Enhancing Permissiveness of Transactional Memory via Time-Warp

Nuno Diegues and Paolo Romano

ndiegues@gsd.inesc-id.pt

INESC-ID/Instituto Superior Técnico

Transactional Memory

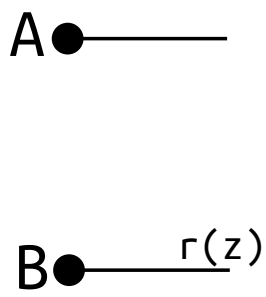
Simple API + guarantee of correctness criterion
Easy to use and reason

Introduction

To guarantee a given correctness level, a TM aborts transactions.

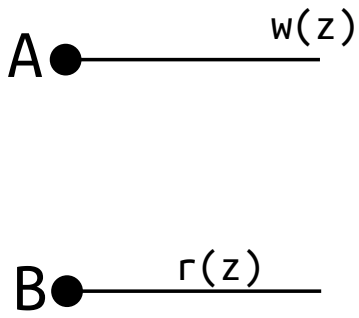
Introduction

To guarantee a given correctness level, a TM aborts transactions.



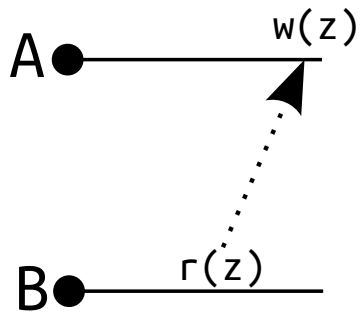
Introduction

To guarantee a given correctness level, a TM aborts transactions.



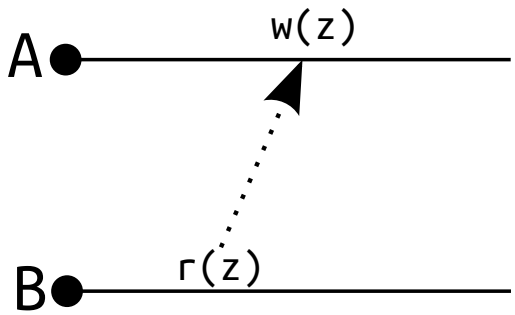
Introduction

To guarantee a given correctness level, a TM aborts transactions.



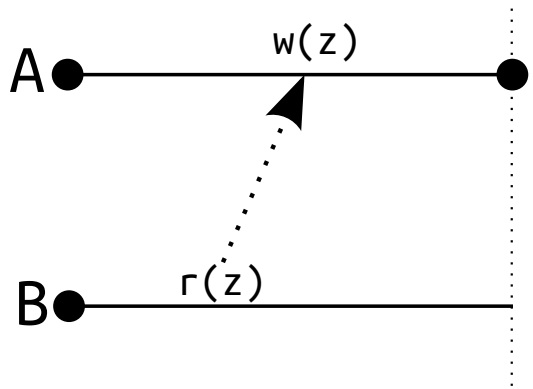
Introduction

To guarantee a given correctness level, a TM aborts transactions.



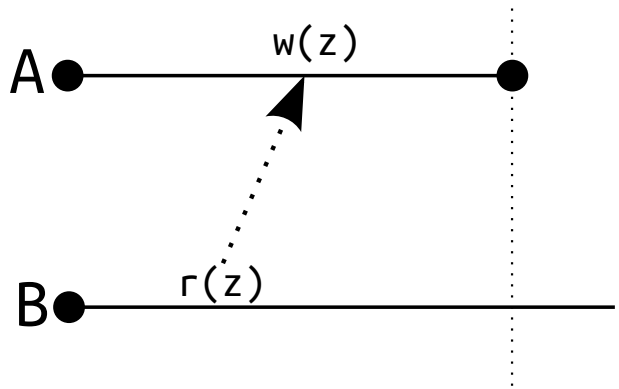
Introduction

To guarantee a given correctness level, a TM aborts transactions.



