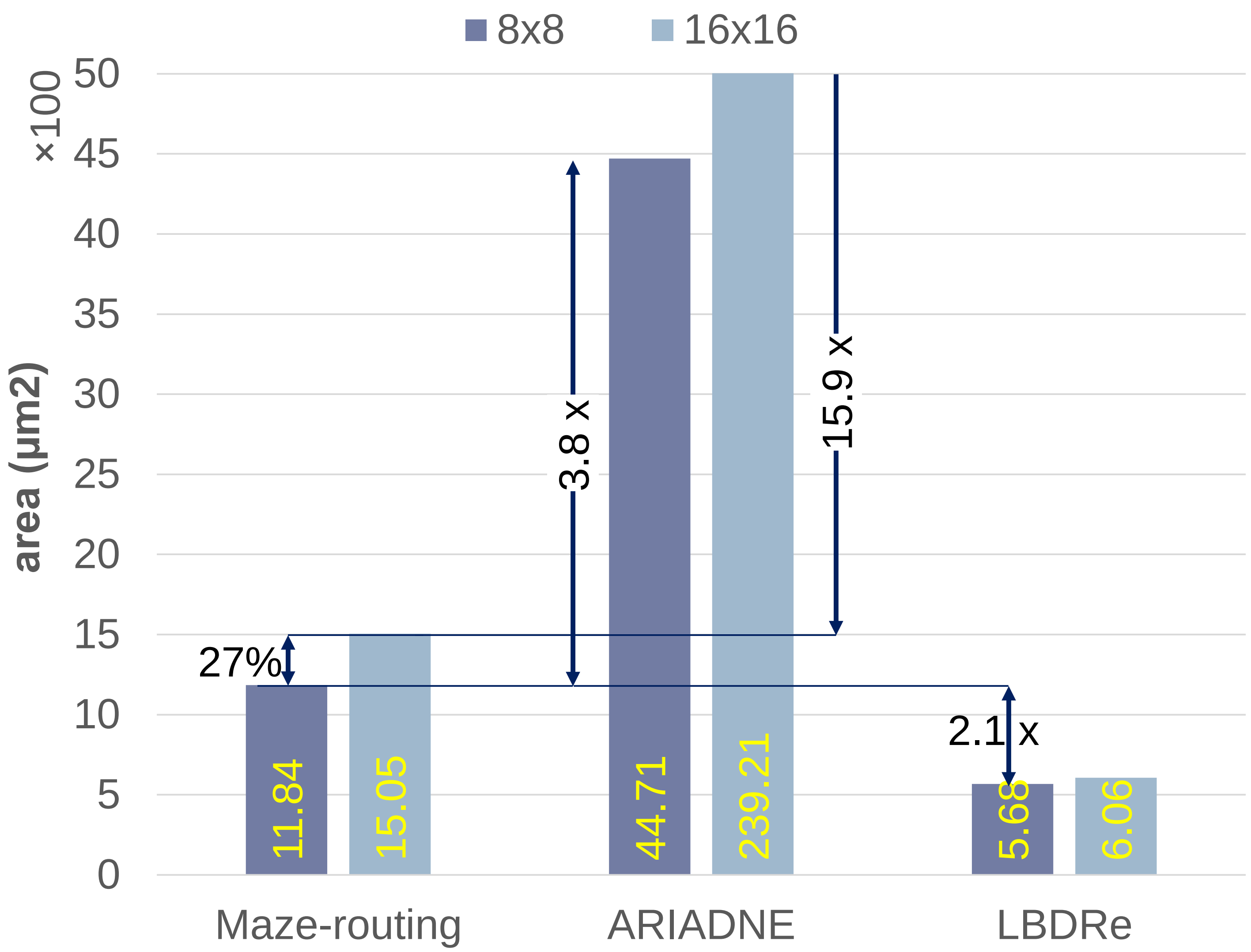
- Faults:
  - Links disabled randomly
  - From 1 to 5 link failures

Workloads

- Synthetic traffic
  - Uniform random traffic with variant injection rates
- SPEC CPU2006 benchmarks
  - Grouped based on L1 misses per kilo instruction (MPKI)
  - 3 groups: High (>50), Low (<5), and Medium (rest) intensity
  - 4 mixes: L (all Low), ML (Medium/Low), M (all Medium), and H (all High).

