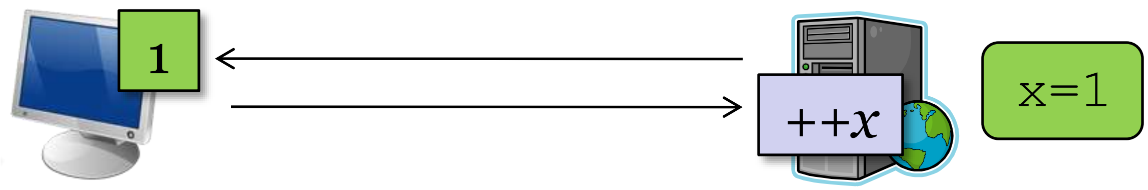
Tolerating Latency in Replicated State Machines through Client Speculation

April 22, 2009

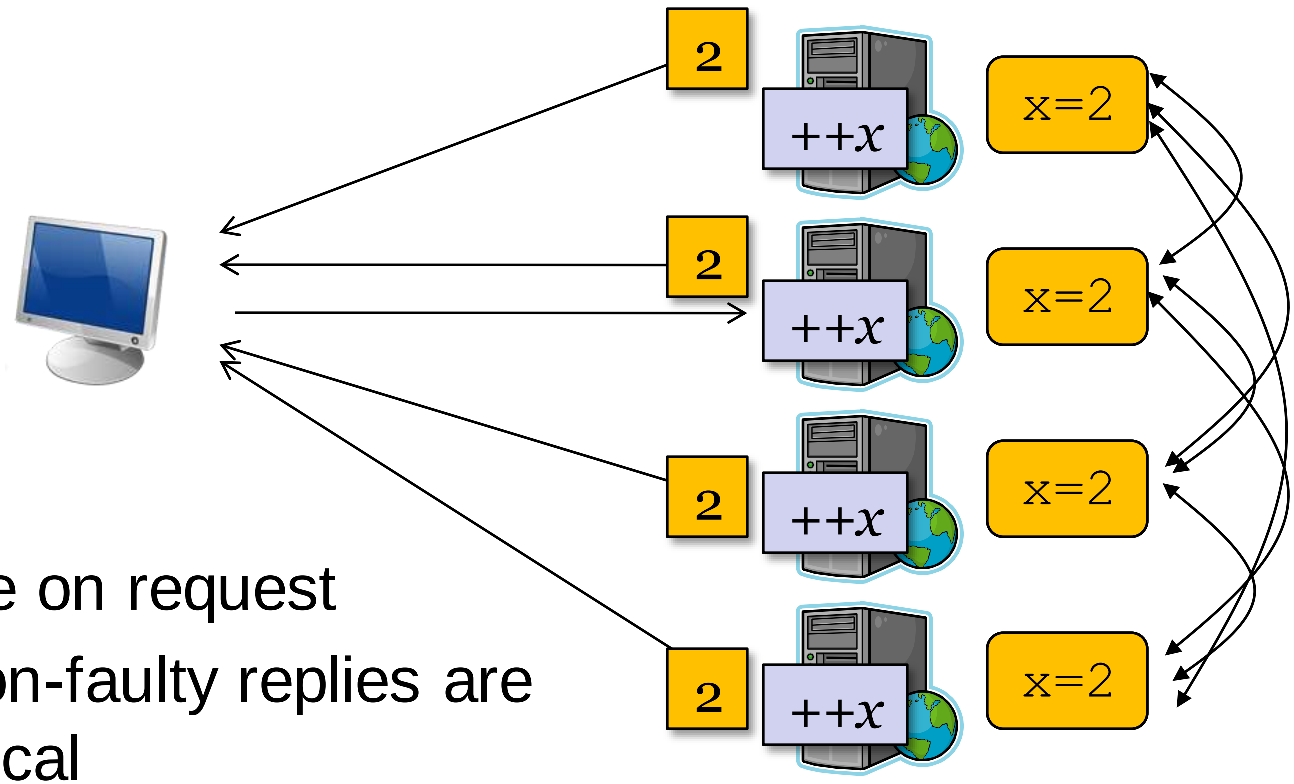
Benjamin Wester[1], James Cowling[2], Edmund B. Nightingale[3], Peter M. Chen[1], Jason Flinn[1], Barbara Liskov[2]

University of Michigan[1], *MIT CSAIL*[2], *Microsoft Research*[3]

Simple Service Configuration

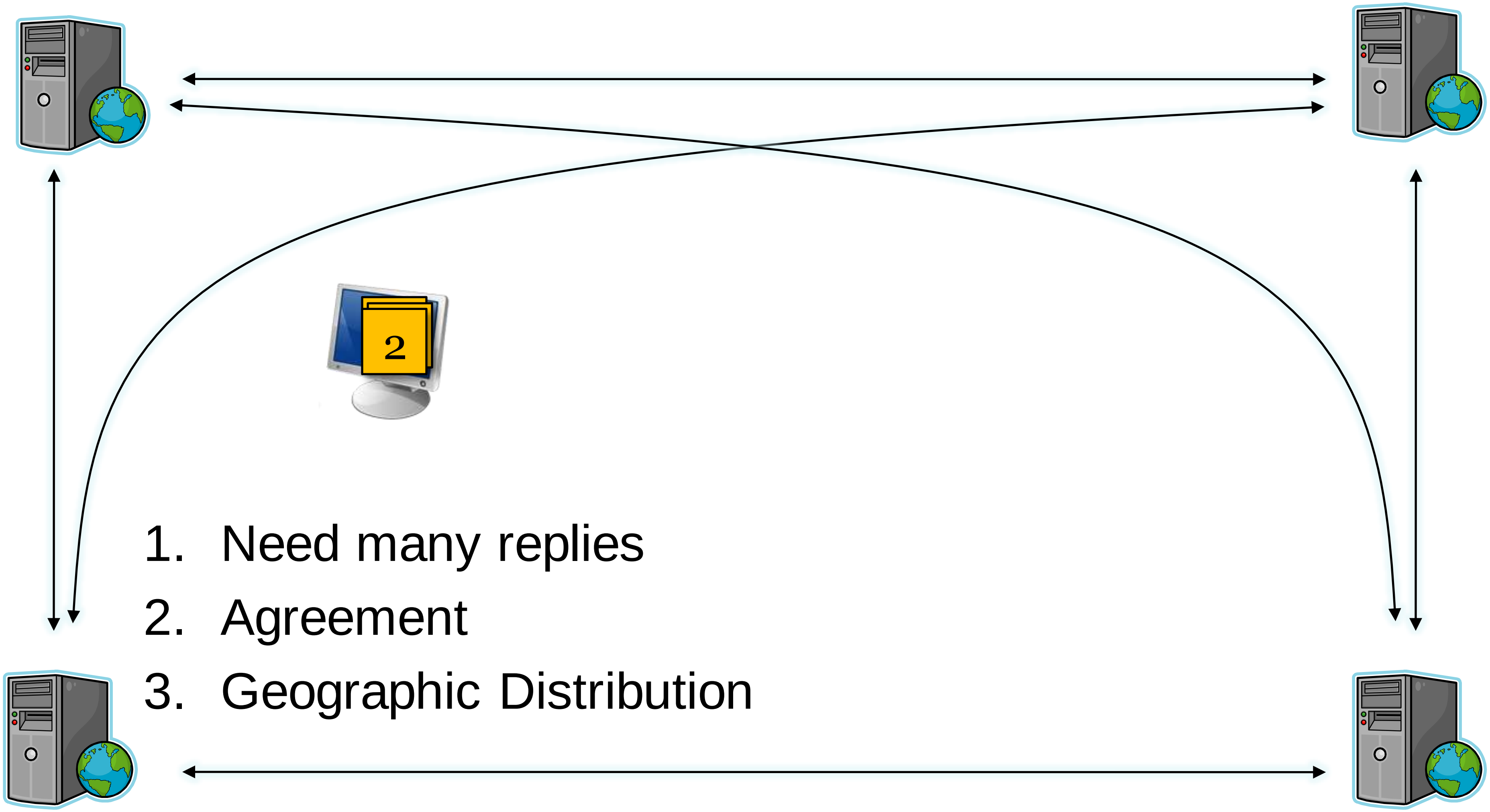


Replicated State Machines (RSM)



