

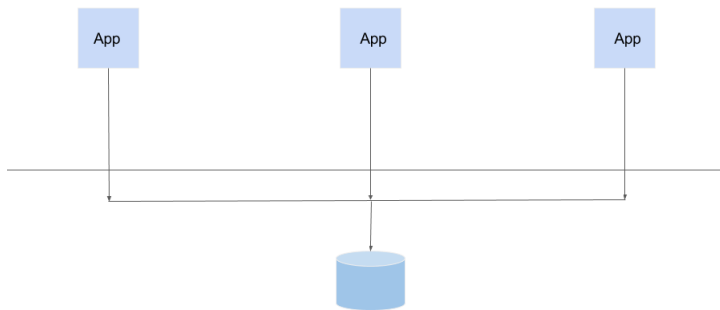
# Highly-scalable Concurrent Objects

Deepthi Akkoorath<sup>1</sup>    Annette Bieniusa<sup>1</sup>

<sup>1</sup>University of Kaiserslautern  
AG Software Technology

18. Apr 2016

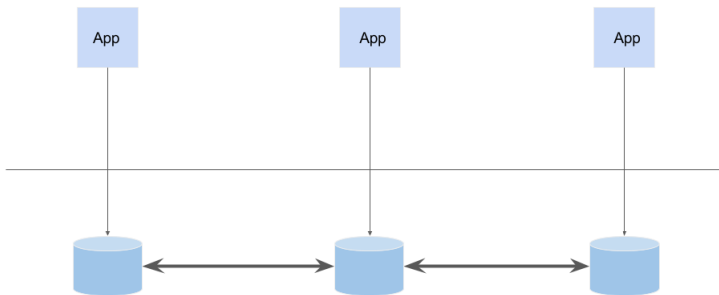
# Concurrent Shared-Memory Systems



Correctness = Linearizability  $\implies$  Synchronization  
Scalable?

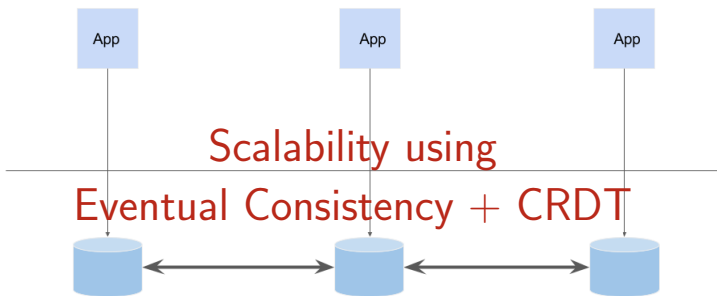
# Shared Objects

## Geo-Replicated Systems



# Shared Objects

## Geo-Replicated Systems



# Shared Objects

Shared Memory Systems / Multi-core

- Many Scalable Data-structures exists
- Lock-free data-structures using Transactional Memory
- Scalable under contention?

# Shared Objects

Shared Memory Systems / Multi-core

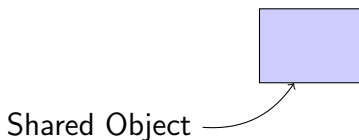
- Many Scalable Data-structures exists
- Lock-free data-structures using Transactional Memory
- Scalable under contention?

Weakening Linearizability  $\implies$  Scalability?

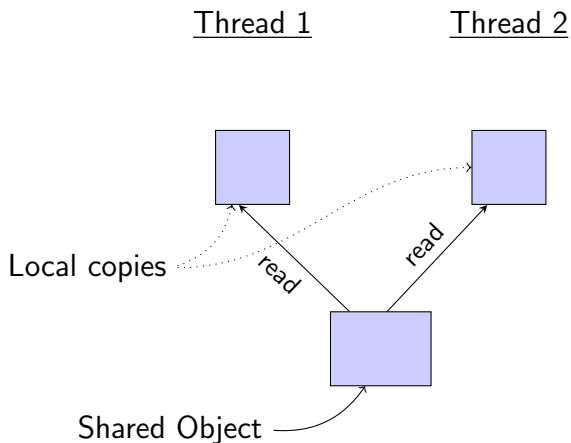
# Mergeable Objects in Shared Memory Systems

