Server-assisted Latency Management for Wide-area Distributed Systems

Wonho Kim (Princeton University),
Kyoungsoo Park (KAIST)
and Vivek Pai (Princeton University)

Low-latency One-To-Many Dissemination

- Existing one-to-many file transfer systems
  - Content Distribution Network (CDN), P2P, Gossip
  - Improve **bandwidth efficiency** or **delivery odds**

- Our focus: Low-latency one-to-many file dissemination
  - Measured as **completion time**
  - Requires different strategies

Use Cases

- Configuration management systems
  - Adapt system's behavior at runtime
  - Akamai CDN: every **10** seconds, **1KB** – **100MB**
- Distributed monitoring systems
  - Coordinate multiple nodes
- Developing/Testing wide-area systems
  - Non-interactive environment (waste N***$90K/y** resource)

Lsync: Low-latency File Transfer System

- Optimized for completion time
  - Not bandwidth efficiency
  - Not average/aggregate throughput
- Provides a folder syncd across remote nodes
  - Pluggable into existing systems
- Benefits
  - Improve overall system responsiveness.
  - Optimize performance under dynamic network conditions
  - Enable interactive development

Lessons and Contributions

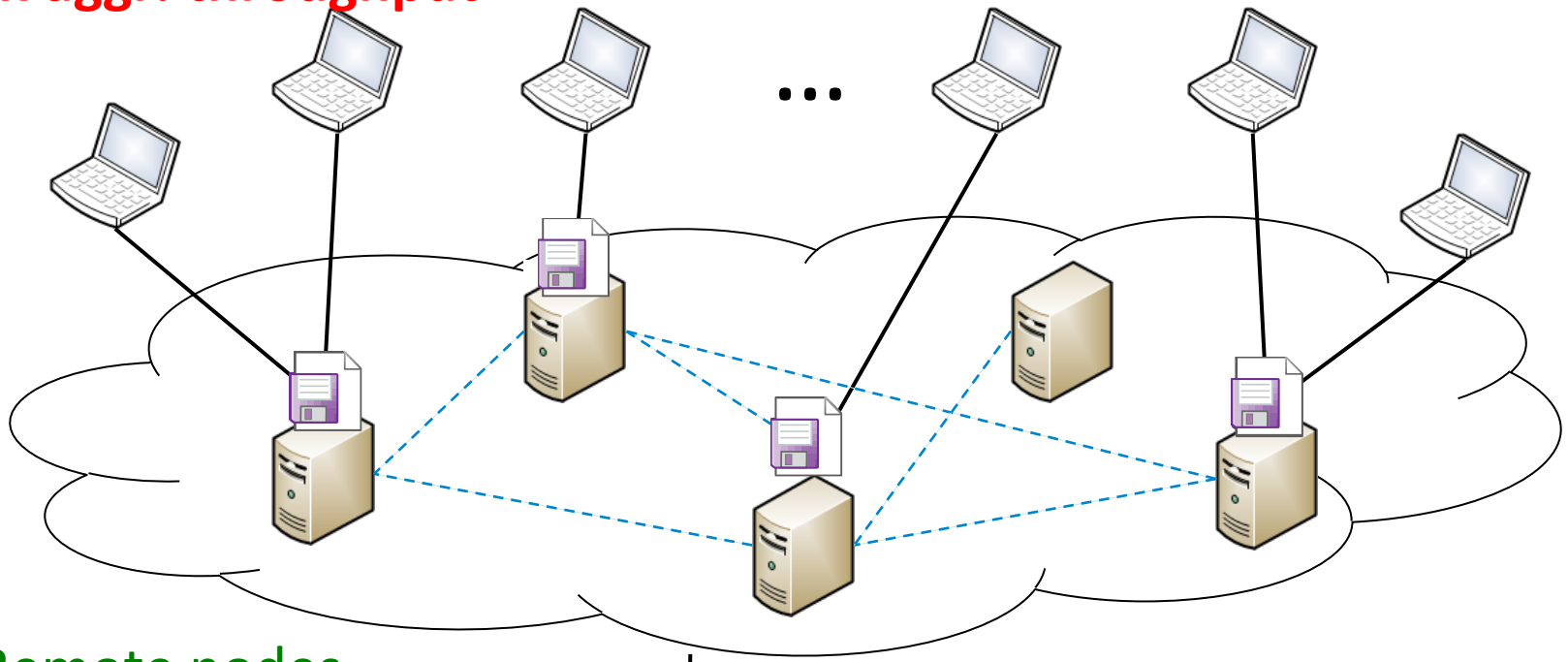
- Existing systems are not favorable to slow nodes
  - Lsync focuses on bottleneck nodes from the beginning
- Completion time depends on target nodes
  - Lsync selects the best set of nodes for user
- End-to-end transfer can be faster than overlay
  - Lsync characterizes overlay and uses only when appropriate
- Overlay performance changes at short time scales
  - Lsync adapts policy while transfer is taking place

Existing File Transfer Systems

Max avg. performance
Max aggr. throughput

Open client population
(no upper bound on clients)

