Performance Evaluation of Adaptivity in STM

Mathias Payer and Thomas R. Gross
Department of Computer Science,
ETH Zürich

Motivation

- STM systems rely on many assumptions
  - Often contradicting for different programs
  - Statically tuned to a baseline
- Use self-optimizing systems
  - Adapt to different workloads
- What parameters can be adapted?
  - How to measure effectiveness?

Outline

- Introduction
- STM System
  - STM Baseline
  - Adaptive Parameters
- Evaluation
- Related work
- Conclusion

Introduction

- Software Transactional Memory (STM) applies transactions to memory
  - (Optimistic) concurrency control mechanism
  - Alternative to lock-based synchronization
- Multiple concurrent threads run transactions
  - Concurrent memory modifications

Introduction

- Concurrent transactions modify memory without synchronization
  - Transaction is verified after completion
  - Conflicts are detected and resolved
  - Changes committed for conflict-free transactions
  - Modifications only visible after commit

Introduction

```
withdraw {
  tmp = balance;
  tmp = tmp – 100
  balance = tmp;
}
```

```
deposit {
  tmp = balance;
  tmp = tmp + 100
  balance = tmp;
}
```

TX starts

balance in read-set

balance in

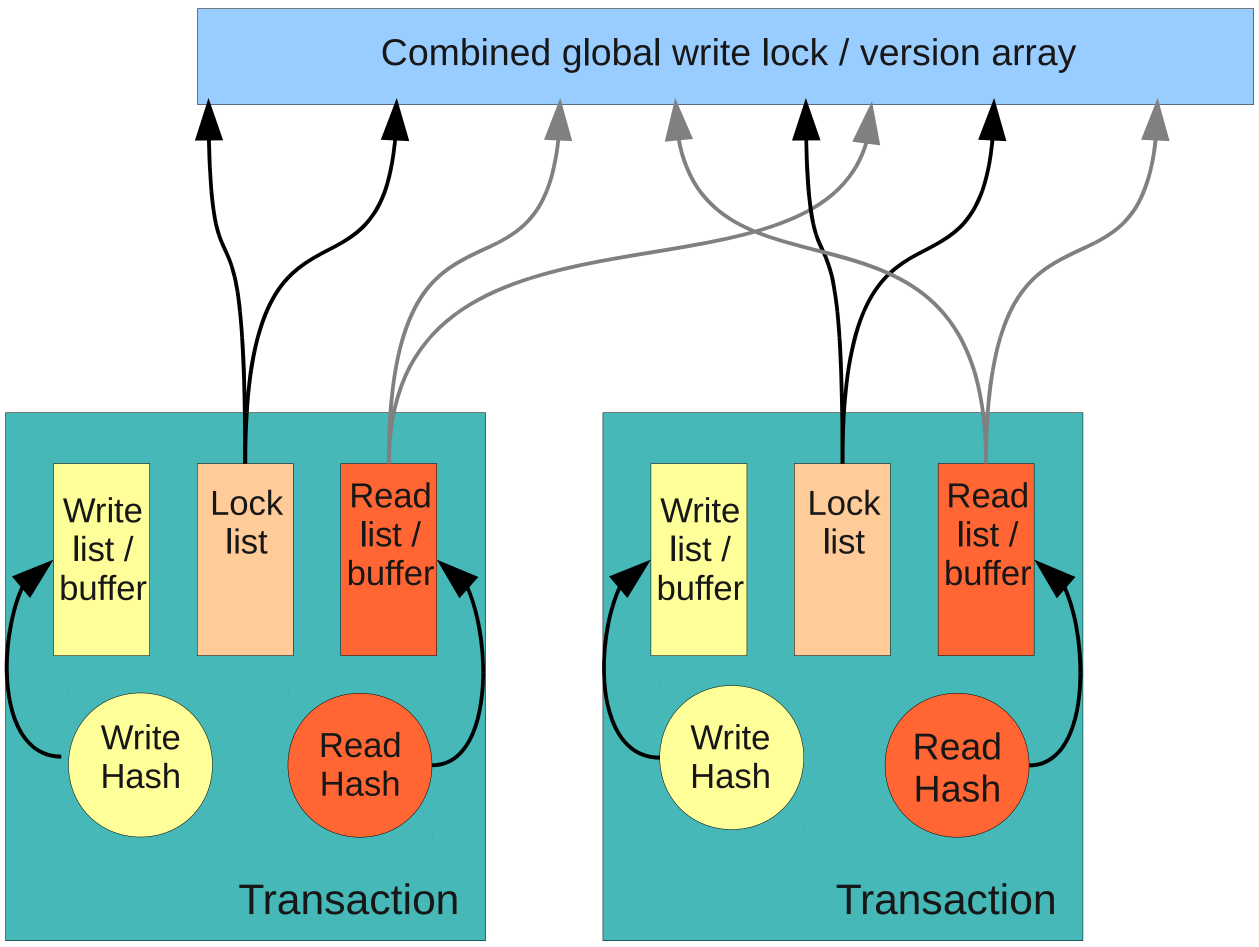
Conflict detection, data committed

- What happens when balance is accessed concurrently?
  - Either locking or STM needed to ensure correct end balance
  - STM system decides which tx is executed first

STM Baseline

- Many efficient STM implementations agree on important design decisions:
  - Word-based locking
  - Global locking / version table
  - Eager locking
  - (Almost) no contention management
  - Simple write-set and read-set implementations

STM Baseline



Adaptive STM Parameters

- Global adaptivity
  - Synchronization needed
  - Optimizes to global optimum
  - Averages over all concurrent transactions

- (Thread-) local adaptivity
  - No synchronization needed
  - Limits adaptable parameters
  - Best parameters for each thread/transaction

Adaptive STM Parameters

- Different adaptive parameters measured:
  - Size of global locking/version-table *G
  - Size of local hash-tables *L
  - Write strategy *L
  - Locality tuning for hash-functions *L
  - Contention management *L

*L – local, *G – global

Adaptive Hash-Table

- Global hash-table: trade-off between over-locking and locality
  - Global strategy: coordinate lock collisions and over-locking between threads
  - Adapt size based on global information

