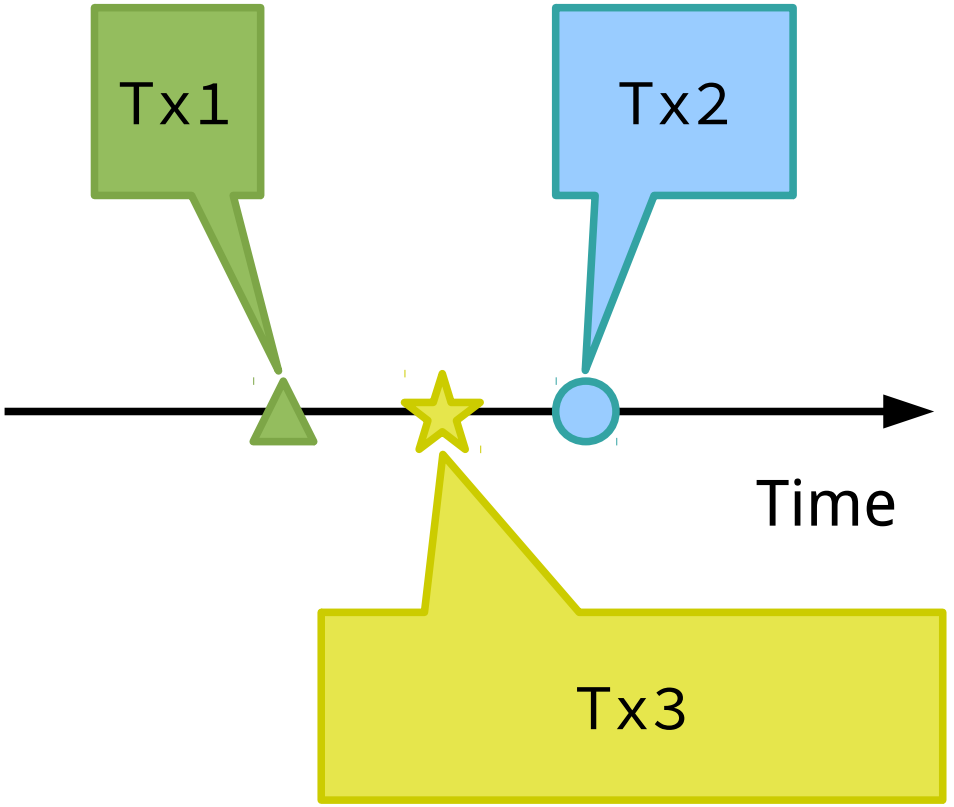
# Transaction Preserialisation in Transactional Memory

Tiago M. Vale, João A. Silva, Ricardo R. Dias, João M. Lourenço
CITI – NOVA University of Lisbon
WTM 2014

