Achieving Lightweight Multicast in Asynchronous Networks-on-Chip Using Local Speculation

Kshitij Bhardwaj **Steven M. Nowick**

Dept. of Computer Science
Columbia University

2016 ACM/IEEE Design Automation Conference (DAC), Austin, TX

Motivation for Networks-on-Chip

- Future of computing is many-core
  - 8 to 22 cores widely available: Intel 22-core Xeon-E5 2699 series
  - Expected progression: hundreds or thousands of cores
- NoC separates communication and computation
  - Improves scalability
    - global interconnects have high latency and power consumption (e.g. buses and point-to-point wiring)
  - Increases performance/energy efficiency
    - share wiring resources between parallel data flows
  - Facilitates design reuse
    - optimized IPs can simply plug in → considerably decrease design efforts
- Key challenge for NoCs = support for new traffic patterns
  - Support communication patterns for advanced parallel architectures
  - Compatibility with emerging technologies for NoCs:
    - wireless, photonics, CDMA

Multicast (1-to-Many) Communication

- Sending packets from one source to multiple destinations
- Widely-used in parallel computing: 3 key applications
  - Cache coherence: sending write-invalidates to multiple sharers
    - For *Token Coherence* protocol, 52.4 % of injected traffic is multicast
  - Shared-operand networks: operand delivery to multiple processors
  - Multi-threaded applications: for barrier synchronization
    - [Jerger/Lipasti et al., "Virtual circuit tree multicasting: a case for on-chip hardware multicast support," ISCA-08]
- Additional applications: multicast in emerging technologies
  - Wireless: mixed wire + millimeter-wave (or surface-wave)
  - Nano-photonics: support for energy-efficient optical broadcast
  - Large-scale neuromorphic CMPs: multicast between 1000s of neurons
- Key challenge for NoCs: performance/energy-efficient multicast

Asynchronous Design: Potential Advantages

- Lower power
  - No clock power
  - Energy-proportional computing: on-demand operation
  - Less overall power than deeply clock-gated sync counterpart
- Comparison with synchronous NoC router [in 40 nm technology]
  - 71% area reduction
  - 39% lower latency, comparable throughput
  - 44% lower energy/flit
    - [Ghiribaldi/Bertozzi/Nowick, "A transition-signaling bundled data NoC switch architecture for cost-effective GALS multicore systems," DATE-13]
- Industrial uptake of asynchronous NoCs

  IBM's TrueNorth neuromorphic chip
  - *5.4 billion* transistors, <u>fully-asynchronous chip</u>, consuming only 63 mW
  - 4096 neurosynaptic asynchronous cores modeling 1 million neurons
  - connected using <u>fully-asynchronous NoC</u>
    - [Merolla et al,. "A million-spiking neuron integrated circuit with a scalable communication network and interface," Science (Aug. 2014), COVER STORY]