- Local hash-table: trade-off between reset cost, and # hash-collisions
  - Local strategy: sample moving average of unique write locations
  - Adapt size based on trend

Adaptive Write Strategy

- Different costs depending on strategy
  - Write-back: cheap abort, expensive commit
  - Write-through: expensive abort, cheap commit
- Adapt strategy to per-thread workload
  - Measure abort rate

Adaptive Locality Tuning

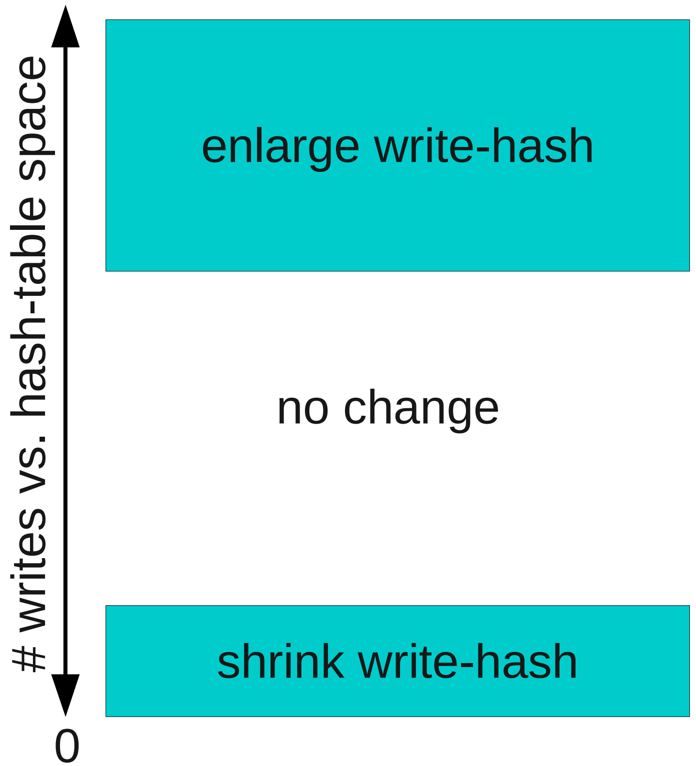
- Different applications have different data access patterns
  - No optimal hash function for all data accesses

- Measure number of hash collisions for thread-local hash tables
  - Circle through different hash functions

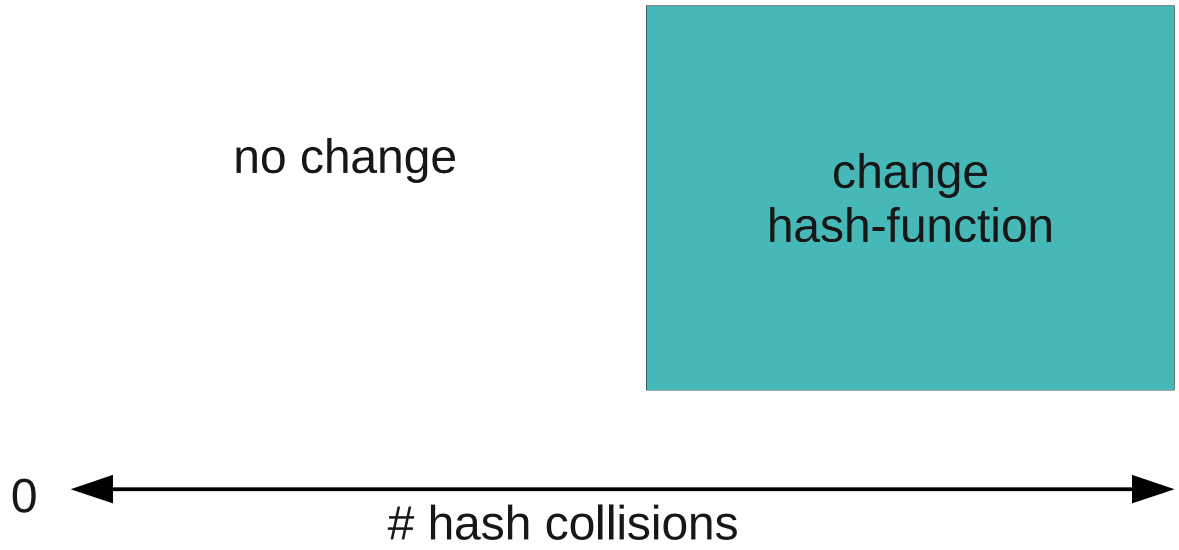
Adaptive Contention Management

- No single strategy works in all environments
- Measure contention and implement an adaptive back-off strategy
  - Wait and retry
  - Abort later

