# Lowering the Overhead of Nonblocking Software Transactional Memory

Virendra J. Marathe
Michael F. Spear
Christopher Heriot
Athul Acharya
David Eisenstat
William N. Scherer III
Michael L. Scott

UNIVERSITY OF
ROCHESTER
COMPUTER SCIENCE

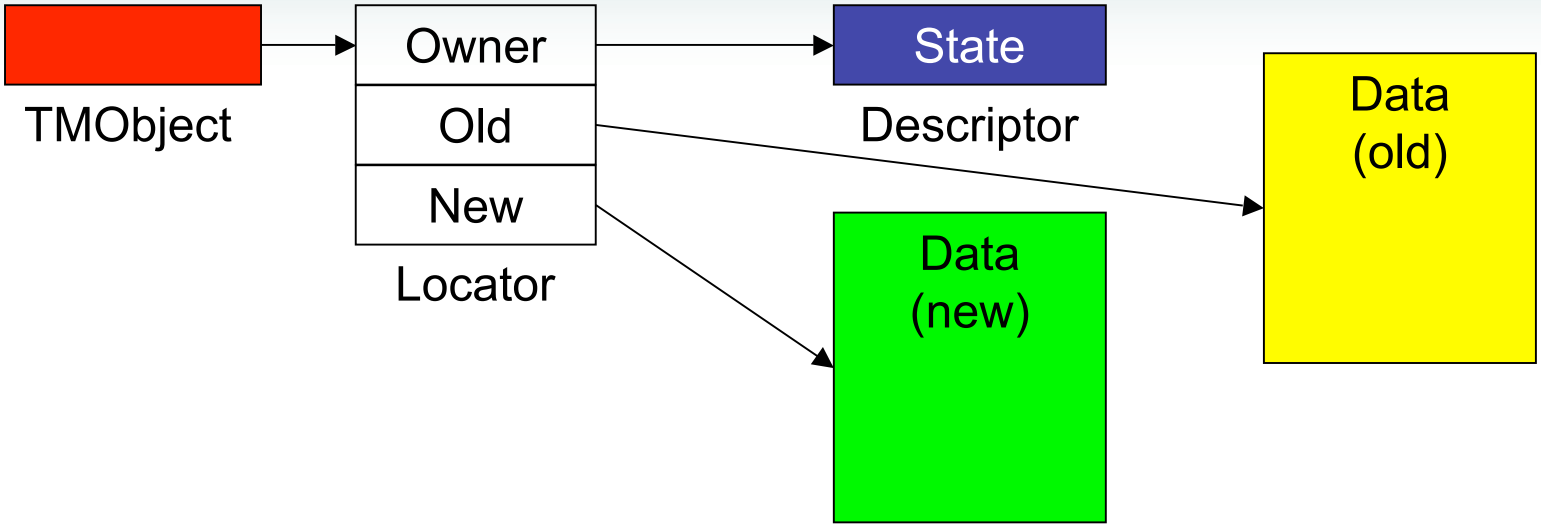
# Background

- Hardware support for managed code STMs is a daunting task
- C/C++ users need a fast nonblocking STM library
- The larger community needs STM libraries that are free and unencumbered by license restrictions
- RSTM:  a fast, free, pthreads STM library