- Agree on request
- All non-faulty replies are identical

RSMs have high latency



1. Need many replies
2. Agreement
3. Geographic Distribution

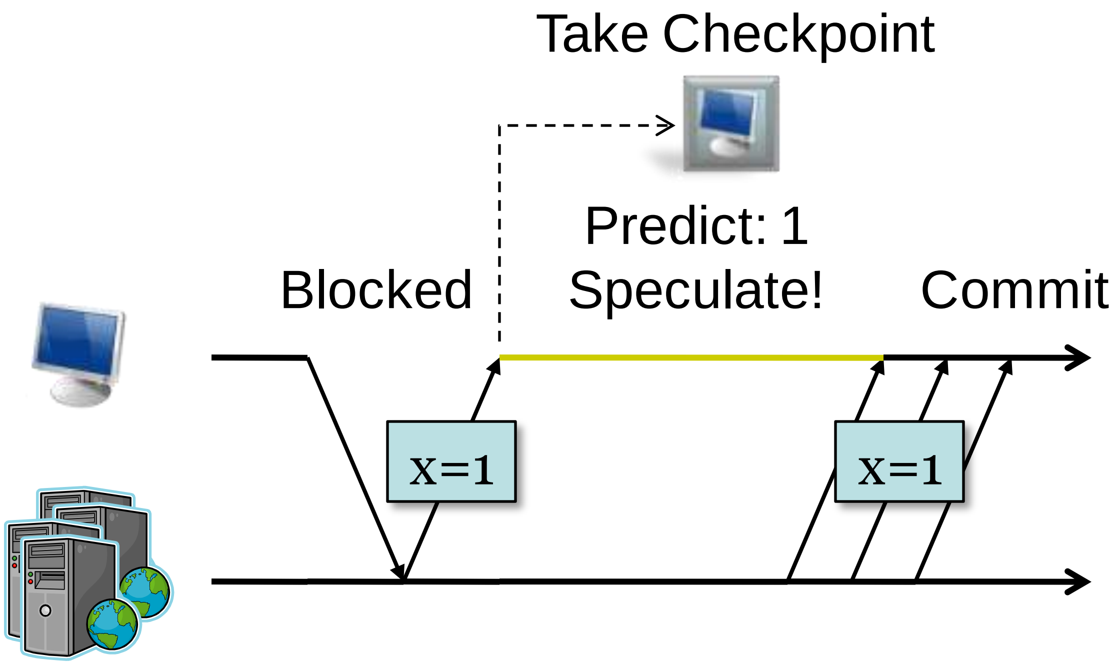
Hide the Latency

- Use speculative execution inside RSM
- Speculate before consensus is reached
  - Without faults, any reply predicts consensus value
  - Let client continue after receiving one reply

Overview

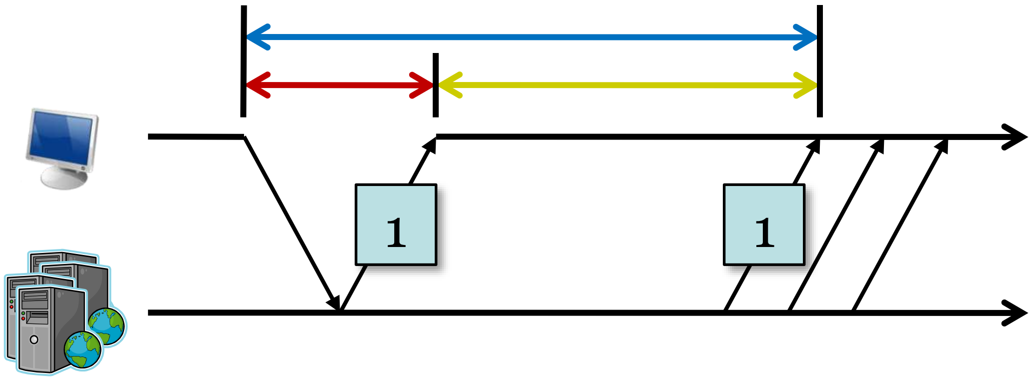
- Introduction
- Improving RSMs with speculation
- Application to PBFT
- Performance
- Conclusion

Speculative Execution in RSM



- Continue processing while waiting

Critical path: first reply

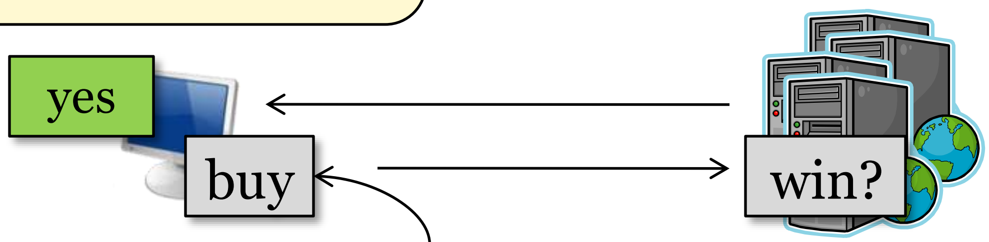


- Completion latency less relevant
- First reply latency sets critical path
  - Speed
  - Accuracy
- Other desirable properties
  - Throughput
  - Stability under contention
  - Smaller number of replicas

Requests while speculative

```
while !check_lottery():
        submit_tps()
buy_corvette()
```

Predict *win?* = *yes*

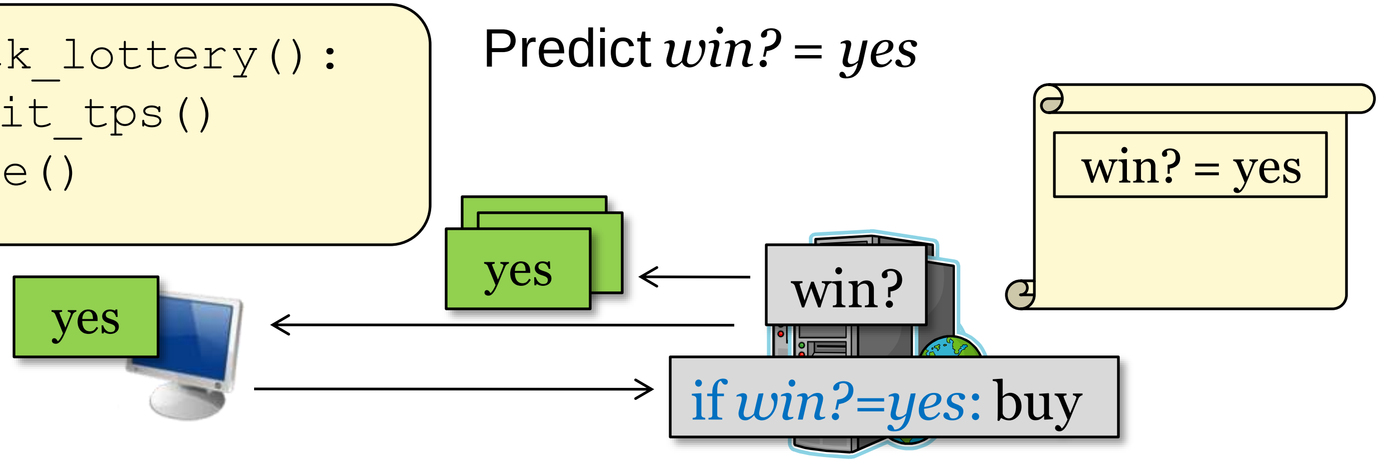


What do we do with this?