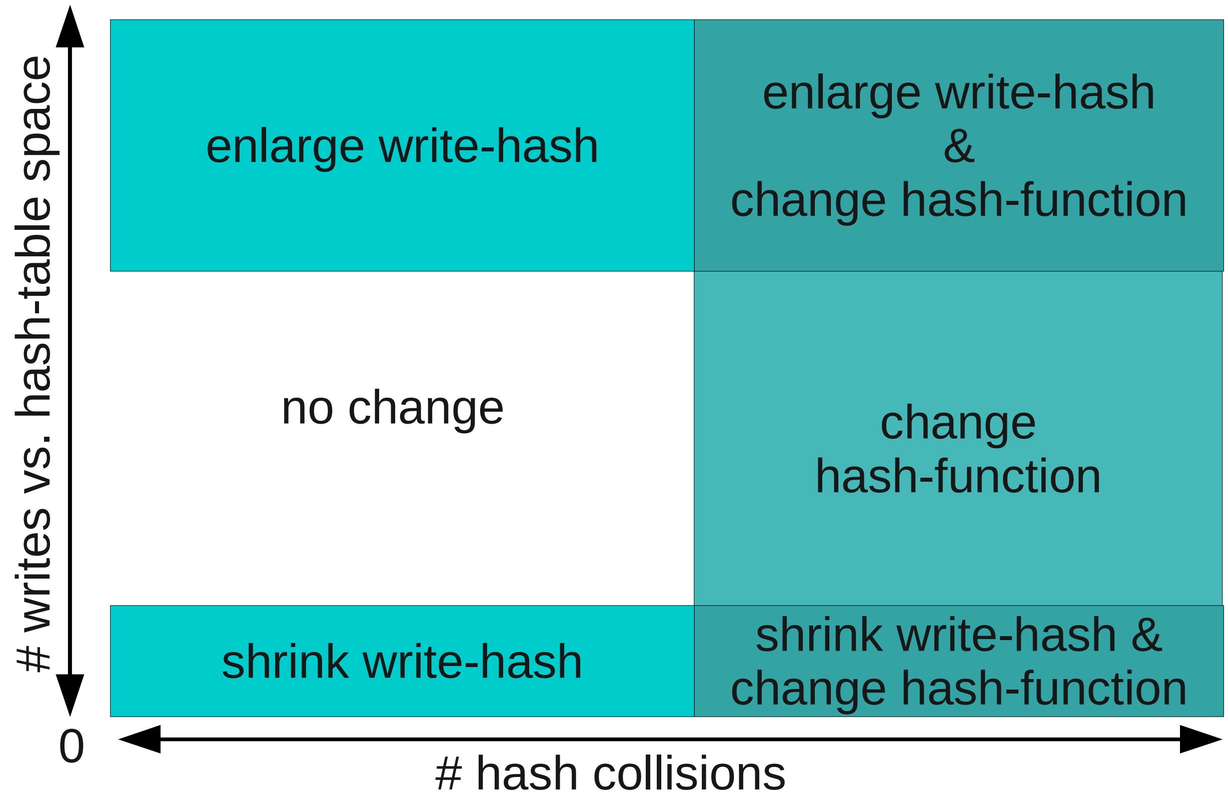
Local Adaptive STM Parameters (for local hash-table)



Local Adaptive STM Parameters (for local hash-table)



Local Adaptive STM Parameters (for local hash-table)



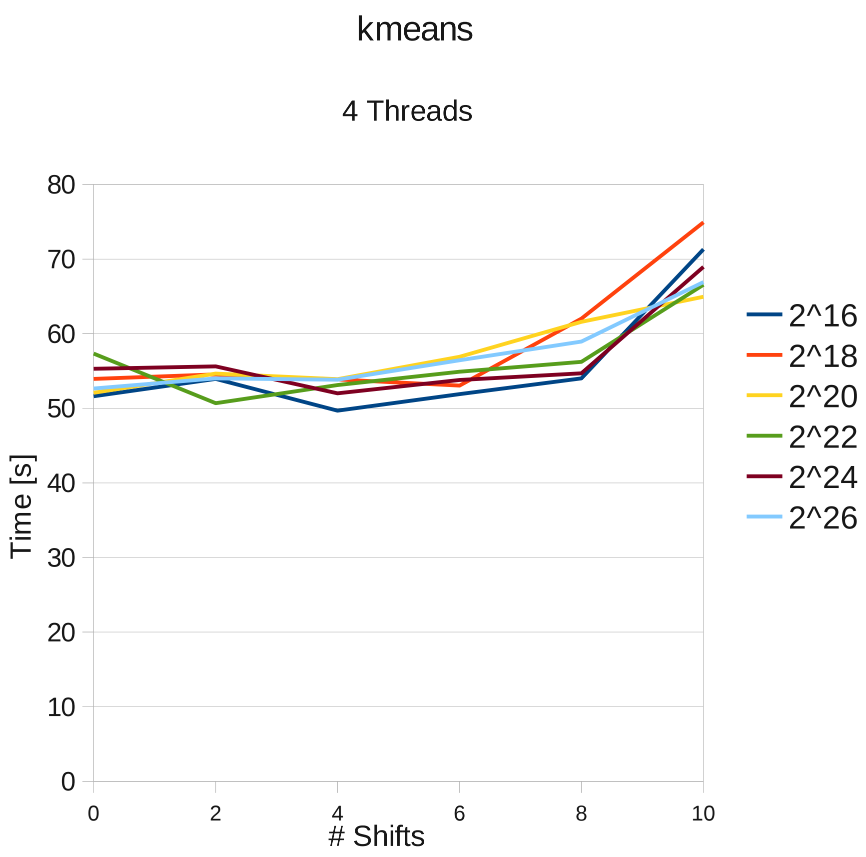
AdaptSTM

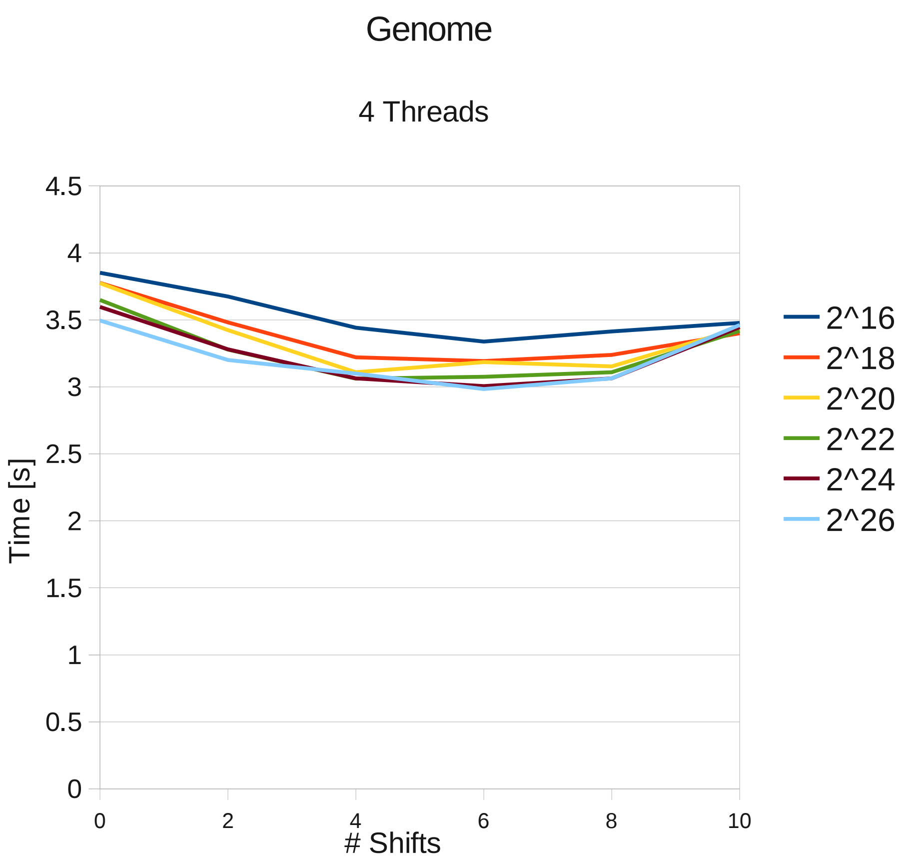
- Adaptive STM system built on presented features
  - Statically tuned competitive baseline
    - Static global hash function and hash table
  - Mature and stable implementation
  - Different local adaptive parameters
    - Write-set hash function and size of hash table
    - Write-through and write-back write strategy
    - Adaptive contention management

