Brief Announcement: Enhancing Permissiveness of Transactional Memory via Time-Warp

Nuno Diegues and Paolo Romano

ndiegues@gsd.inesc-id.pt

Introduction

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

Introduction

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

function **commit**(`Transaction` *tx*):

  for each ‹*datum*, *version*›∈ *tx*.readSet **do**
    if not latestVersion(*datum*, *version*) **then**
      abort(*tx*)

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

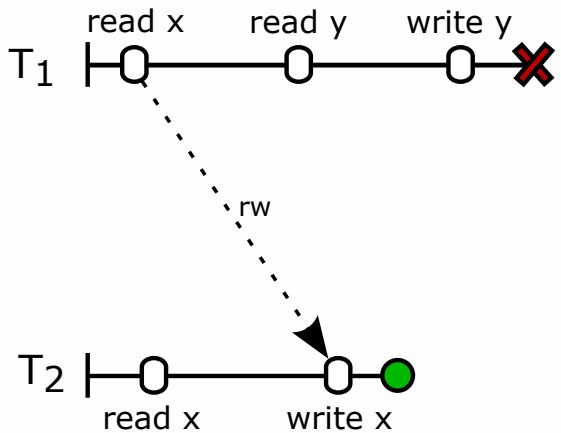
Condition:
Abort T if its reads are not up-to-date when it attempts to commit.

Serializability:

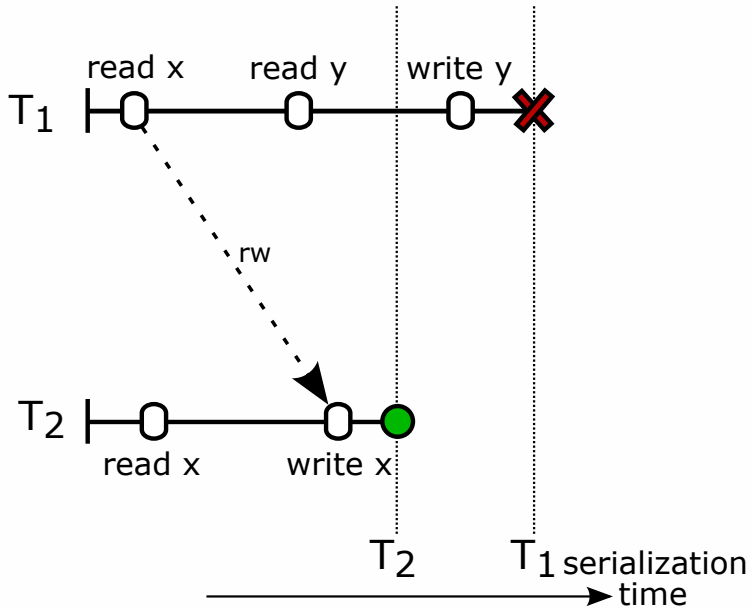
- Necessary condition
- But **not** sufficient
- Deemed to be practical