1. Hold request
   – Bad performance
2. Distributed commit/rollback
   – State tracking complex

Resolve speculations on the replicas

```
while !check_lottery():
        submit_tps()
buy_corvette()
```



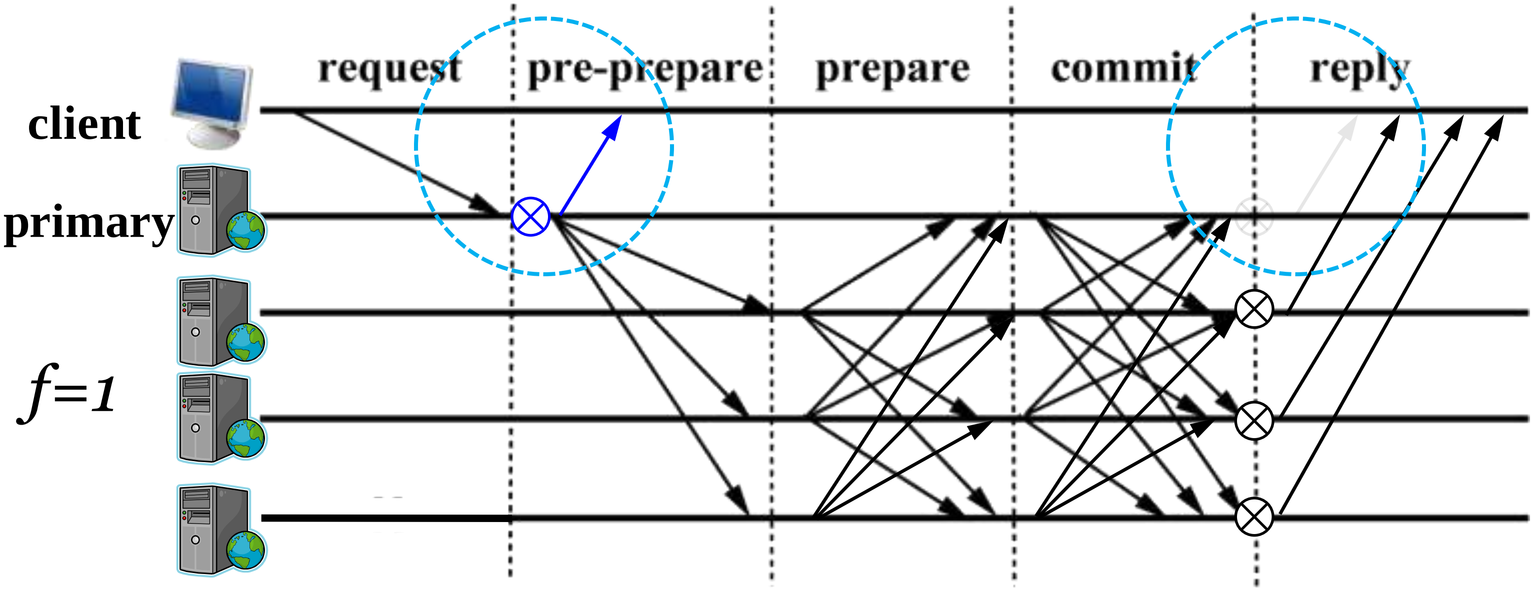
- Explicitly encode dependencies as predicates
- No special request handling needed
- Replicas need to log past replies
- Local decision at replicas matches client

Overview

- Introduction
- Improving RSMs with speculation
- Application to PBFT
- Performance
- Conclusion

Practical BFT-CS

[Castro and Liskov 1999]



Additional Details

- Tentative execution
  - PBFT/PBFT-CS complete in 4 phases
- Read-only optimization
  - Accurate answer from backup replica
- Failure threshold
  - Bound worst-case failure
- Correctness

Overview

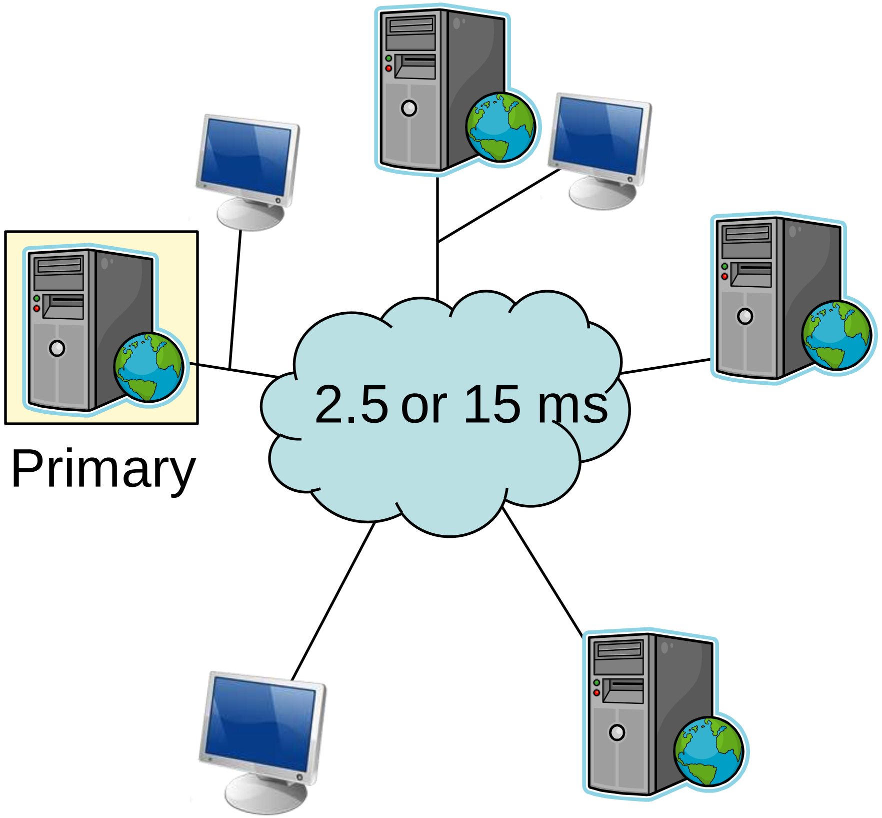
- Introduction
- Improving RSMs with speculation
- Application to PBFT
- Performance
- Conclusion

Benchmarks

- Shared counter
  - Simple checkpoint
  - No computation
- NFS: Apache httpd build
  - Complex checkpoint
  - Significant computation

