

# Token Tenure: PATCHing Token Counting Using Directory-Based Cache Coherence

Arun Raghavan, Colin Blundell, Milo Martin

University of Pennsylvania

*{arraghav, blundell, milom}@cis.upenn.edu*



This work licensed under the Creative Commons



## Attribution-Share Alike 3.0 United States License

- **You are free:**

- to **Share** — to copy, distribute, display, and perform the work
- to **Remix** — to make derivative works

- **Under the following conditions:**

- **Attribution.** You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).
- **Share Alike.** If you alter, transform, or build upon this work, you may distribute the resulting work only under the same, similar or a compatible license.

- For any reuse or distribution, you must make clear to others the license terms of this work. The best way to do this is with a link to:

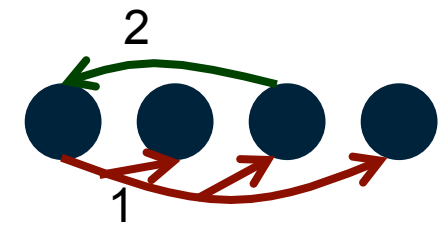
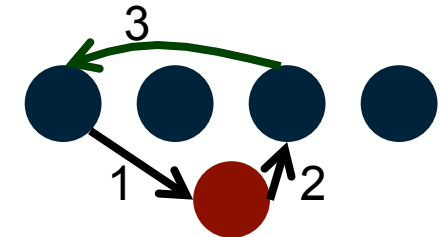
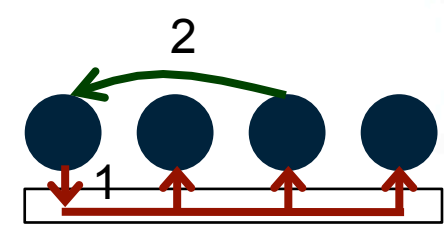
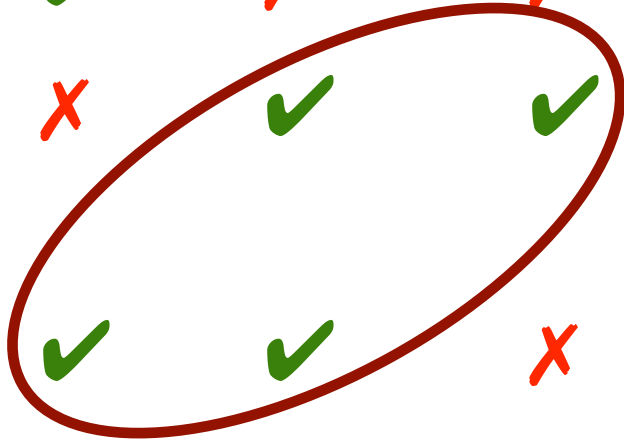
<http://creativecommons.org/licenses/by-sa/3.0/us/>

- Any of the above conditions can be waived if you get permission from the copyright holder.
- Apart from the remix rights granted under this license, nothing in this license impairs or restricts the author's moral rights.



# Why Yet Another Coherence Protocol?

	Fast sharing	Scalable interconnect	Avoids broadcast
<b>Snoopy</b>	✓	✗	✗
<b>Directory</b> •Track sharers	✗	✓	✓
<b>Token Coherence</b> •Token counting	✓	✓	✗
<b>Our goal</b>	✓	✓	✓



?

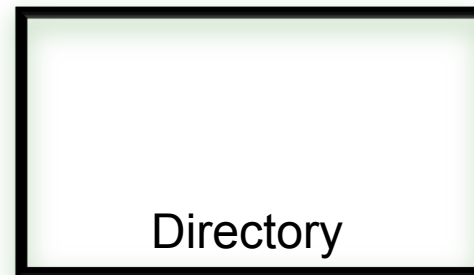
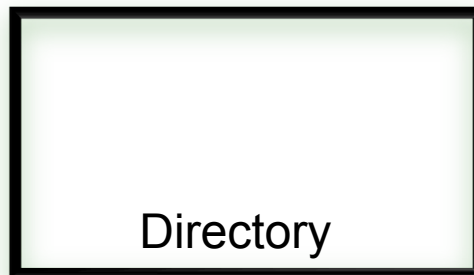
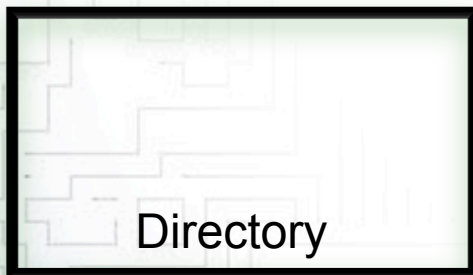
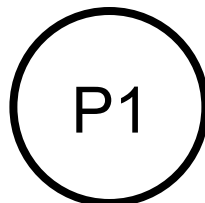
**This work: combining directory and token counting**

# Overview

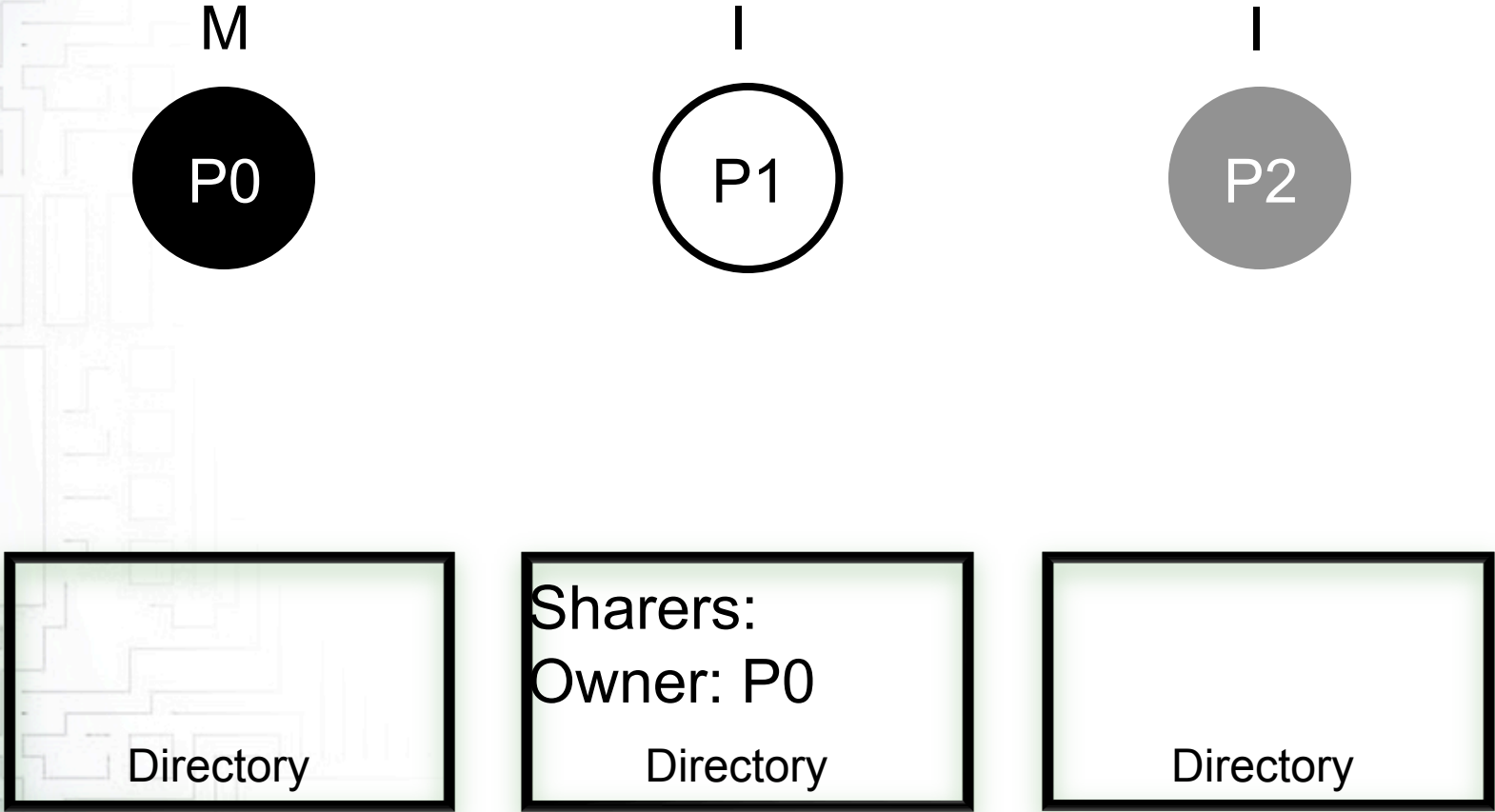
- Begin with a standard directory protocol
- Fast sharing misses? **Direct requests**
- Ensure safety? **Token counting**
- Broadcast-free forward progress? **Token Tenure**
  - Directory selects one requestor to retain tokens
  - Requestors give up tokens after a timeout interval
- **PATCH**: Predictive, Adaptive Token Counting Hybrid
  - Send request “hints” directly to predicted sharers
  - Retain scalability? Lowest-priority, best-effort delivery