End users



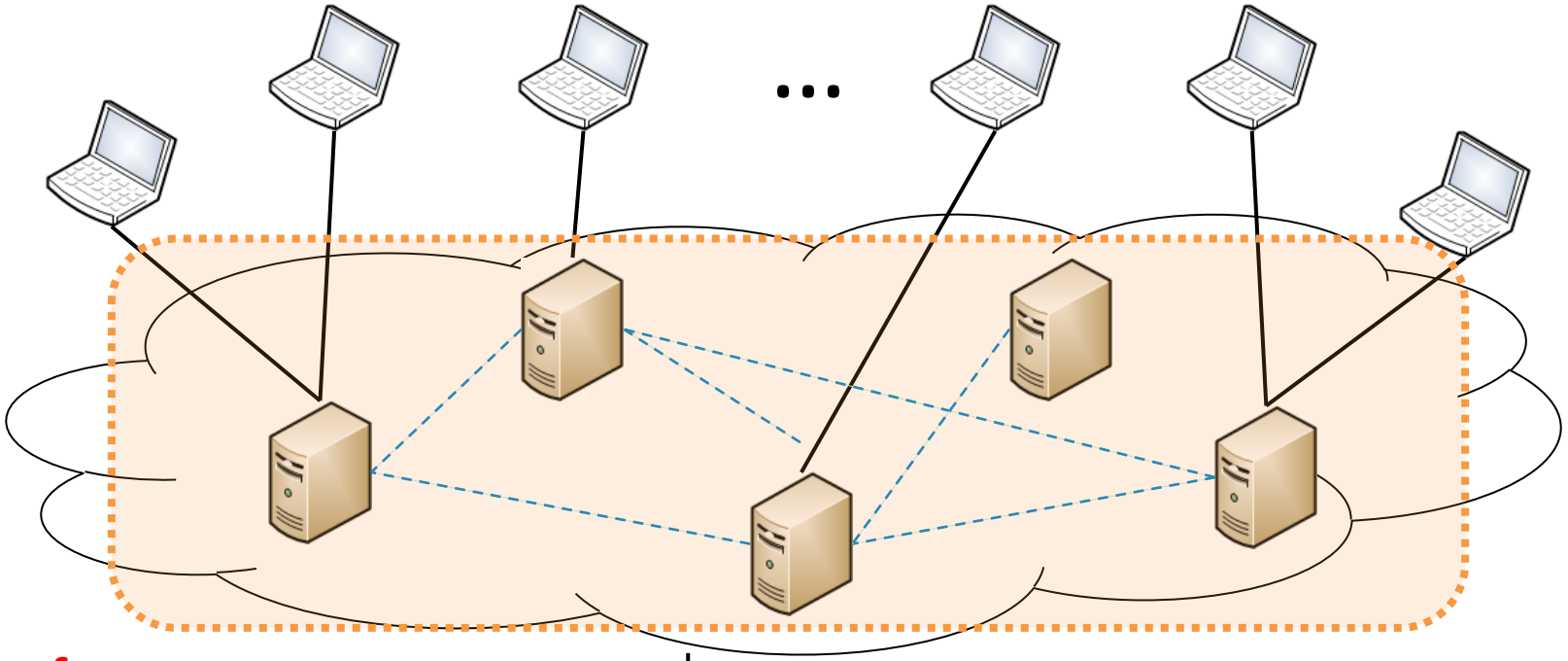
Remote nodes

Steady state:
Pulled from server AND
Cached in remote nodes

Origin server

Min load on origin server

Our Focus: Remote Nodes



Our focus
- Remote nodes

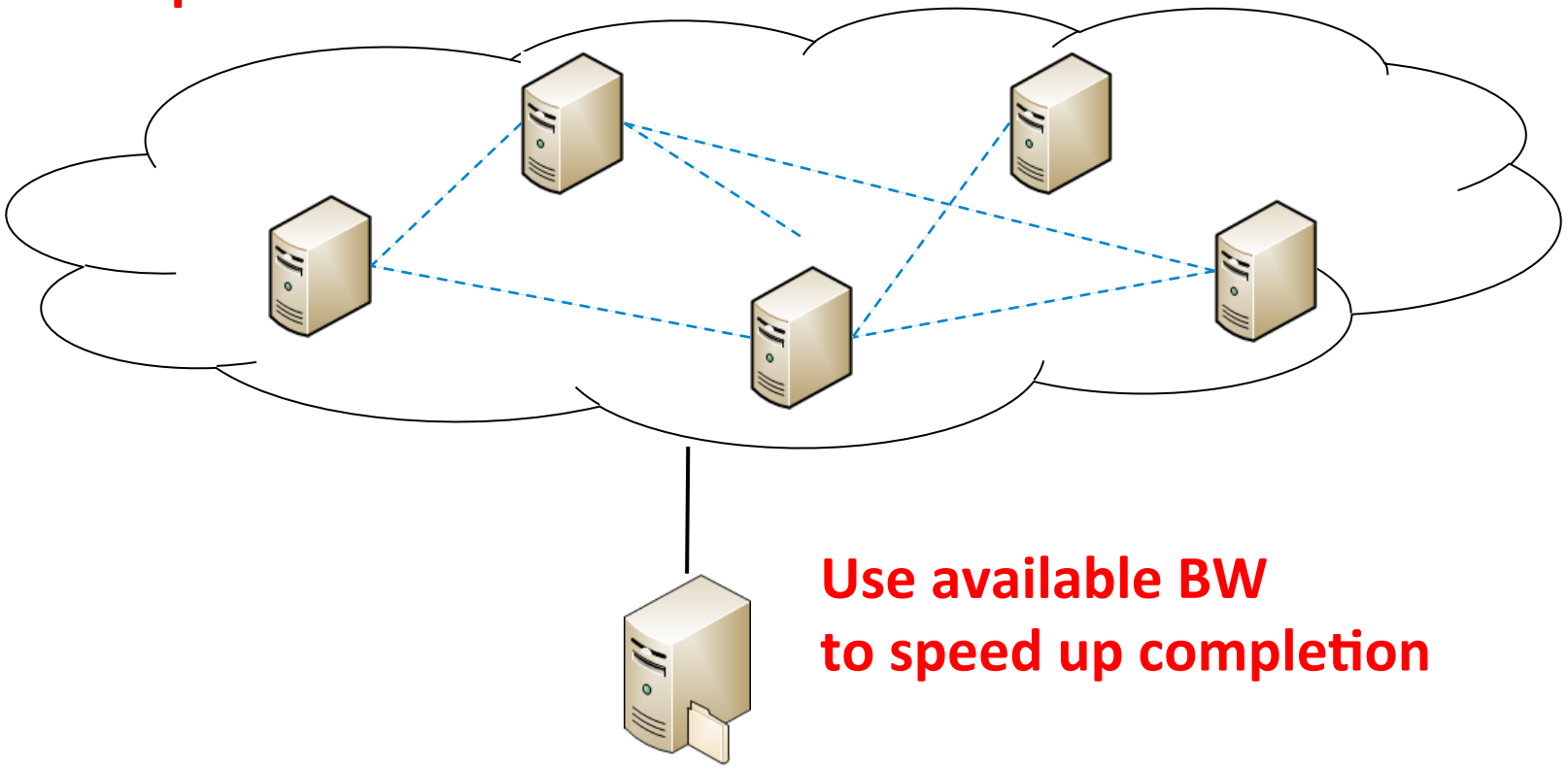
Configurations/Policies
Updated modules
Coordinated task lists

Low-latency File Transfer to Remote Nodes

Completion time is dominated by "Slow Nodes"

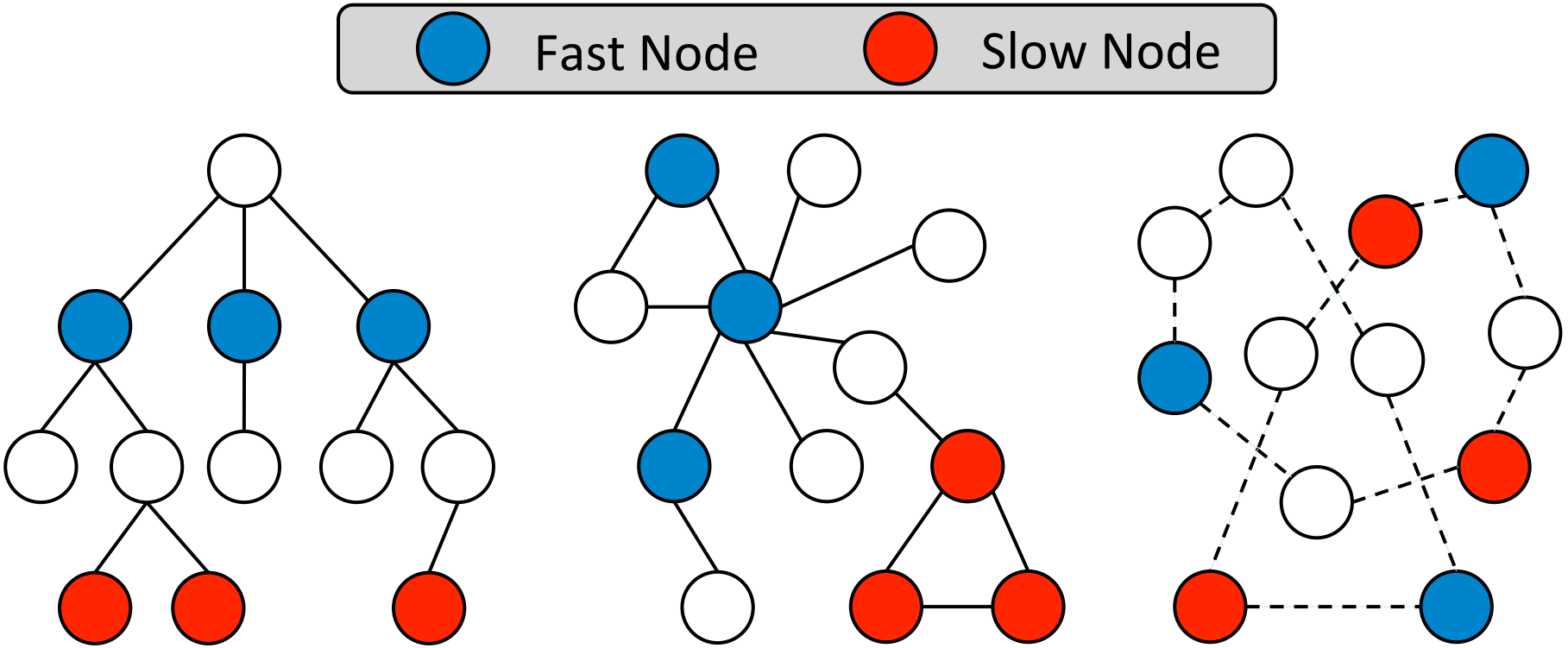
Fixed client population

Min completion time



Use available BW
to speed up completion

Slow Nodes in Existing Systems



Overlay Multicast **P2P** **Epidemic Routing**

- Peering strategies are not favorable to slow nodes
- Scheduling slow nodes last makes the problem worse