# Outline

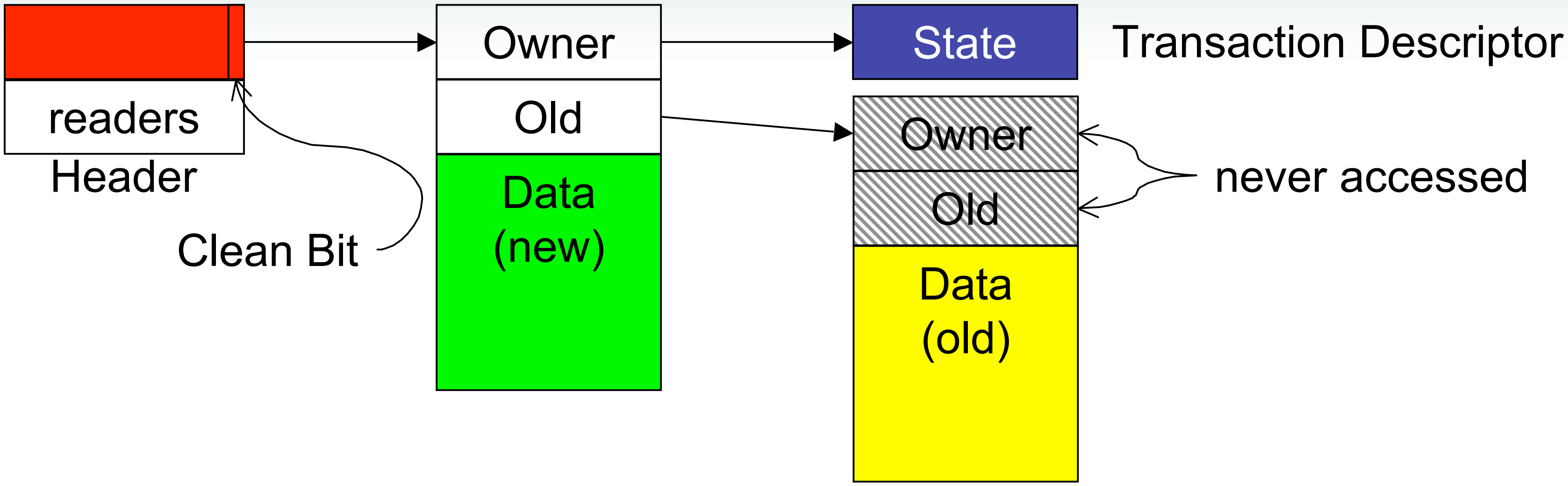
- Reducing indirection
- Limiting heap use
- Fast, flexible conflict detection
- Performance
- Future work
- Conclusions

# Indirection Costs



- Basic DSTM / ASTM / SXM organization
  - Adds 2 levels of indirection
  - Adds 3 pointer dereferences to access data
    - Up to 4 cache misses to determine valid version

# Reducing Indirection

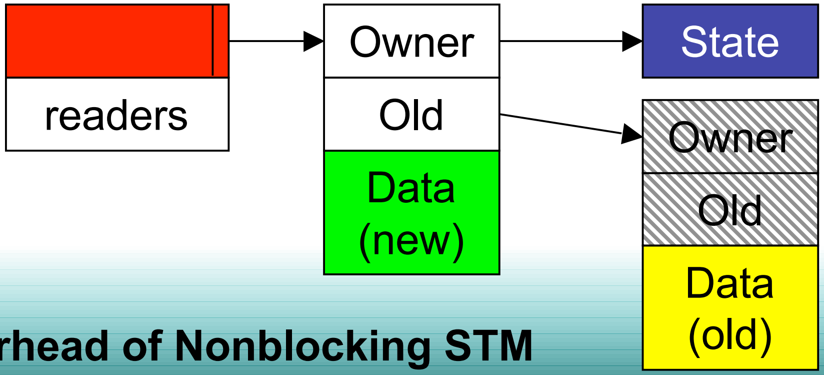


- Adds up to 2 levels of indirection
- Adds up to 3 dereferences
  - Unacquired objects: 1 dereference
  - Committed owner: 2 dereferences
  - Aborted owner: 3 dereferences

4 cache misses only on dirty, aborted owner

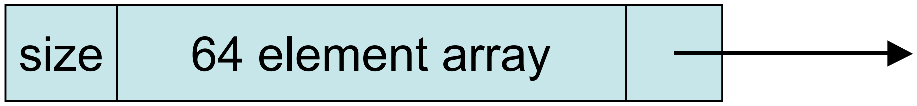
# Reusing Heap Objects

- Reference counting descriptors risks a cache miss on every decrement
- At transaction end, RSTM cleans up all pointers to the descriptor
  - If abort, install clean header pointing to old object
  - If commit, install clean header pointing to new object
  - Most headers will be in cache
  - Appropriate data objects marked for lazy reclamation