**Fast sharing misses, scales as directory**

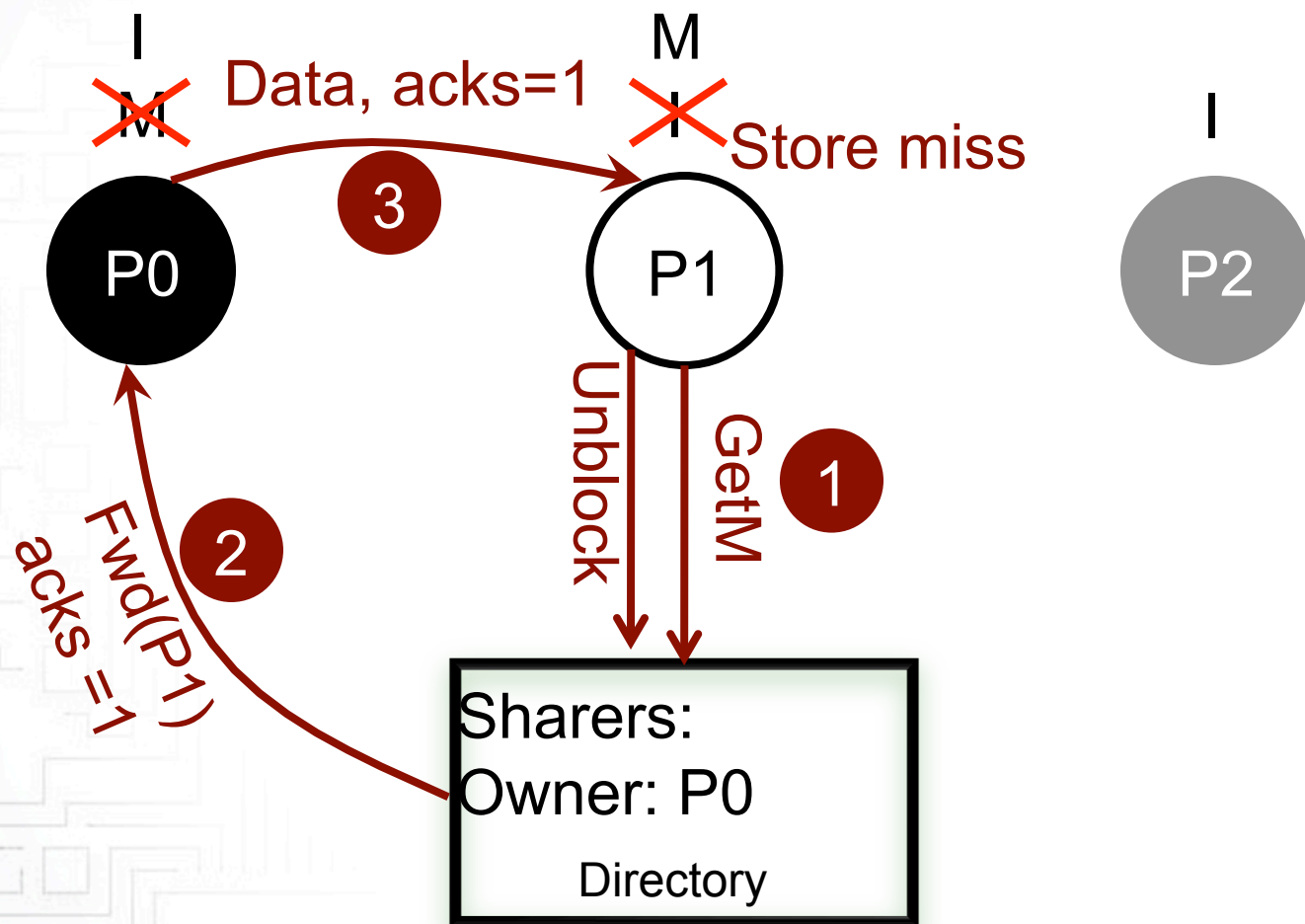
# Directory Operation



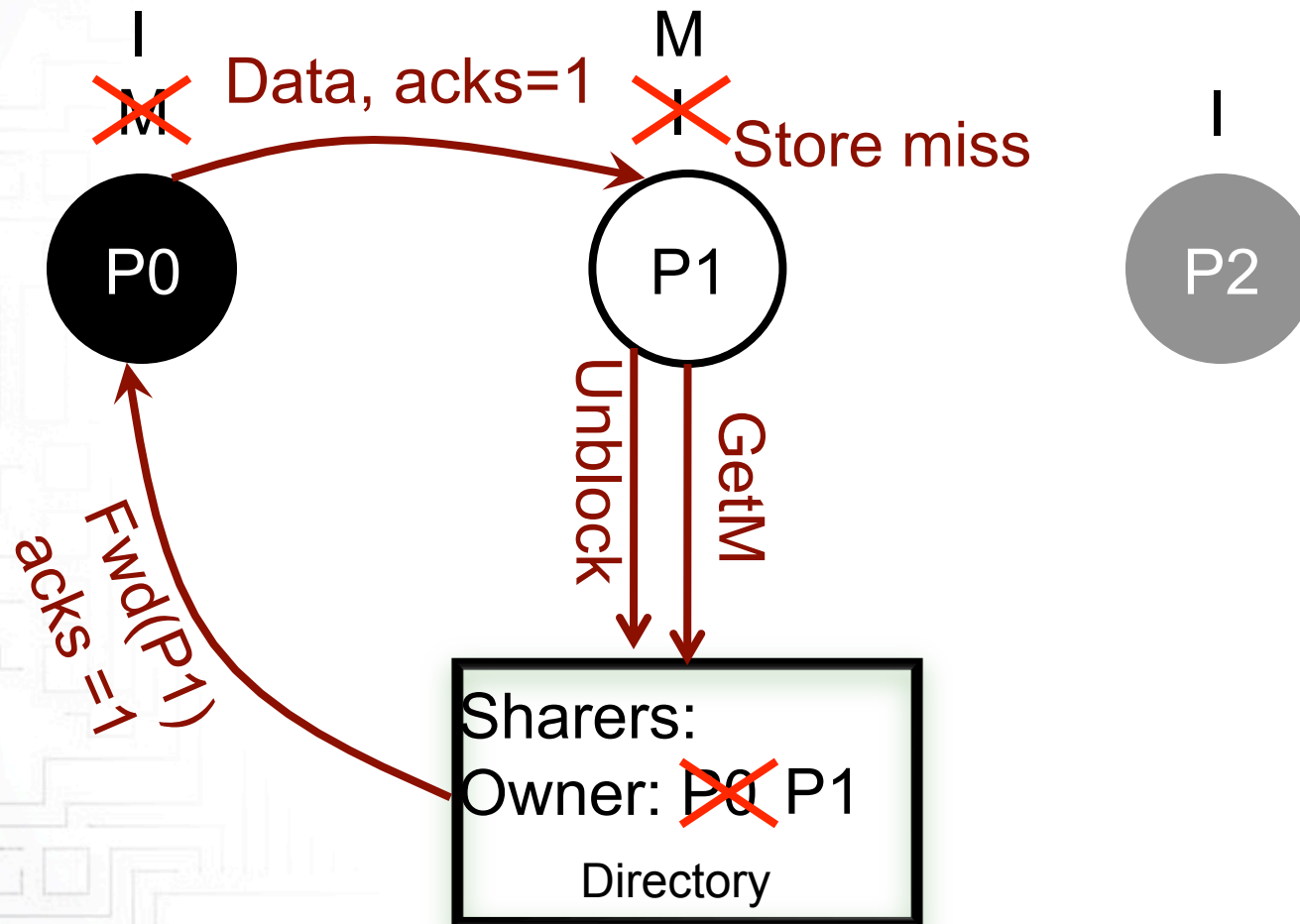
# Directory Operation



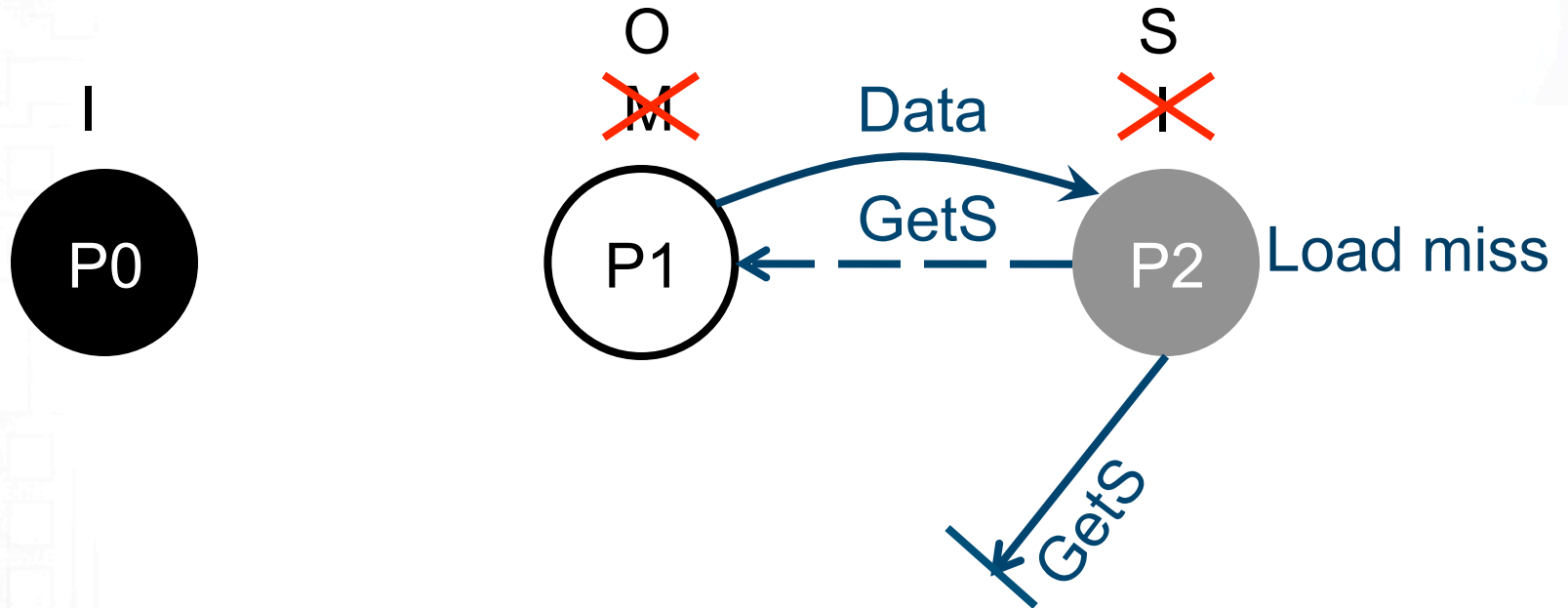
# Directory Operation



# Directory Operation

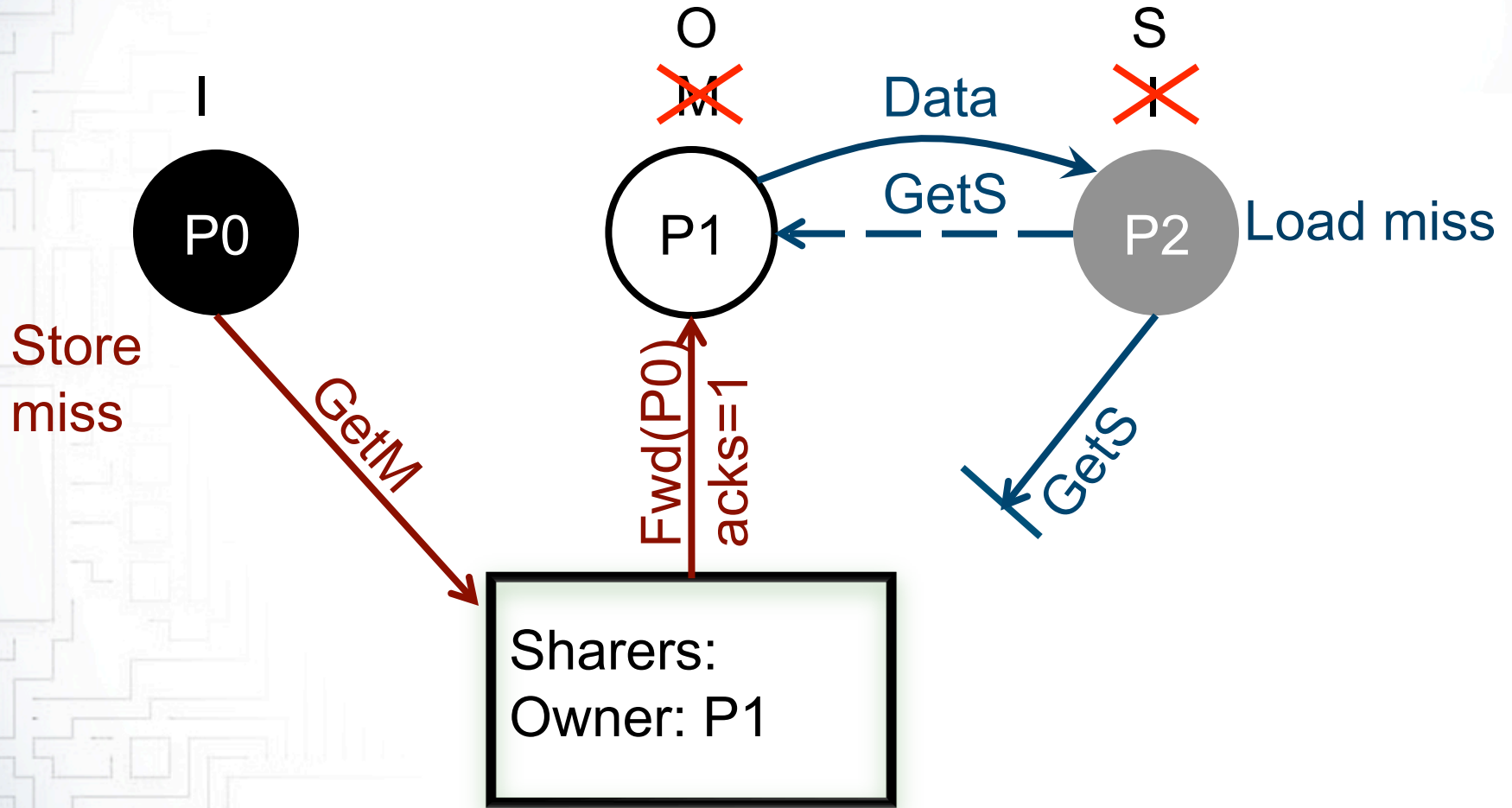


# Directory with Direct Requests?

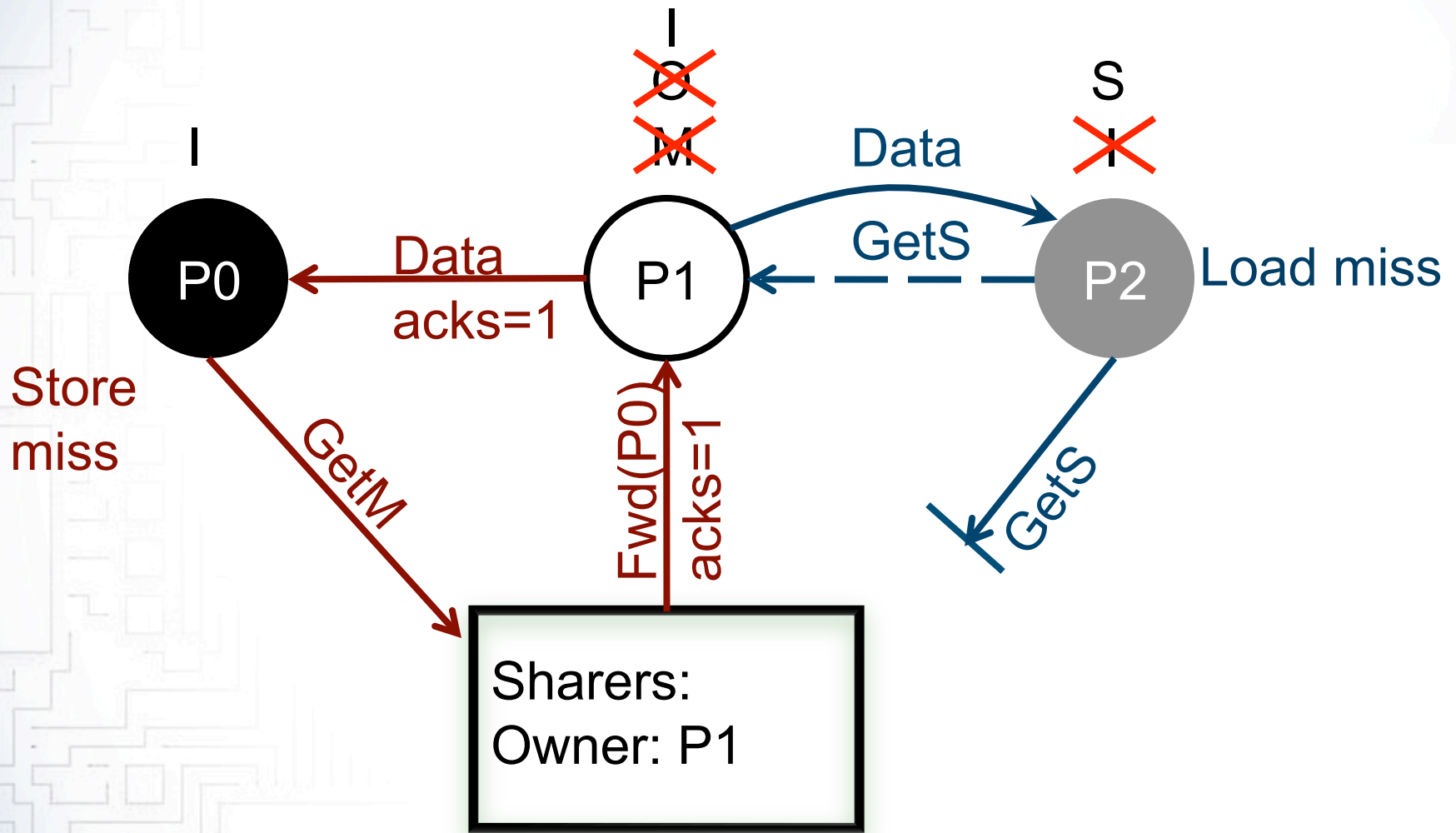


Sharers:  
Owner: P1

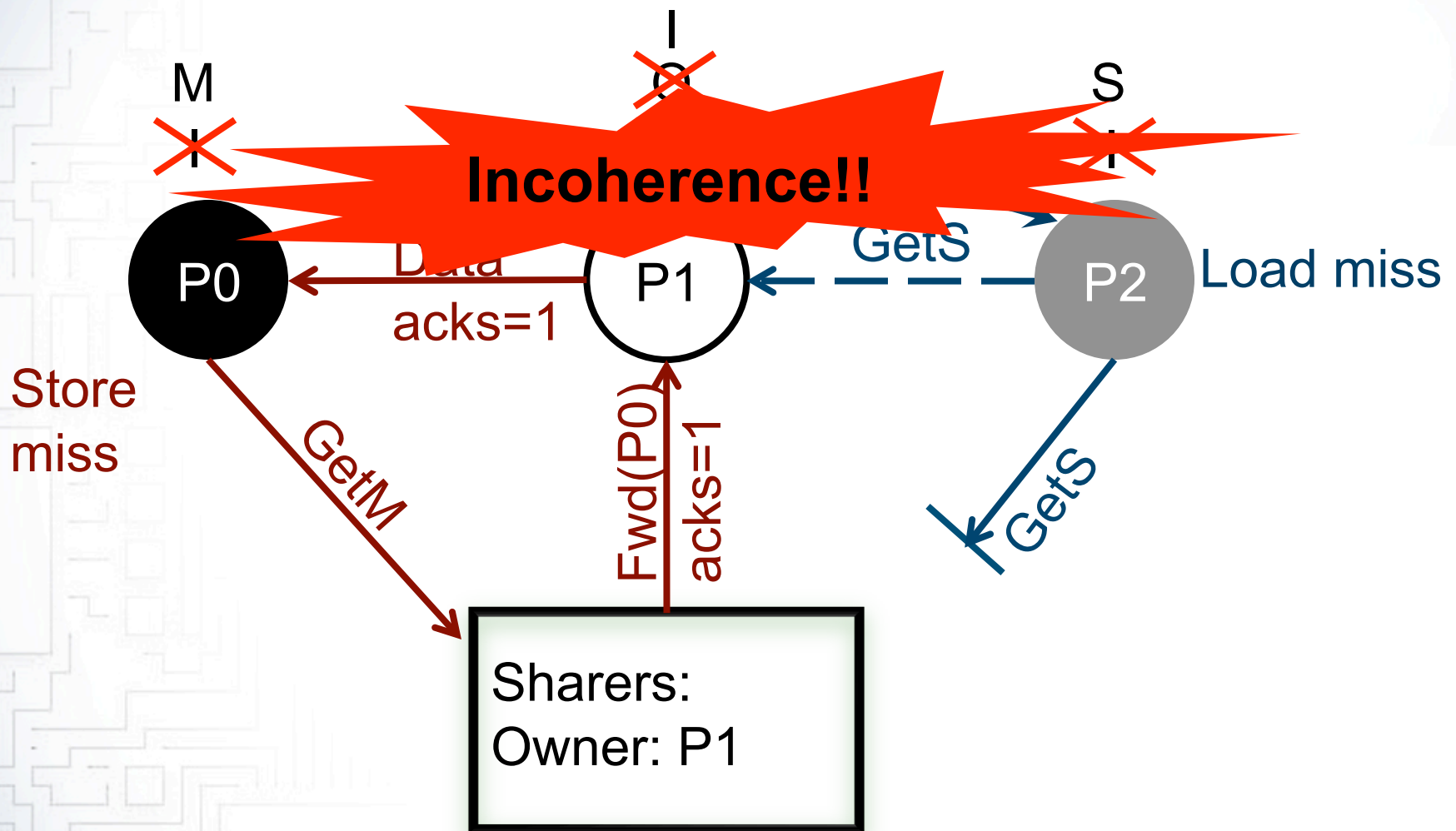
# Directory with Direct Requests?



# Directory with Direct Requests?



# Directory with Direct Requests?



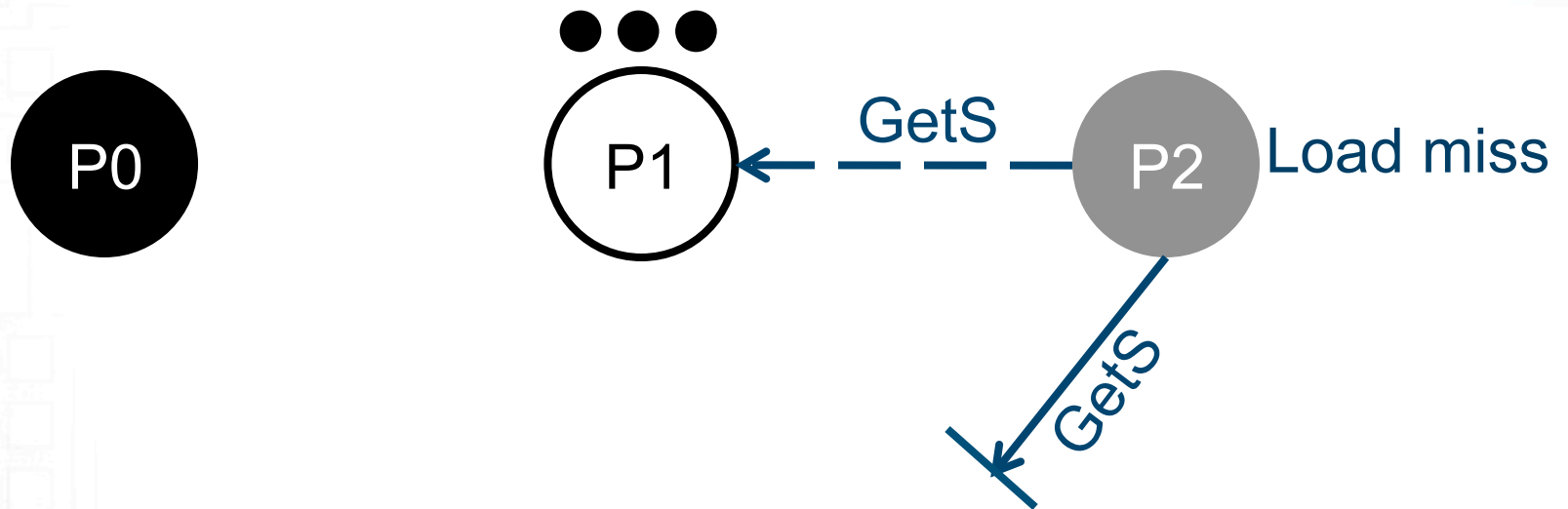
**Why? Direct requests break key directory assumption**

# Restoring Coherence

- Coherence invariant: **one writer or many readers**
- Directory: enforces implicitly by distributed algorithm
  - Assumes complete state information at the directory
- Alternative: encode permission with **token count**
  - Fixed number of tokens per cache block
  - **Need all tokens to write**
  - **One or more tokens to read**
- **Explicitly** enforces coherence invariant
  - Without regard to races, protocol details