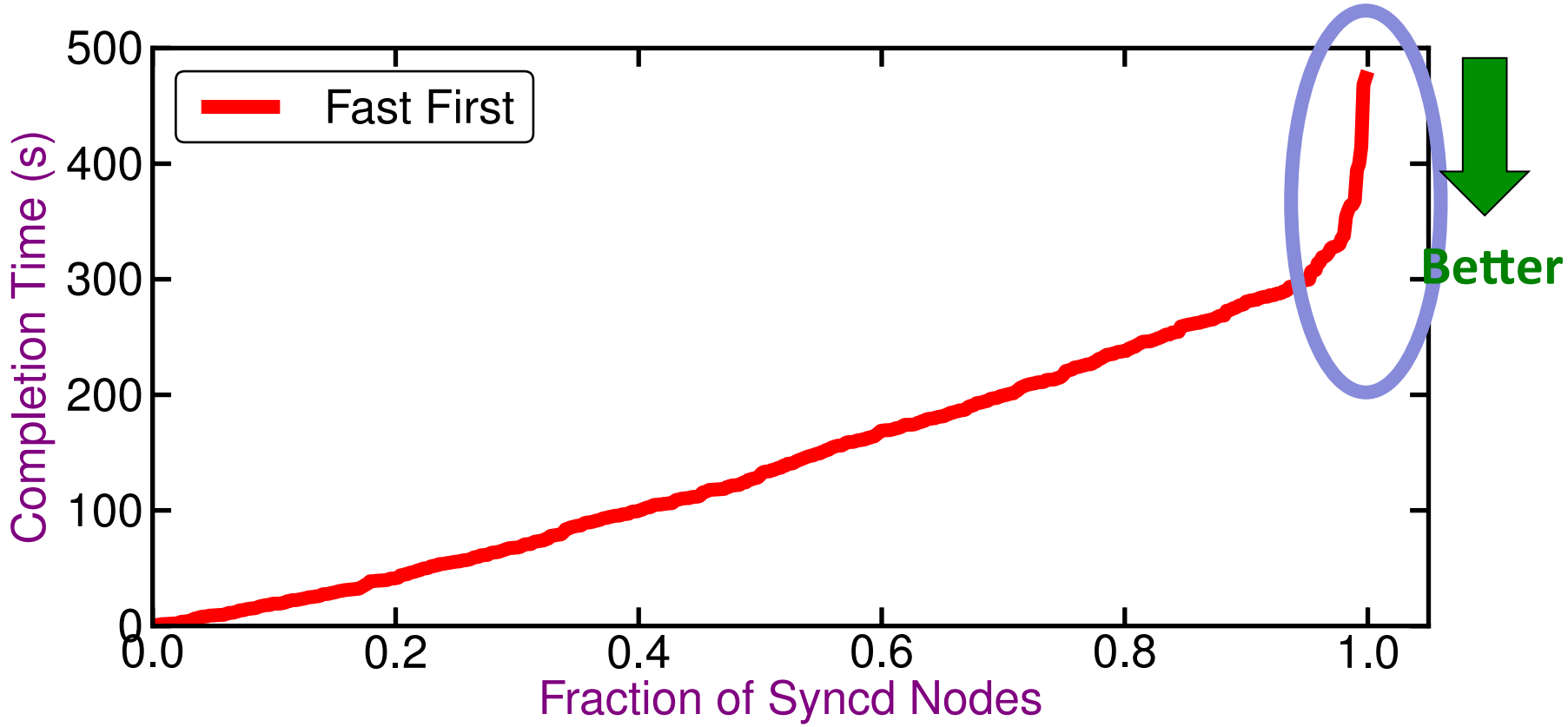
Server Bandwidth Allocation

- Lsync uses server's spare BW to assist slow nodes
- How to allocate server's BW to reduce latency
- Compare different BW allocation policies

Node Scheduling

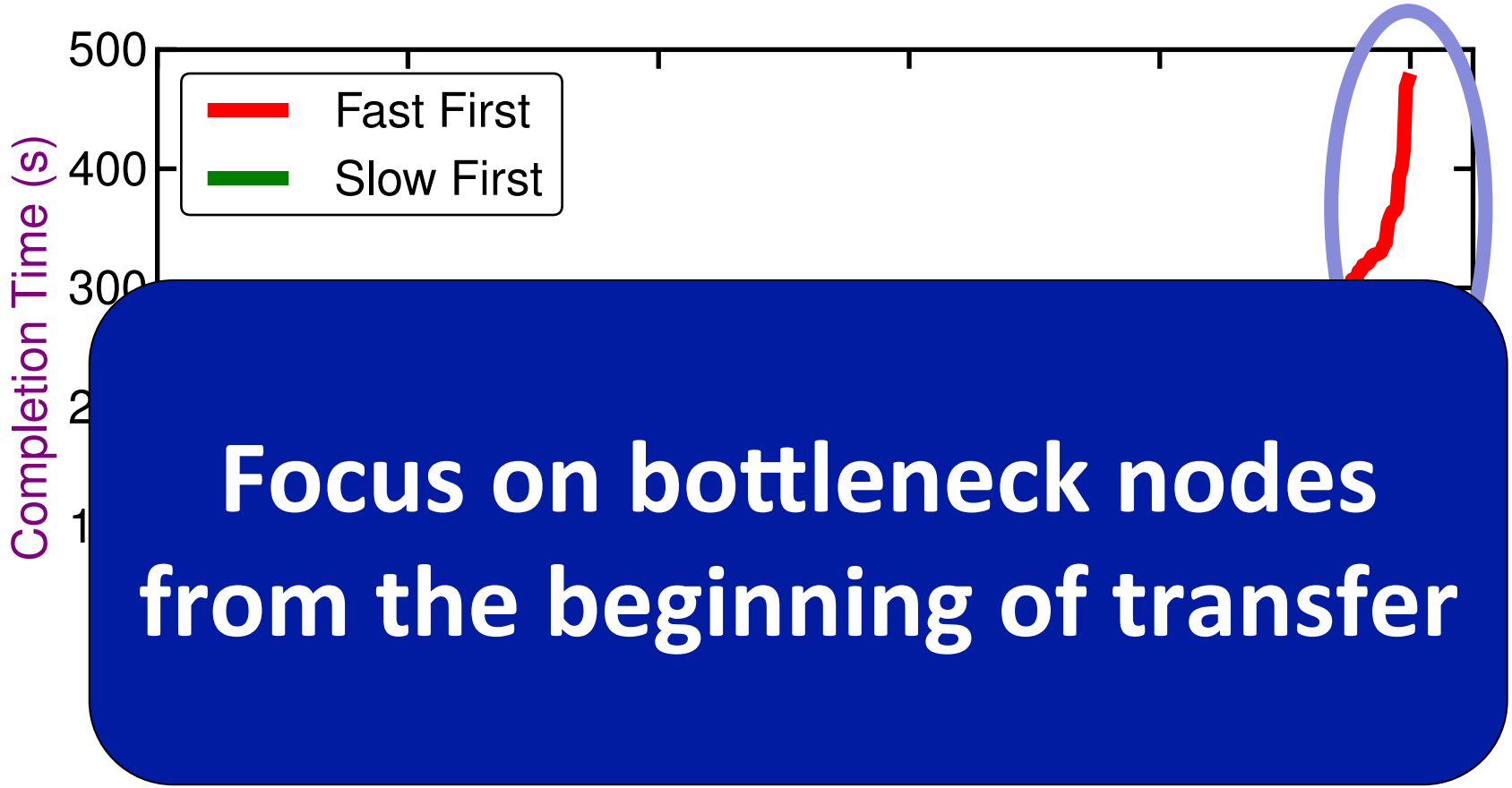
- Fast First
  - Allocate BW to nodes with shortest expected completion time
  - Shortest Remaining Processing Time (SRPT)
  - Optimal for mean response time
- Slow First
  - Opposite of Fast First
  - Give preferences to the slowest node when BW is available

Fast First vs. Slow First



- All live PlanetLab nodes (**556**), 100 Mbps server bandwidth
- Slow nodes become bottleneck in Fast First