# Preallocation

- Initial read/write sets are fields in descriptor
  - Dynamic allocation only if set > 64 items
- Sets optimized for iteration
  - Every method that may do a lookup also does a full validation
  - Predict result of lookup, then verify it during the validation
  - High locality during iteration
  - Similar to McRT's Sequential Store Buffers [PPoPP 06]



size

64 element array

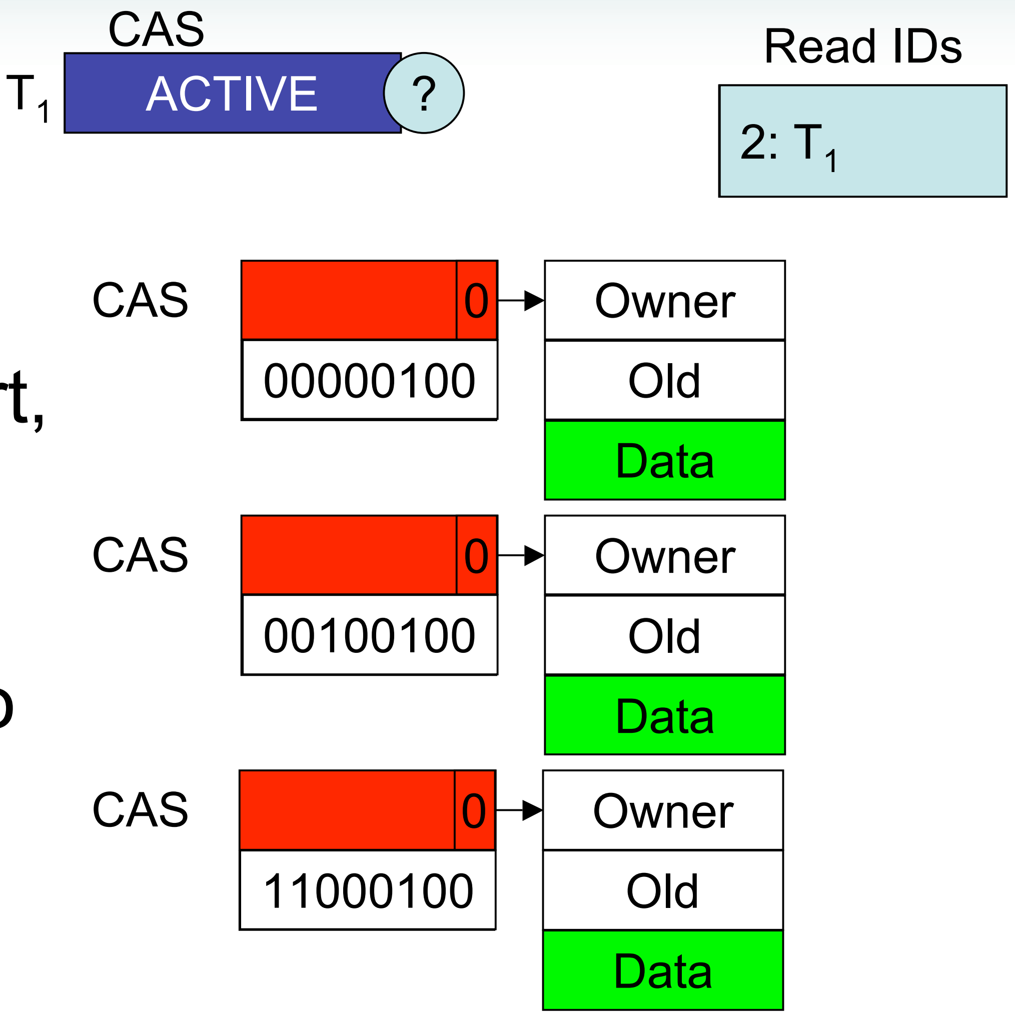
# Conflict Detection

- “Eager” and “Lazy” acquire are straightforward
- What about “Visible” readers?
  - Saves validation overhead, allows writer-reader arbitration
  - Typical implementation is as field in locator; visible reader list is modified atomically as part of header
    - Increases heap use and takes time to get memory, construct locator, and CAS it in
- Simpler solution via bitmap
  - Limits # visible readers
  - Allows (rare) spurious aborts
  - No memory management required