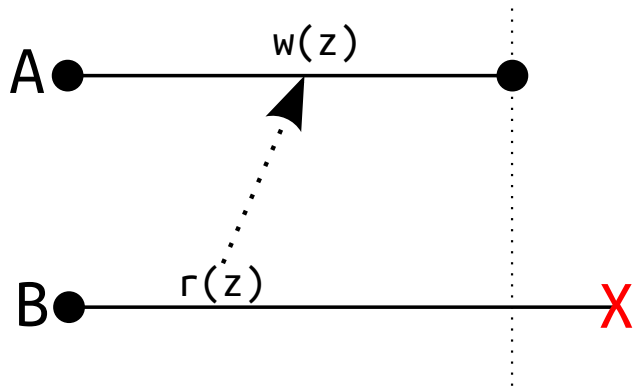
Introduction

To guarantee a given correctness level, a TM aborts transactions.



Introduction

To guarantee a given correctness level, a TM aborts transactions.

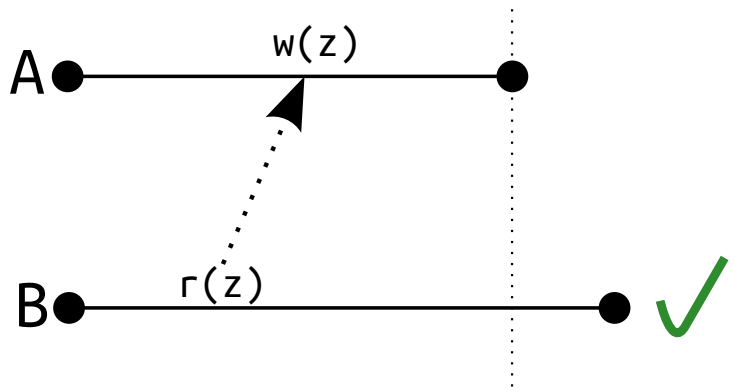


Problem at hands

But why are we aborting the transaction in the example?

Problem at hands

But why are we aborting the transaction in the example?
Instead, we could do the following:



Serialization order: B, A

Problem at hands

Condition:
Abort T if its reads are not up-to-date when it attempts to commit.

Problem at hands

Condition:
Abort T if its reads are not up-to-date when it attempts to commit.

Serializability:

- Necessary condition
- But **not** sufficient

Problem at hands

Condition:
Abort T if its reads are not up-to-date when it attempts to commit.

Serializability:

- Necessary condition
- But **not** sufficient
- Deemed to be practical

Permissiveness

Notion of avoiding unnecessary aborts

Permissiveness

Notion of avoiding unnecessary aborts:

- If it only aborts a transaction when the resulting history (without the abort) does not respect some target correctness criterion

Permissiveness

State of the art TMs far from being permissive.

Permissiveness

State of the art TMs far from being permissive.

- Cost is **not** negligible.

Permissiveness

State of the art TMs far from being permissive.

- Cost is **not** negligible.

What do we have to counter that?

- One way is to track the graph of dependencies — DATM
  - inconsistent reads and zombie transactions

Permissiveness

State of the art TMs far from being permissive.

- Cost is **not** negligible.

What do we have to counter that?

- One way is to track the graph of dependencies — DATM
  - inconsistent reads and zombie transactions
- Maintain interval of possible serialization points — AVSTM
  - Moderate bookkeeping
  - Aborts read-only transactions

Goal

Enhance permissiveness without such costs

Goal

Enhance permissiveness without such costs:

- More restrictive abort condition
- Always read consistently
- Wait-free read-only transactions (mv-permissiveness)

Time-warping

Allow transactions to commit in the past

Time-warping

Allow transactions to commit in the past:

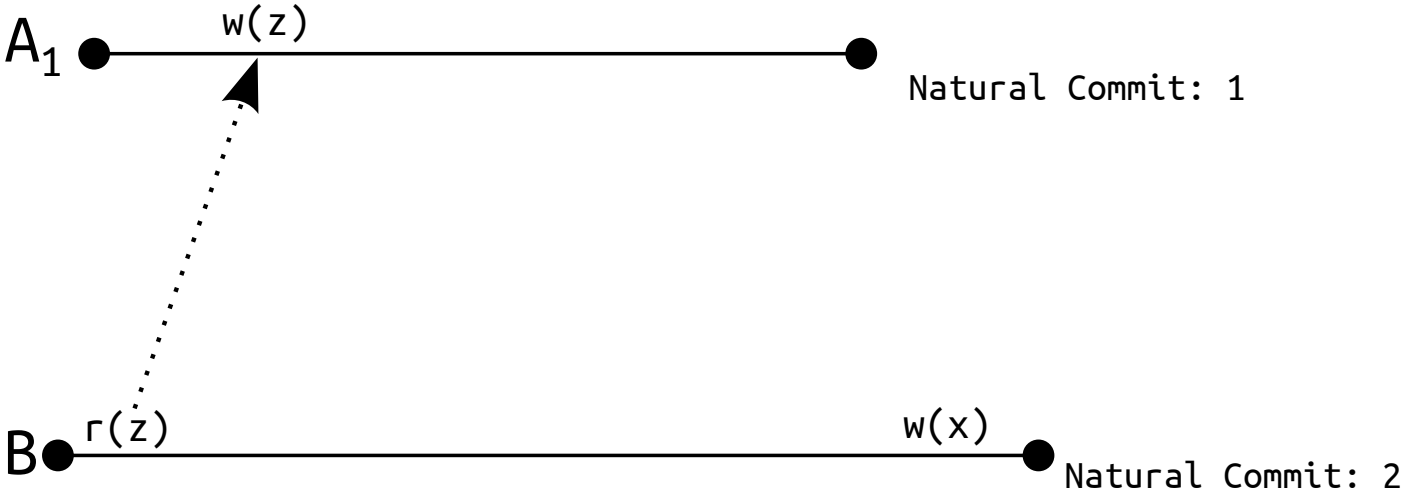
- Pick the condition defined previously
  - Abort T if its reads are not up-to-date when it attempts to commit.

Time-warping

Allow transactions to commit in the past:

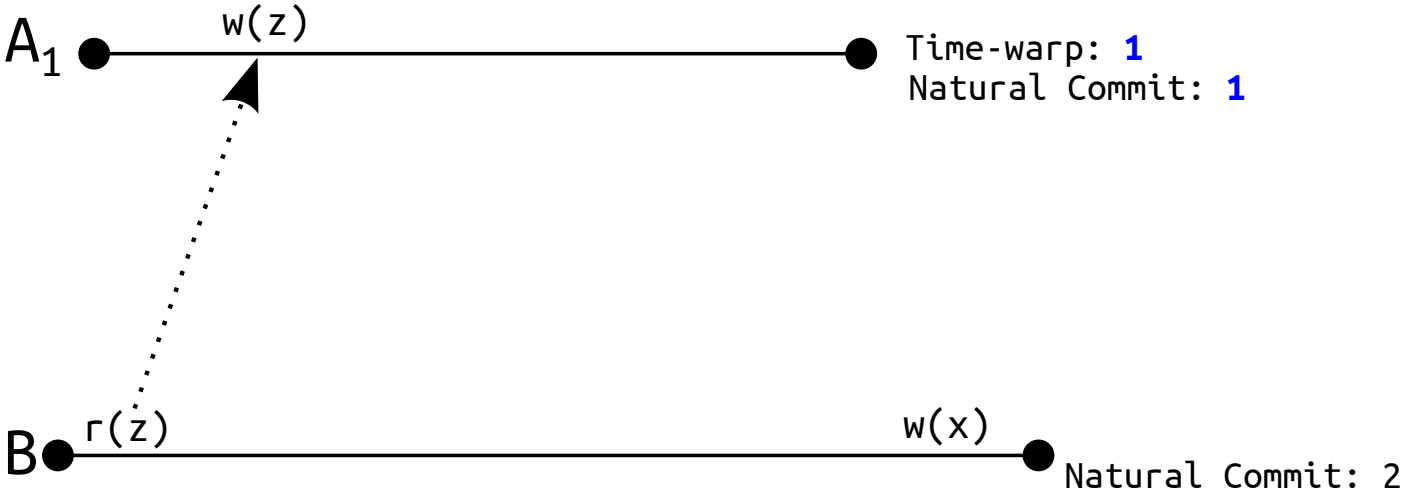
- Pick the condition defined previously
  - Abort T if its reads are not up-to-date when it attempts to commit.
- Commit T and serialize it before the writes it missed
  - Time-warp commit