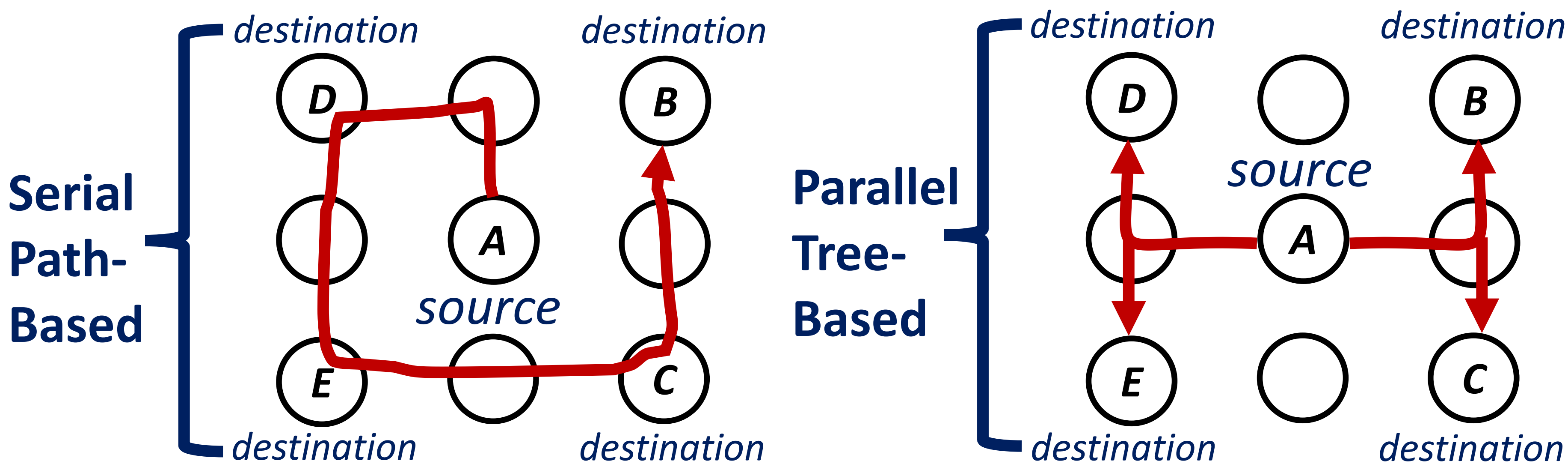
Related Work: Techniques for Multicast

1) Path-based serial multicast [Ebrahimi/Daneshtalab/Tenhunen IEEE TC-14]

- Packet routed to first destination, from there to next, and so on
- Expensive if large number of destinations – latency overheads

2) Tree-based parallel multicast: high-performance, widely-used

- First route packet on a common path from source to all destinations
  - When common path ends, replicate packet and diverge
- Earlier works set up tree in advance using multiple unicasts [Jerger/Lipasti ISCA-08]
- Recent works do not use unicast-based set up: tree constructed dynamically
  - [Krishna/Reinhardt MICRO-11]



Major Contributions

1) First general-purpose asynchronous NoC to support multicast

- Initial solution: uses simple tree-based parallel multicast

2) Novel strategy called *Local Speculation* for parallel multicast

- Always broadcast at subset of very fast speculative routers
- Neighboring non-speculative routers:
  - Quickly **throttle** misrouted packets from speculative nodes
  - Correctly route the other packets based on **source-routing** address
- New multicast protocol ➡ relaxed variant of tree-based multicast

3) New hybrid network architecture

- Mixes speculative and non-speculative routers
- 17.8-21.4% improvement in network latency
  - over basic non-hybrid tree-based solution

4) Additional contributions:

- Two more architectures with extreme degrees of speculation:
  - *no speculation* and *full (global) speculation*
- Router-level protocol optimizations for multi-flit packets
  - Further improve power and performance

Variant Mesh-of-Trees Topology

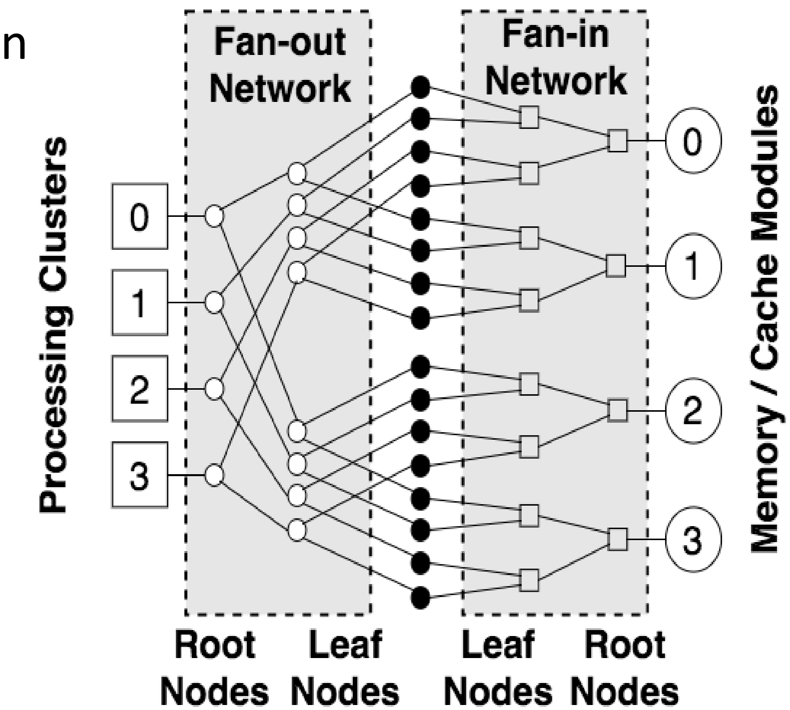
- Variant MoT: contains *two binary trees*
  - Fanout tree: *1-to-2 routing nodes*
  - Fanin tree: *2-to-1 arbitration nodes*
- Recently used for core-to-cache network
  - In shared memory parallel processors
- Several advantages of variant MoT:
  - Small hop count from source to destination
    - constant: log (n)
  - Unique path from source to destination
    - Minimize network contention
    - Challenge: lack of path diversity