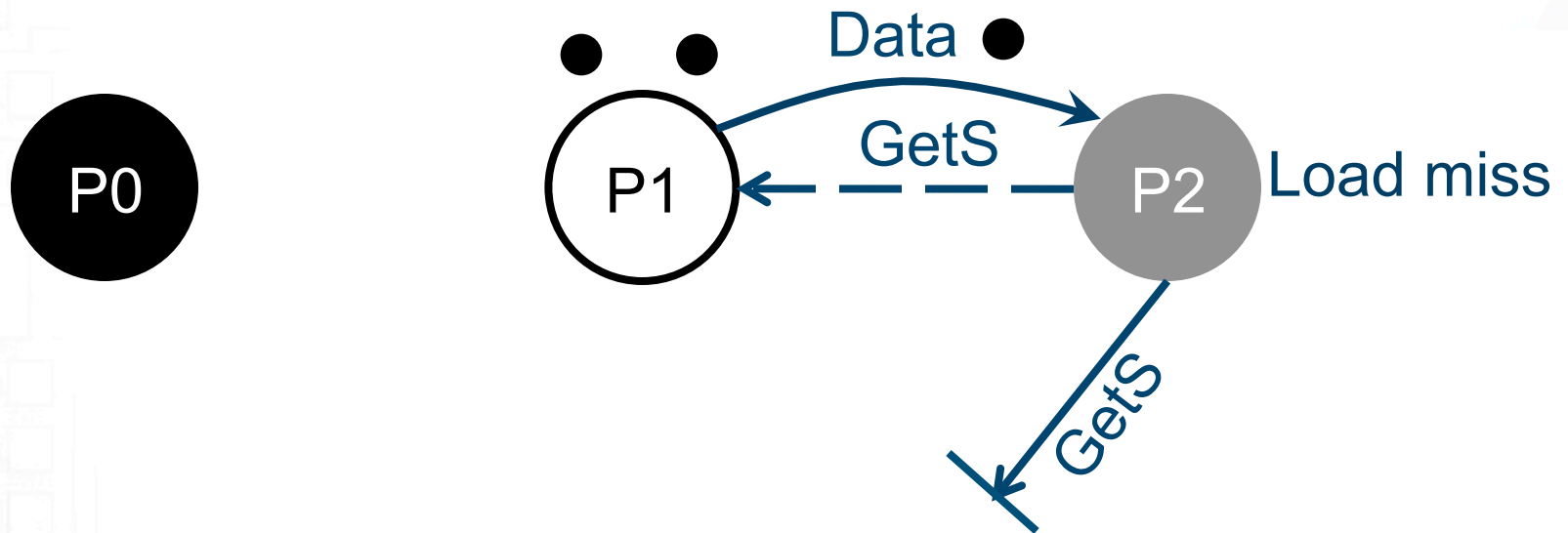
Token Coherence  
[ISCA '03]