Thread 1

Thread 2

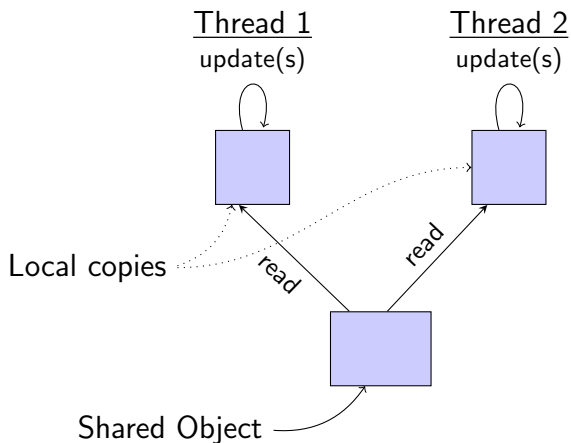


# Mergeable Objects in Shared Memory Systems

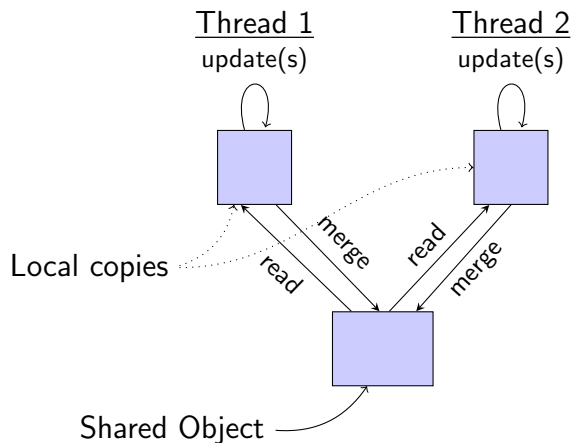




# Mergeable Objects in Shared Memory Systems

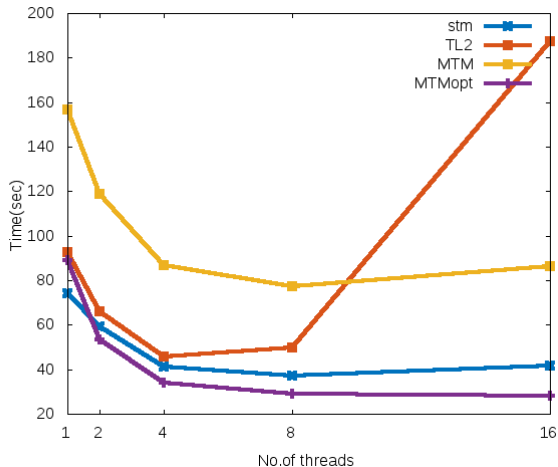


# Mergeable Objects in Shared Memory Systems



# Mergeable Transactions

## Mergeable Objects in STM



# CRDTs as Mergeable Objects

- Example: G-Set
  - a Set
  - Merge = union of two sets
- Example: G-Counter
  - A vector
  - Merge = max of each element in the vector

# CRDTs as Mergeable Objects

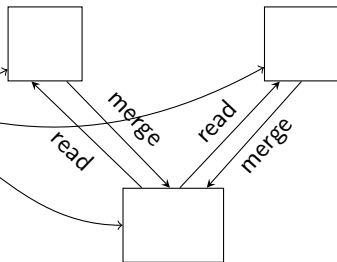
- Example: G-Set
  - a Set
  - Merge = union of two sets
- Example: G-Counter
  - A vector
  - Merge = max of each element in the vector

CRDT Merge is Expensive

# Properties of Mergeable Objects

## Persistence

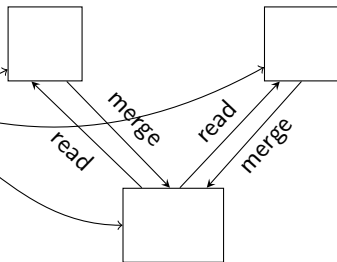
- Multiple Versions co-exists
  - Local Versions
  - Global Version
- A **data structure is persistent** if multiple versions of it are accessible



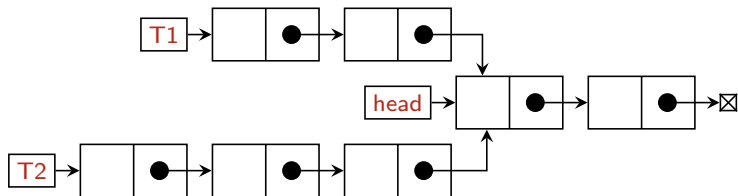
# Properties of Mergeable Objects

## Persistence

- Multiple Versions co-exists
  - Local Versions
  - Global Version
- A **data structure is persistent** if multiple versions of it are accessible
- Copy entire Object?