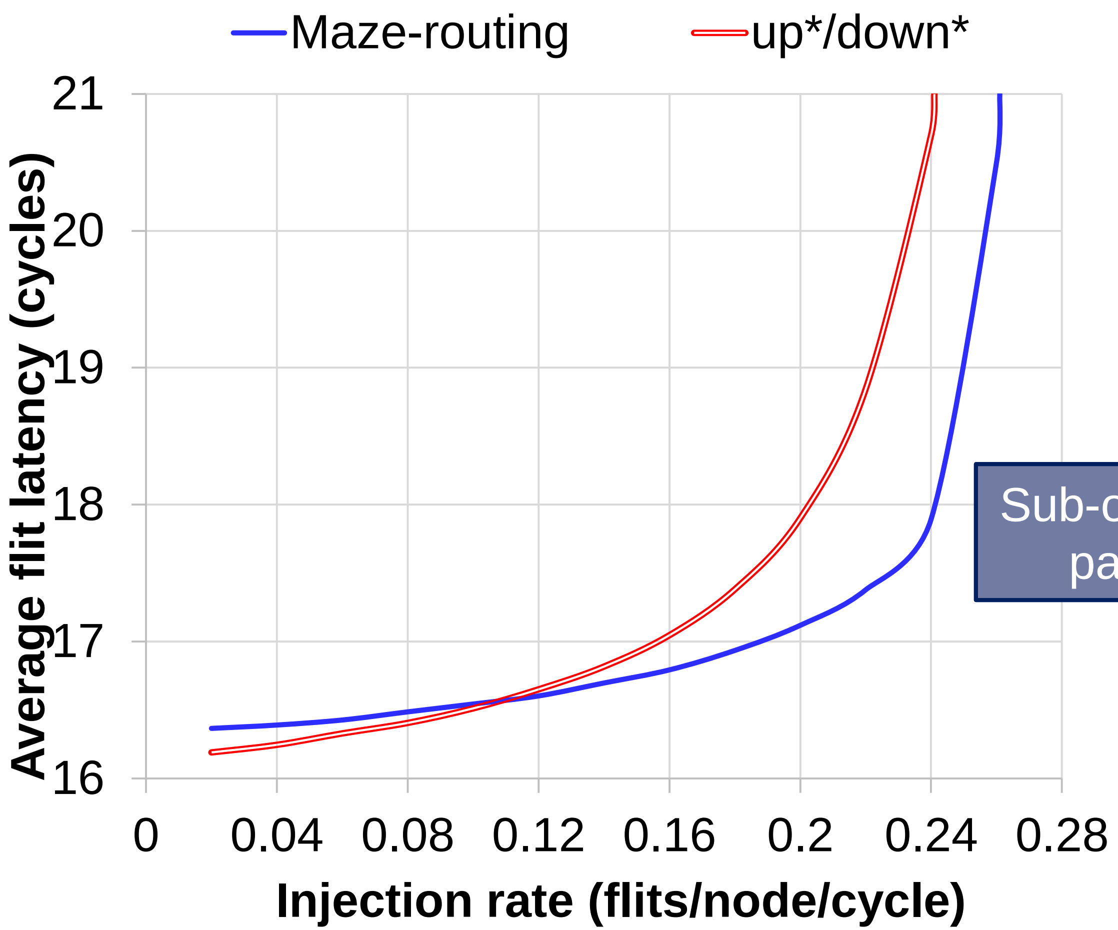
Area Overhead

- STMicro 60nm technology node
- Maze-routing:
  - 5 copies of alg., 1 per port
- ARIADNE:
  - Smallest table
  - Reconfiguration logic is not implemented
  - 5 read ports
- LBDRe:
  - Logic-based method
  - Central approach
  - Limited coverage