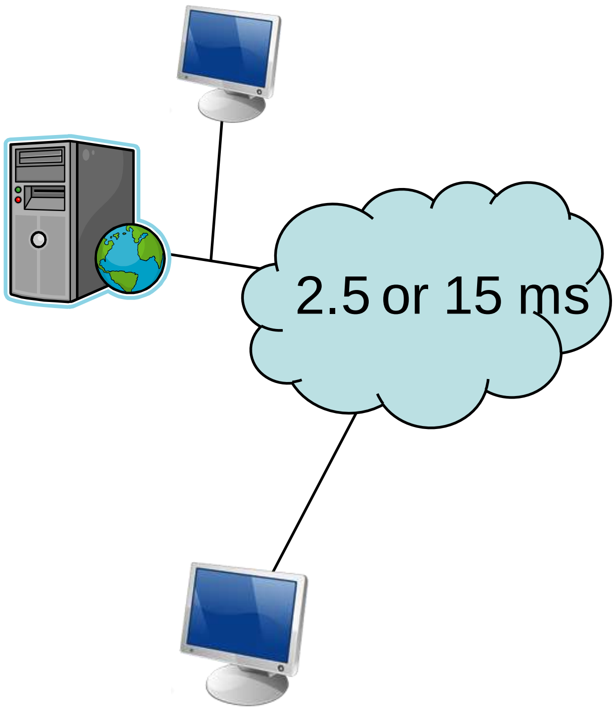
Topology

1. Primary-local
2. Primary-remote
3. Uniform

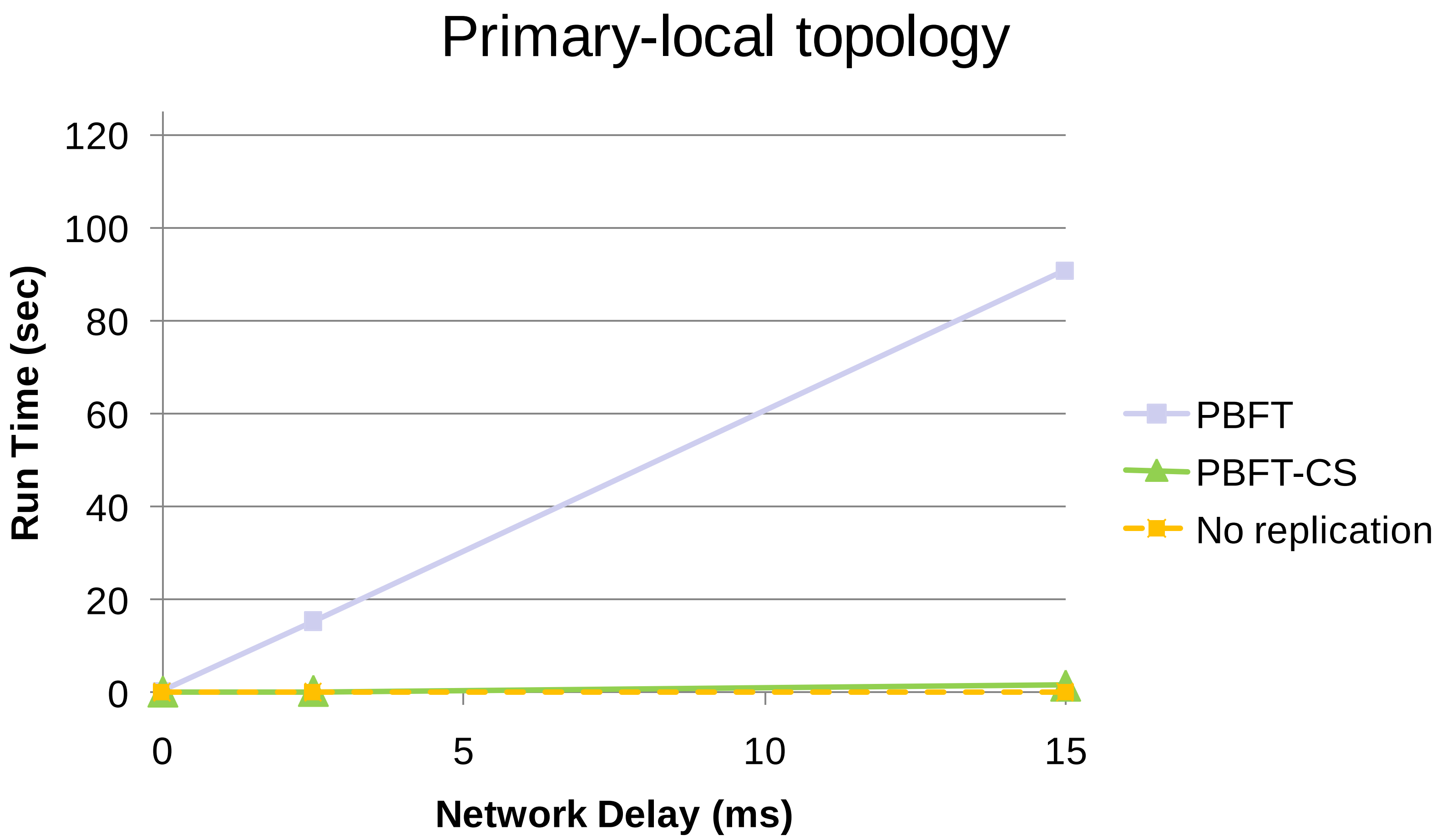


Base case: no replication

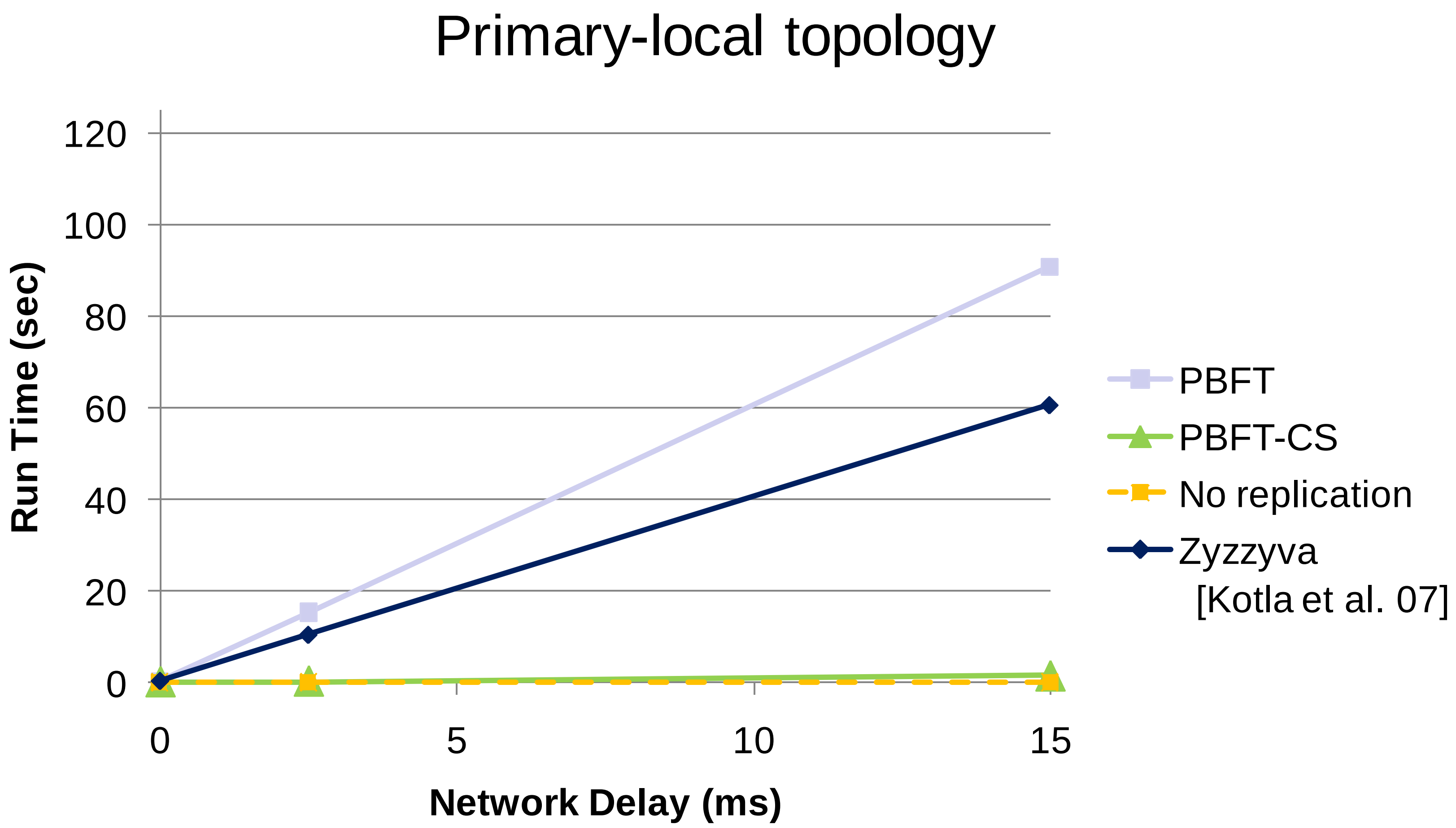
1. Primary-local
2. Primary-remote
3. Uniform



