A Locality-Aware Software Transactional Memory

Patrick Marlier
Anita Sobe
Pierre Sutra

University of Neuchâtel

Context

State of the game

- time-based validation (global clock)
- at least 2 global operations in the critical path

Observations

- cache coherency is expensive
- locality exists in
  - in the architecture (e.g., NUMA)
  - in the workload

Our goal

- improve STM performance
  - minimize cache invalidations,
  - move data closer to where it is computed

Our approach - core components

Time reference

- non-global
- clustered clocks (core or socket)

Deferred update

- write locks
- buffer writes until commitment

Lazy snapshot based reads

- invisible reads
- try extending at each invalidated read

Our approach - algorithmic detail

At commit time:

```
if (not extend(clock(T), T)) then
        return ABORT
clock(T) = max(clock(T), clock(cluster(p))) + 1
clock(cluster(p)) = max(clock(T), clock(cluster(p)))
for all o in ws(T) do
        o  = (update, clock(T))
        unlock(o)
return COMMIT
```

Our approach - algorithmic detail

At commit time:

```
if (not extend(clock(T), T)) then
        return ABORT
clock(T) = max(clock(T), clock(cluster(p))) + 1
clock(cluster(p)) = max(clock(T), clock(cluster(p)))
for all o in ws(T) do
        o  = (update, clock(T))
        unlock(o)
return COMMIT
```

Our approach - algorithmic detail

At commit time:

```
if (not extend(clock(T), T)) then
        return ABORT
clock(T) = max(clock(T), clock(cluster(p))) + 1
clock(cluster(p)) = max(clock(T), clock(cluster(p)))
for all o in ws(T) do
        o  = (update, clock(T))
        unlock(o)
return COMMIT
```

conjecture: non-global time + invisible reads
requires O(r) operations at commit time

Results

Settings

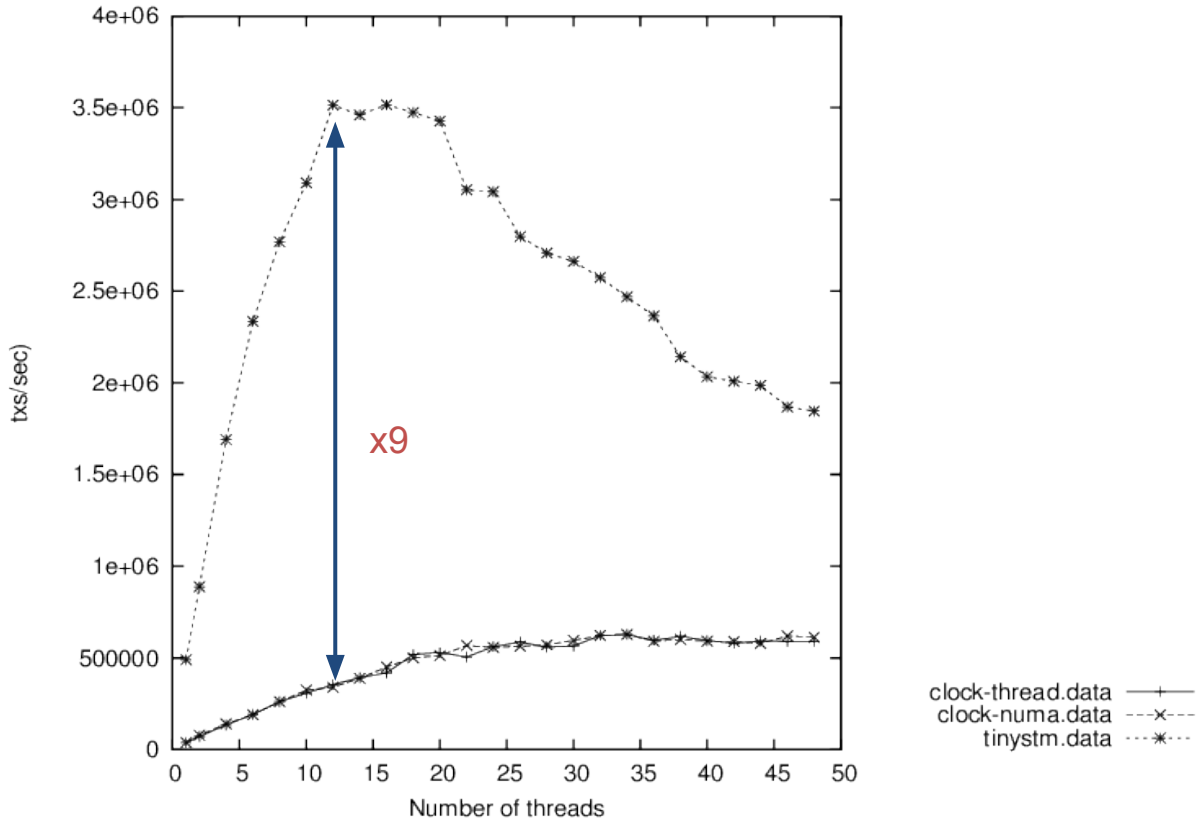
- 48 cores = 4 sockets x 12, NUMA architecture