# Directory with Direct Requests: Tokens



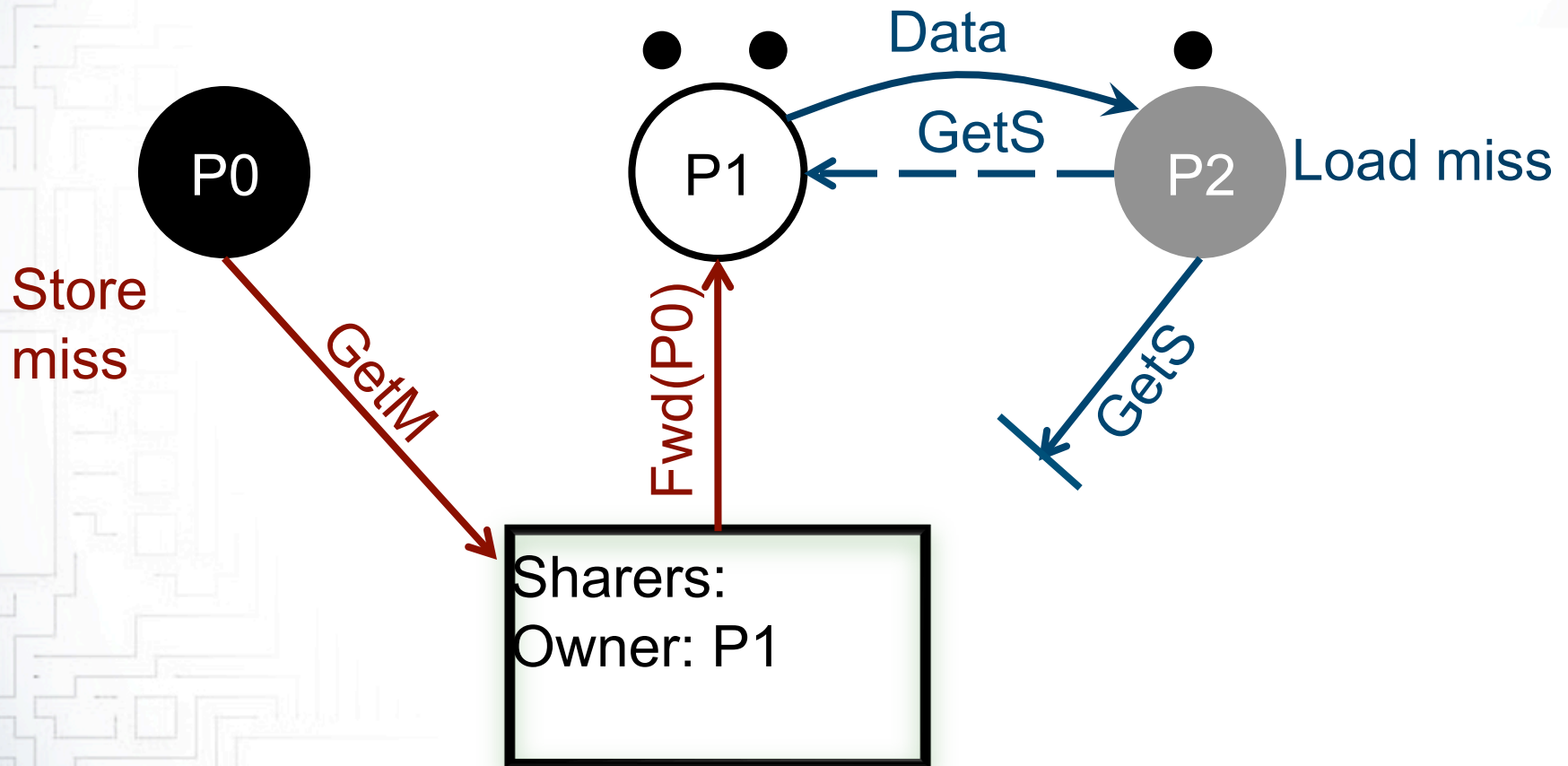
Sharers:  
Owner: P1

# Directory with Direct Requests: Tokens

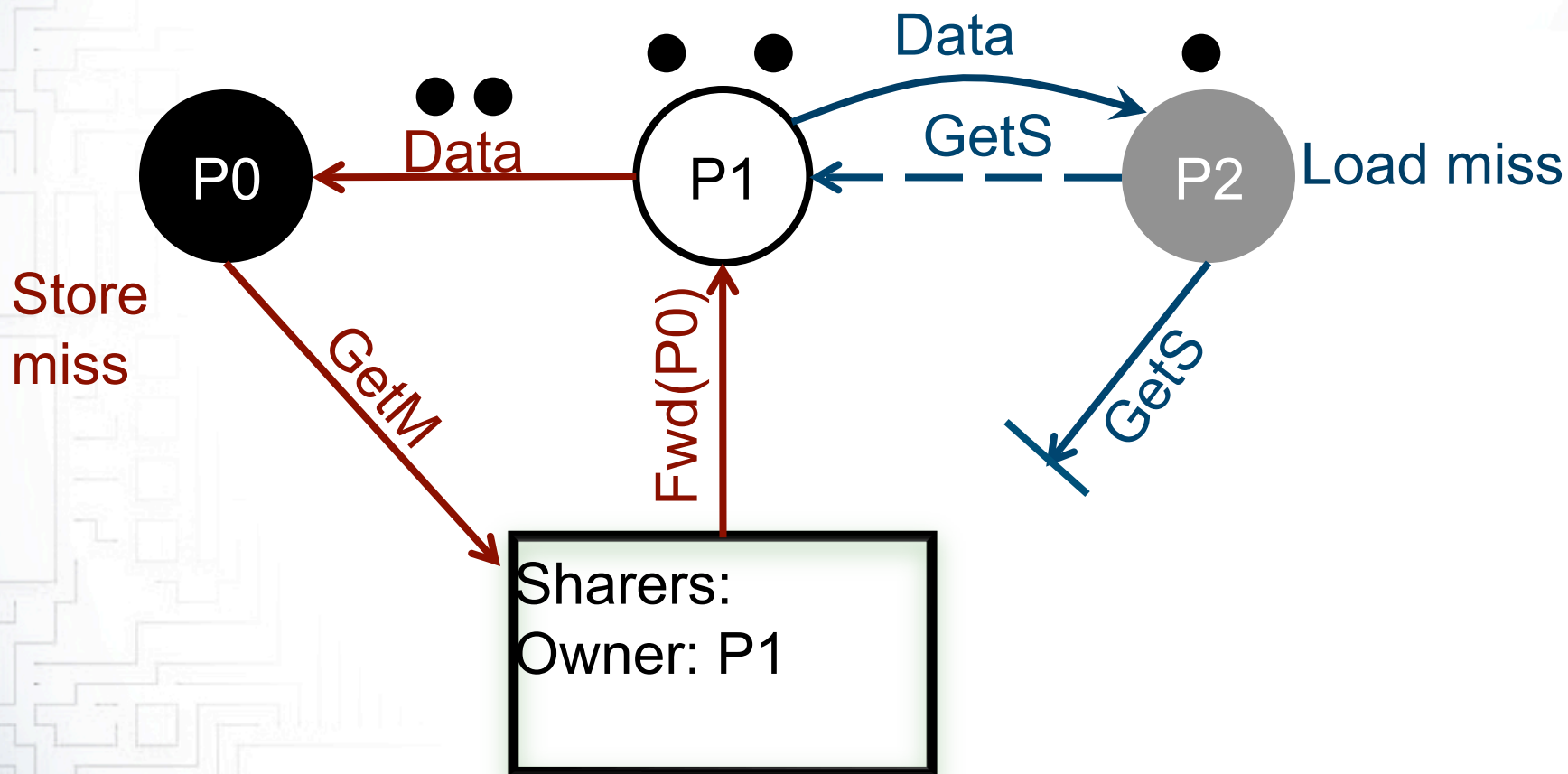


Sharers:  
Owner: P1

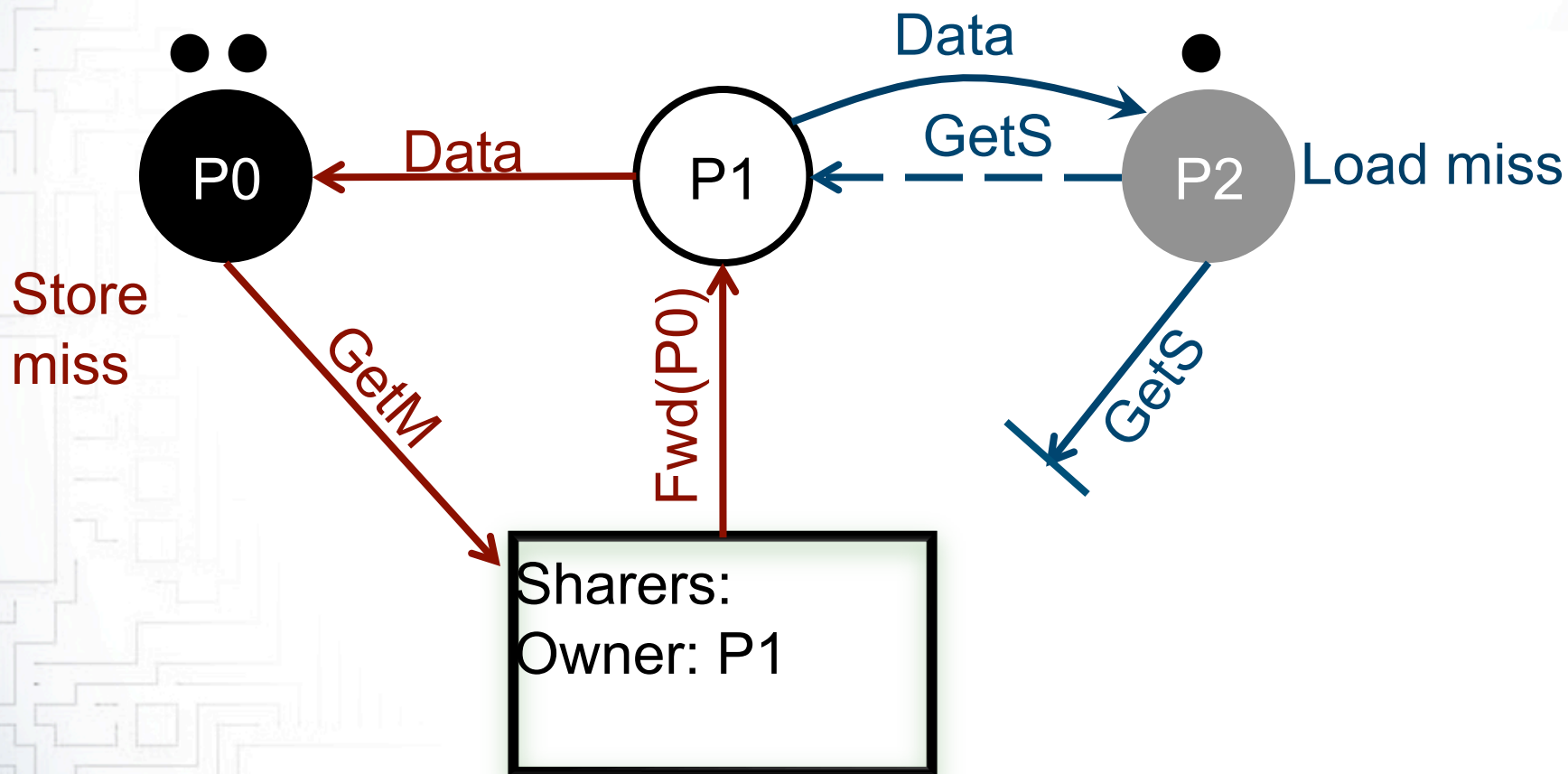
# Directory with Direct Requests: Tokens



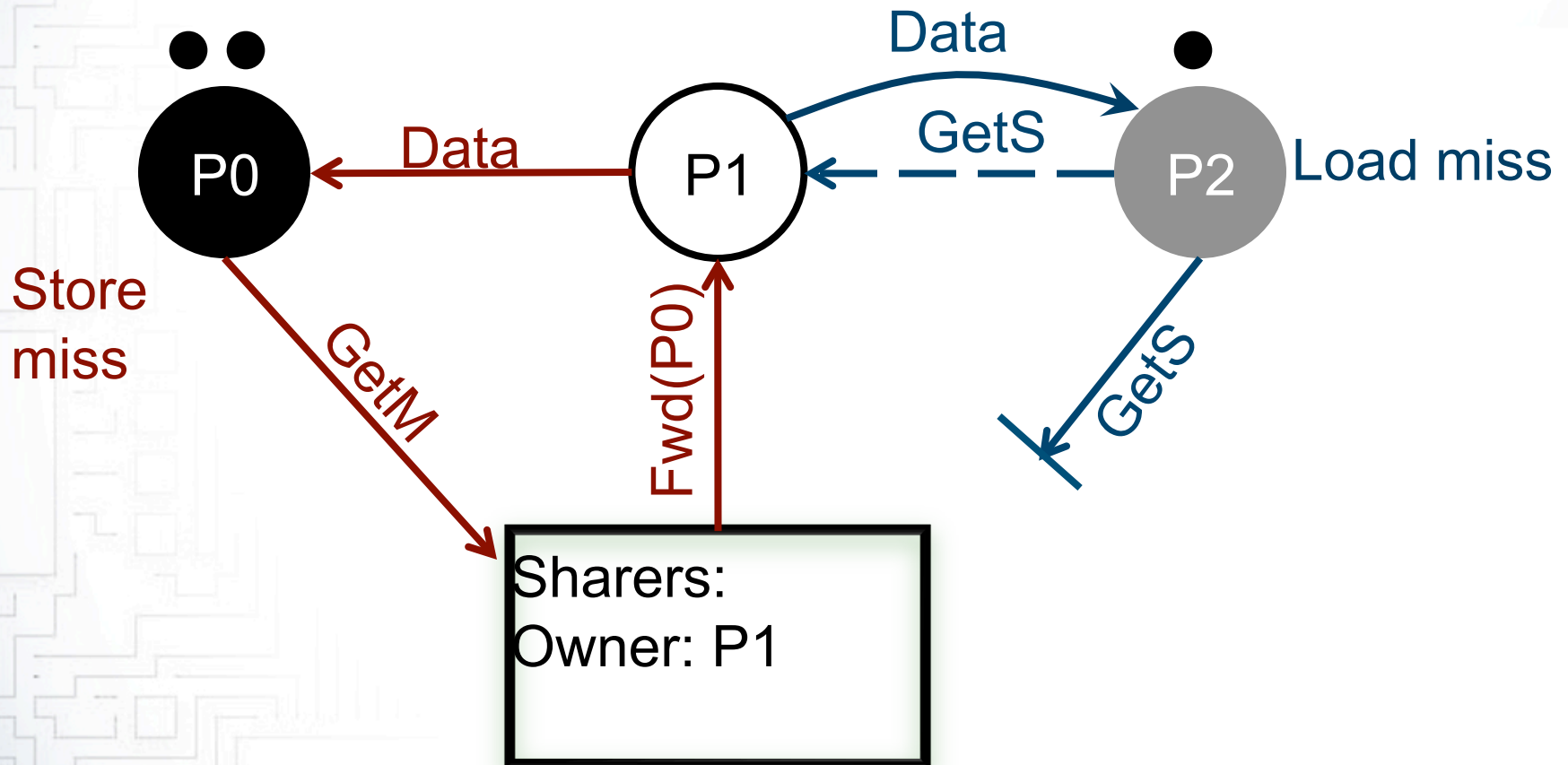
# Directory with Direct Requests: Tokens



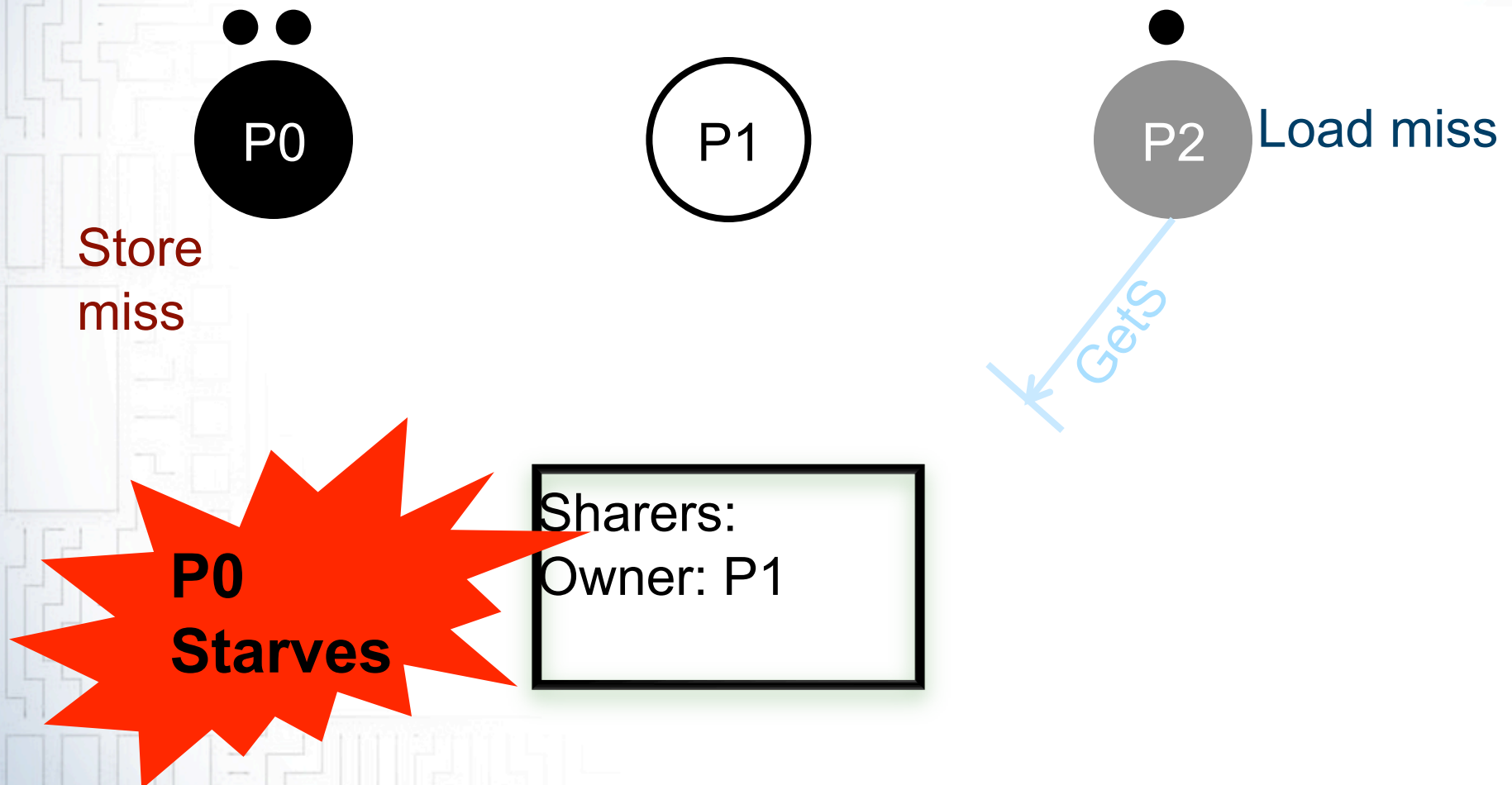
# Directory with Direct Requests: Tokens