# RSTM Visible Readers

1. Get ReaderID
2. On open_RO(), set bit
3. On commit/abort, clear read bits

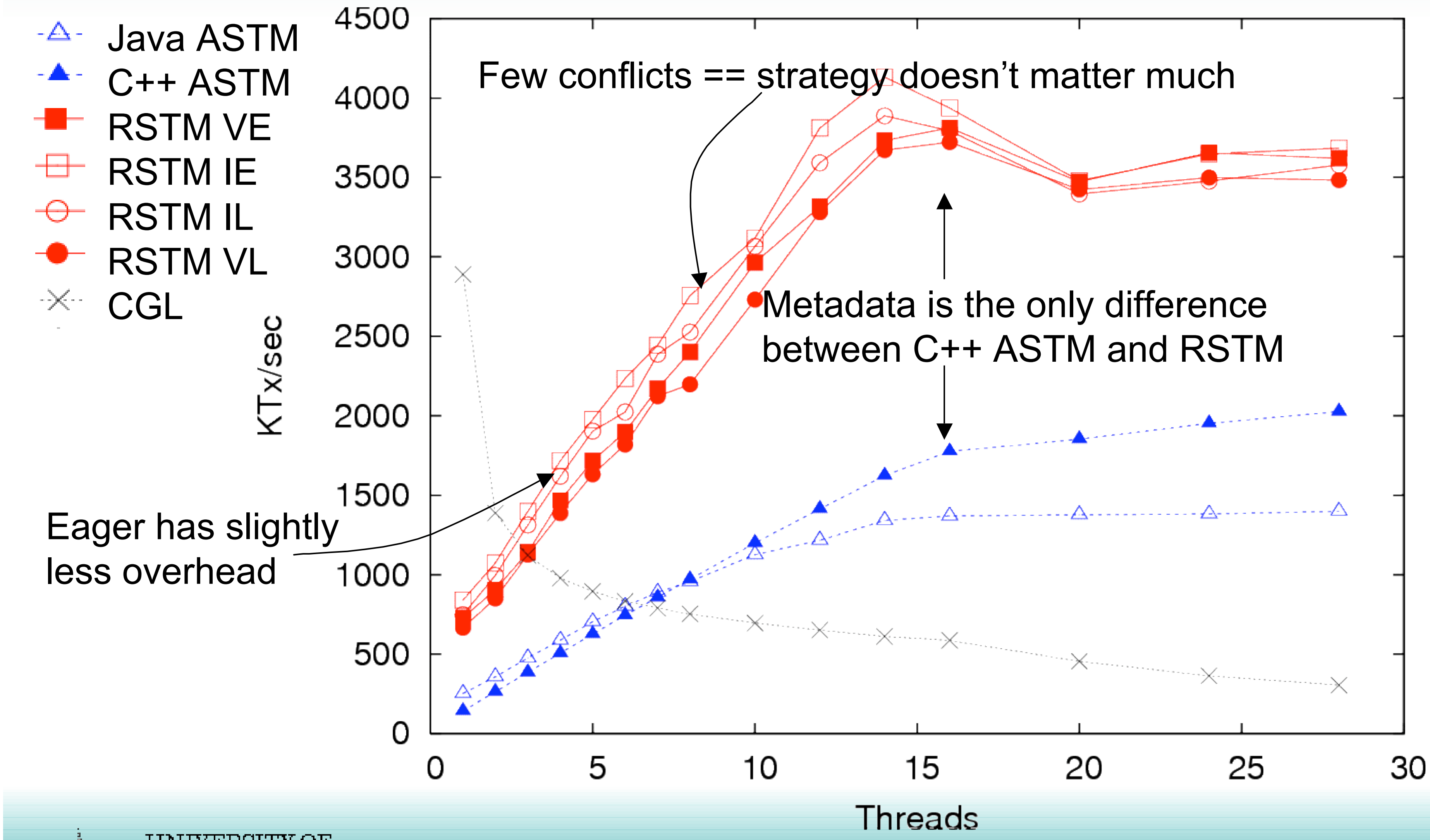
2n CAS instrs to read n objects



# RSTM Performance

- Tests conducted on 16-processor SunFire 6800
- Always outperforms Java ASTM
- C++ ASTM implementation shows that language is less important than metadata and conflict detection policy
- No single conflict detection policy is best