Evaluation

- Benchmark: STAMP 0.9.10
  - ++ configuration (increased workload for kmeans)
- AdaptSTM version 0.5.1
- Intel 4-core Xeon E5520 CPU
  - 8 cores @ 2.27GHz, 12GB RAM
  - 64bit Ubuntu 9.04

Evaluation: Global Hash-Table





Evaluation: Global Adaptivity

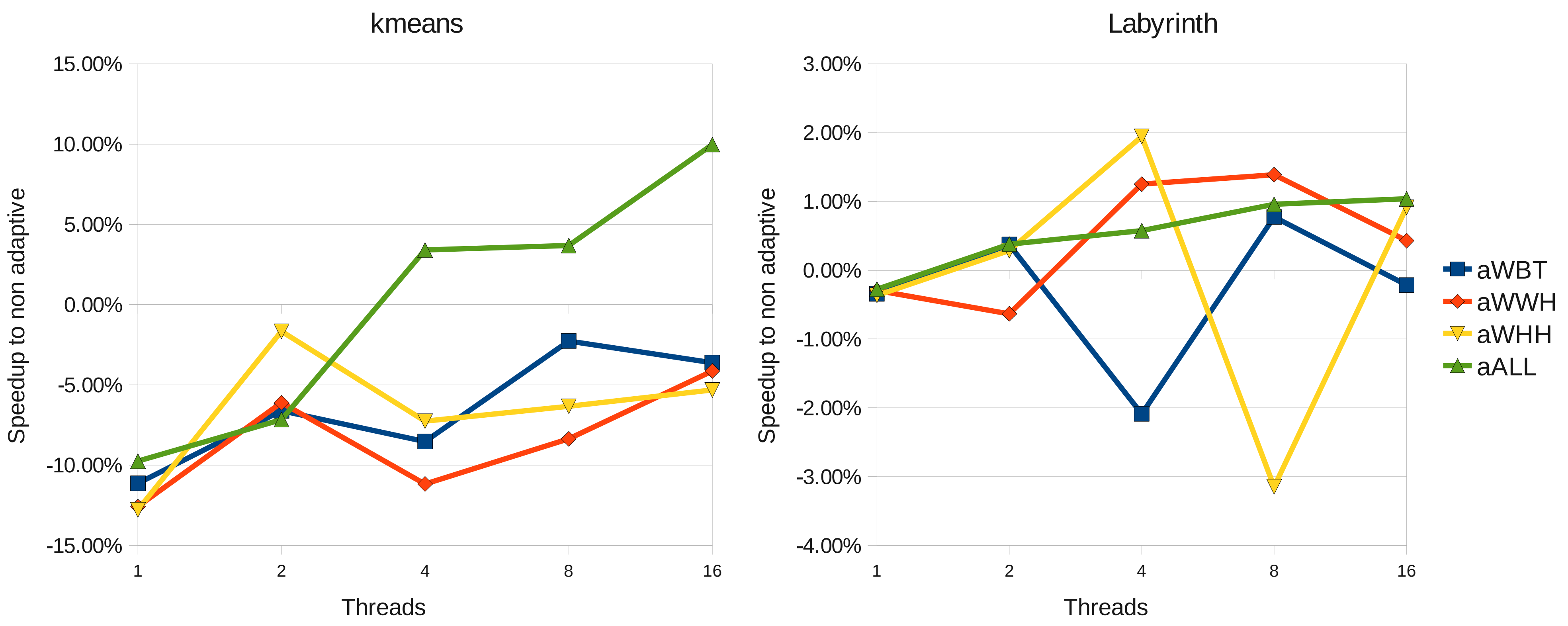
- Global optimizations have limited potential
  - Small optimization potential
  - High synchronization cost
  - Reasonable baseline outperforms global optimization

Evaluation: Local Adaptivity

- Different configurations:
  - naWB: no adaptivity, use write-back
  - aWBT: adaptivity, adjust write-through / write-back
  - aWWH: aWBT plus an adaptive hash-table for the write-set
  - aWHH: aWWH plus different hash functions
  - aALL: all adaptive parameters plus Bloom filter for write-entries