      Can be bottleneck for unbalanced traffic
    - But overall, significant benefits for improved saturation throughput

      - [Horak/Nowick et al. "A low-overhead asynchronous interconnection network for GALS chip multiprocessor," TCAD-11]

[Balkan/Vishkin et al. "Layout-accurate design and implementation of a high-throughput interconnection network for single-chip parallel processing,"HOTI-07]

[Rahimi/Benini et al. "A fully-synthesizable single-cycle interconnection network for shared-L1 processor clusters," DATE-11]



Baseline Asynchronous NoC

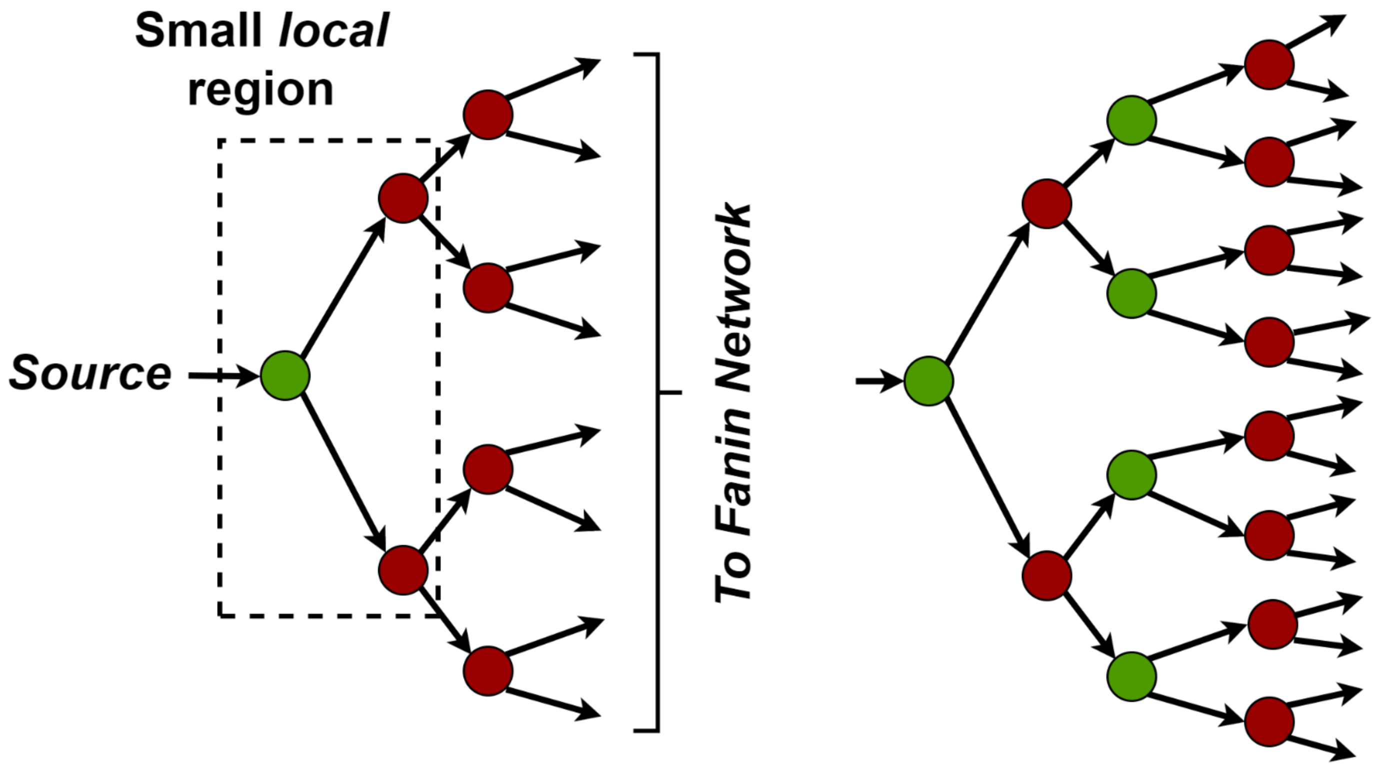
- New approach builds on recent async NoC: supports only unicast
  - [Horak/Nowick et al. "A low-overhead asynchronous interconnection network for GALS chip multiprocessor," TCAD-11]
- Comparison with synchronous 8x8 MoT network
  - Network latency: 1.7x lower (vs. 800 MHz synchronous)
  - Node-level metrics: significantly lower area, energy/packet than 1GHz sync
- Key design decisions: async communication + packet addressing
  - Uses 2-phase handshaking protocol instead of 4-phase
    - Only 1 round trip communication per data transfer
  - Data encoding: single-rail bundled data encoding
    - High coding efficiency and low area/power
  - Source routing: header contains address for every fanout node on its path
    - Allows simple fanout node
- Due to lack of multicast support
  - Multicast packet serially routed using multiple unicasts
- Our focus only on fanout nodes
  - Only fanout nodes will be modified to support parallel multicast
    - Enhancements to support parallel replication, new multicast addressing
  - No changes needed to fanin nodes for multicast: use baseline ones