Fast First vs. Slow First

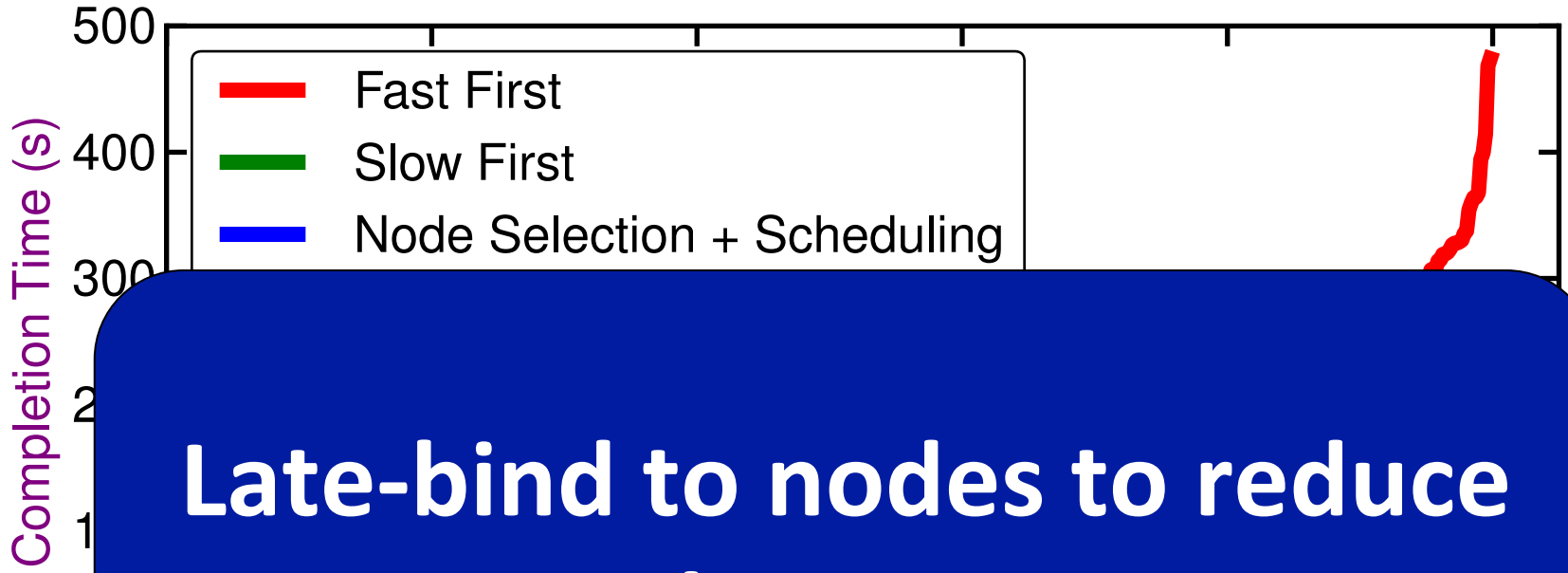


- All live PlanetLab nodes (556), 100 Mbps server bandwidth
- Slow nodes become bottleneck in Fast First

Node Selection

- Not every scenario requires waiting until 100% of nodes are syncd
  - Too costly and unpredictable
- Target sync ratio (***r***)
  - The fraction of nodes that need to be syncd fast
  - Incremental rollouts, fast development, etc
- Integrate node selection with scheduling
  - Sort nodes by estimated remaining completion time
  - Select N·***r*** nodes
  - Use Slow First scheduling

Integrating Scheduling with Selection



Late-bind to nodes to reduce latency

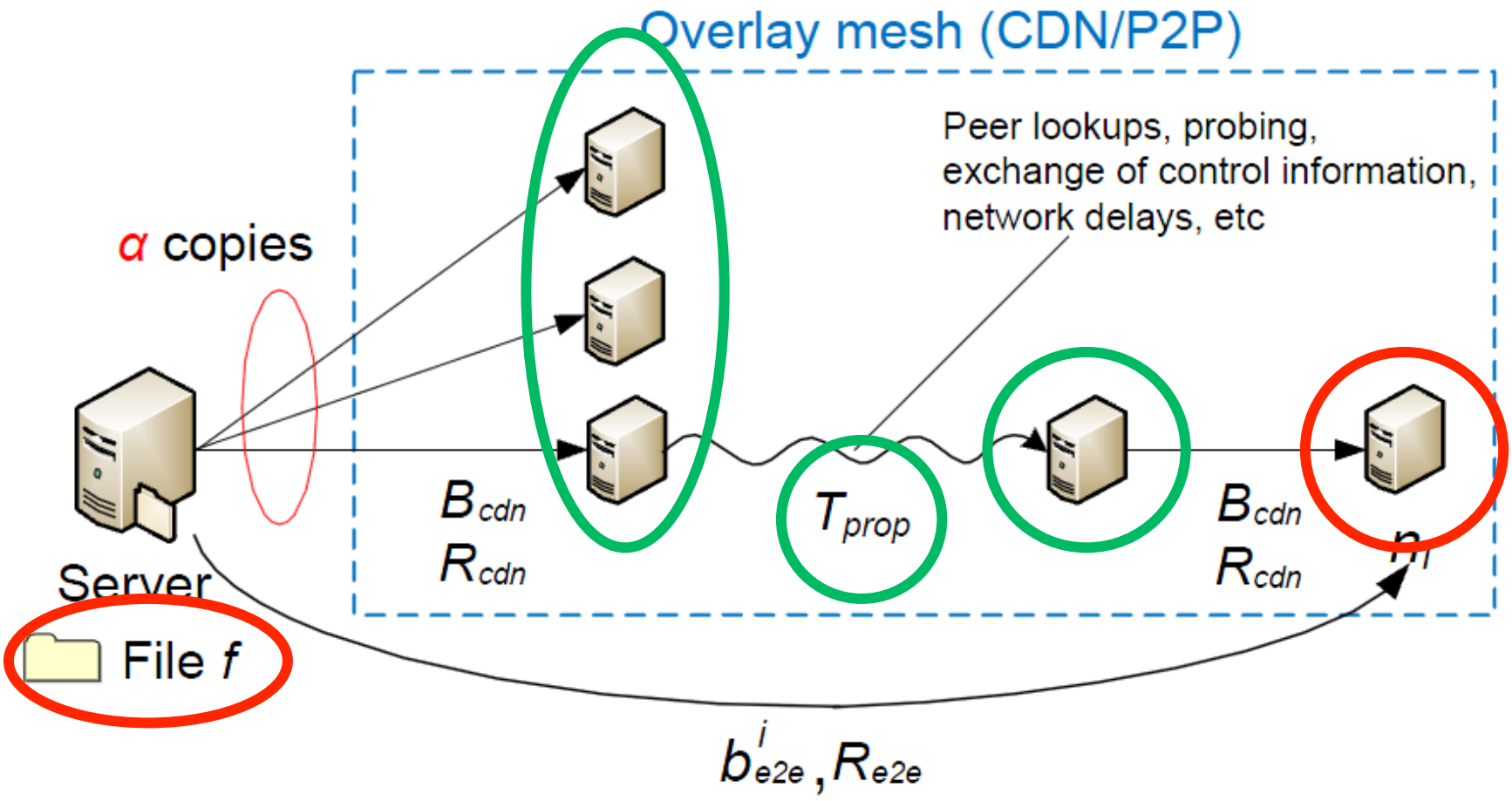
- Reduce latency for all target ratios

Leverage Overlay Mesh

- Wide-area systems construct overlay mesh
  - Scalable data transfer to end users
  - Also used for internal data dissemination

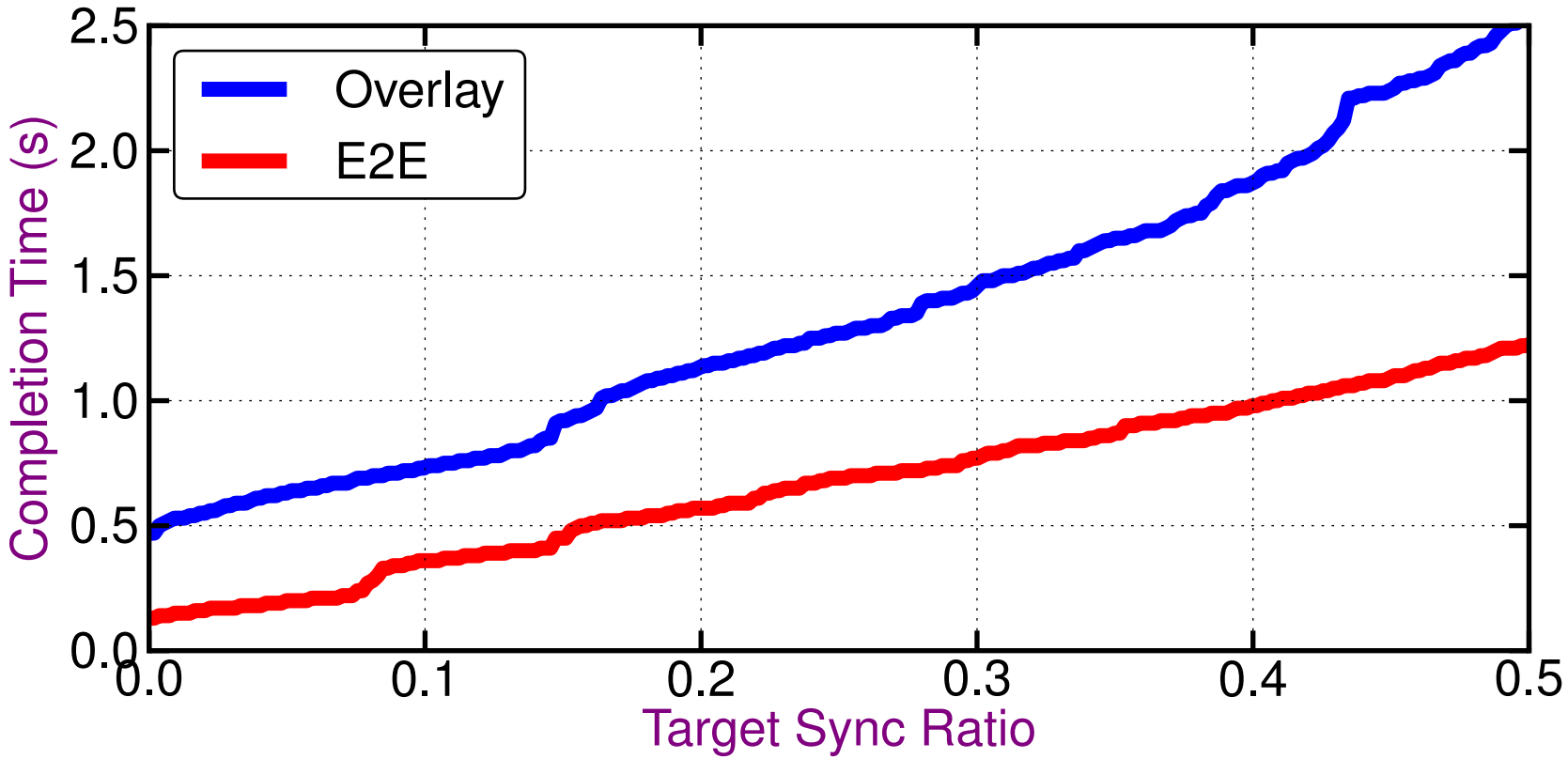
- Scalable but not focusing on startup latency