Problem

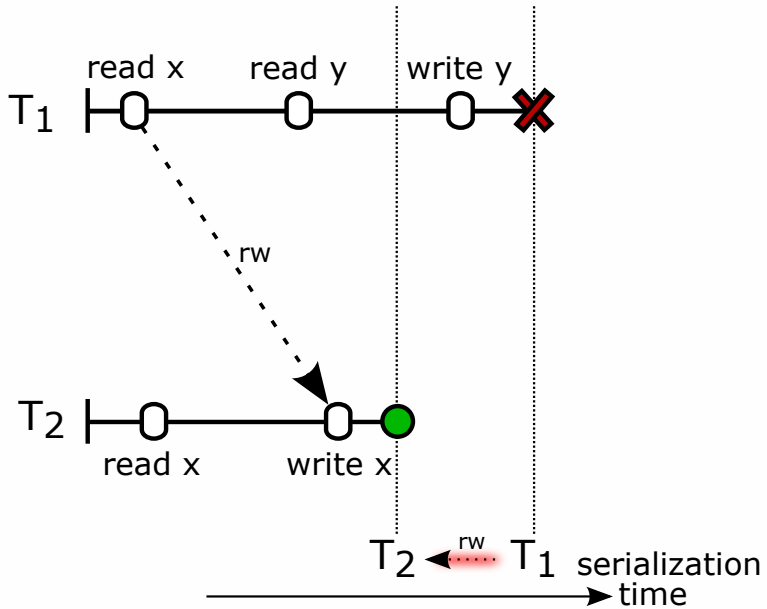
Problem



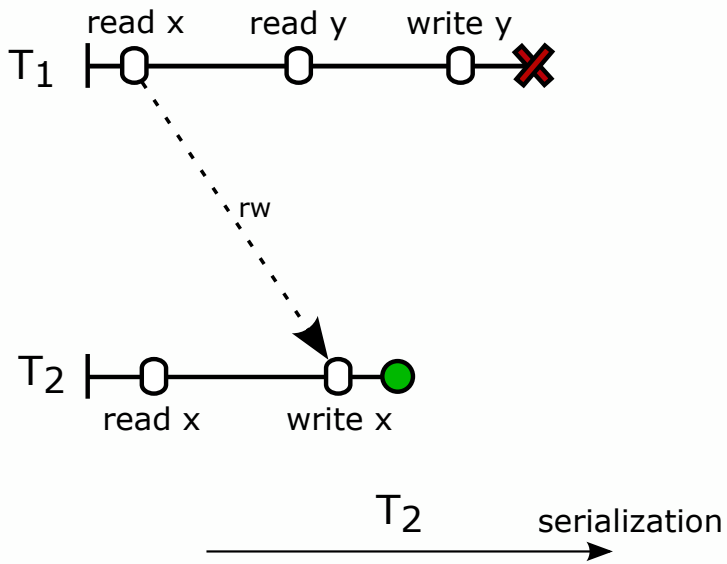
Problem


read x
read y
write y
T1
rw
T2
read x
write x
T2
T1
serialization
time

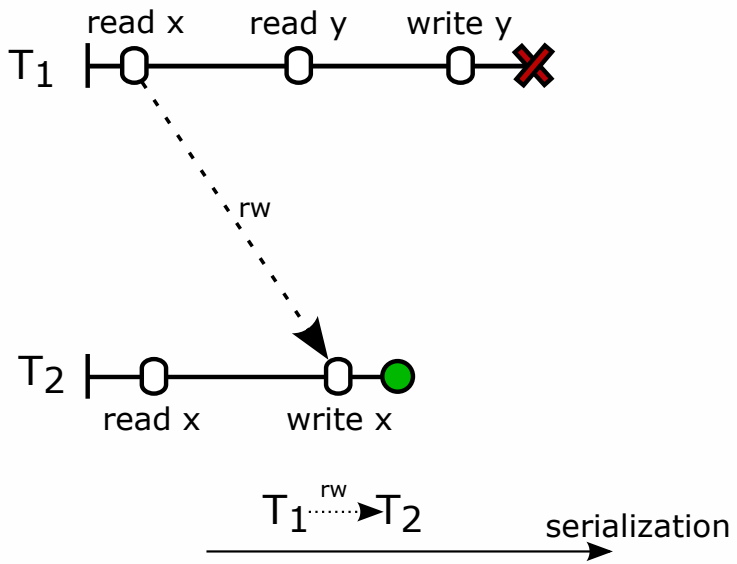
Problem


read x
read y
write y
T1
rw
T2
read x
write x
T2
rw
T1
serialization
time

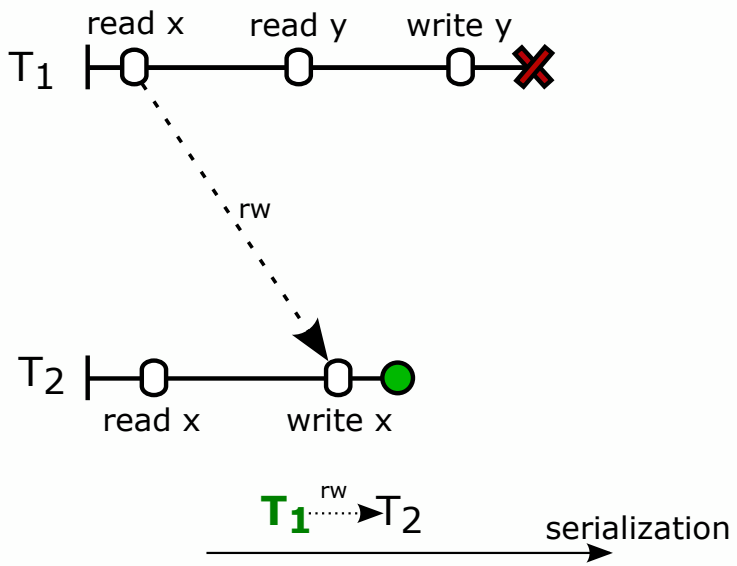
Problem



Problem


read x
read y
write y
T1
rw
T2
read x
write x
T1 rw T2
serialization

Problem



Time-warp commit

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

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
- State of the art TMs are far from being permissive

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

- State of the art TMs are far from being permissive
- Cost is **not** negligible.