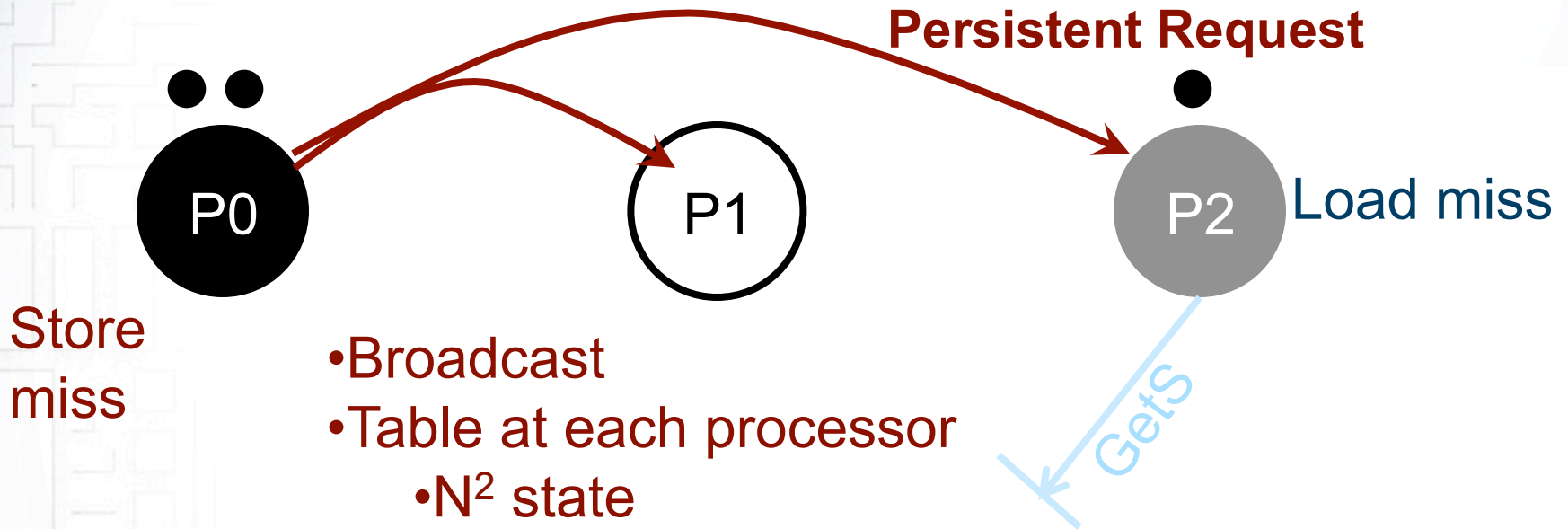
# Directory with Direct Requests: Tokens



# Directory with Direct Requests: Tokens



# Token Coherence Solution: Persistent Requests

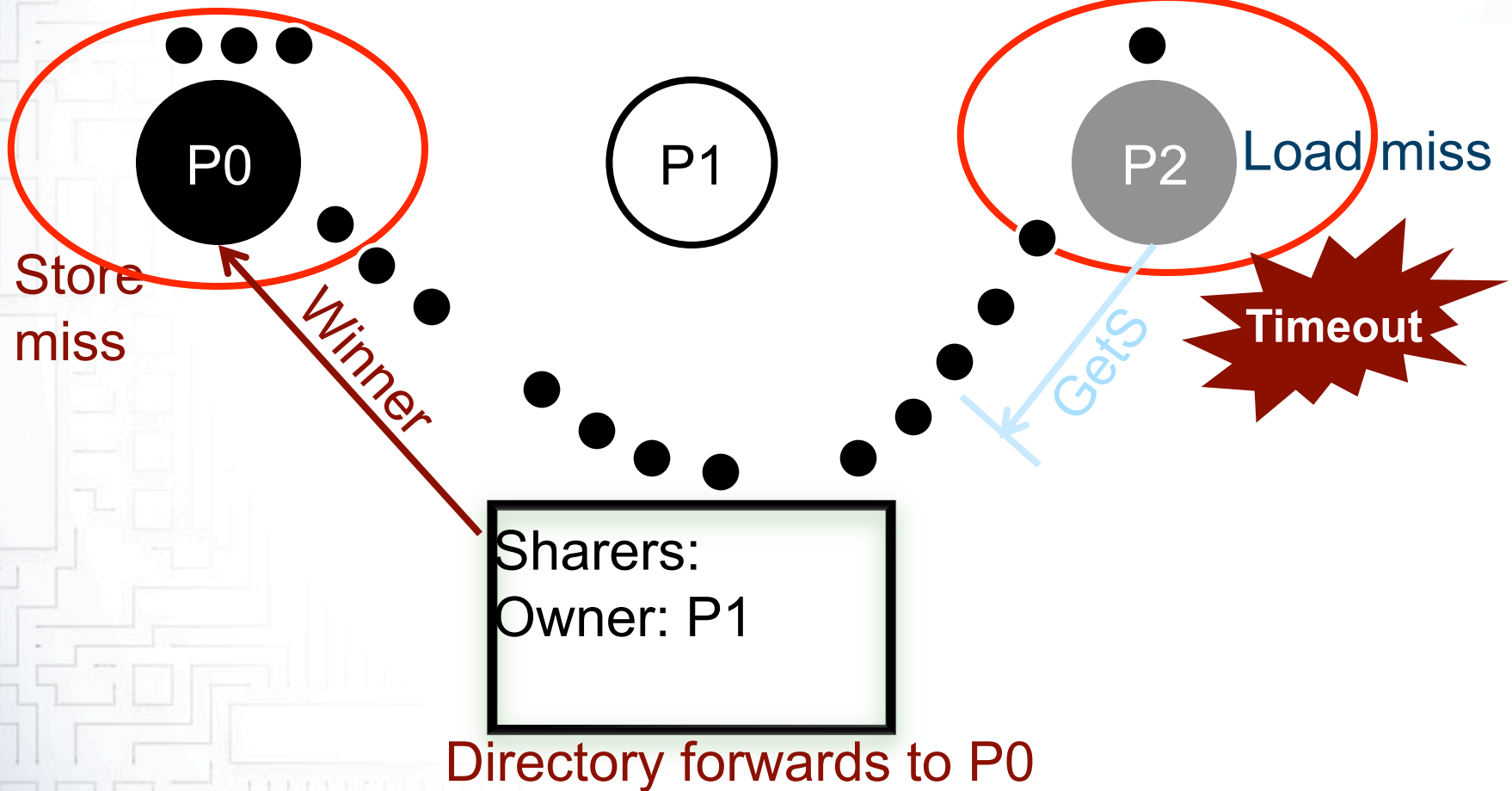


Sharers:  
Owner: P1

# Our Solution

P0's request reached directory first  
•Directory declares P0 winner

P2 non-winner  
•Inferred after timeout



# Token Tenure



UNIVERSITY of PENNSYLVANIA  
ARCHITECTURE + COMPILERS GROUP

PATCH - Arun Raghavan - MICRO 2008 [ 23 ]

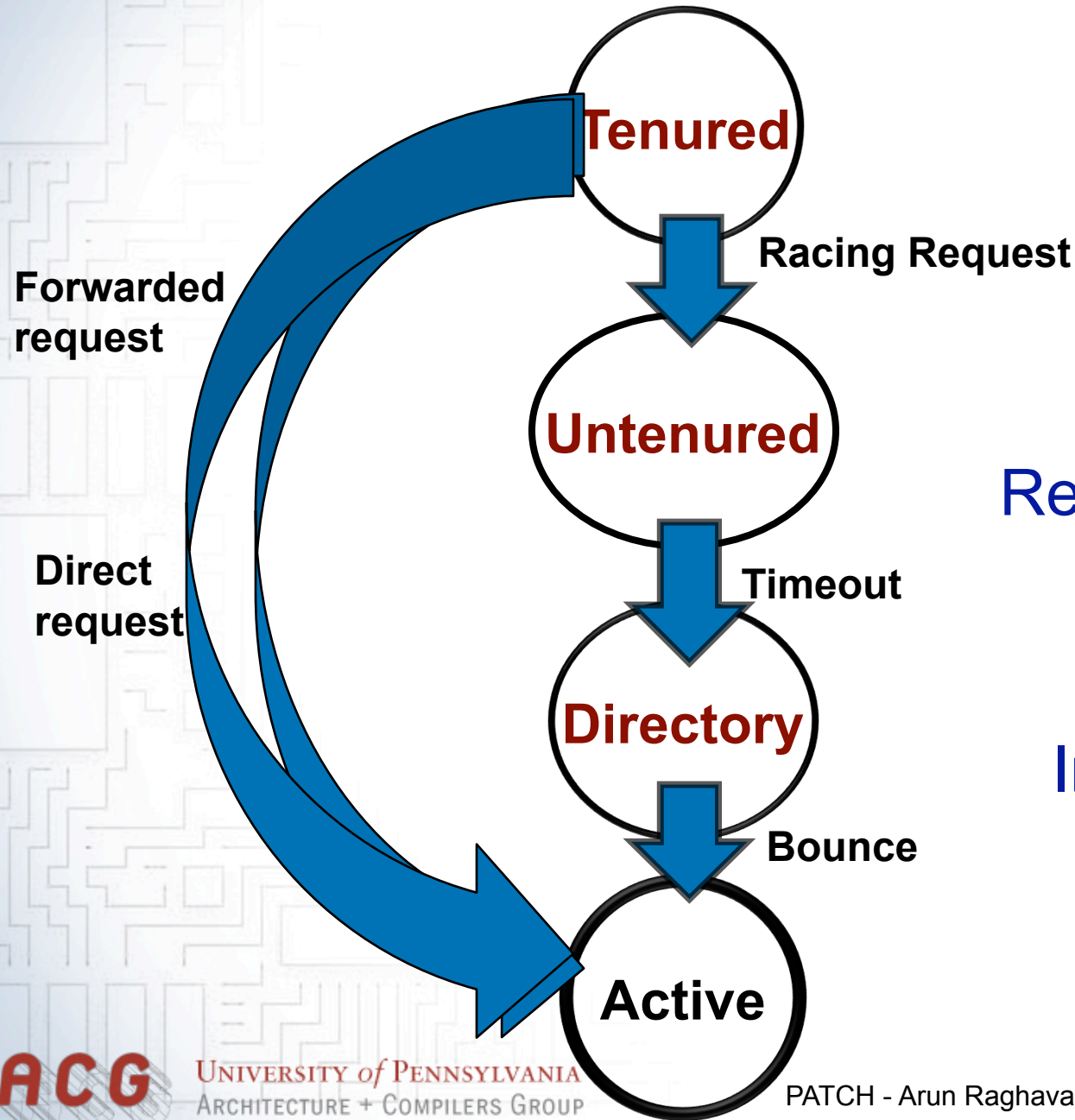


# Token Tenure

- Tokens can be **tenured** or **untenured**
  - **Tokens by default untenured**
  - Untenured tokens must be sent to the directory...
  - ... unless tenured within **timeout** window
- Active (winner) requestors tenure tokens
  - Directory activates one request at a time
  - Directory explicitly informs active requestor
- Multiple processors can hold tenured tokens