Overview of Proposed Approach

Local Speculation: Basic Idea

- Goal of research
  - High-performance parallel multicast: improve latency/throughput
- Basic strategy = speculation
  - Fixed subset of fanout nodes are always speculative
    - Speculative nodes ➡ *always broadcast* every packet
      - *Lightweight, very fast:* no route computation or channel allocation steps
    - Novel approach: does not follow classic speculation
  - Hybrid network: non-speculative nodes surround speculative
  - Non-speculative nodes: always route based on address
    - Support parallel replication capability for multicast
    - Throttle any redundant copies received from speculative nodes
  - Redundant copies restricted to small *local* regions
- Net effect:
  - High performance due to speculation
  - Minimum power overhead due to local restriction

New Hybrid Network Architecture

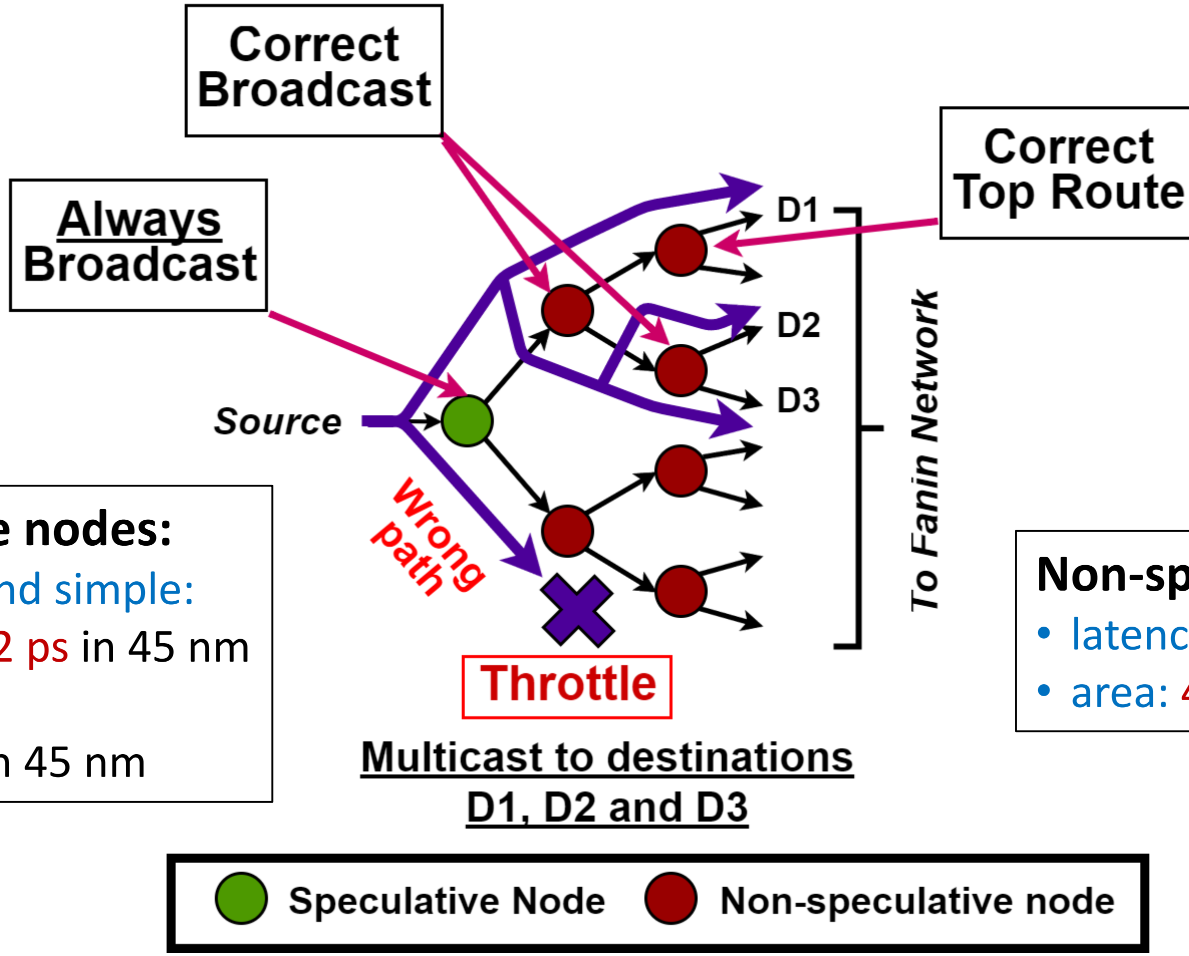


One possible hybrid fanout network for 8x8 MoT

One possible hybrid fanout network for 16x16 MoT

Speculative node   Non-speculative node

Local Speculation: Multicast Operation



Speculative nodes:
- very fast and simple: latency: 52 ps in 45 nm