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-warp

When can we **not** apply this idea?

Time-warp

When can we **not** apply this idea?

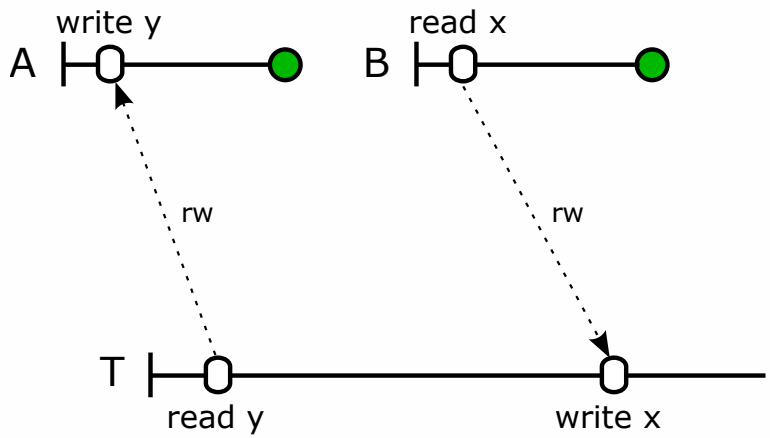
Look out for a specific structure

Time-warp

When can we **not** apply this idea?

Look out for a specific structure:

- Three transactions connected
  - a triad

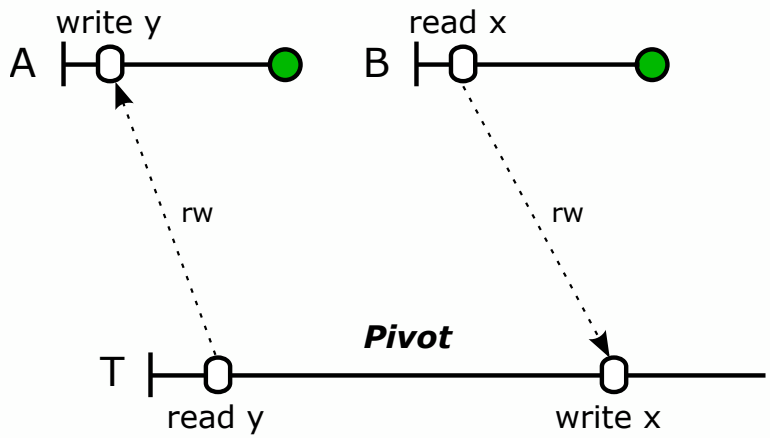


Time-warp

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

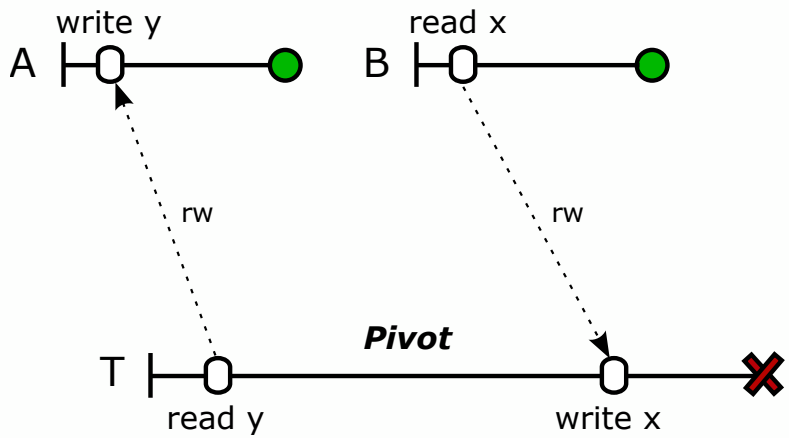


Time-warp

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 if:
  - Completes a triad
  - Whose pivot time-warp commits



Theoretical and Practical results

- Virtual World Consistency

Theoretical and Practical results

- Virtual World Consistency
  - Serializability for committed transactions
  - Prevent a range of phenomena for running transactions