# HashTable (embarrassingly parallel)

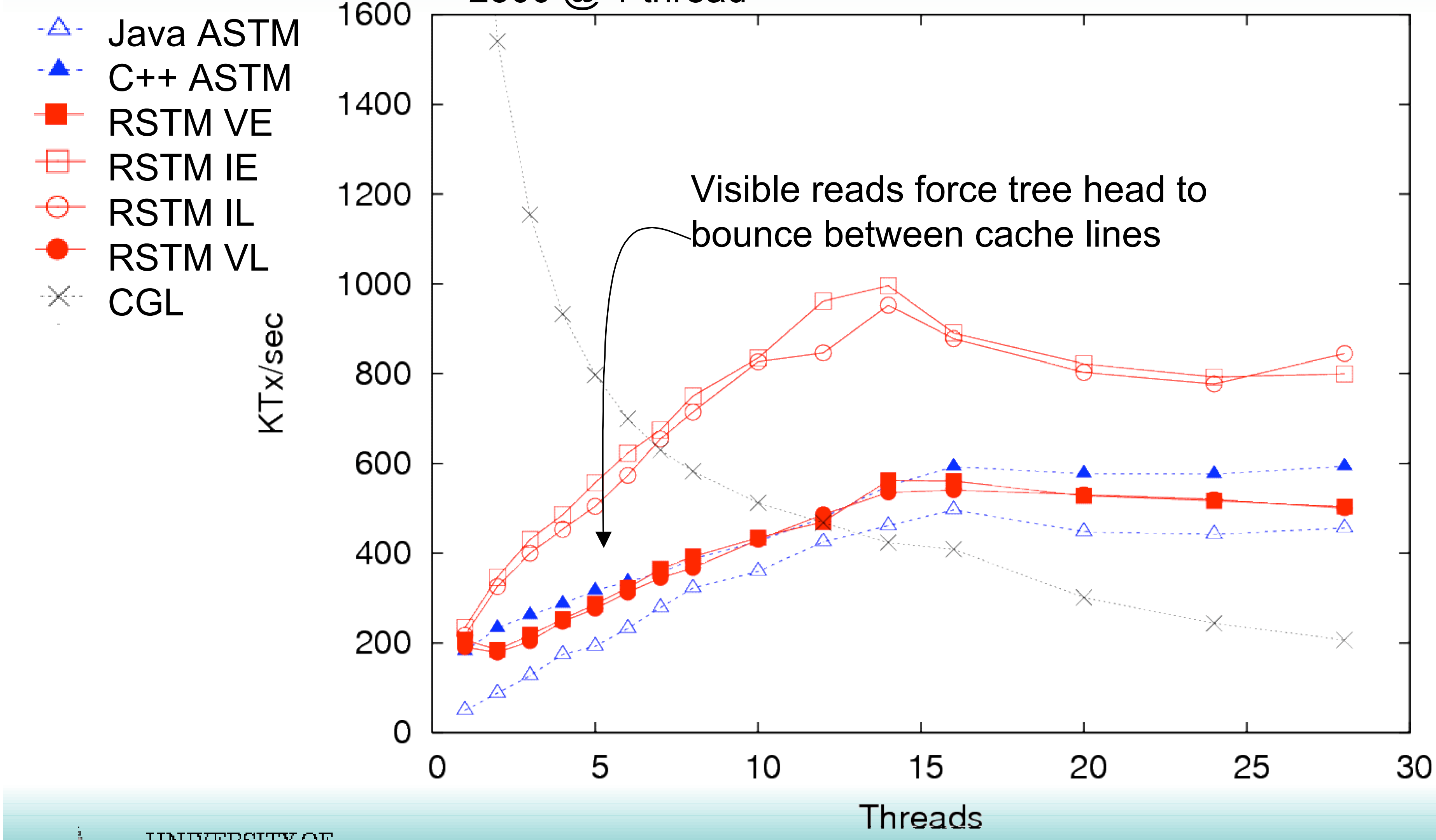


Few conflicts == strategy doesn’t matter much