Startup Latency in Overlay Mesh



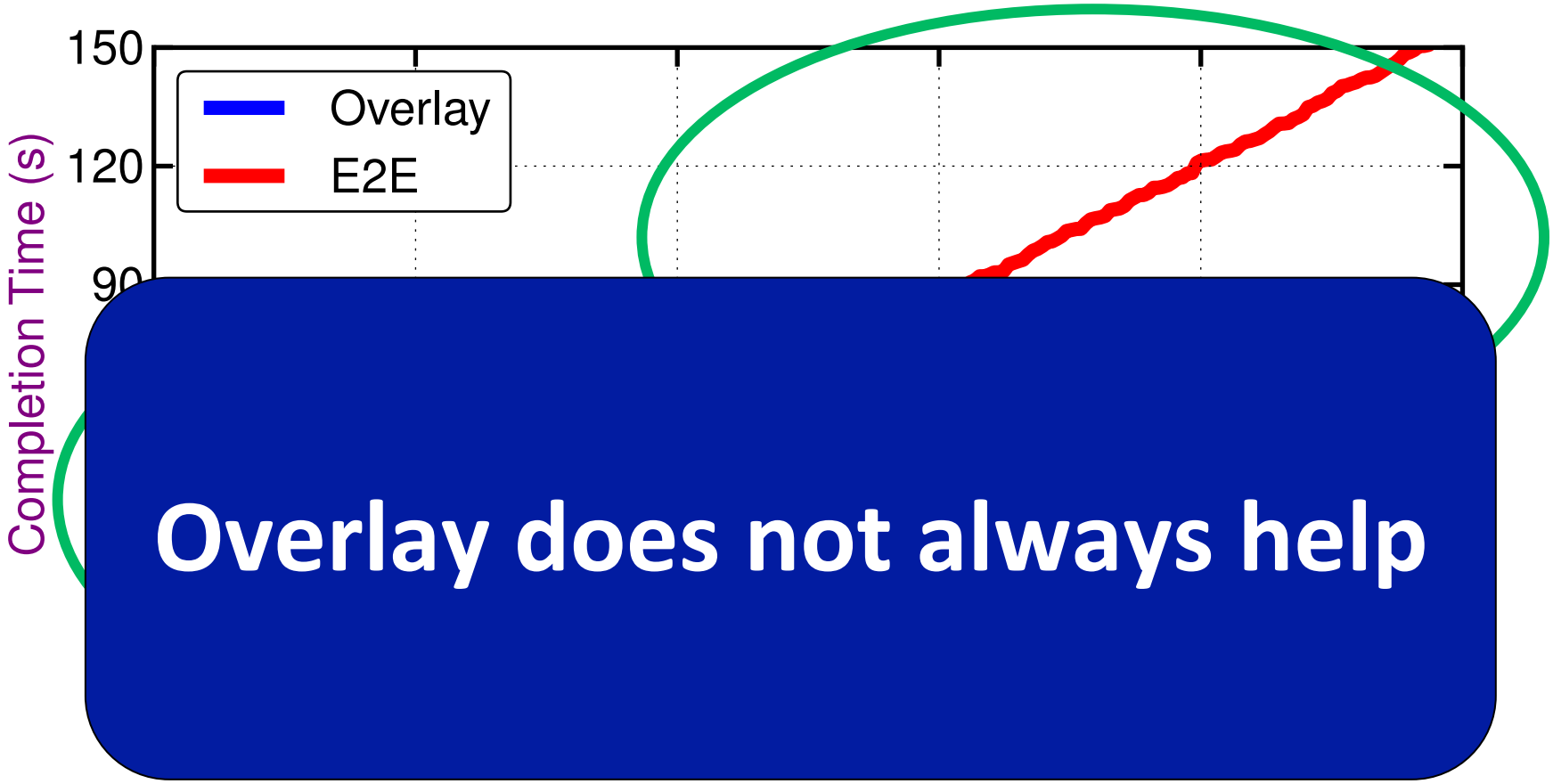
- Measure first-byte latency ($2 \cdot R_{cdn} + T_{prop}$)
- Measure overlay bandwidth (B_{cdn})

E2E Transfer vs. Overlay (5KB)



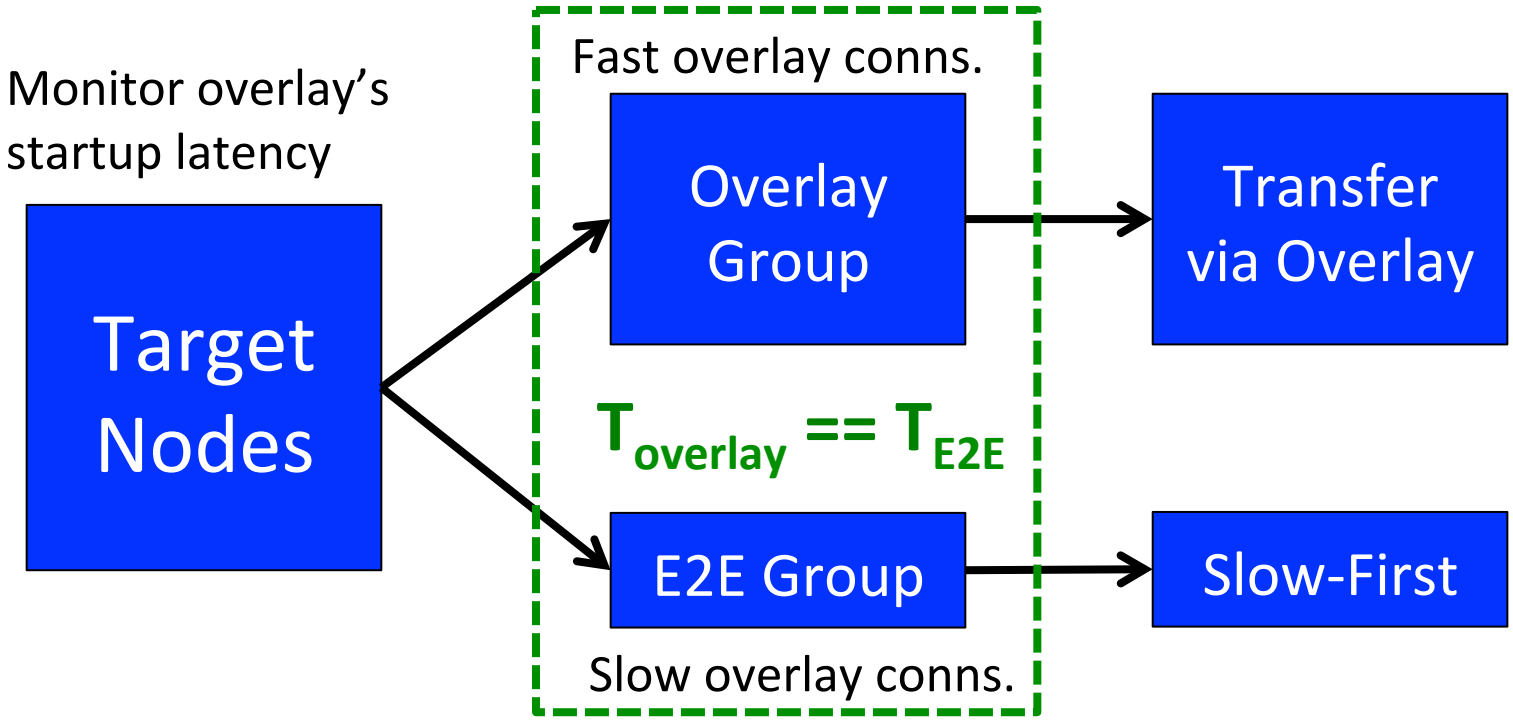
- Used **CoBlitz** [NSDI'06] (but not limited to CoBlitz)
- Overlay can be slower than E2E because of startup latency