Theoretical and Practical results

- Virtual World Consistency
  - Serializability for committed transactions
  - Prevent a range of phenomena for running transactions

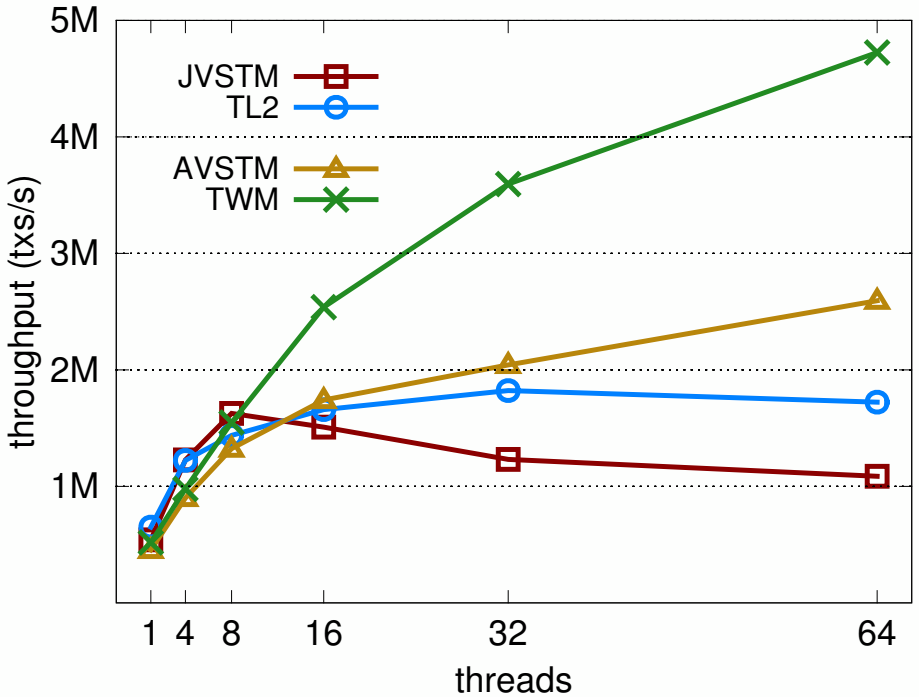
- Lock-free prototype in Java: TWM

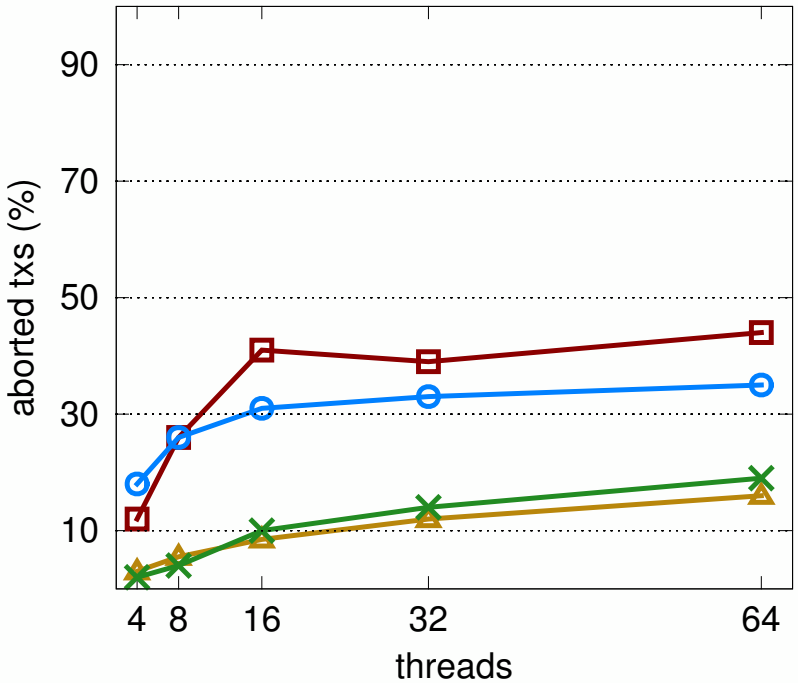
Theoretical and Practical results

- Virtual World Consistency
  - Serializability for committed transactions
  - Prevent a range of phenomena for running transactions

- Lock-free prototype in Java: TWM
- 64 core machine
- Comparison with TL2, JVSTM, AVSTM

Ordered Skip-List





Thank you

Enhancing Permissiveness in Transactional Memory via Time-Warping,
INESC-ID, TR 10/2013