Metadata is the only difference between C++ ASTM and RSTM

Eager has slightly less overhead

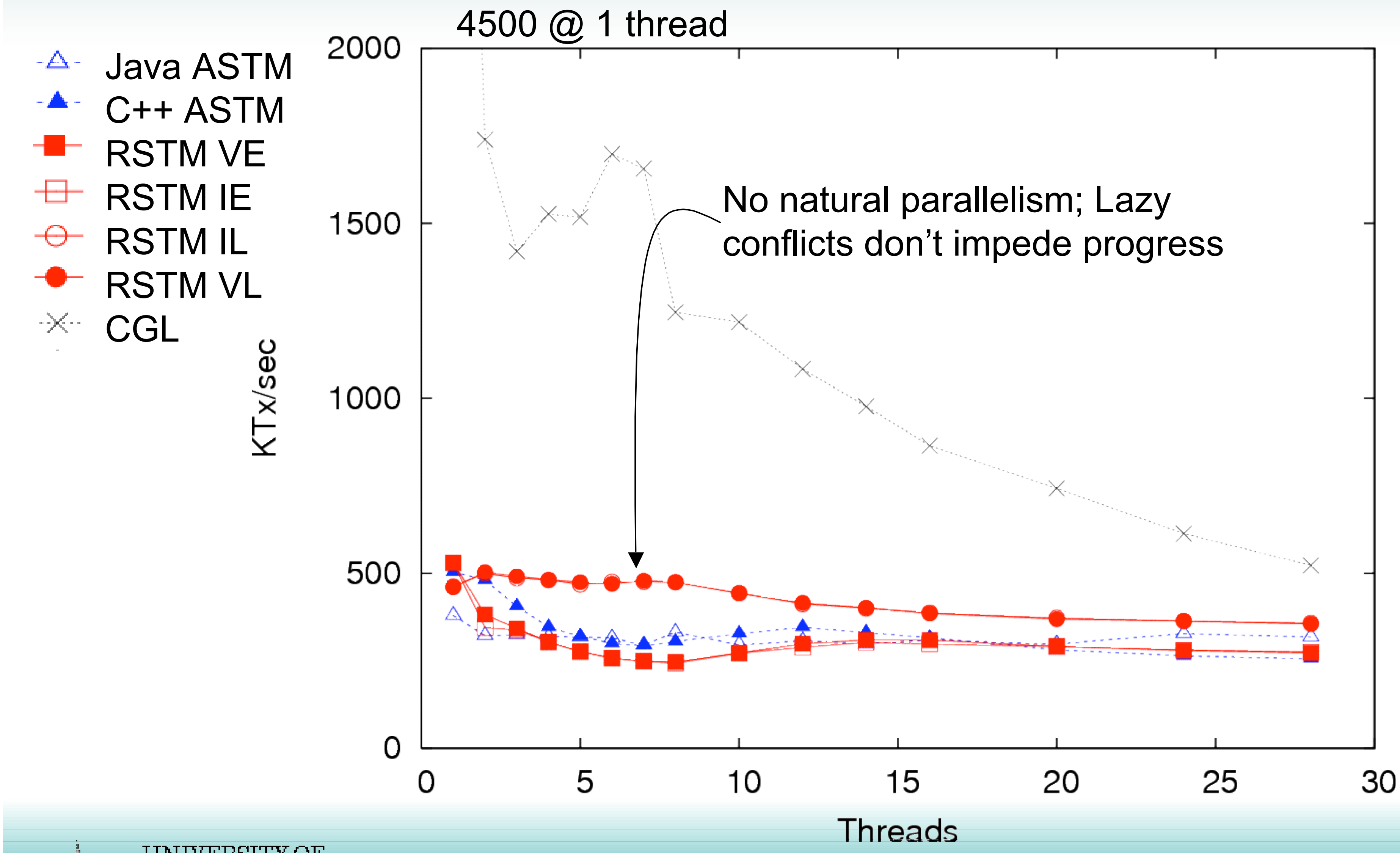
# RBTree (some conflicts)