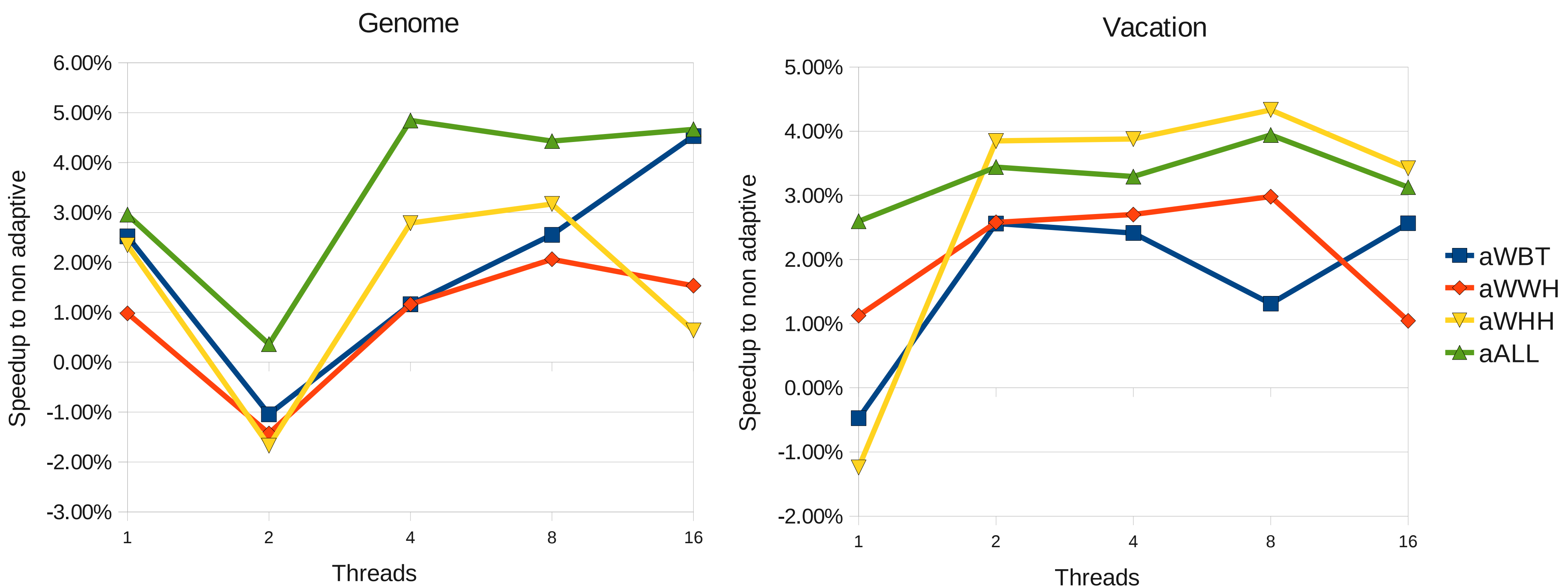
- Adaptation system starts with best 'average' parameters, improves from there

Evaluation: Local Adaptivity



- aWBT: adaptive, write-back/-through
- aWWH: adaptive, write-back/-through, write-hash
- aWHH: adaptive, write-back/-through, write-hash, hash-function
- aALL: adaptive, write-back/-through, write-hash, hash-function, Bloom filter

Evaluation: Local Adaptivity



- aWBT: adaptive, write-back/-through
- aWWH: adaptive, write-back/-through, write-hash
- aWHH: adaptive, write-back/-through, write-hash, hash-function
- aALL: adaptive, write-back/-through, write-hash, hash-function, Bloom filter

Evaluation: Local Adaptivity

- No single optimization works for all benchmarks
- Combination of all options leads to best performance
- Impressive speed-ups for individual benchmarks compared to the globally optimized case

Related Work

- TL2 (Dice et al.): baseline STM system
- Different related work on static tuning of global parameters (Harris, Dice, Ennals, Felber)
  - Crucial for efficient baseline
- TinySTM (Felber et al.): adapts size and hash function of global locking table
- ASTM (Marathe et. al.): adapts lazy-eager locking strategies and different meta-formats

Conclusions

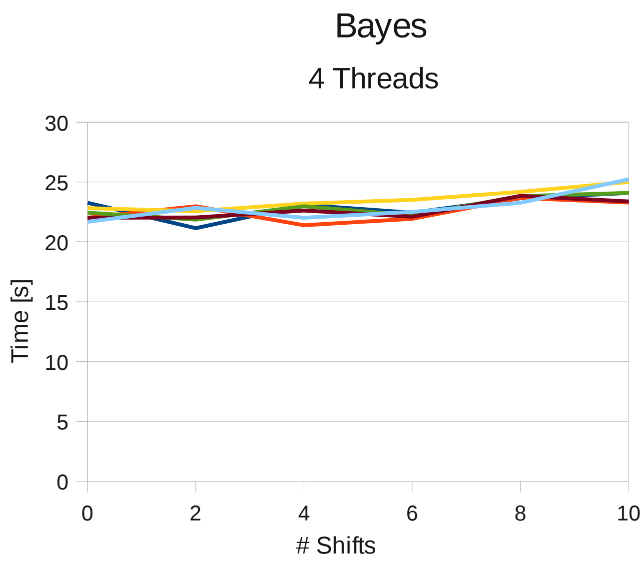
- Adaptivity in STM is important for good performance
  - Speedups up to 10% possible
- Global optimization are limited
  - Low potential, high synchronization cost
- Local optimizations tune thread-local parameters
  - High correlation with workload