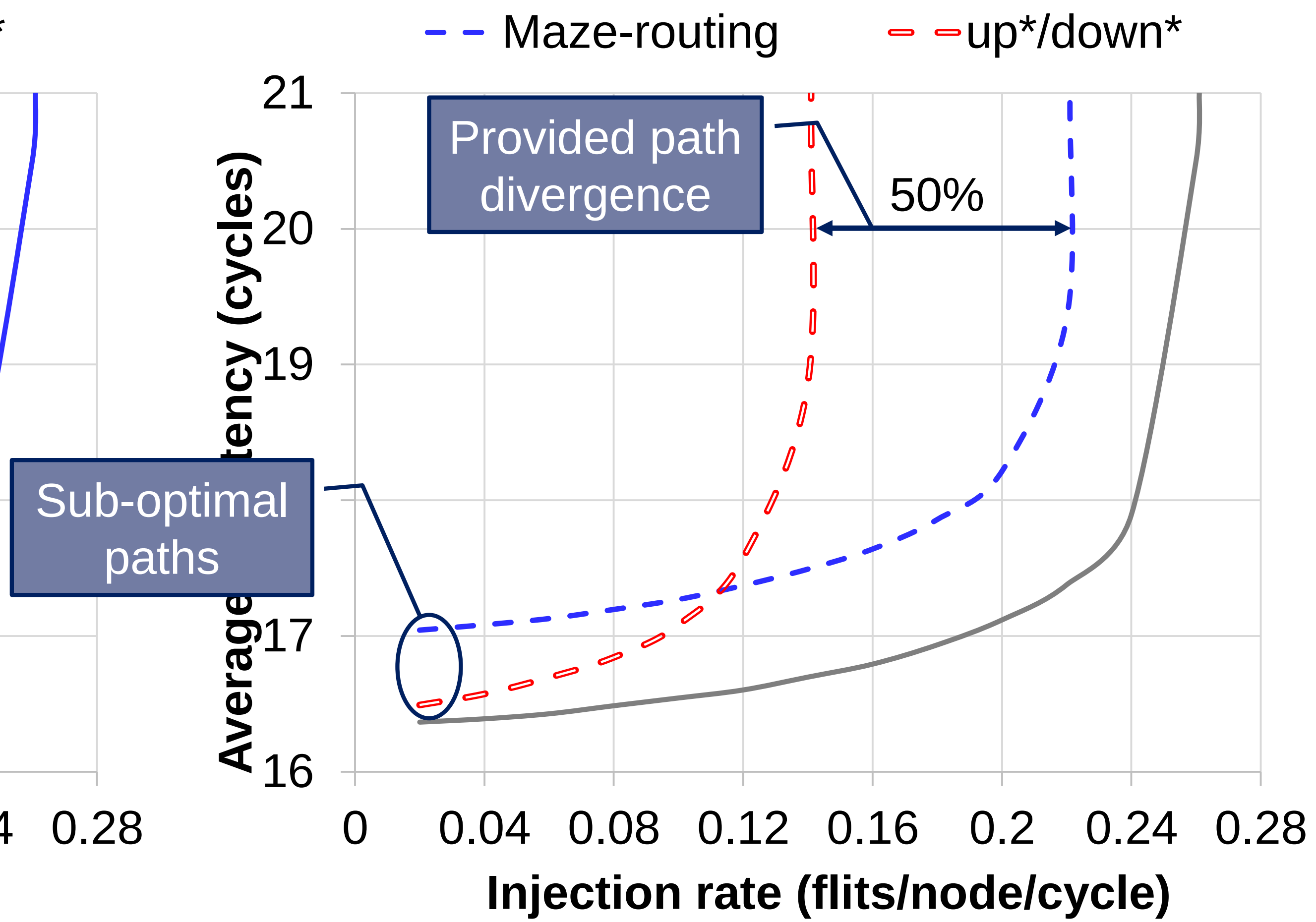


Throughput: Uniform Random Traffic

1 disabled link



5 disabled links



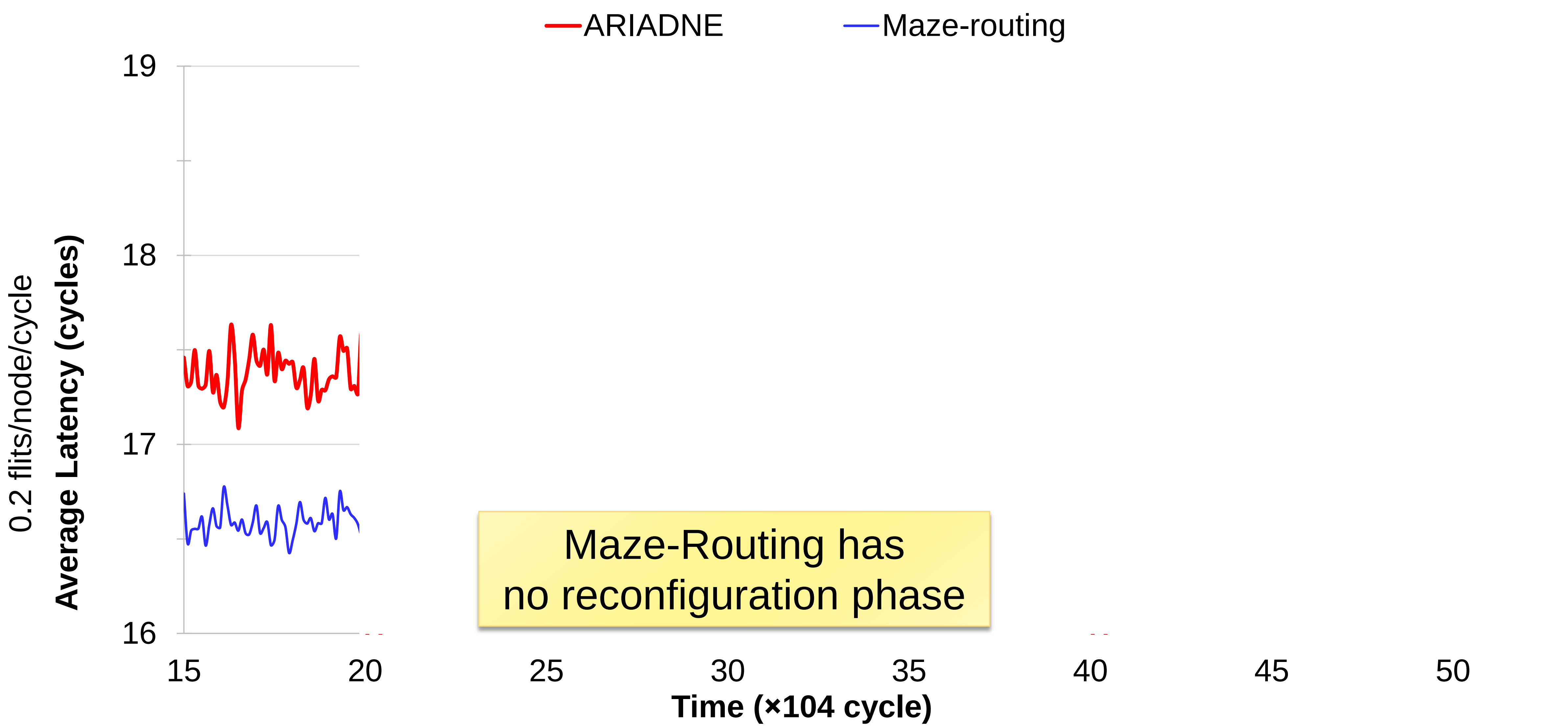
Throughput: SPEC CPU

Average packet latency

workload mix	Up*/Down*		Maze-routing	
	5 failures	no failure	5 failures	no failure
L	16.7	16.4	17.8	16.4
ML	18.8	18.2	18.9	17.2
M	27.7	25.7	21.6	19.2
H	54.4	50.5	25.8	23.1
AVG	**29.4**	**27.7**	**21**	**19**

30% latency reduction in average case

Reconfiguration Overhead



Summary

- A practical fault-tolerant routing algorithm must
  - Provide full coverage with guaranteed delivery
  - Operate in fully-distributed manner
  - Impose low area overhead
  - Have low reconfiguration overhead

- Maze-Routing is the first work to meet all the above goals

- NOCulator and Maze-Routing are available on GitHub
  - https://github.com/CMU-SAFARI/NOCulator
  - https://github.com/CMU-SAFARI/NOCulator/tree/Maze-routing

A Low-Overhead, Fully-Distributed, Guaranteed-Delivery Routing Algorithm for Faulty Network-on-Chips