E2E Transfer vs. Overlay (5MB)



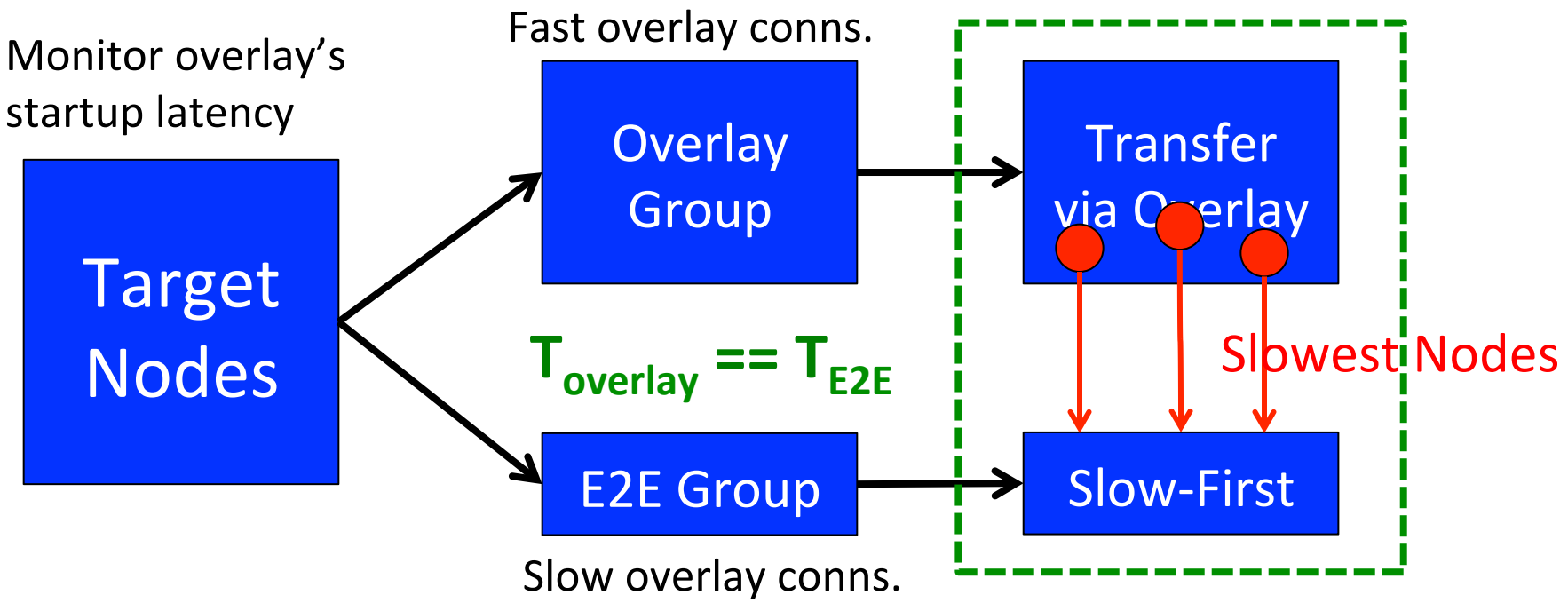
- CDN's efficient bandwidth usage outweighs its setup cost
- Should adapt to target ratio, file size, bandwidth, etc.

Nodes Division (before transfer)



Factors to be considered:
Target ratio, Spare BW in server, File size,
Startup latency and BW in overlay

Adaptive Switching (during transfer)



Variations in overlay performance are common (BW fluctuation, multi-hop traverse, etc)

Dynamically switch to end-to-end transfer

Lsync Implementation

- Lsync server
  - Detects file changes in the background (inotify in Linux)
  - Plans file transfer process
  - Perform node selection, node scheduling, nodes division
- Remote daemons
  - Save file changes in local folder
  - Measure overlay performance
  - Perform adaptive switching
- Written in C (6.5K lines), run in user space

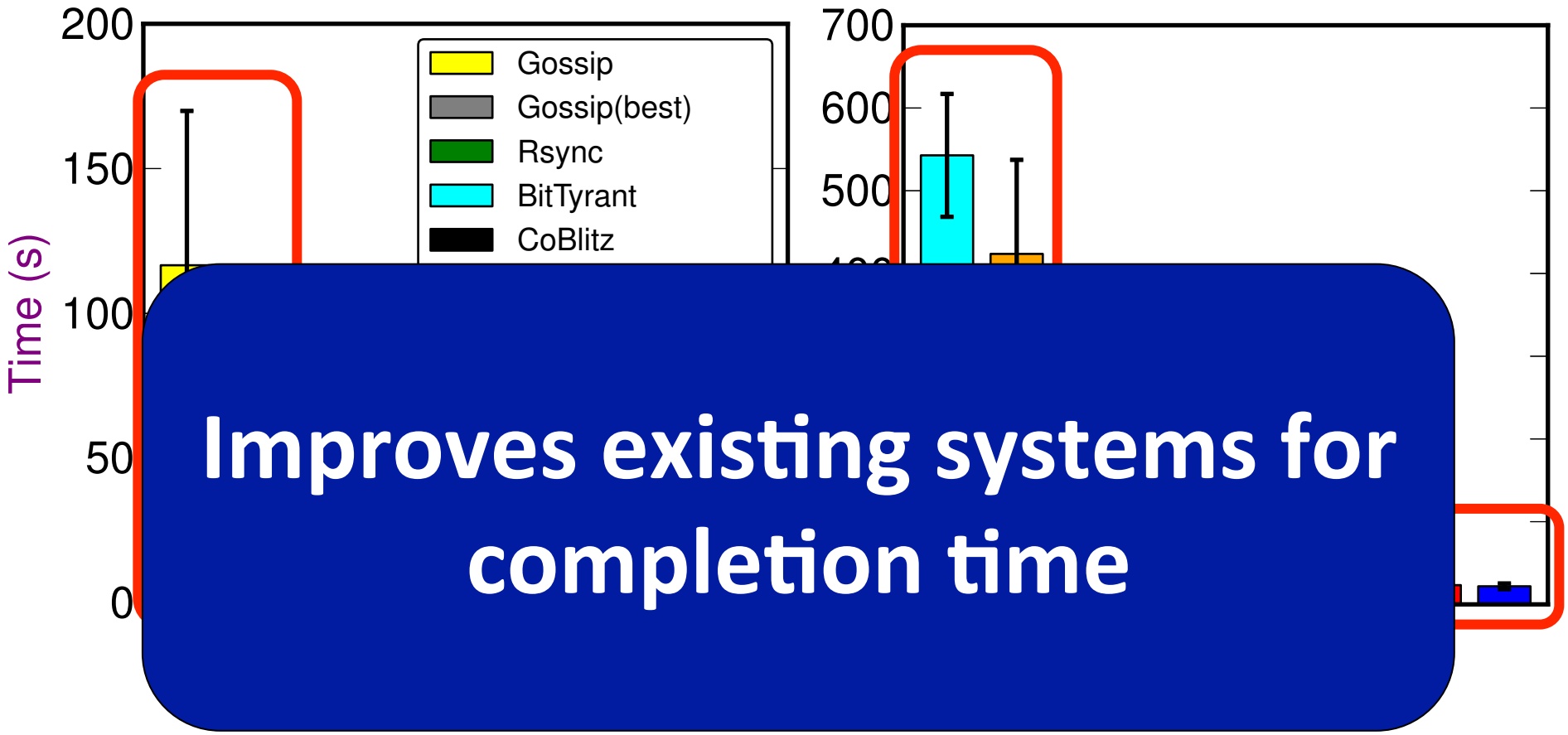
Evaluation

- Setting
  - Deploy Lsync on all live PlanetLab nodes (**556**)
  - Run a dedicated origin server with **100Mbps** BW
- Experiments
  - Compare Lsync with different types of data transfer systems
  - Examine how Lsync adapts its file transfer
  - Repeat **10** times and show 95th confidence intervals