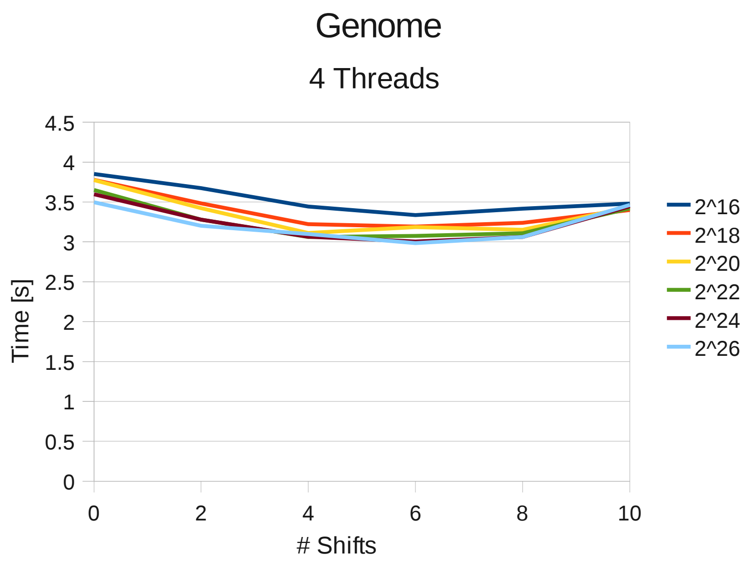
Questions

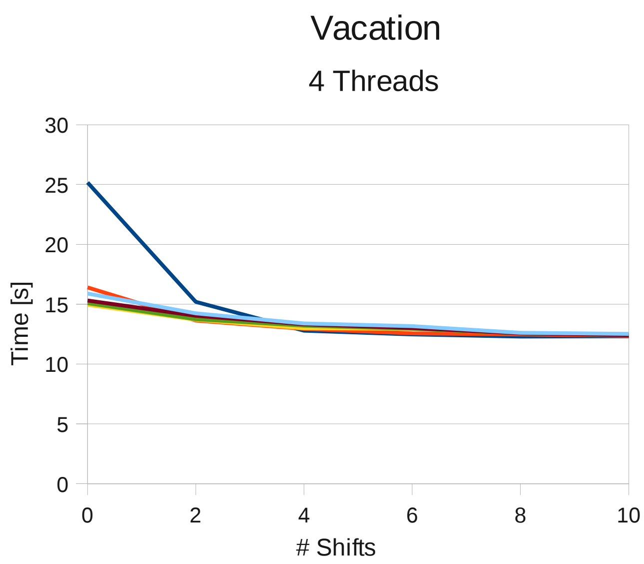
?

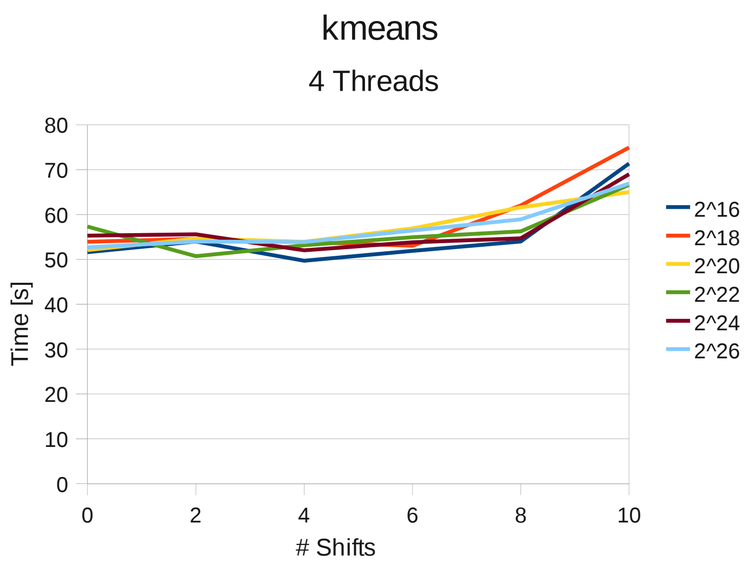
- Contact: mathias.payer@nebelwelt.net
- Source: http://nebelwelt.net/projects/adaptSTM/