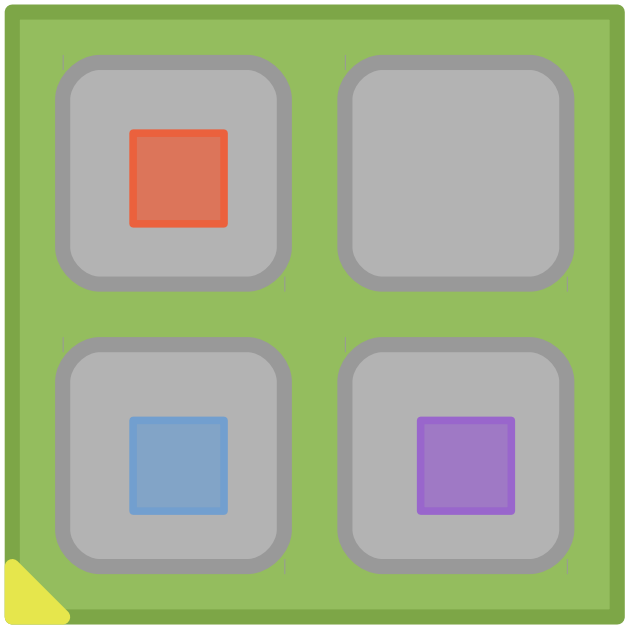
# Transactions



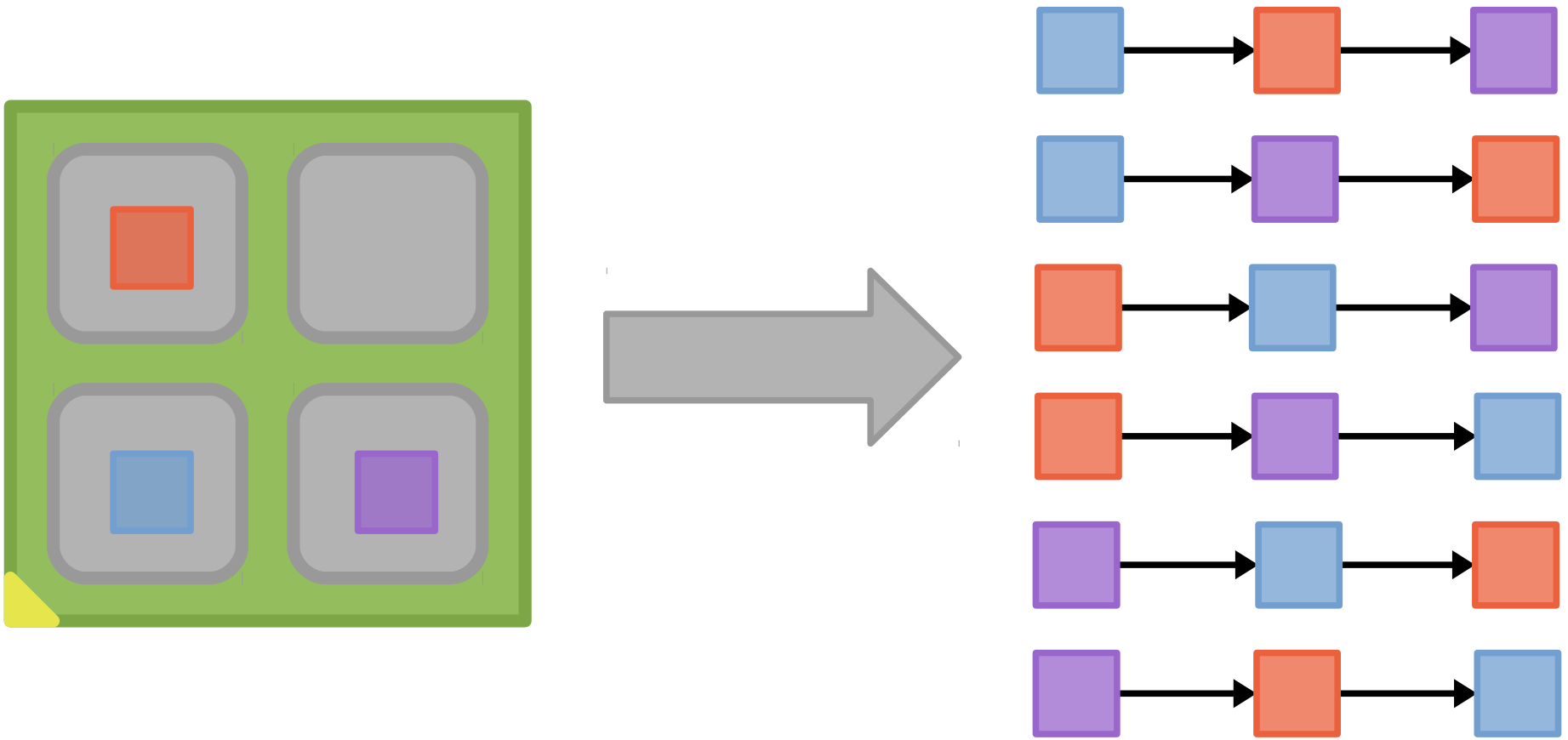
# Serializability

"The result of concurrently executing a set of transactions is equivalent to **some** sequential execution of the same transactions."