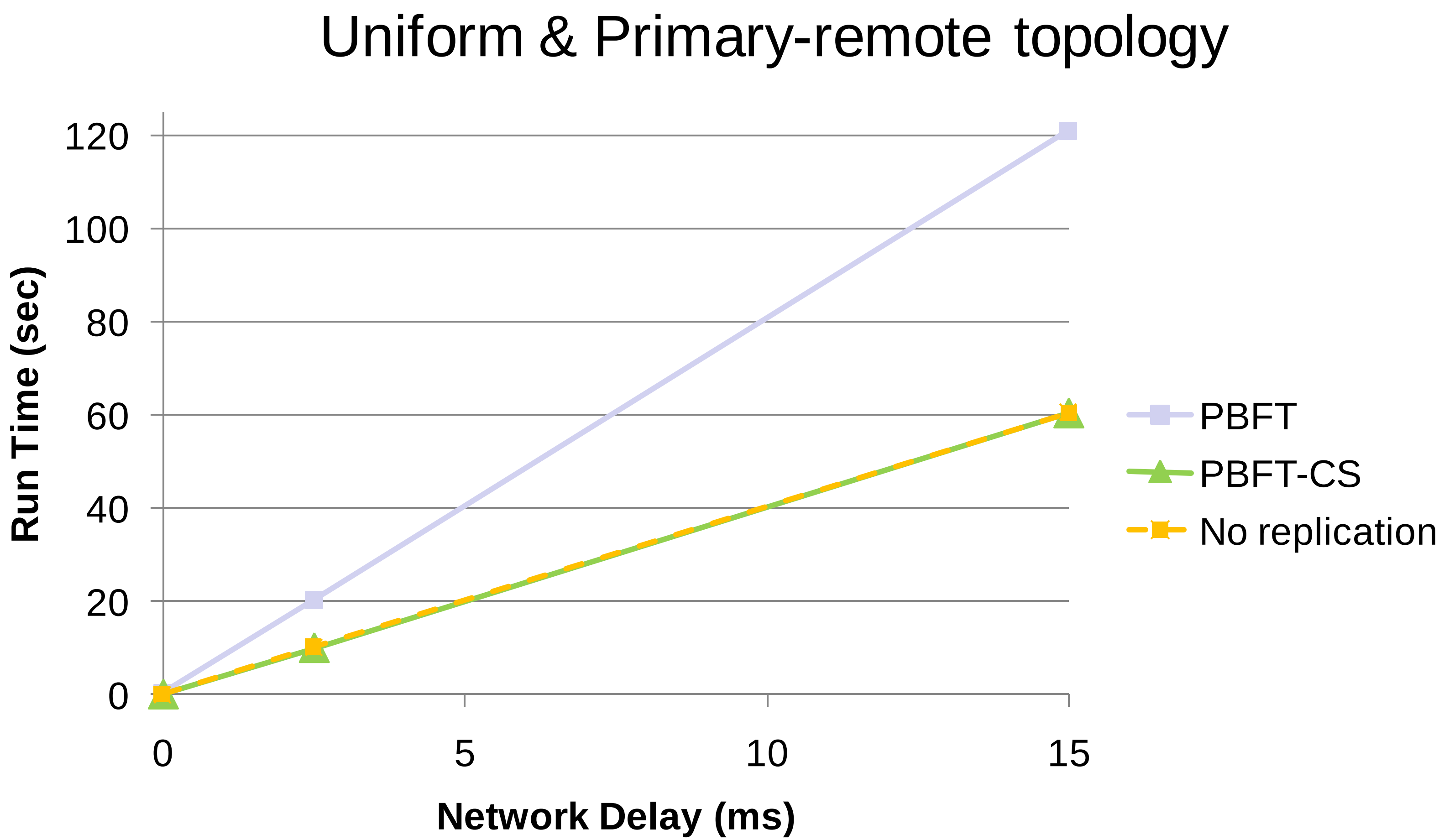
Shared Counter



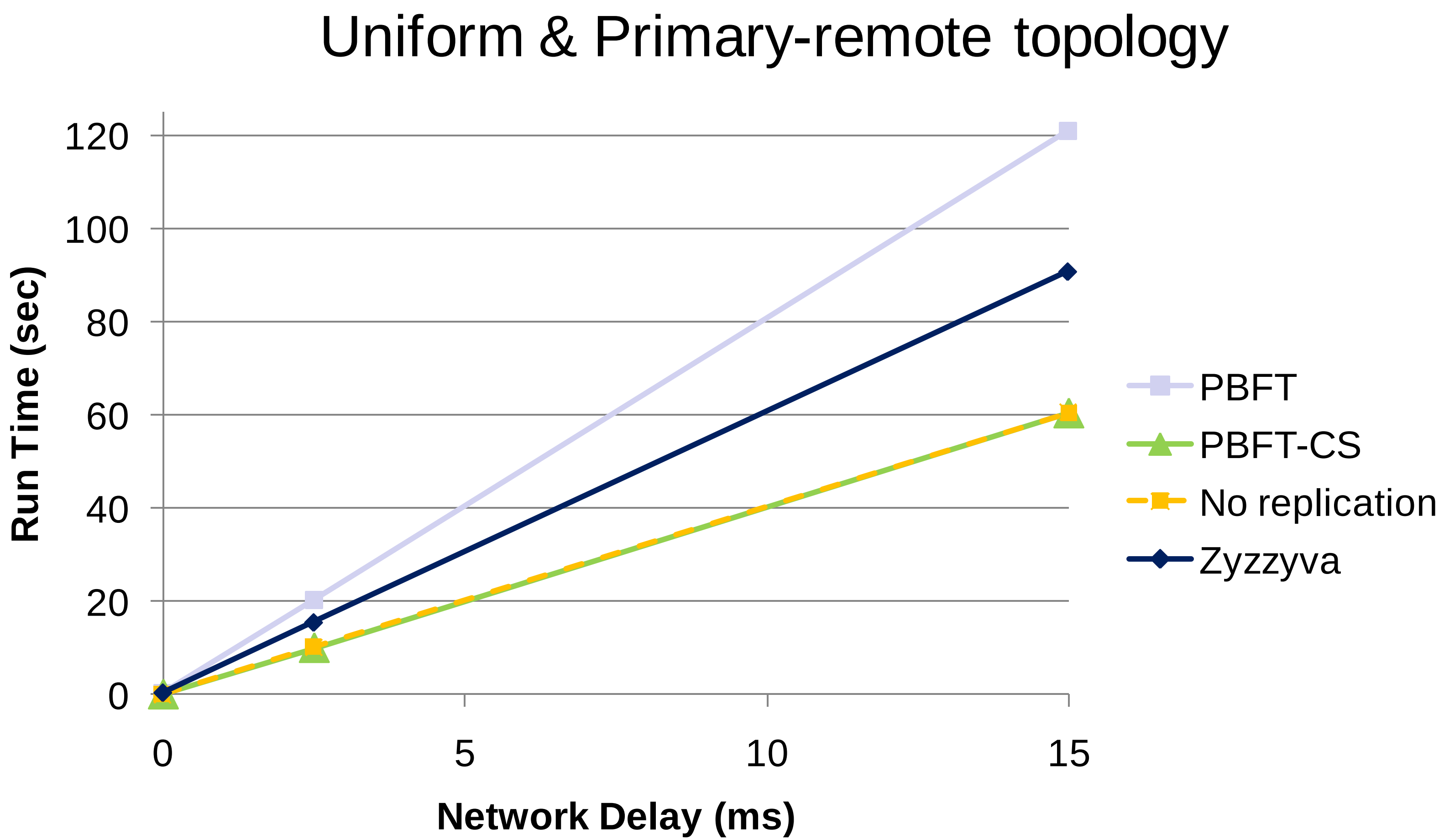
Shared Counter



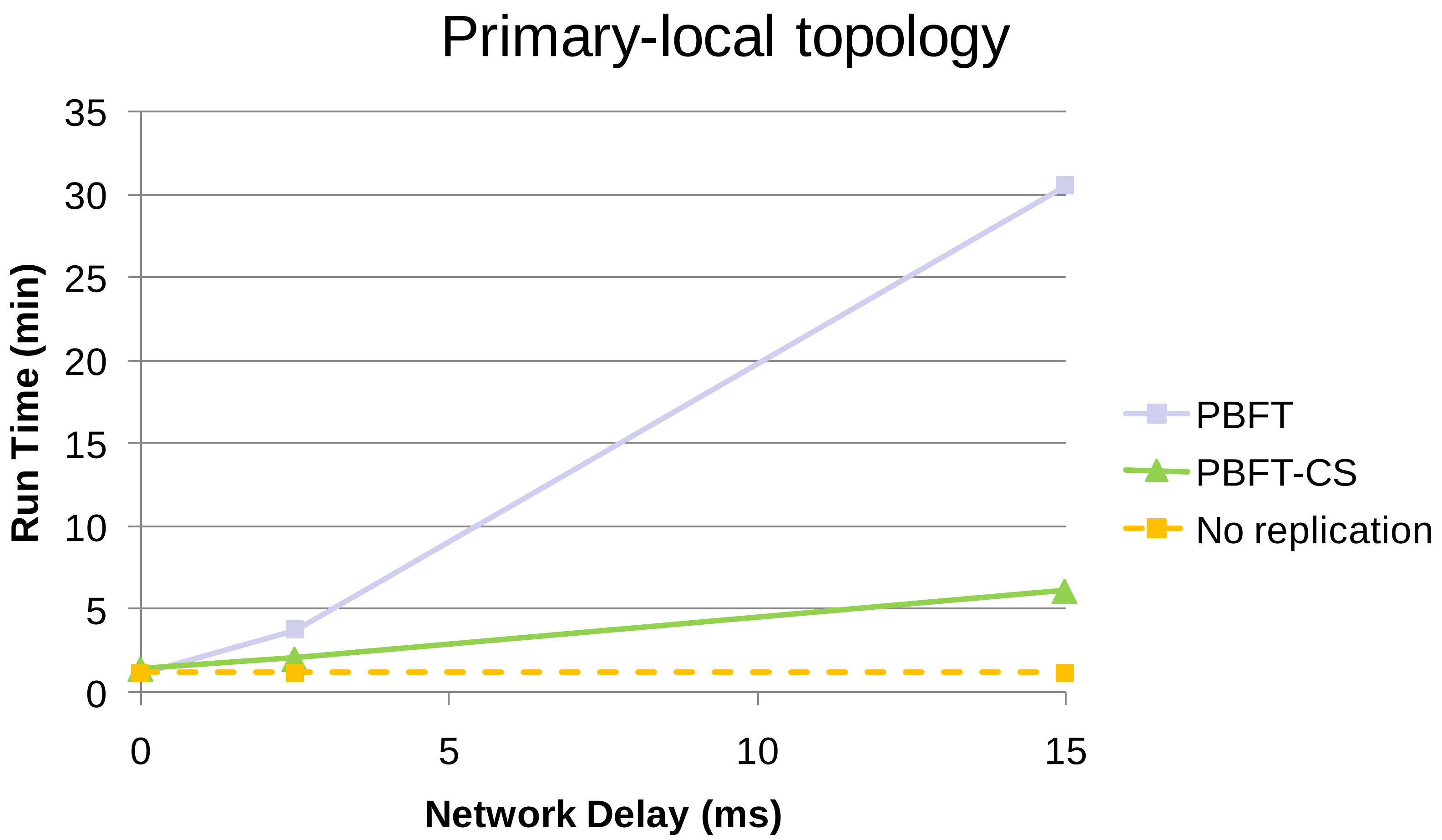
Shared Counter