2500 @ 1 thread

Visible reads force tree head to bounce between cache lines

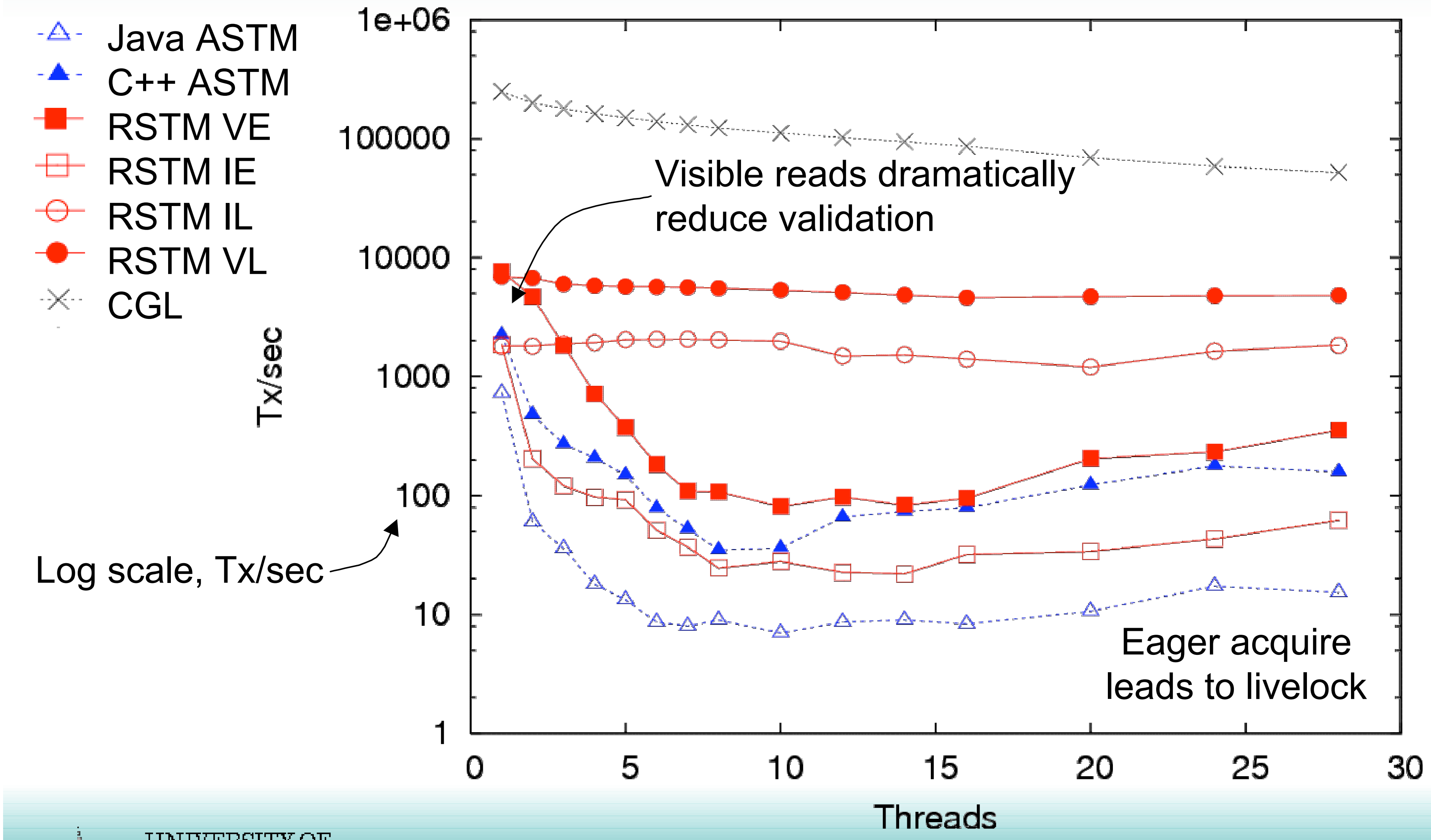
# LFUCache (no parallelism)