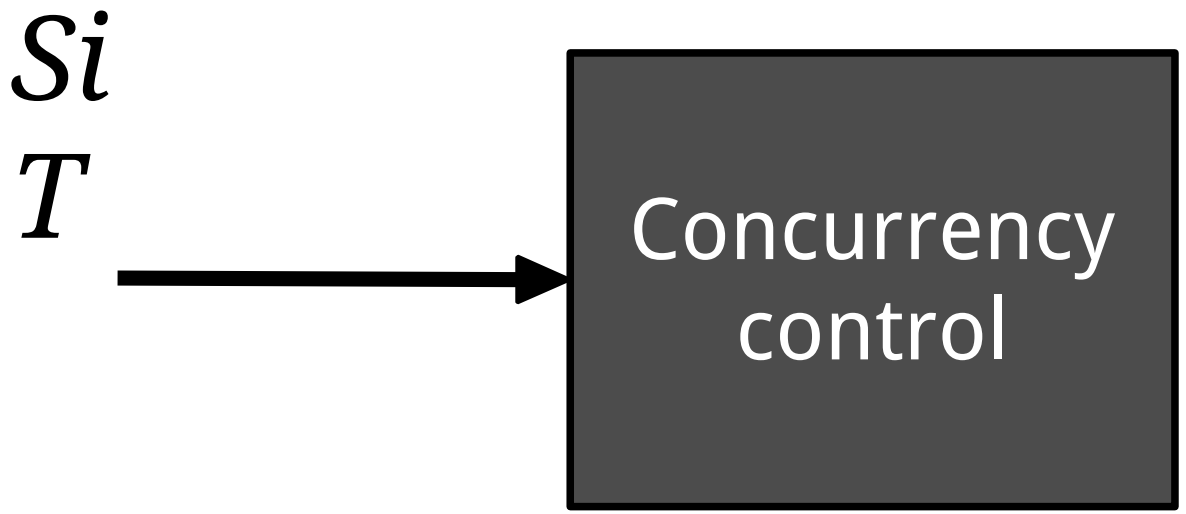
# Serializability



# Serializability

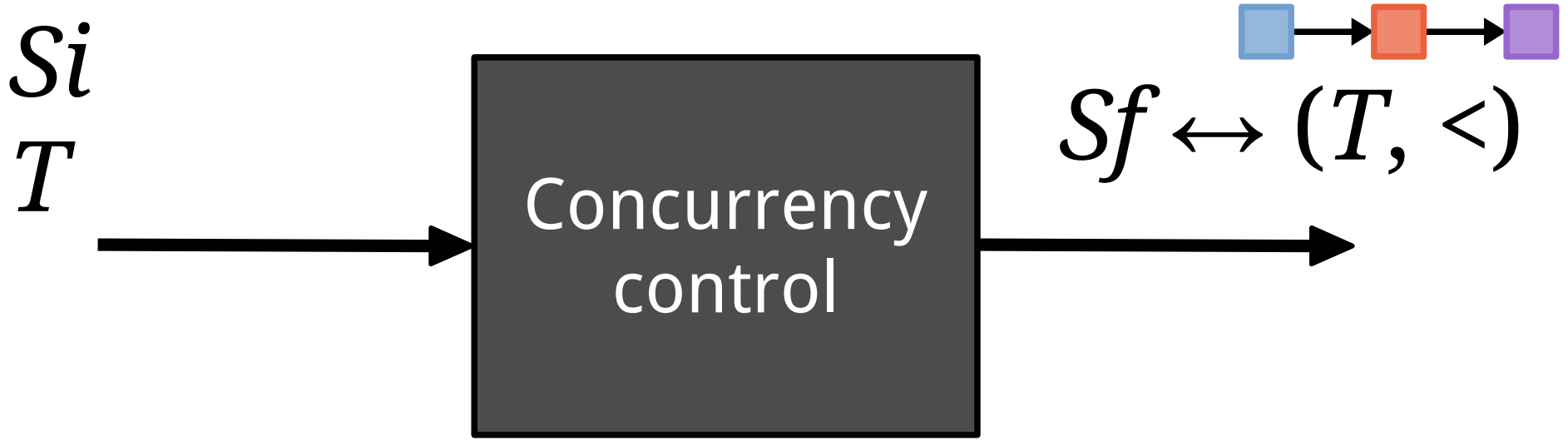


# Traditional concurrency control

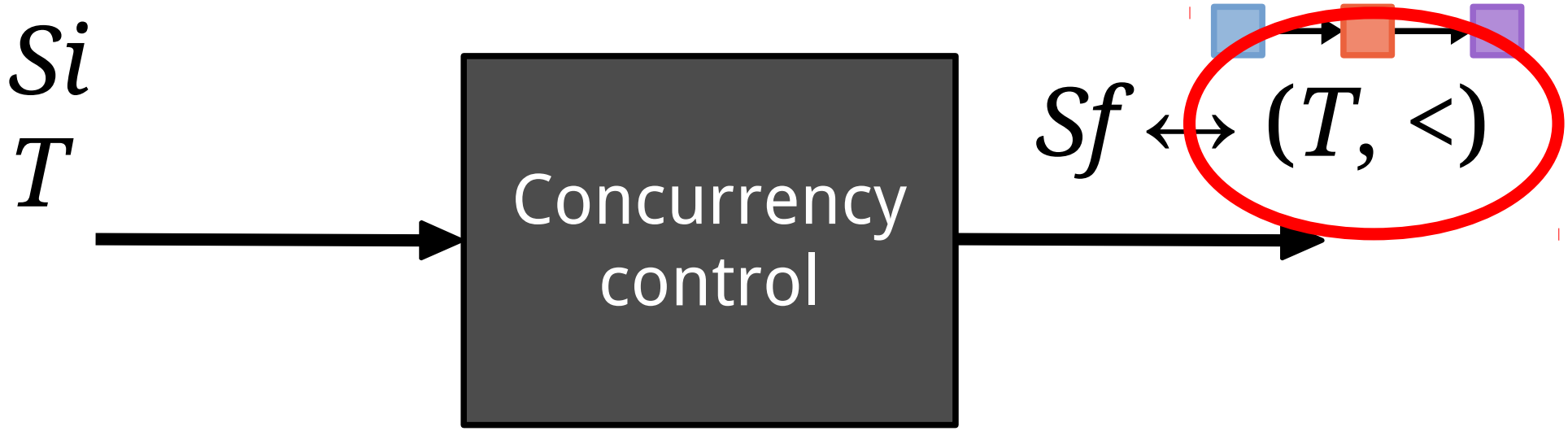