- low area: 247 um^2 in 45 nm

Non-speculative nodes:
- latency: 299 ps in 45 nm
- area: 406 um^2 in 45 nm

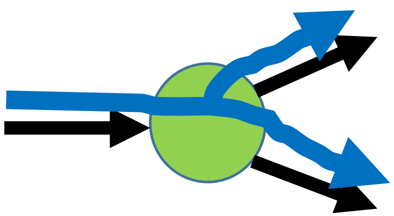
- **Similar operation for unicast traffic**
- Simplified source routing:
  - Only encode non-speculative nodes on paths to destinations
  - *No addressing* for speculative nodes: improves packet coding efficiency

Node-Level Protocol Optimizations

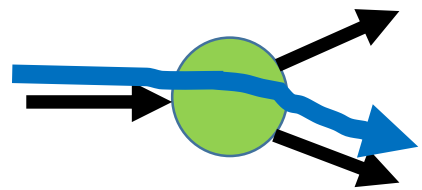
Optimize power and performance for <u>multi-flit packets</u>

1) <u>Speculative nodes</u> – extra power due to redundant copies
   - **Optimize power** ➡ switch to non-speculative mode for body flits
     - <u>After header:</u> no need for speculation as correct route known

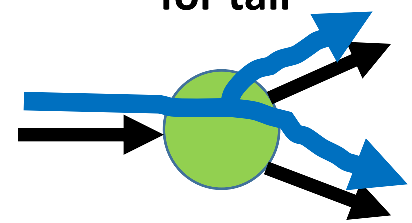
Speculative for head



<u>Switch</u> to non-speculative for body going to one port



<u>Back</u> to speculative for tail



2) <u>Non-speculative nodes</u> – slow, compute route + allocate channel per flit
   - **Optimize latency/throughput** using <u>channel pre-allocation</u>
   - Routing of head used to <u>pre-allocate</u> correct output channel(s) for body/tail
   - Body/tail <u>fast forwarded</u> after arrival