*Why does this ensure forward progress?*

# Flow of Tokens to Active Requestor



Restore directory's ability to resolve races

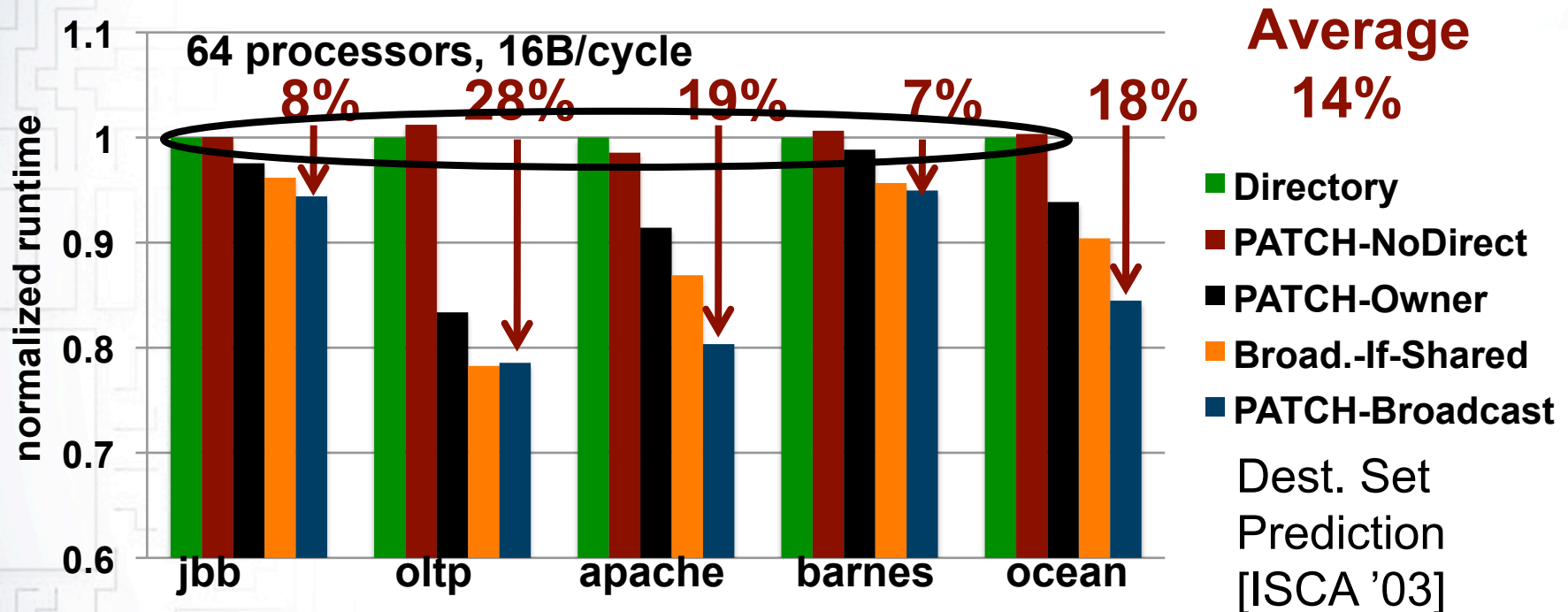
Implementation: add timeout

# Token Tenure: Implementation

- No common-case performance impact
- Activation off critical path of miss
  - Token count still determines permissions
- No additional traffic
  - Activation piggybacked on forwarded messages
- Set timeout to twice average roundtrip latency
  - Avoid early timeout....
  - ...but minimize slowing down winner in races

# Using Direct Requests

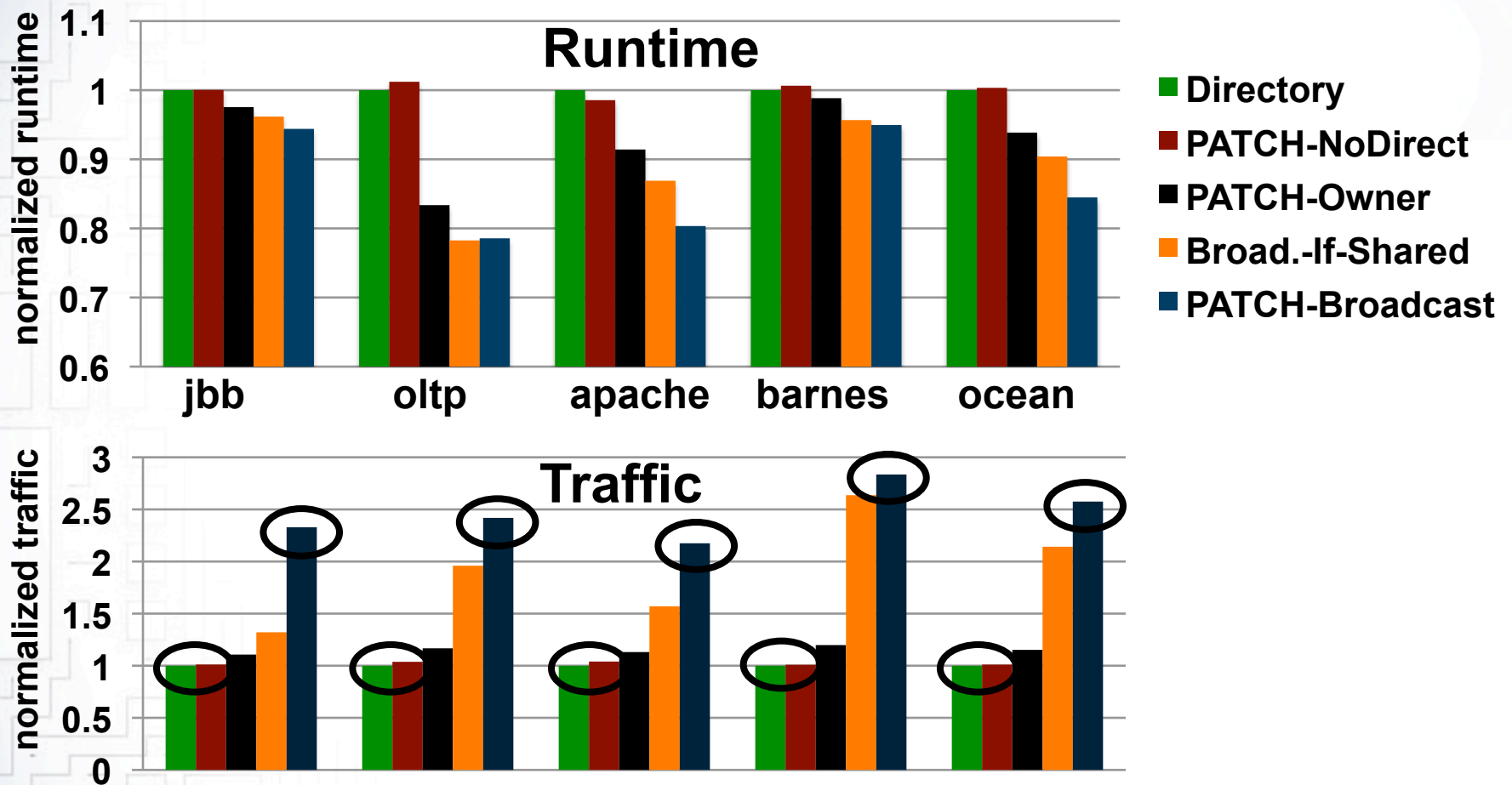
- Direct requests to **no**, **some** or **all** processors



**Direct requests improve performance**

**But at what cost?**

# Direct Requests: Runtime and Traffic



**PATCH-NoDirect and Directory have identical traffic**

**PATCH-Broadcast has >100% overhead**

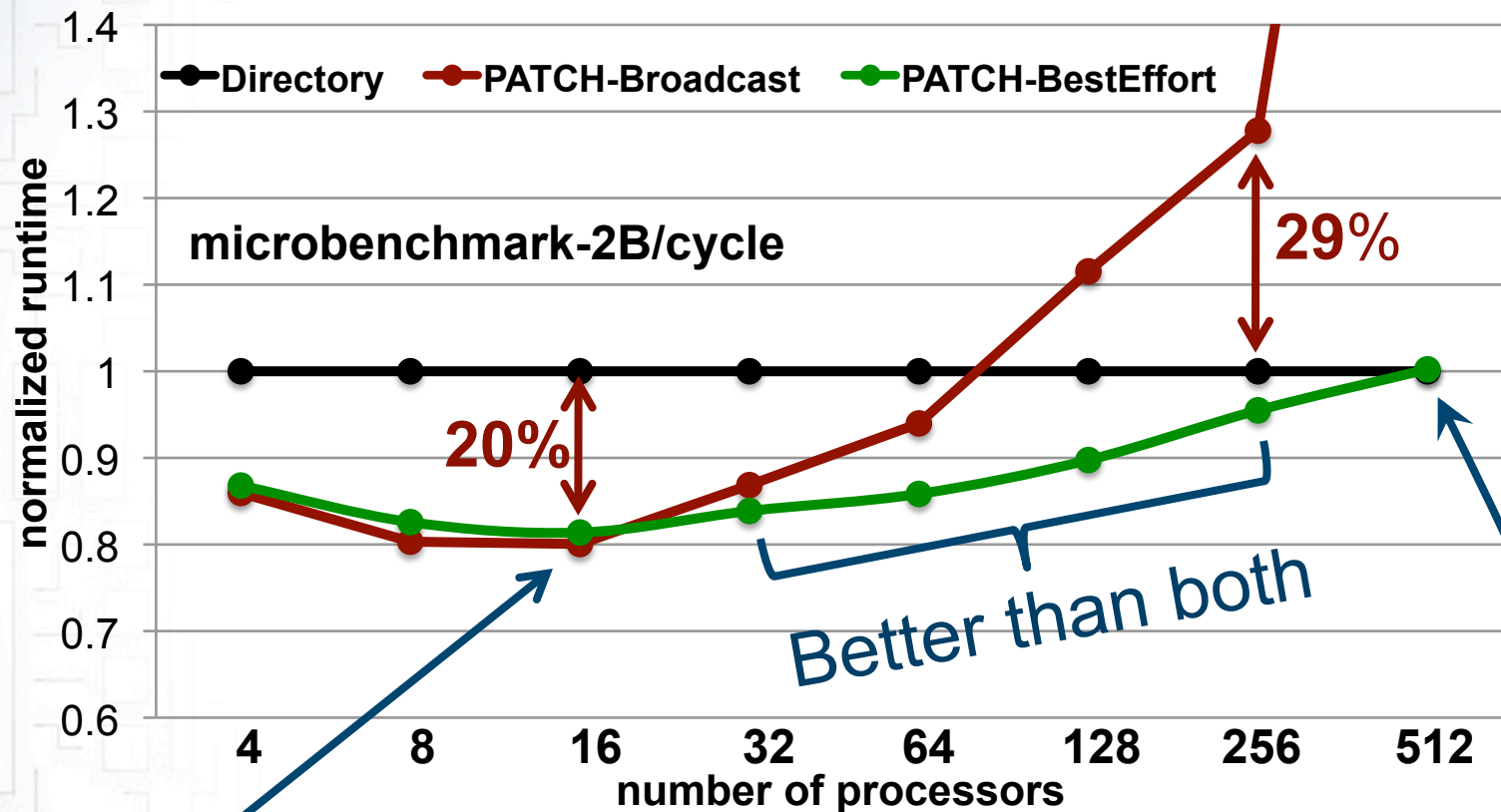
# Best-Effort Direct Requests

- Direct requests in PATCH
    1. Strictly in addition to directory requests
    2. Don't need explicit acks

→ direct requests can be dropped arbitrarily
  - **Best-effort delivery**
    - Lowest priority, deliver strictly on “do-no-harm-basis”
    - If queued up too long in switches, controller: drop

→ lower-bound: PATCH-NoDirect performance
  - **Adequate bandwidth?** drop no requests
  - **Scarce bandwidth?** drop all requests
- Never worse than directory**

# Best-Effort Direct Requests



Broadcast performance with plentiful bandwidth  
Converges with directory performance at 512