# Traditional concurrency control

# Traditional concurrency control



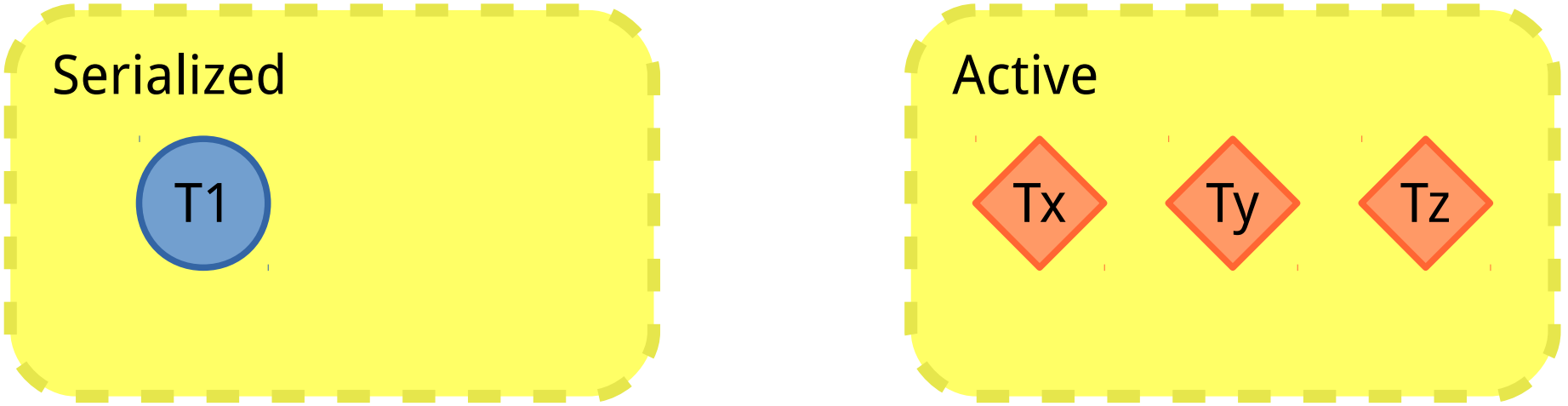
# Traditional concurrency control



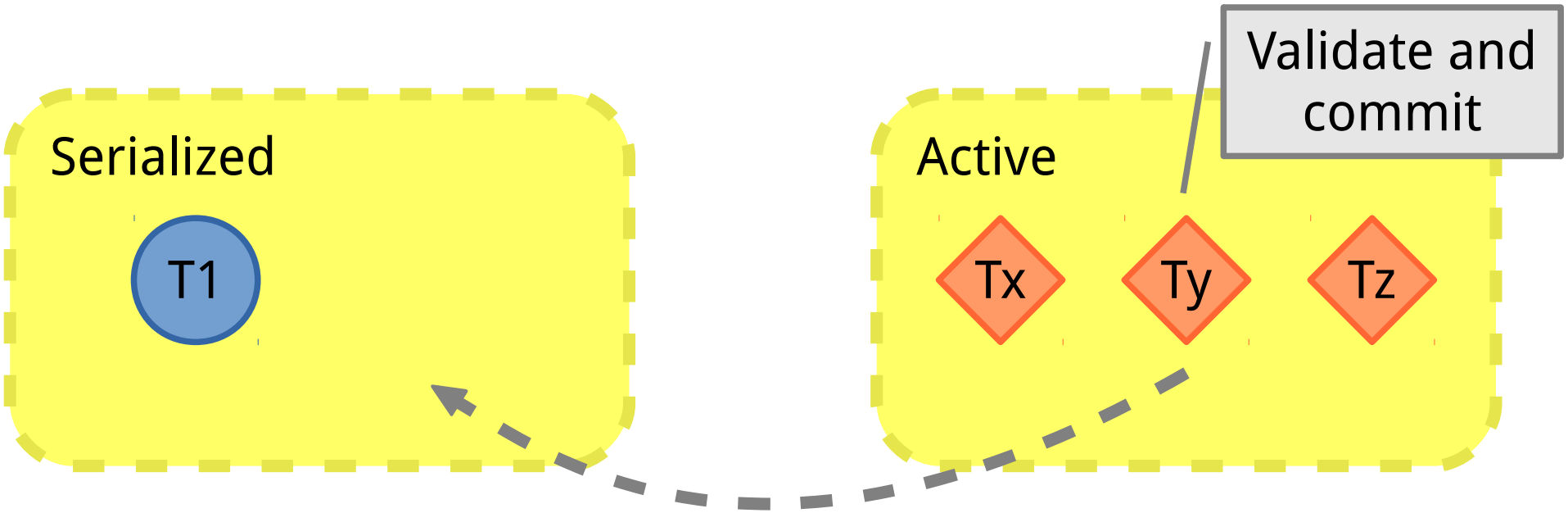

Computed nondeterministically during transaction runtime