Header: Route Compute → Allocate Channel → Traverse Switch → Traverse Link

Body/Tail: Traverse Switch → Traverse Link

Experimental Results

Experimental Setup

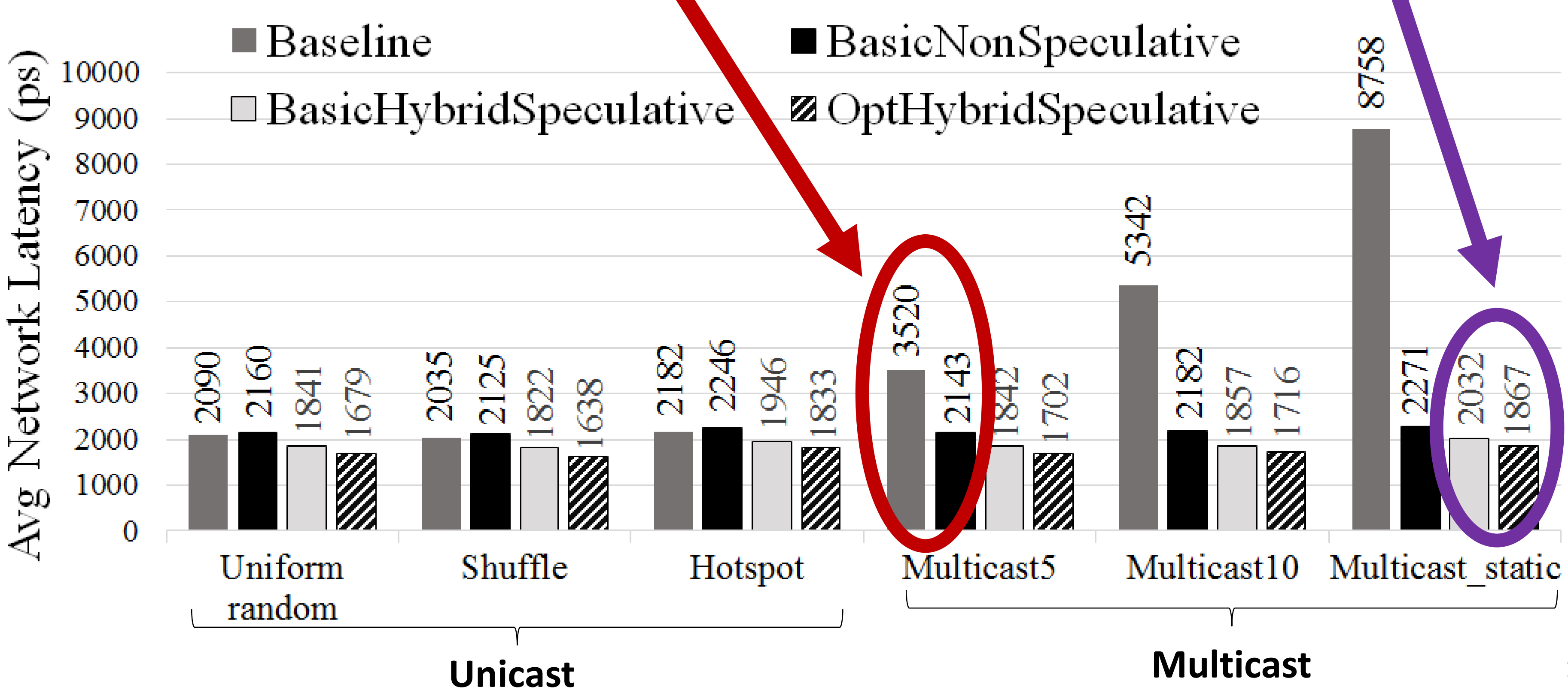
- Compare 5 new parallel multicast networks with serial *Baseline*
  - **BasicNonSpeculative:** tree-based multicast/unoptimized fanout nodes
  - **BasicHybridSpeculative:** local speculation/unoptimized fanout nodes
  - **OptNonSpeculative:** tree-based multicast/optimized fanout nodes
  - **OptHybridSpeculative:** local speculation/optimized fanout nodes
  - **OptAllSpeculative:** full (global) speculation/optimized fanout nodes
- Six 8x8 MoT networks: one for each configuration
  - Technology-mapped pre-layout implementation using structural Verilog
  - Implemented using FreePDK Nangate 45 nm technology
- Six synthetic benchmarks
  - 3 unicast: *Uniform Random (UR)*, *Bit Permutation*, and *Hotspot*
  - 3 multicast:
    - *Multicast5/10* – 5% or 10% of injected packets are multicast
    - *Multicast_static:* 3 sources perform multicast, remaining: UR unicast

Network Latency

- **Latency measured at 25% saturation load of respective network**
- **<u>Significant improvements</u> for *new hybrid* networks over *tree-based* and *Baseline***

39.1-74.1% improvement for *basic parallel tree-based* multicast over *Baseline serial*