Mohammad Fattah[1], Antti Airola[1], Rachata Ausavarungnirun[2], Nima Mirzaei[3], Pasi Liljeberg[1], Juha Plosila[1], Siamak Mohammadi[3], Tapio Pahikkala[1], Onur Mutlu[2] and Hannu Tenhunen[1]

Turun yliopisto
University of Turku

Carnegie Mellon University

Backup slides

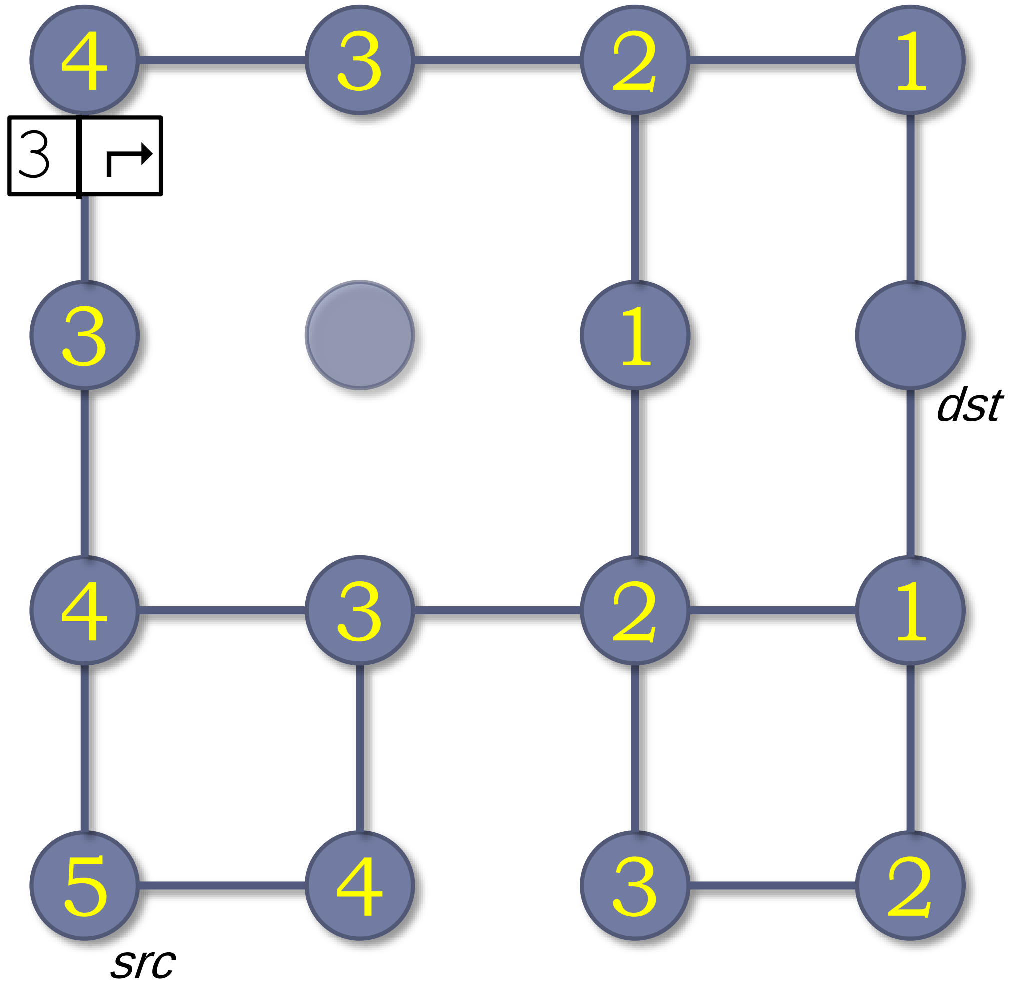
Area Overhead

- Header fields can be coded in 14/17 bits in 8x8/16x16 meshes.
- Assuming a baseline router with 144-bit channel width, we need to widen the channel by 10%/12%.
- Results in almost 20%/25% increase in the router area.