Comparison with Other Systems

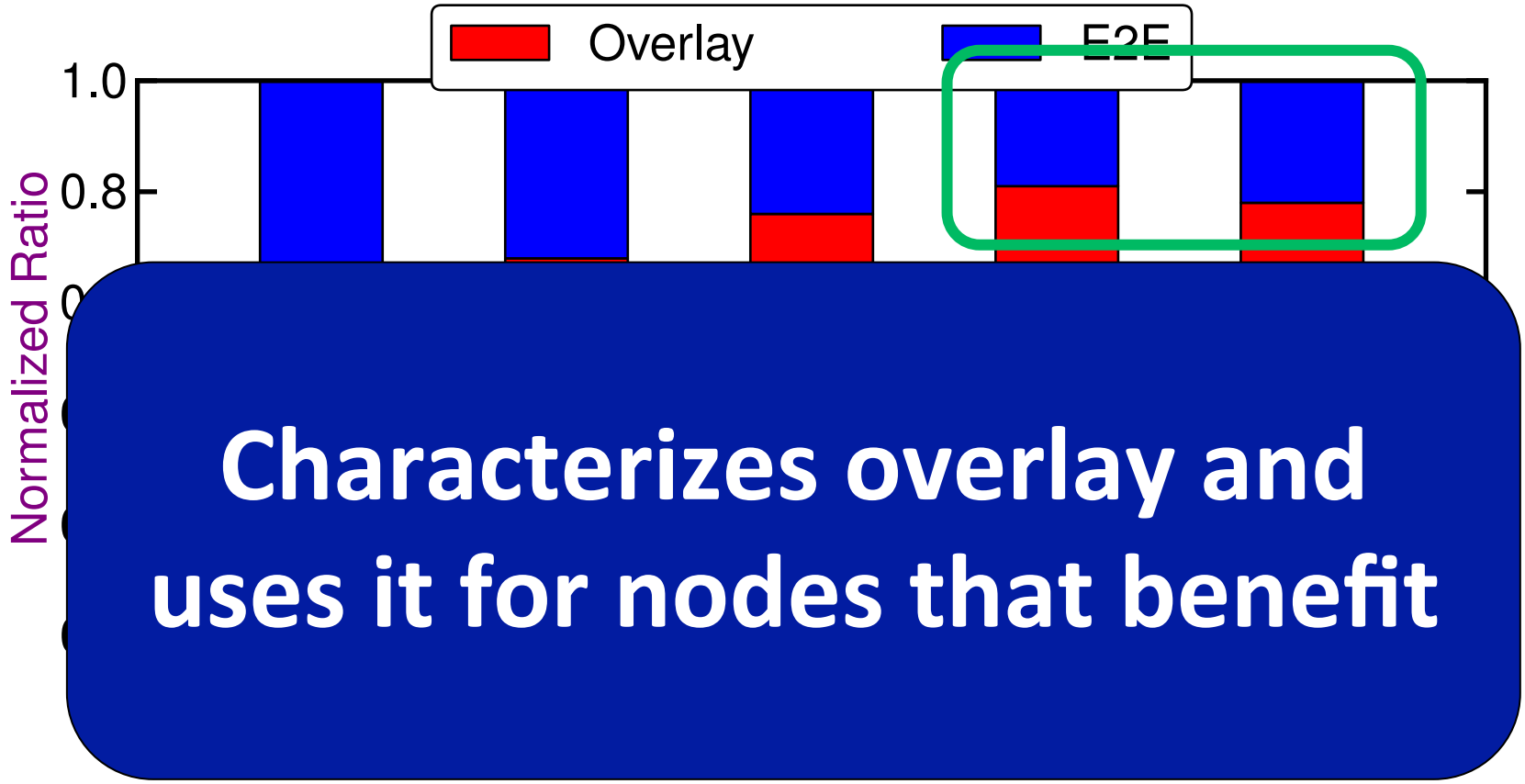
- End-to-end copying tool
  - Rsync: widely used for cloning files across remote machines
- P2P systems
  - BitTorrent and BitTyrant: high throughput and BW efficiency
- Epidemic routing (Gossip)
  - Parameters: step-interval (how often) & fanout (how many)
  - Tune parameters: measure best and average results
- CDNs
  - CoBlitz (Academic), CloudFront (Commercial)
- Lsync
  - Two versions: Lsync-CoBlitz and Lsync-CloudFront

Comparison with Other Systems



- Transfer CoBlitz web proxy file (600KB)
- Measure completion time for target ratio 0.5 and 0.98