***Additional* 10.5-21.4% improvement** using *local speculation* (unoptimized/optimized)

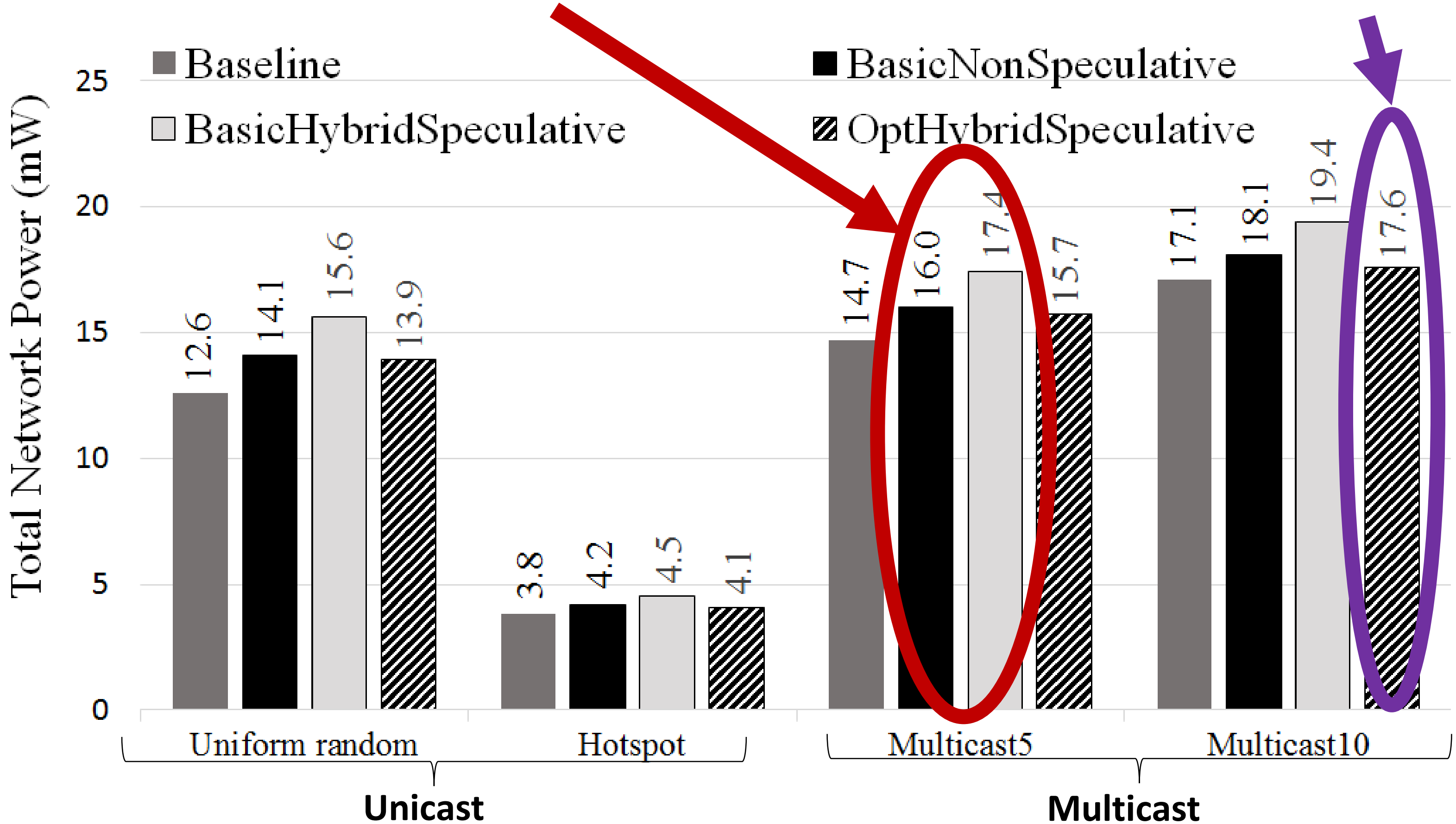


Network Power

- **Power measured at 25% saturation load of *Baseline***
- **Optimized network with local speculation (*OptHybridSpeculative)* has <u>minimal overhead</u> vs. *Baseline***

Both *tree-based and speculative (unoptimized)* incur moderate (5.8-23.8%) overhead over *Baseline*

Optimized speculative network *(OptHybridSpeculative)* **reduces this overhead** to 2.9-10.3% vs Baseline