**Adapt dynamically; one-size-fits-all**

# Enhancing Directory Scalability

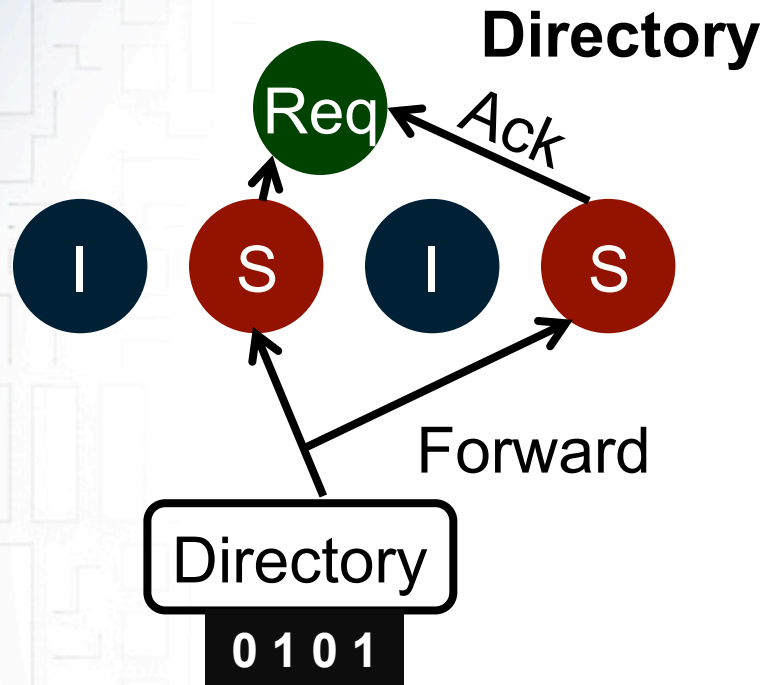


UNIVERSITY of PENNSYLVANIA  
ARCHITECTURE + COMPILERS GROUP

PATCH - Arun Raghavan - MICRO 2008 [ 31 ]

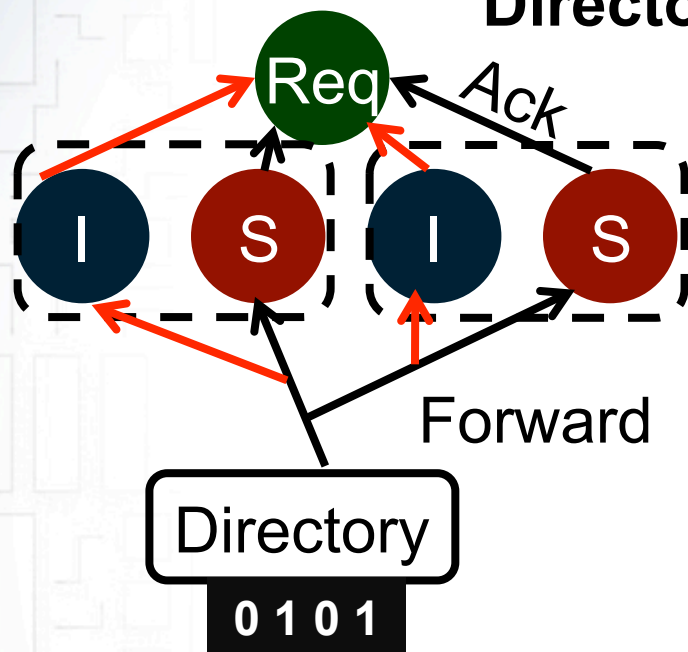


# Enhancing Directory Scalability



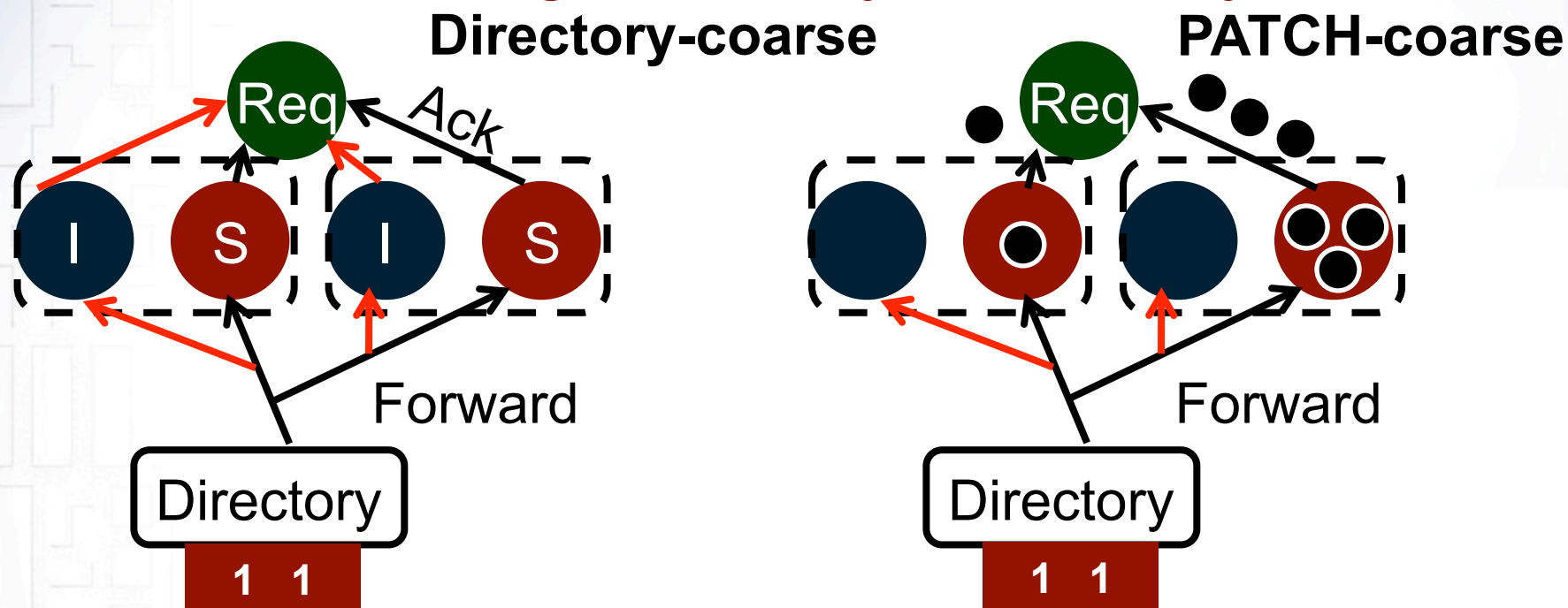
# Enhancing Directory Scalability

## Directory-coarse



- Coarse directories: 1-bit for k sharers
  - Fan-out delivery of forwards: worst case  $O(N)$  traffic
  - Requires acks from non-sharers too
    - Multiple unicast messages (no ack combining)
    - Worst case  $O(N\sqrt{N})$  on 2D torus interconnect

# Enhancing Directory Scalability

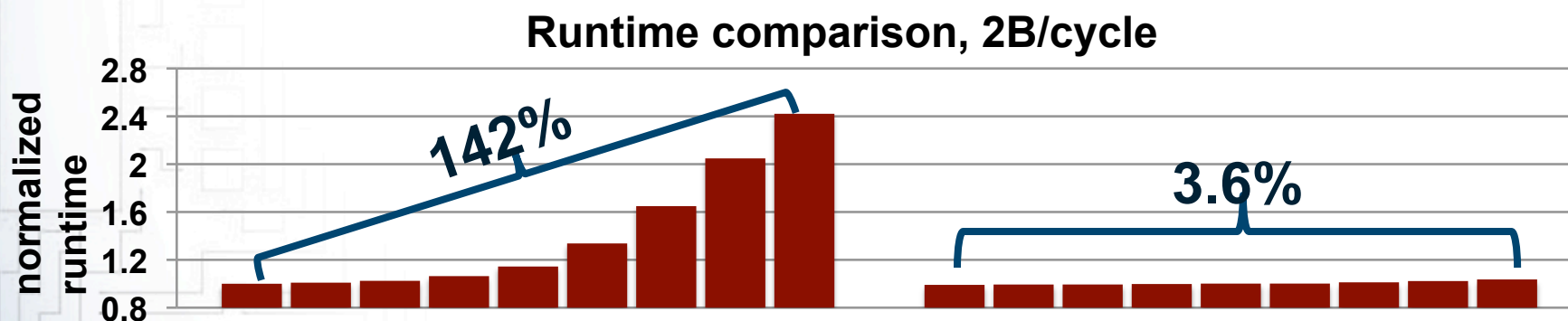
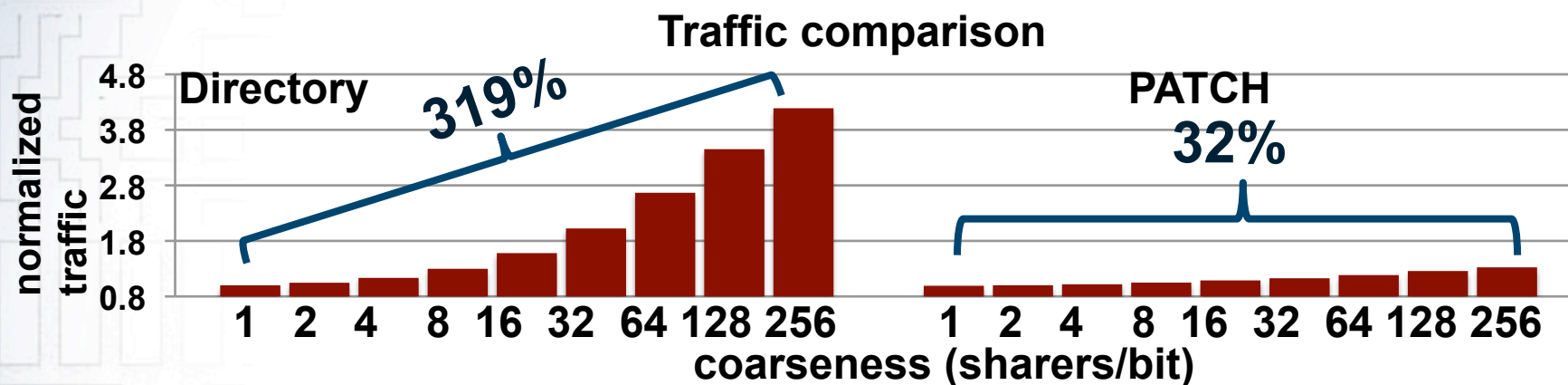


- With PATCH only token holders need respond
  - Avoid “unnecessary acknowledgements”
  - When # of sharers small, prevents ack from dominating

**Even more scalable than directory**

# Coarse Directory: Runtime and Traffic

microbenchmark @ 256 processors



**PATCH has high tolerance to inexactness**

## Related Work

- Token counting
  - Token Coherence [Martin+, ISCA '03]
  - Priority Requests [Cuesta+, PDP '07]
  - Virtual Hierarchies [Marty+, ISCA '07]
  - Ring Order [Marty+, MICRO '06]
- Predictive direct requests
  - Multicast snooping [Bilir+, ISCA '99]
  - Owner Prediction [Acacio+, SC '02]
  - Producer-Consumer sharing [Cheng+, HPCA '07]
  - Virtual Circuit Tree Multicast [Jerger+, ISCA '08]
- Bandwidth Adaptive Snooping [Martin+, HPCA '02]
- Embedded ring snooping
  - Uncorq [Strauss+, MICRO '07]

# Conclusion

- PATCH
  - Directory protocol foundation
  - Fast sharing? Direct requests
  - Safety? Token counting
  - Forward progress? **Token tenure**
    - Broadcast-free
  - Retain scaling of directory? Best-effort delivery
- Resulting properties
  - One-size-fits-all
  - Opportunistically uses bandwidth for performance
  - Yet scales no worse than directory

ACG

UNIVERSITY *of* PENNSYLVANIA  
ARCHITECTURE + COMPILERS GROUP



Penn  
UNIVERSITY *of* PENNSYLVANIA