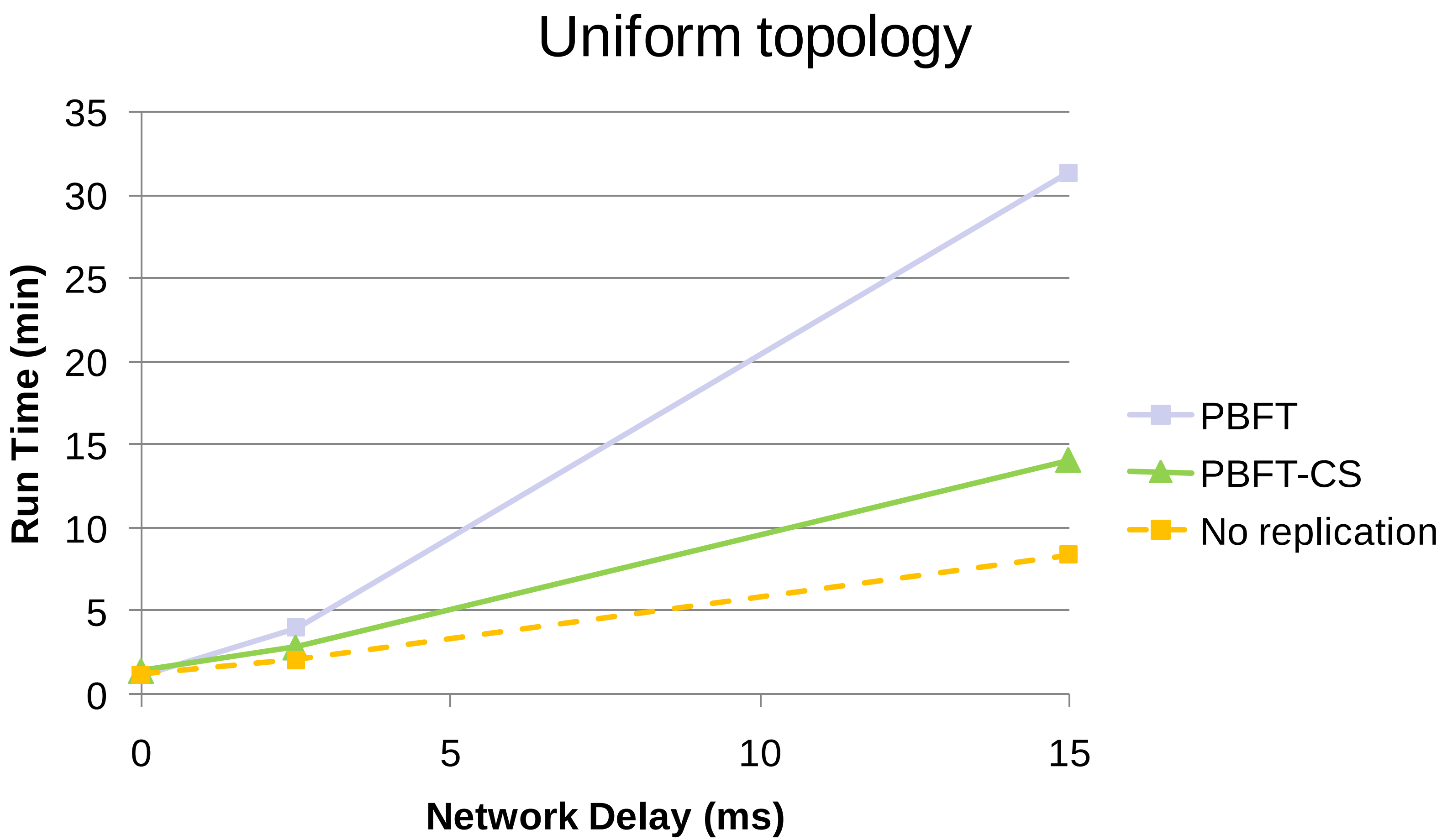
Shared Counter



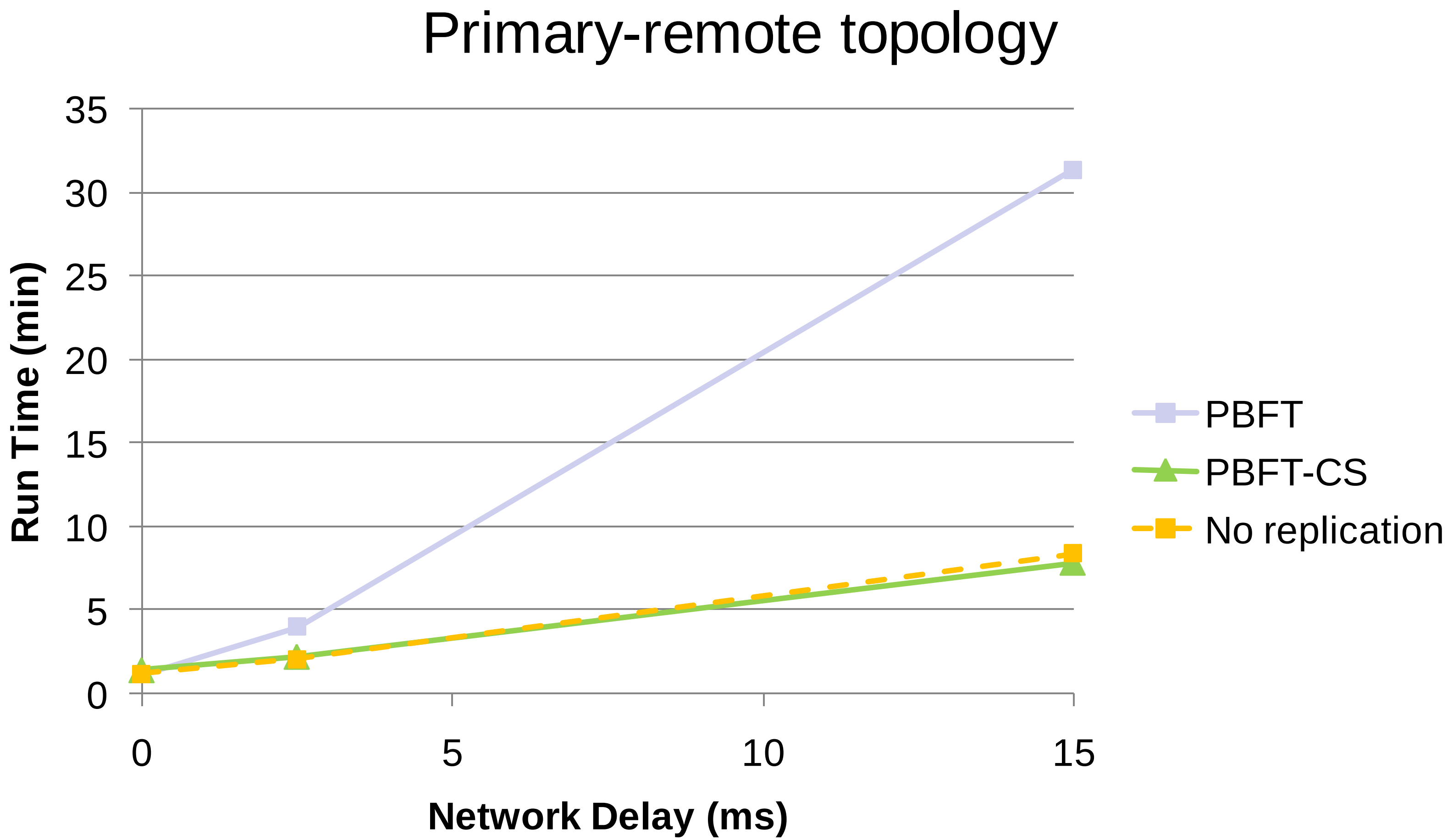
NFS: Apache build



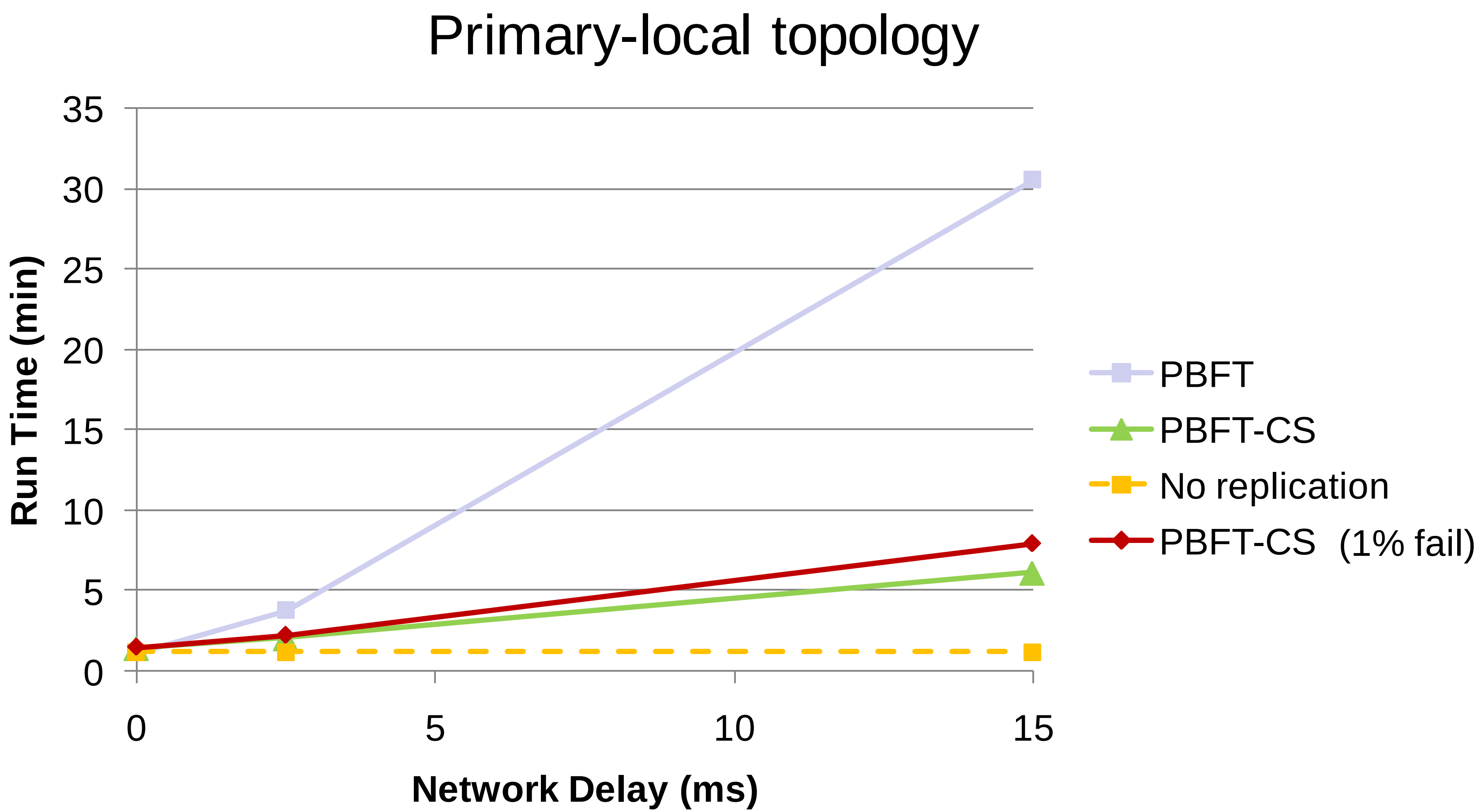
NFS: Apache build