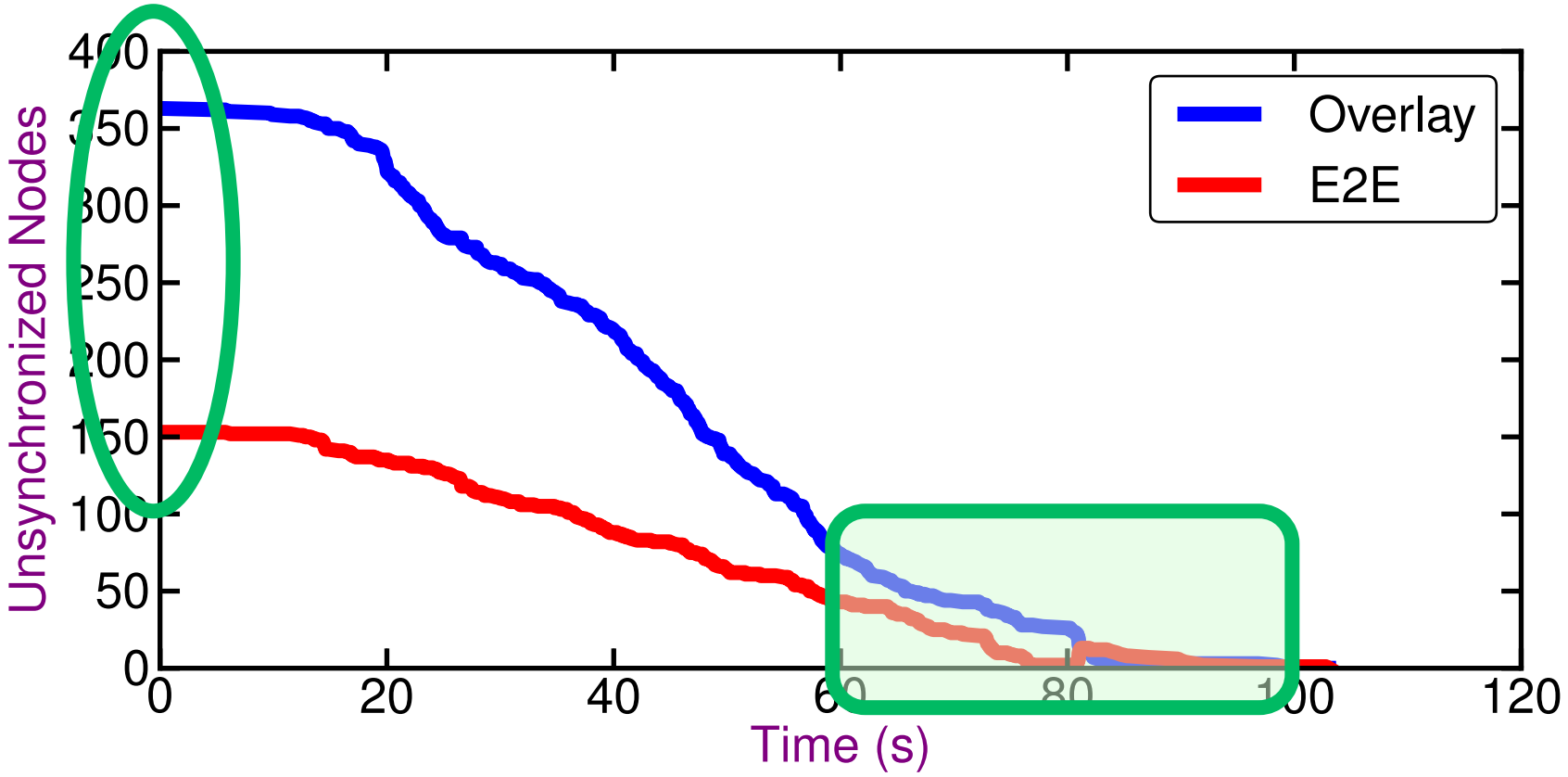
Nodes Division



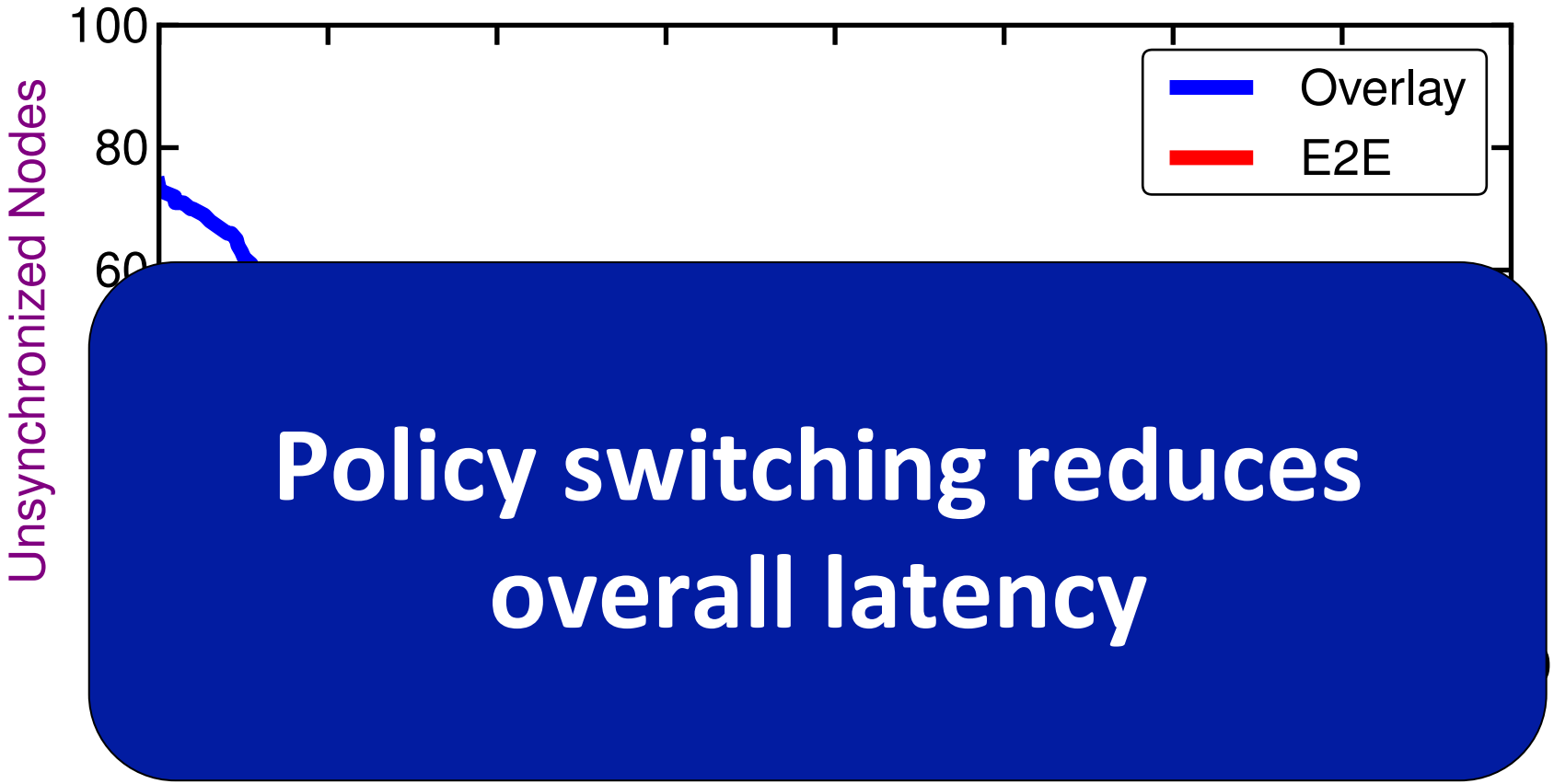
Characterizes overlay and uses it for nodes that benefit

- Lsync adjust workload across server and overlay

Adaptive Switching

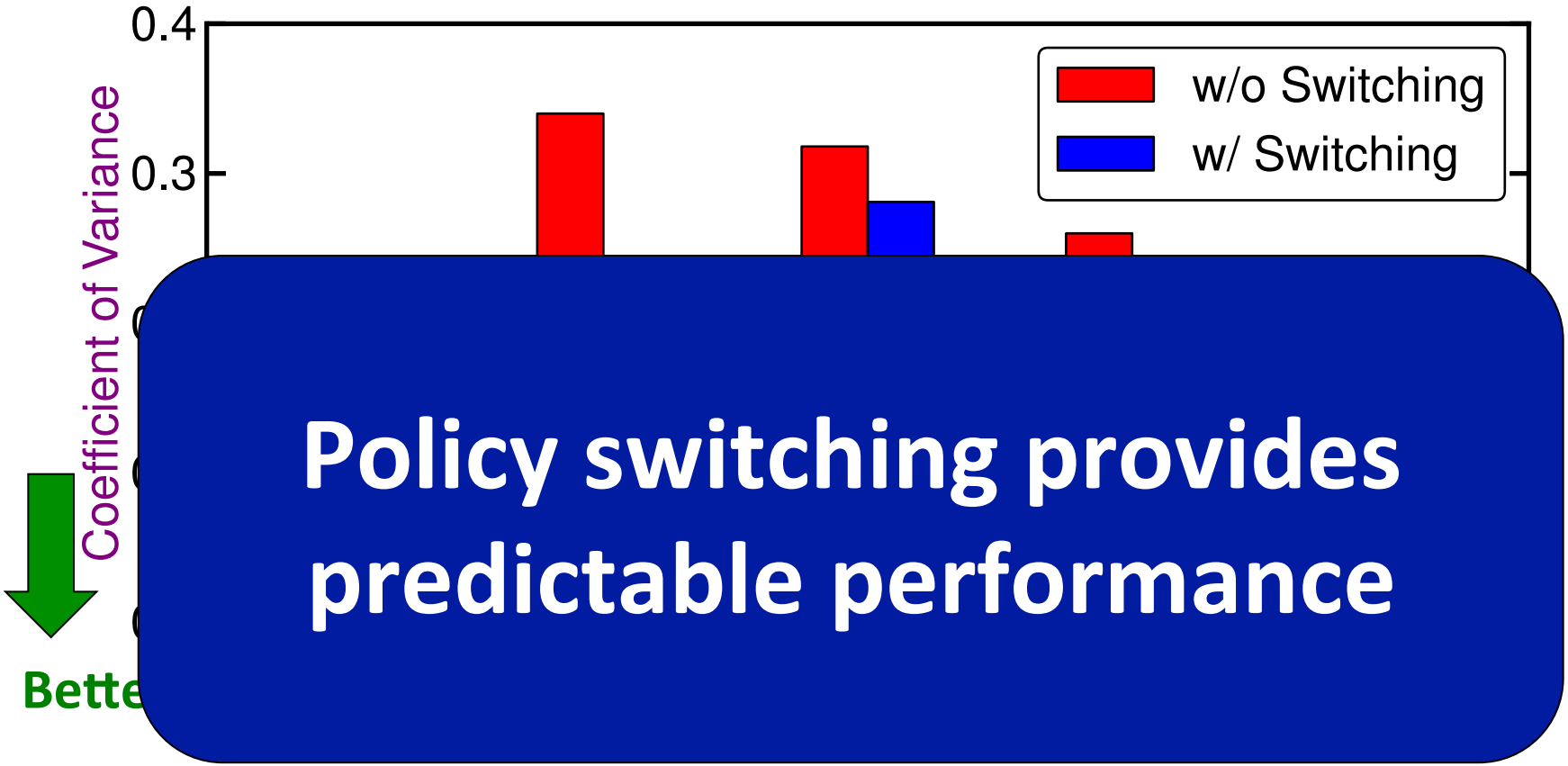


Adaptive Switching



- An average of **7%** nodes switched to E2E
- Reduce completion time by **16%**

Stable Transfer



Policy switching provides predictable performance

- Adaptive switching provides stable file transfer
- Mask the variations in overlay performance at runtime

Conclusion

- Low-latency dissemination is essential for managing nodes in wide-area systems
- Existing data transfer systems are suboptimal for completion time
- Lsync identifies the sources of the latency and implements policies to reduce their impact
- Lsync can be used as building block where latency matters

Thank you

Wonho Kim (wonhokim@cs.princeton.edu)