4500 @ 1 thread

No natural parallelism; Lazy conflicts don't impede progress

# RandomGraph (torture test)



Visible reads dramatically reduce validation

Log scale, Tx/sec

Eager acquire leads to livelock

# Future Work

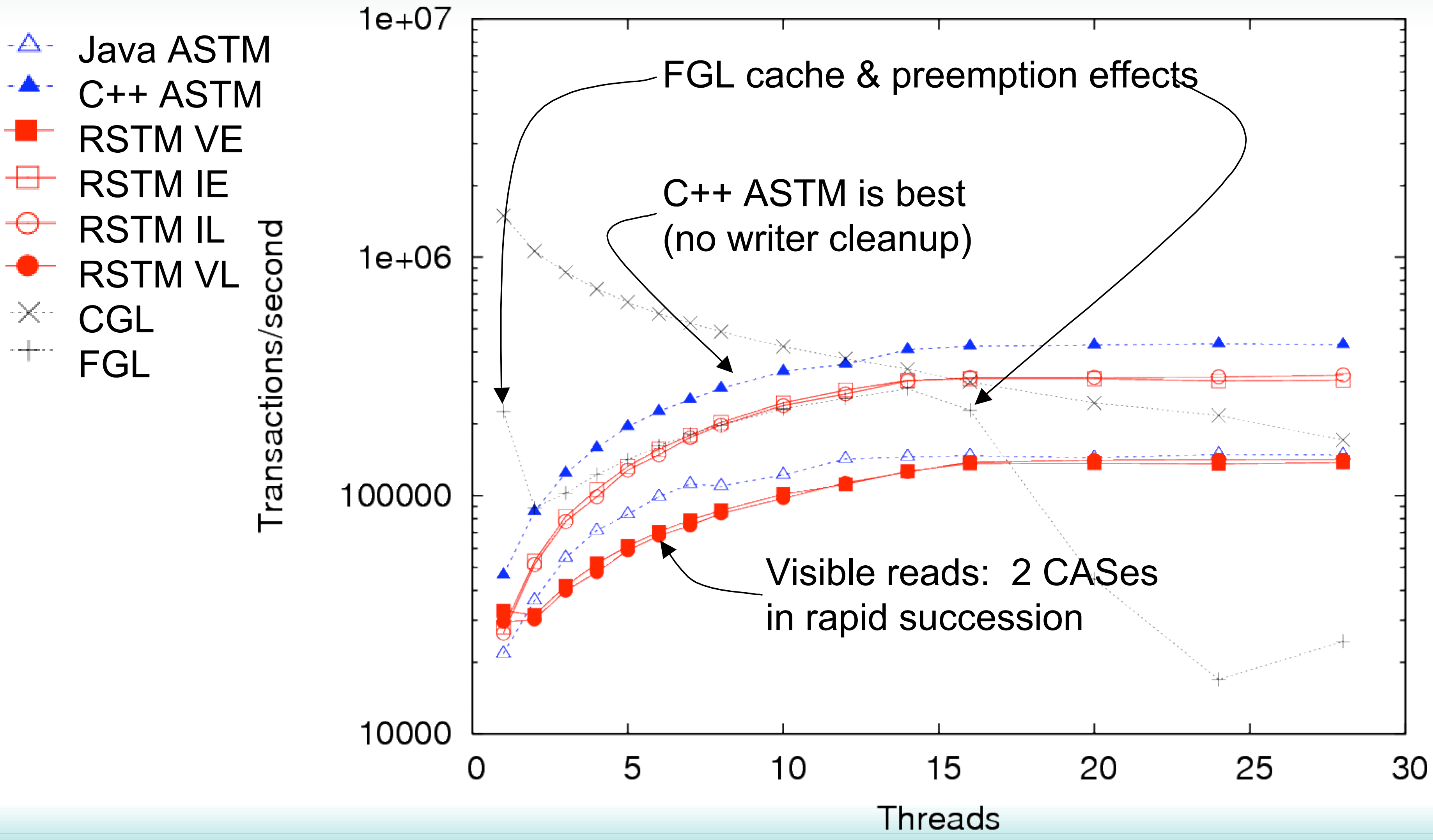
- Adaptation between lazy and eager, visible and invisible
  - Architectural implications…Intel Xeon, Sun Niagara have very different CAS overheads
- Avoiding validation with heuristics
- Mixed invalidation
- Hardware assistance

# Summary

- Better metadata organization reduces cache misses
- Limiting dynamic memory management helps
- Conflict detection is workload dependent
- Download RSTM for SPARC/Solaris at http://www.cs.rochester.edu/research/synchronization/rstm/ (check back soon for x86/Linux version)

# Supplemental Material

# Linked List with Early Release



FGL cache & preemption effects

C++ ASTM is best
(no writer cleanup)

Visible reads: 2 CASes
in rapid succession