Time-warp



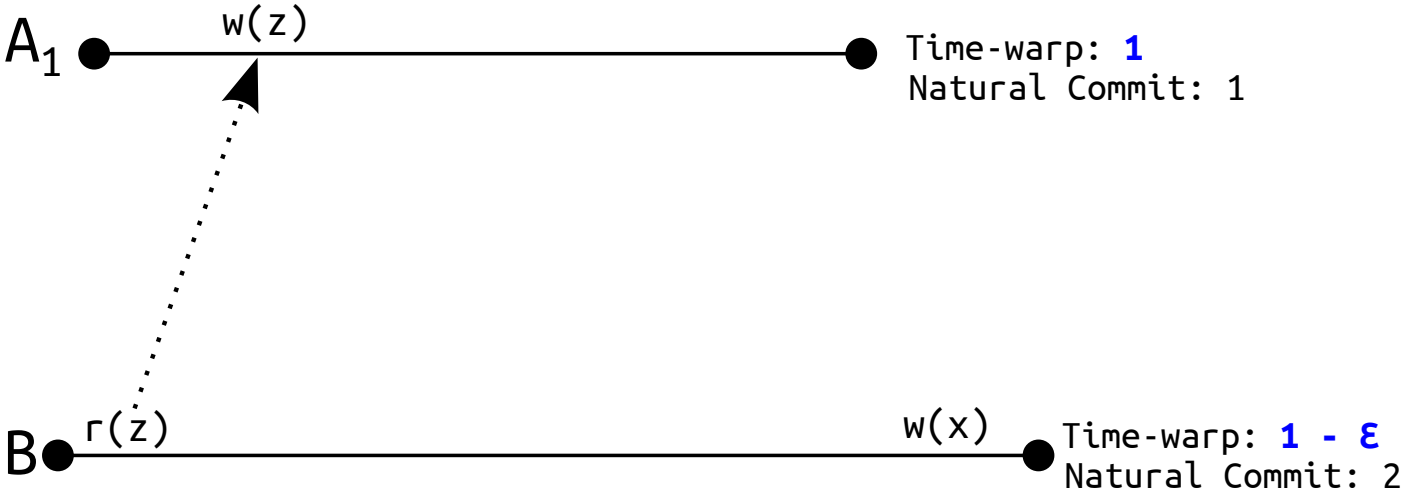
Same history as before

Time-warp



Time-warp order

Time-warp



B time-warp commits

Time-warp

When can we not apply this idea?

Time-warp

When can we not apply this idea?

Look out for a specific structure

Time-warp

When can we not apply this idea?

Look out for a specific structure:

- Three transactions connected — a triad

Time-warp

When can we not apply this idea?

Look out for a specific structure:

- Three transactions connected — a triad
- The link between all three — the pivot

Time-warp

When can we not apply this idea?

Look out for a specific structure:

- Three transactions connected — a triad
- The link between all three — the pivot
- Abort T if

Time-warp

When can we not apply this idea?

Look out for a specific structure:

- Three transactions connected — a triad
- The link between all three — the pivot
- Abort T if:
  - Completes a triad

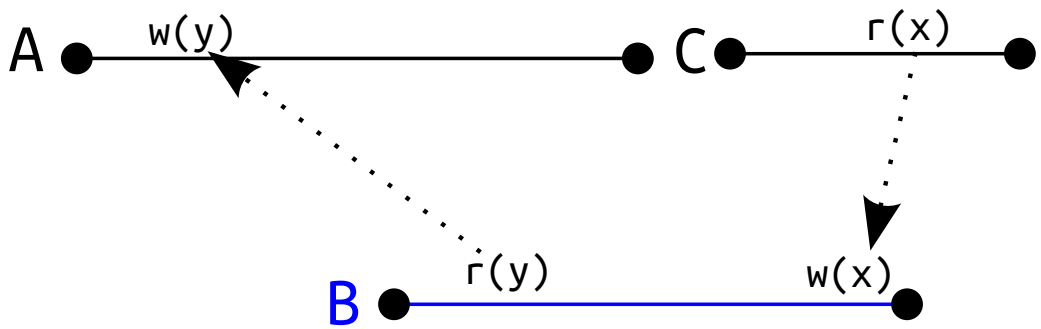
Time-warp

When can we not apply this idea?