# Persistence in List





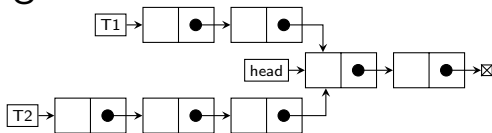
# Mergeability

- Semantic Mergeability
  - Exploit object semantics (e.g. idempotence, commutativity) to obtain merge operation
  - Similar to CRDTs [Shapiro et. al '12]
- Structural Mergeability
  - How efficiently two versions can be merged
  - Share (parts of) the data structure between concurrent threads
  - Merge by modifying the data structure

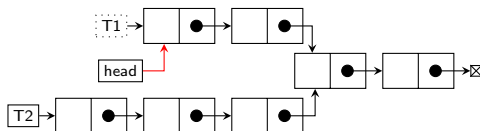
# Mergeable Counter

- 2 Integers:  $(G,L)$
- increment =  $(G, L+1)$
- merge  $(G,0) (G', L) = (G+L, 0)$
  
- Persistence by “Copy object”
- Semantic Merge by arithmetic properties

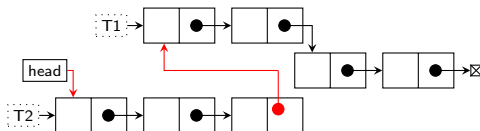
# Add only Bag



After  $T1$  commits:

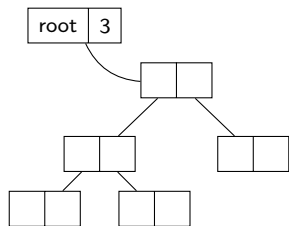


After  $T2$  commits:



# OR-Set

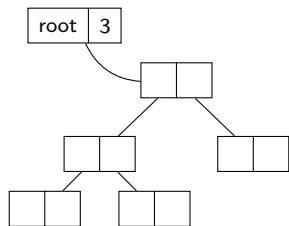
## Binary Search Tree



- Node = (Val, Version info)
- Version info =  $\langle first, last, removed \rangle$
-

# OR-Set

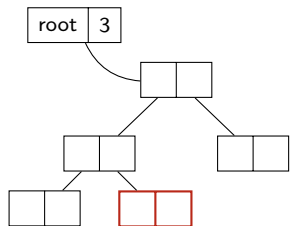
## Binary Search Tree



- Node = (Val, Version info)
- Version info =  $\langle first, last, removed \rangle$
- Lookup

# OR-Set

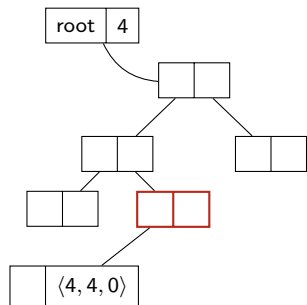
## Binary Search Tree



- Node = (Val, Version info)
- Version info =  $\langle first, last, removed \rangle$
- Add

# OR-Set

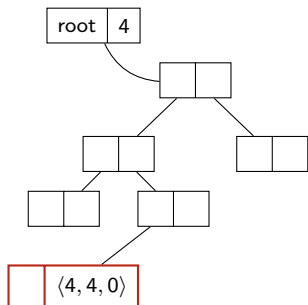
## Binary Search Tree



- Node = (Val, Version info)
- Version info =  $\langle first, last, removed \rangle$
- Merging Adds

# OR-Set

## Binary Search Tree

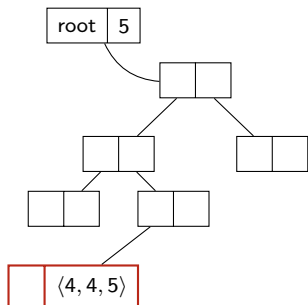


- Node = (Val, Version info)
- Version info =  $\langle first, last, removed \rangle$
- Remove



# OR-Set

## Binary Search Tree



- Node = (Val, Version info)
- Version info =  $\langle first, last, removed \rangle$
- Merging Removes

# Conclusion

- Mergeable Objects analogous to CRDTs
  - Non-linearizable
  - Meaningful merge semantics
- Properties for efficiency
  - Persistence
  - Mergeability
- Scalable?



akkoorath@cs.uni-kl.de