Evaluation: Global Hash-Table

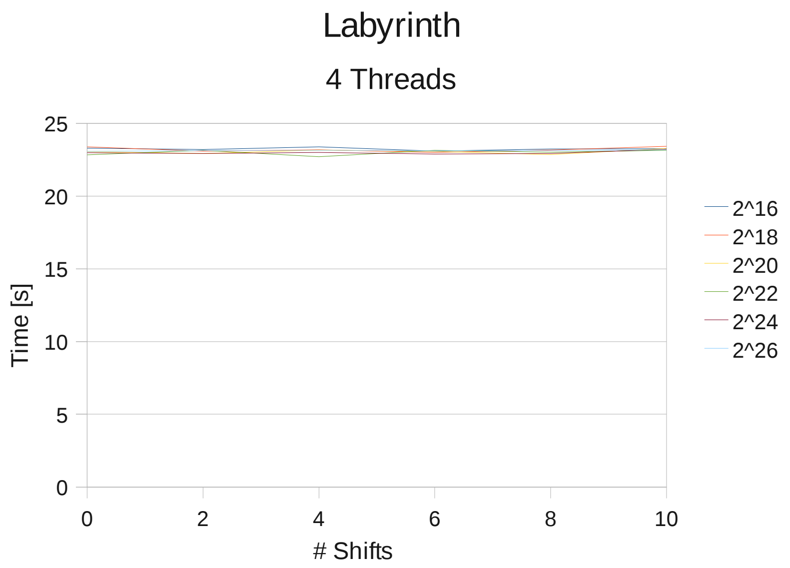


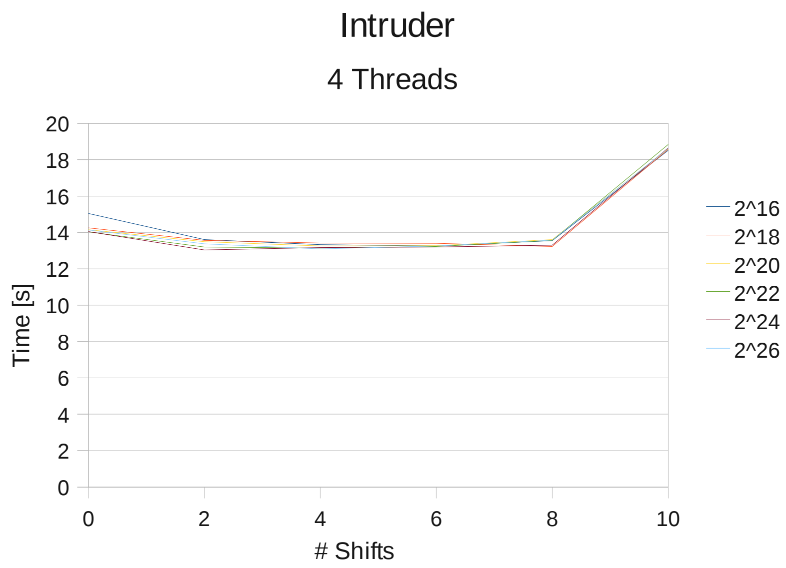


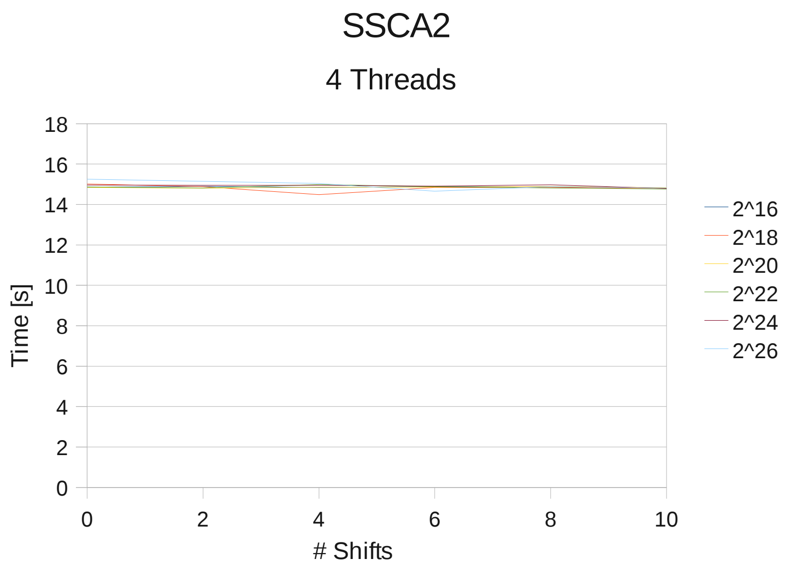


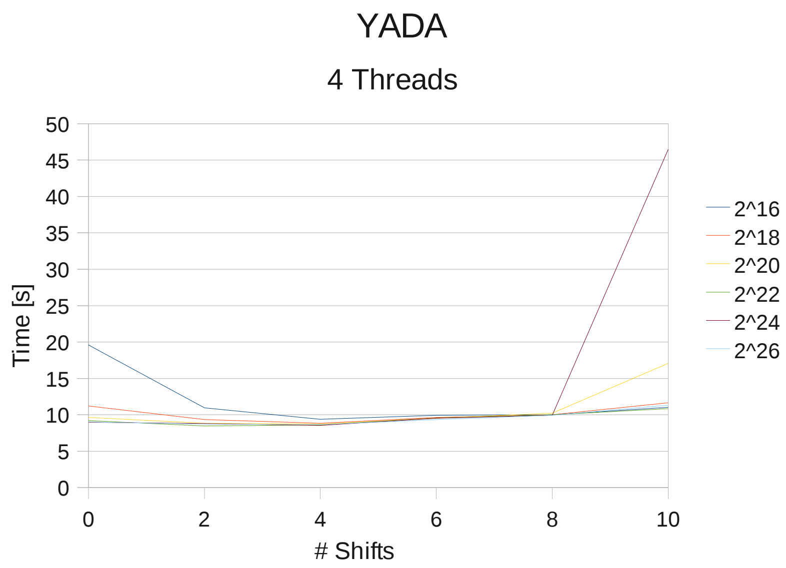


Evaluation: Global Hash-Table