Look out for a specific structure:

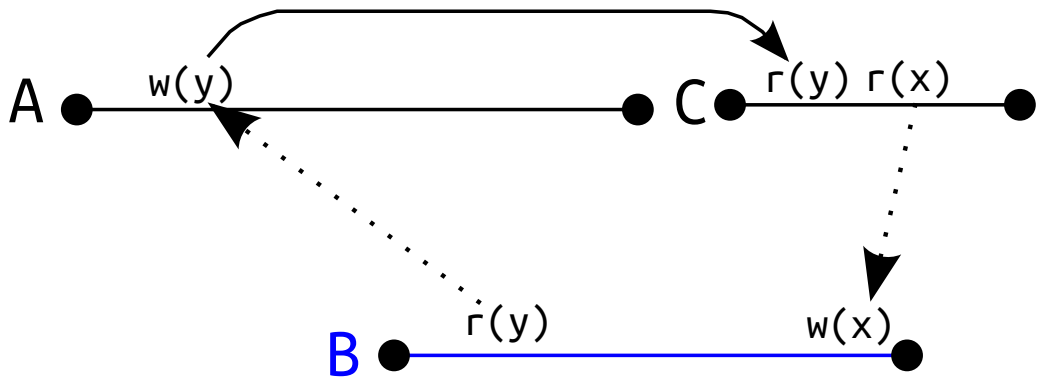
- Three transactions connected — a triad
- The link between all three — the pivot
- Abort T if:
  - Completes a triad
  - Whose pivot time-warp committed

Abort rule



C aborts here

Abort rule



(rather strict) necessary condition for a cycle

Theoretical and Practical results

- Serializability for committed transactions

Theoretical and Practical results

- Serializability for committed transactions
- Virtual World Consistency (stronger) for running transactions
  - Prevent a range of phenomena to avoid sandboxing

Theoretical and Practical results

- Serializability for committed transactions
- Virtual World Consistency (stronger) for running transactions
  - Prevent a range of phenomena to avoid sandboxing

- Lock-free prototype in Java

Theoretical and Practical results

- Serializability for committed transactions
- Virtual World Consistency (stronger) for running transactions
  - Prevent a range of phenomena to avoid sandboxing

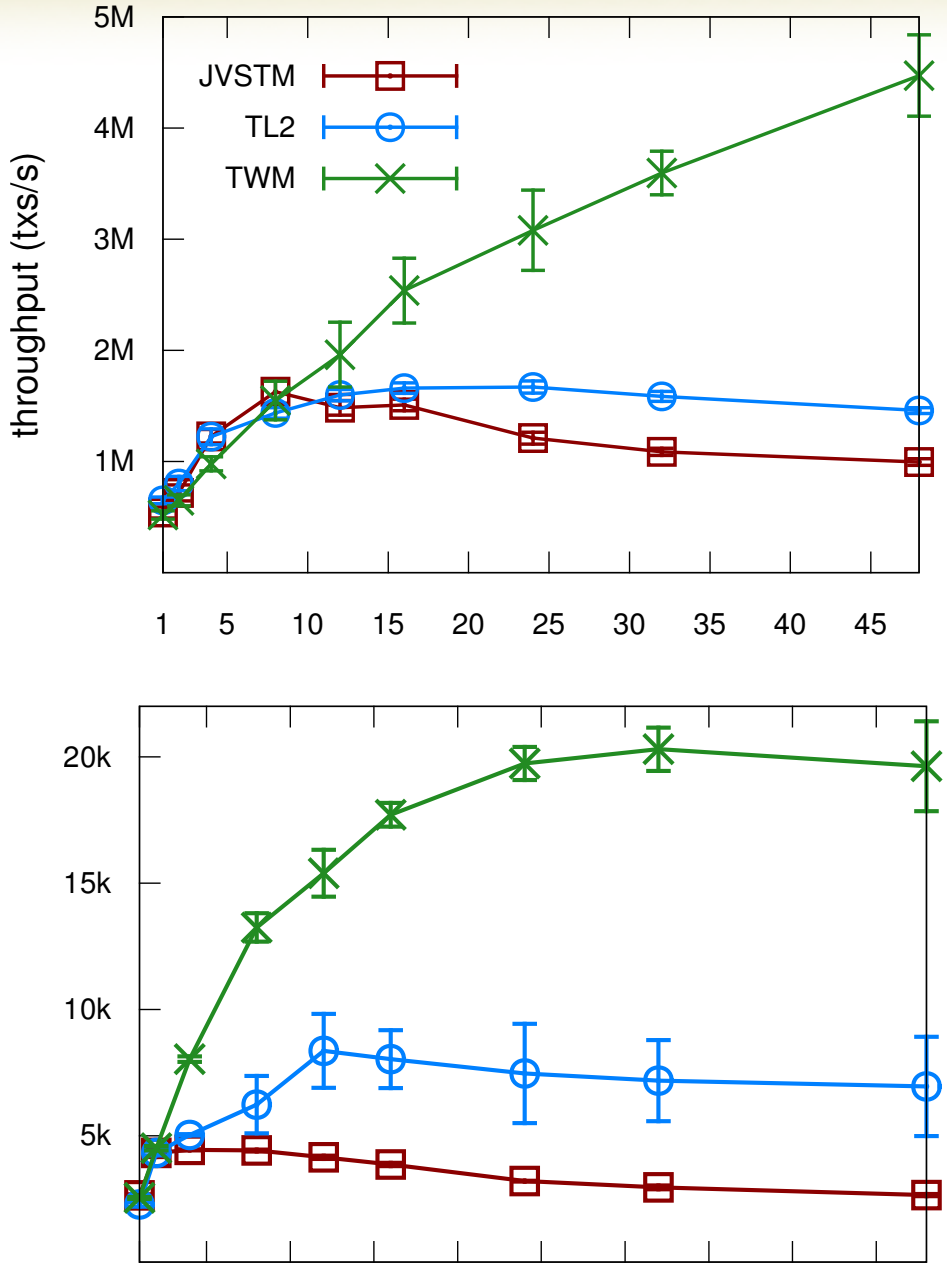
- Lock-free prototype in Java
- 48 core machine