The benchmarks

- Linked-list
  - with random search
  - probability p to write
- RB tree
  - same as above
- Bank
  - transfers between two random accounts

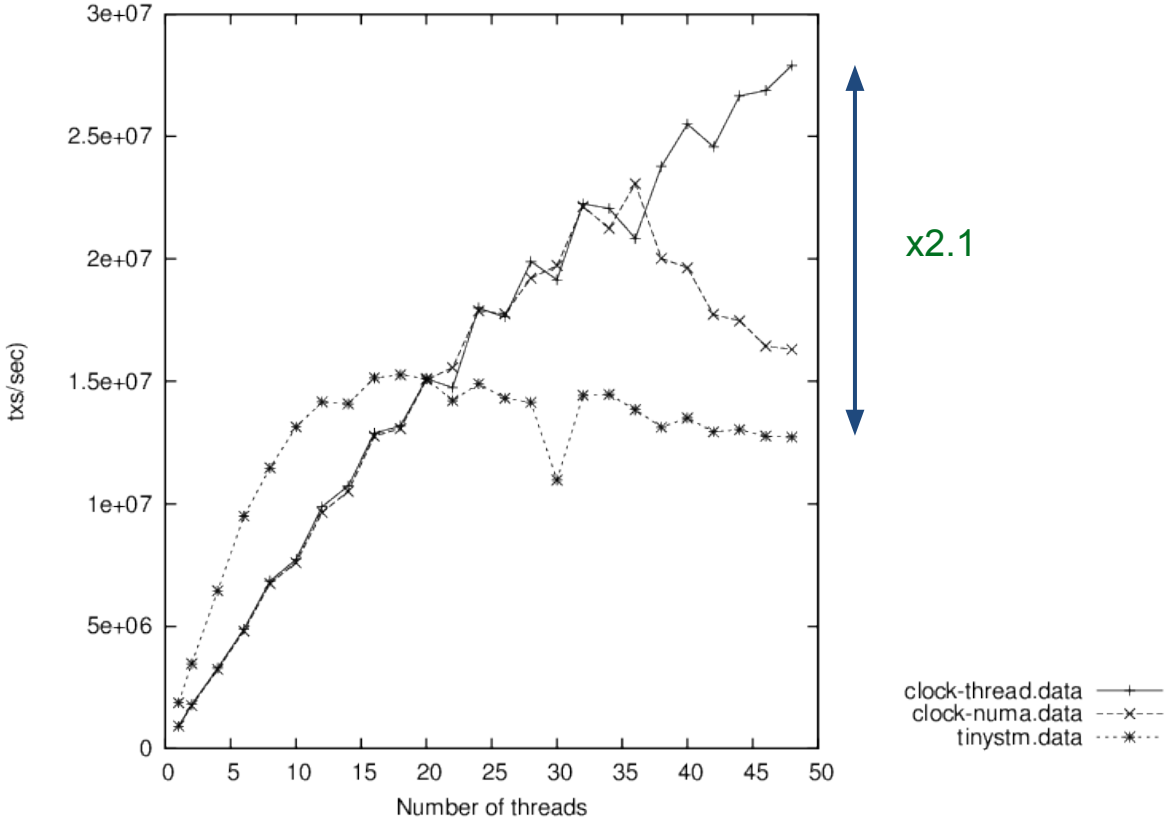
Linked list

256 elements
5% updates



RB-Tree

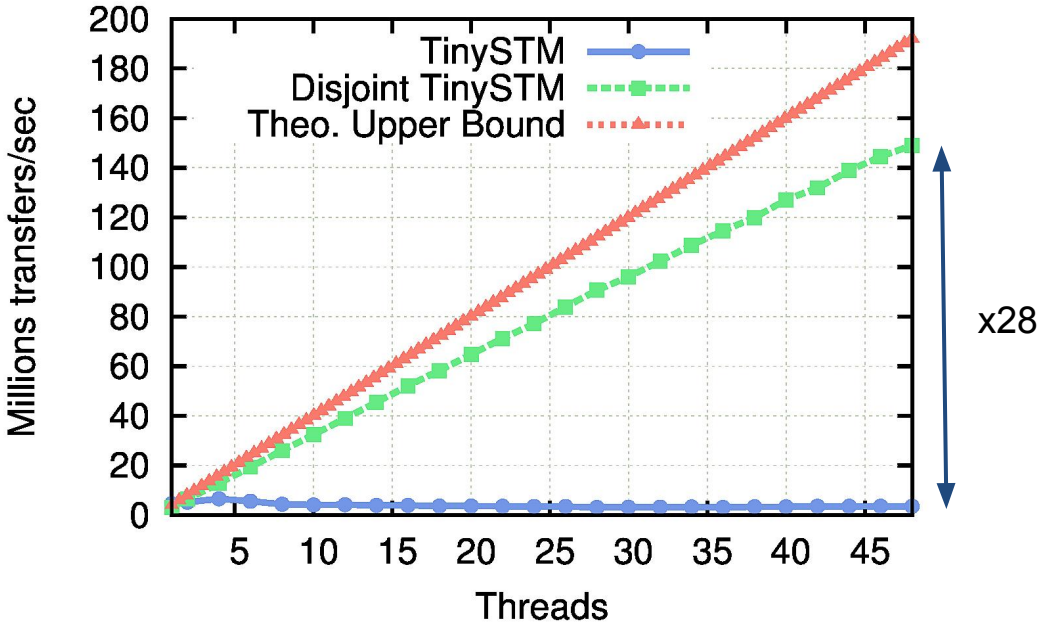
8192 elements
5% updates



Bank: motivation

Random transferts

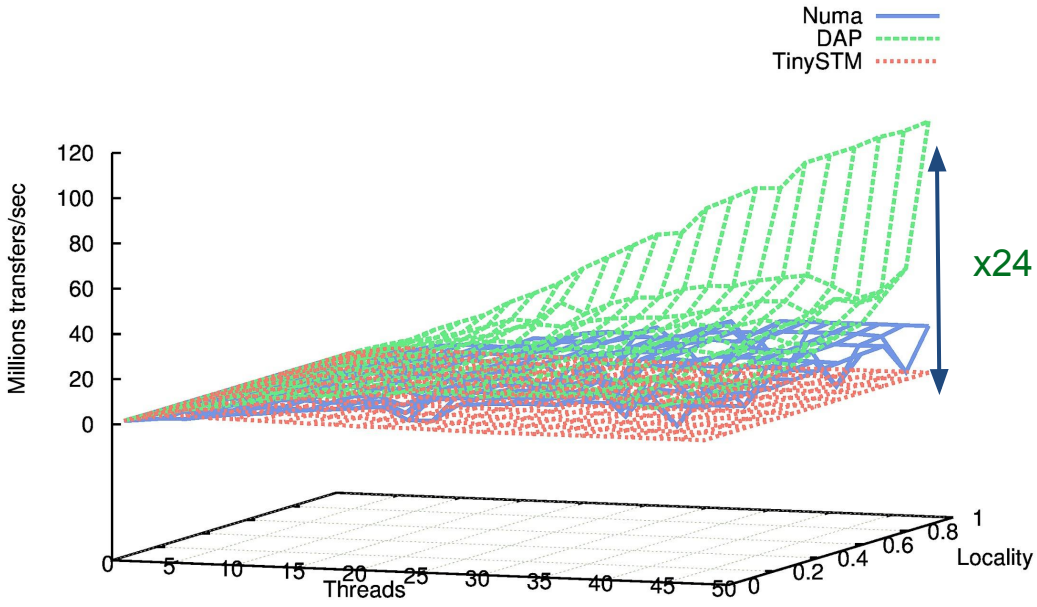
Disjoint TinySTM
- 1 STM per core
- ~ optimal parallelism



Bank: results

2M accounts

Locality = ratio of accounts
local to each core



Conclusion

Current state

- locality-aware variation of TinySTM
- clock is local to core, socket or global
- up to x24 performance of TinySTM when locality is high on 48 cores

Future work

- dynamically change time reference
- move locks (in-progress)

(find what a global clock costs)