Deflection Implications

- When a packet is deflected
  - Header values are not valid anymore

- We need to reset the header values:
  - Mode ➔ Normal
  - MD_{best} ➔ MD (next router, dst)



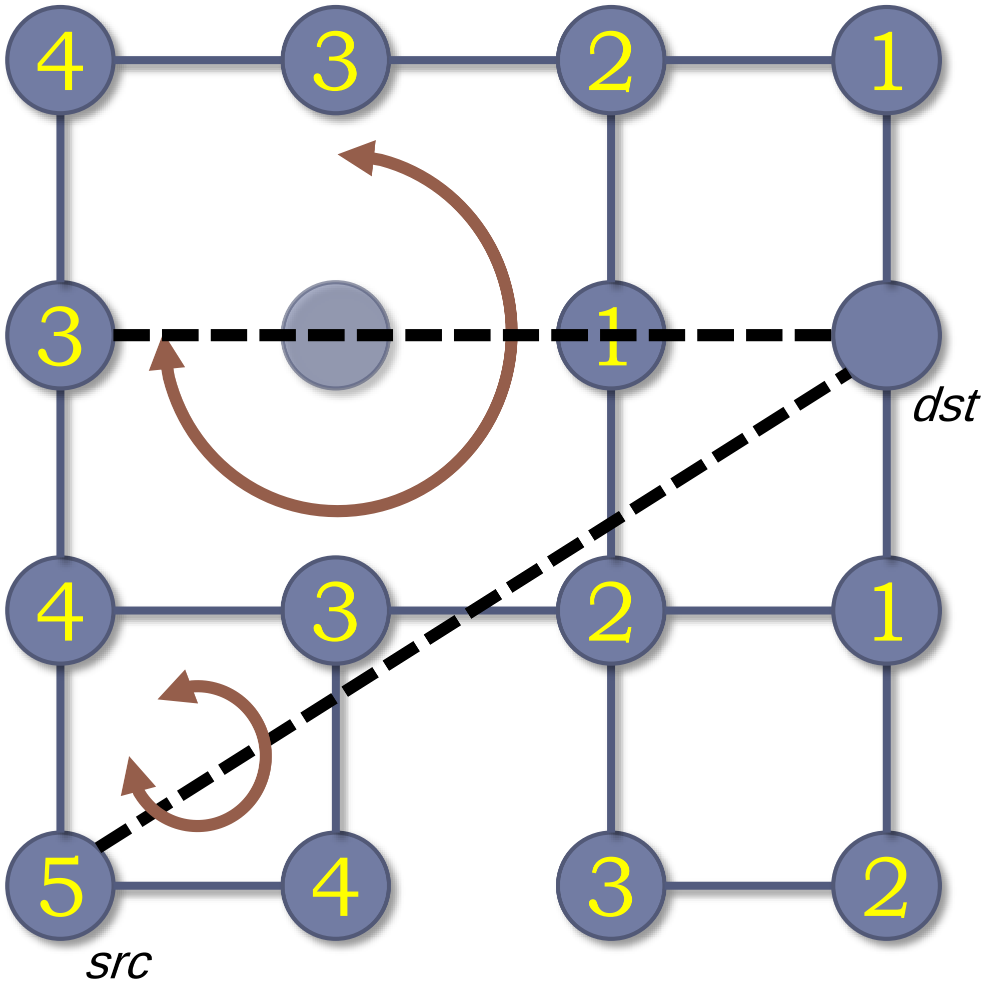
Delivery Proof

- Property: Given there is a path between *src* and *dst*, starting from *src*, by traversing the face underlying line$_{(src,dst)}$, the packet will definitely intersect the line at some point (p) other than *src*

- The MD(p,*dst*) is definitely smaller than MD(src,*dst*).

- In traversal mode: If $MD_{cur,\ dst} = MD_{best}$ with productive output?
  - Return to (and act as in) normal mode

➔ we definitely exit to normal mode



A Low-Overhead, Fully-Distributed, Guaranteed-Delivery Routing Algorithm for Faulty Network-on-Chips

Mohammad Fattah[1], Antti Airola[1], Rachata Ausavarungnirun[2], Nima Mirzaei[3], Pasi Liljeberg[1], Juha Plosila[1], Siamak Mohammadi[3], Tapio Pahikkala[1], Onur Mutlu[2] and Hannu Tenhunen[1]

Turun yliopisto
University of Turku

Carnegie Mellon University