Theoretical and Practical results

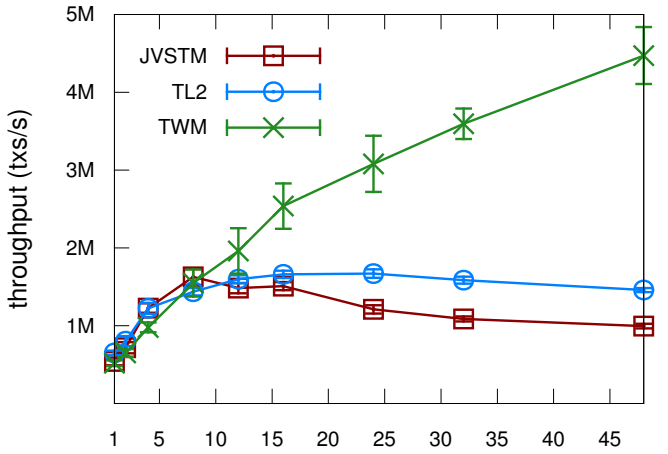
- Serializability for committed transactions
- Virtual World Consistency (stronger) for running transactions
  - Prevent a range of phenomena to avoid sandboxing

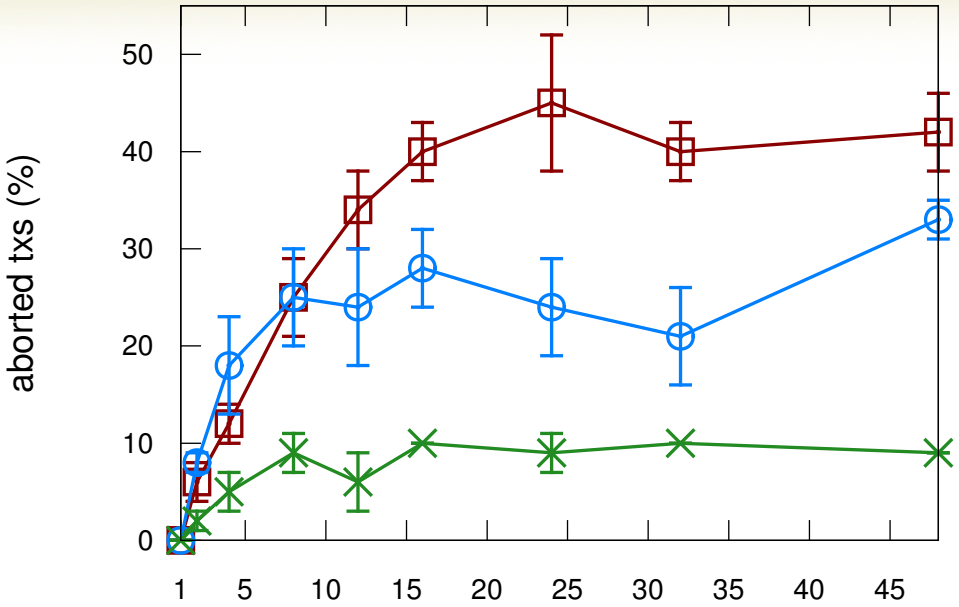
- Lock-free prototype in Java
- 48 core machine