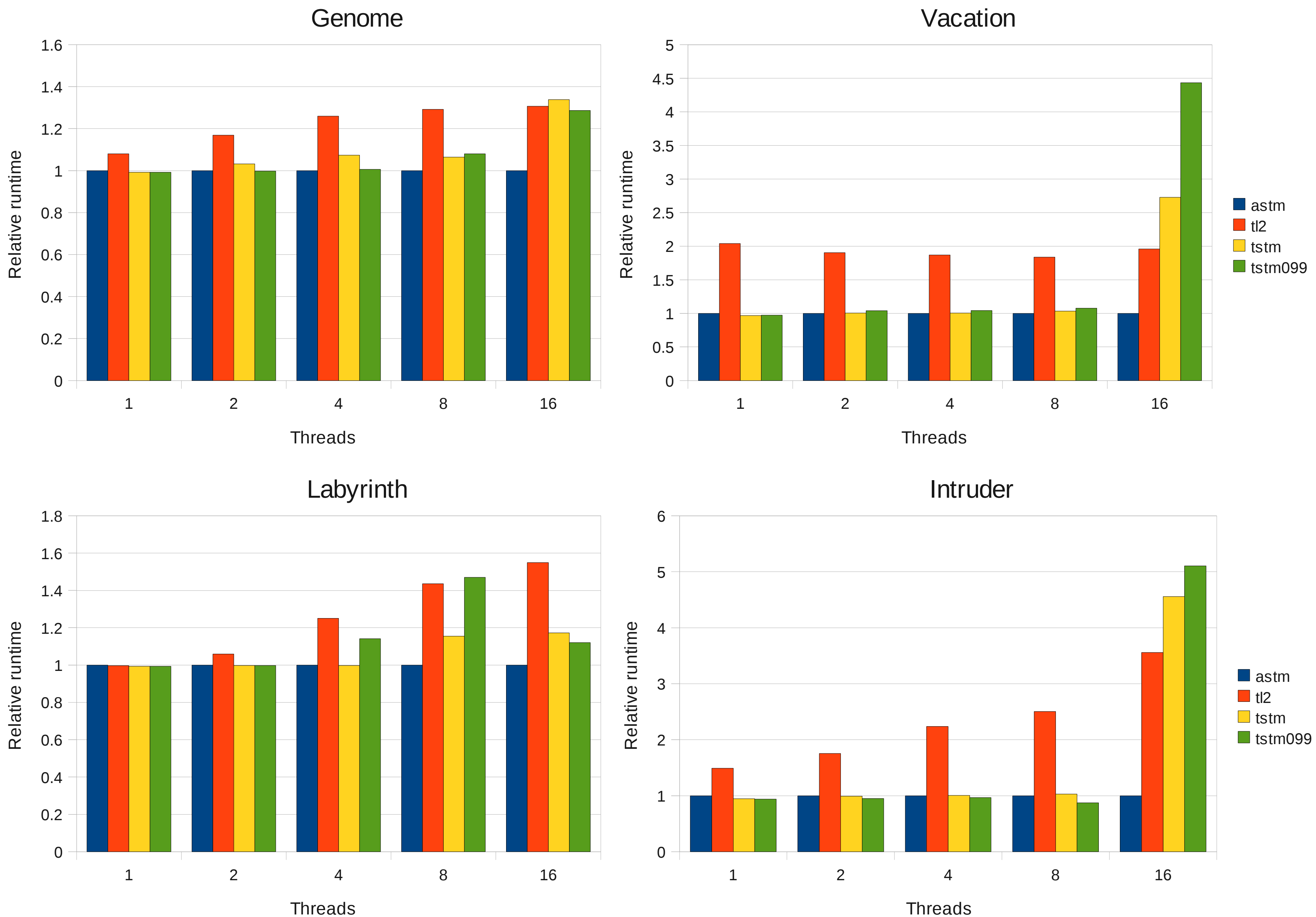




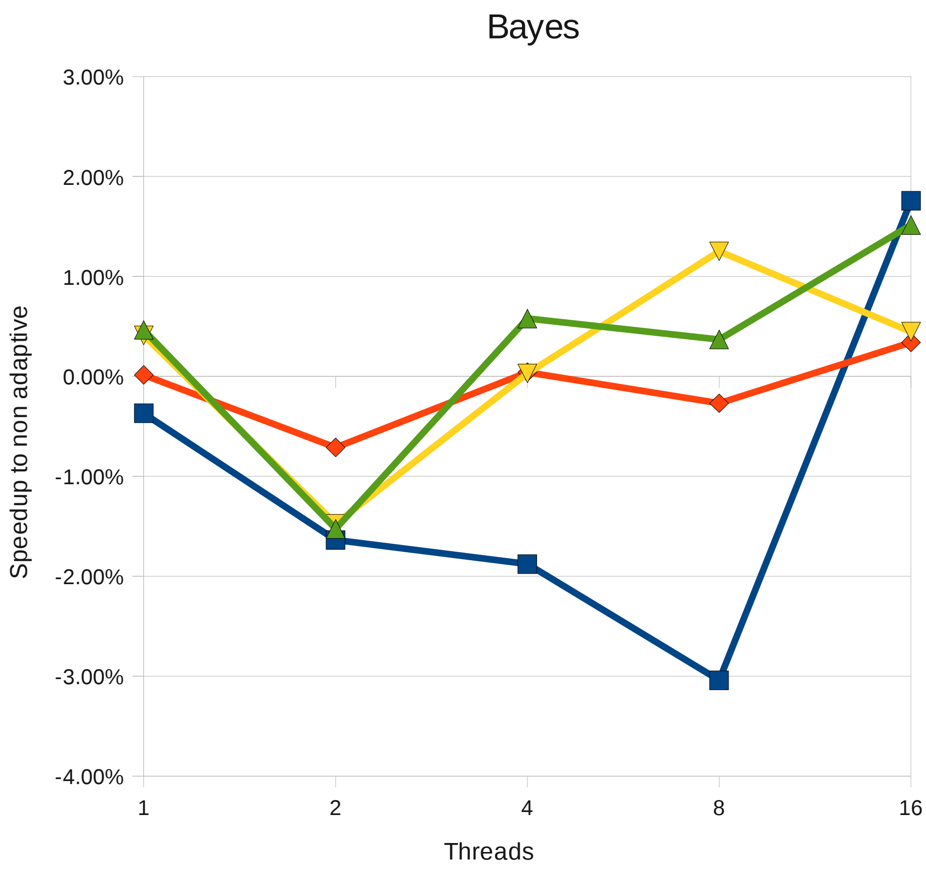


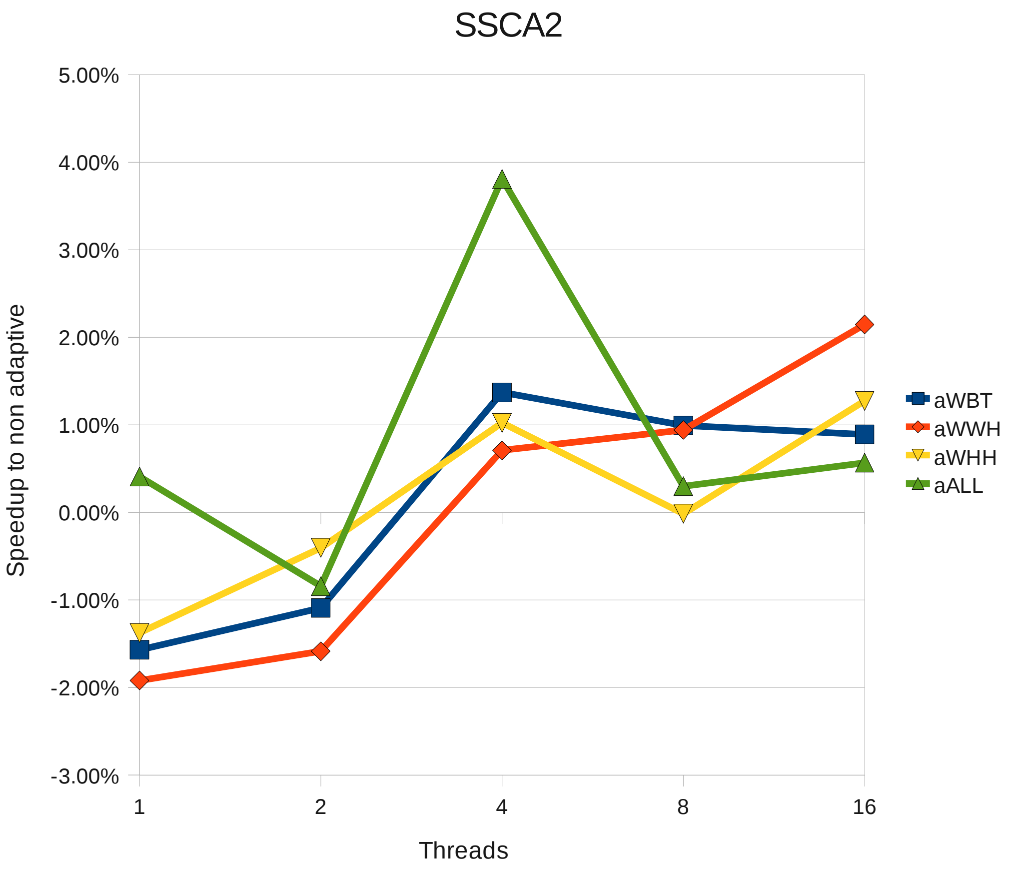


STM Comparison



Evaluation: Local Adaptivity





Evaluation: Local Adaptivity