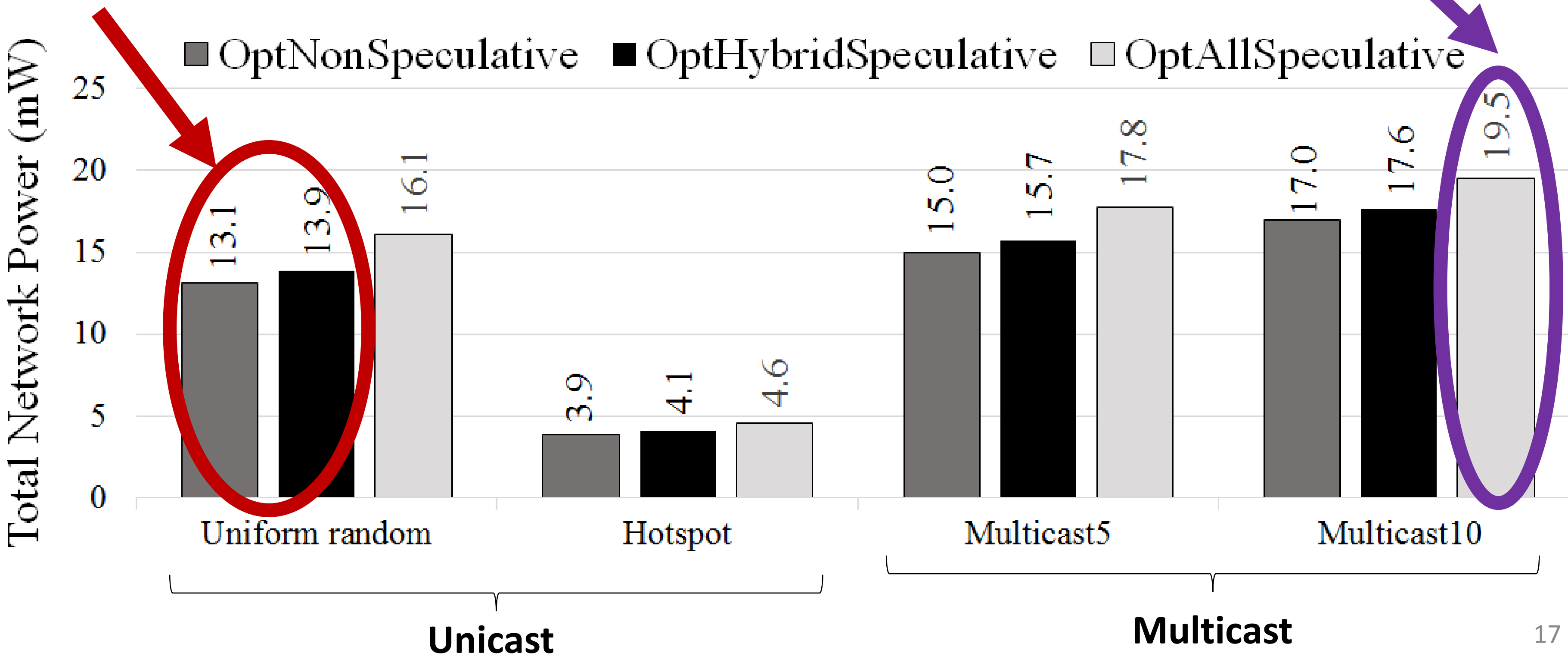


Different Degrees of Speculation: Effect on Network Power

- **Power measured at 25% saturation load of Baseline**
- **Fully-speculative network (*OptAllSpeculative)* incurs <u>significant overhead</u> due to global speculation**

Optimized network with *local speculation (OptHybridSpeculative)* has **almost the same power** as *non-speculative network*

However, fully-speculative network *(OptAllSpeculative)* incurs 14.7-22.9% extra power over non-speculative network



Conclusions and Future Work

- New parallel multicast approaches for asynchronous NoCs
  - *First general-purpose asynchronous NoC to support multicast*
  - New routing strategy called *Local Speculation* for parallel multicast
    - Fixed high-speed speculative switches: *always broadcast*
      - Extremely simple and fast
    - Non-speculative switches: rapidly throttle incorrect traffic locally
      - Redundant copies restricted to neighboring *local regions*
  - New hybrid network architecture
    - Mixes speculative and non-speculative switches
- Experimental design-space exploration
  - New basic tree-based parallel multicast network achieves:
    - 39.1-74.1% latency reduction over unicast-based serial multicast baseline
    - Incurs only small power overheads over serial multicast baseline
  - Additionally, new local speculation based hybrid network achieves:
    - 17.8-21.4% latency improvements over our basic tree-based solution
    - Small power reductions over our basic tree-based approach
- Future Work
  - Extend approach to larger MoTs, 2D-mesh topology, synchronous NoCs