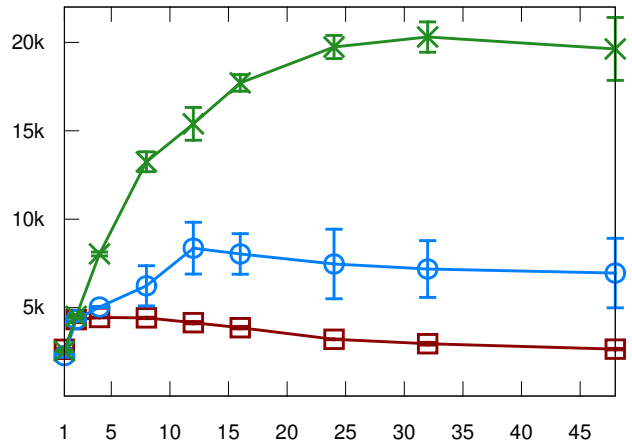
SkipList and Vacation from STAMP

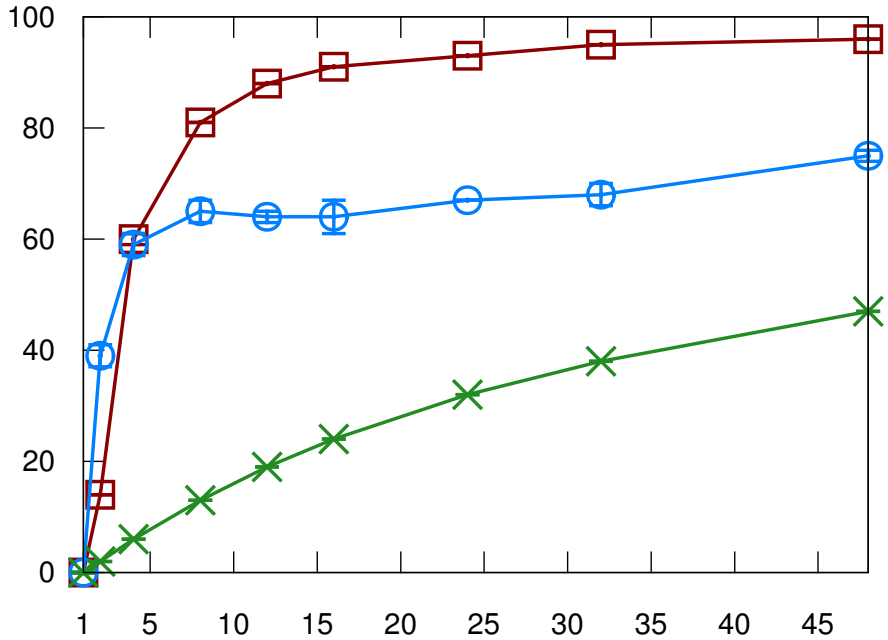


SkipList and Vacation from STAMP









Ongoing work 1/2

Ongoing work 1/2

Time-warp in a distributed transactional setting

Ongoing work 1/2

Time-warp in a distributed transactional setting:

- Generalizable to a variety of protocols

Ongoing work 1/2

Time-warp in a distributed transactional setting:

- Generalizable to a variety of protocols
- Partially replicated key-value store with transactions

Ongoing work 1/2

Time-warp in a distributed transactional setting:

- Generalizable to a variety of protocols
- Partially replicated key-value store with transactions
- Scalable solution
  - Now also with update workloads

Ongoing work 2/2

Partially replicated large collections

- Scalable computations
- Transaction friendly despite structural conflicts

Thank you