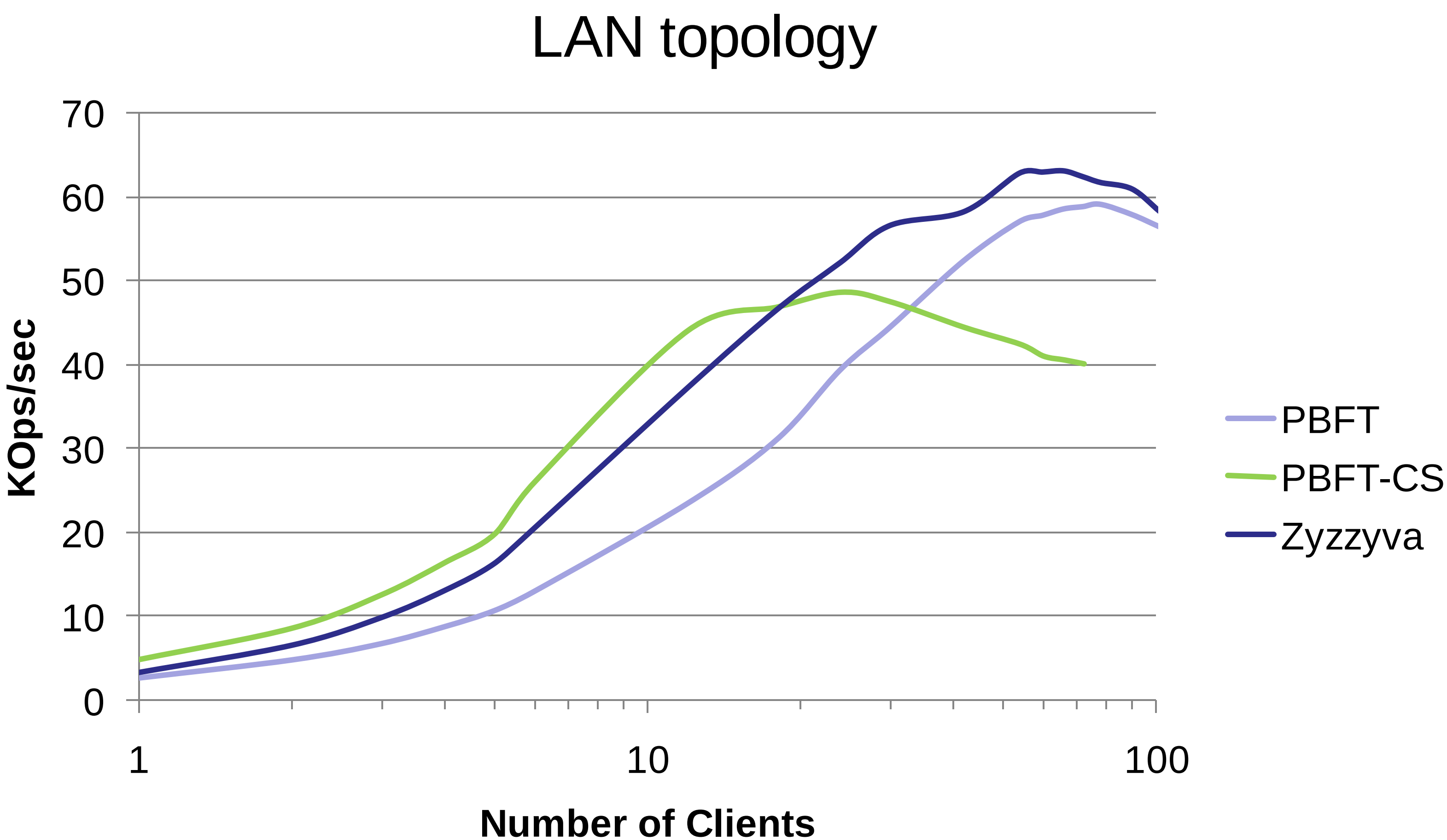
NFS: Apache build



NFS: With Failure



Throughput (Shared Counter)



Conclusion

- Integrate client speculation within RSMs
- Predicated requests: performance without complexity
- Clients less sensitive to latency between replicas
- 5x speedup over non-speculative protocol

Makes WAN deployments more practical