# Optimistic concurrency control (OCC)

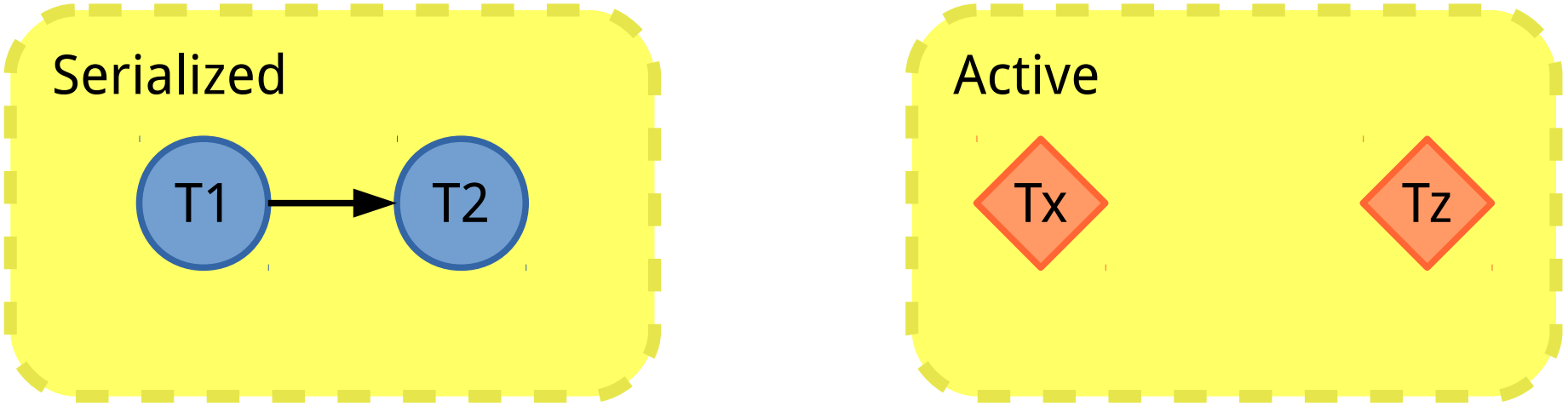
# Optimistic concurrency control



# Optimistic concurrency control



# Optimistic concurrency control



Two distinct tasks at the same time:

(1) defining the serialization order;

Two distinct tasks at the same time:

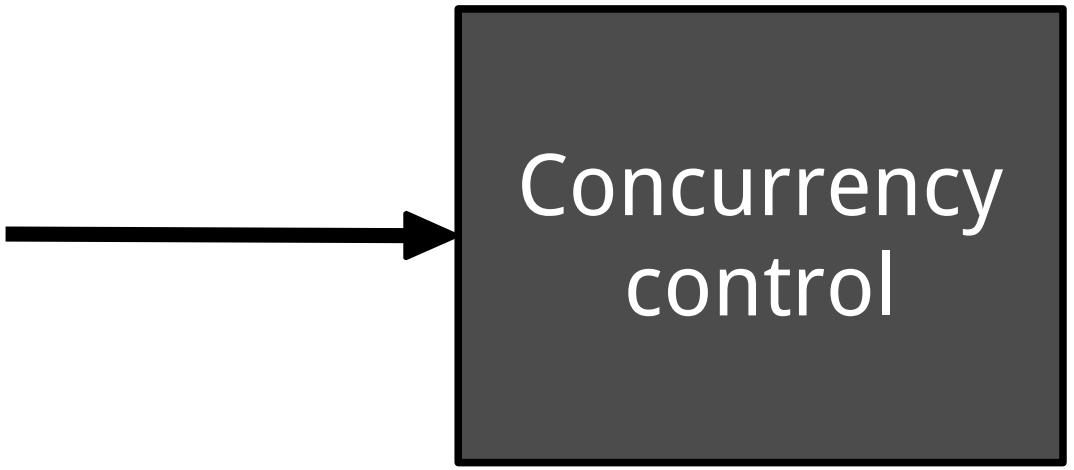
(1) defining the serialization order;

(2) controlling the concurrent execution to respect the defined order.

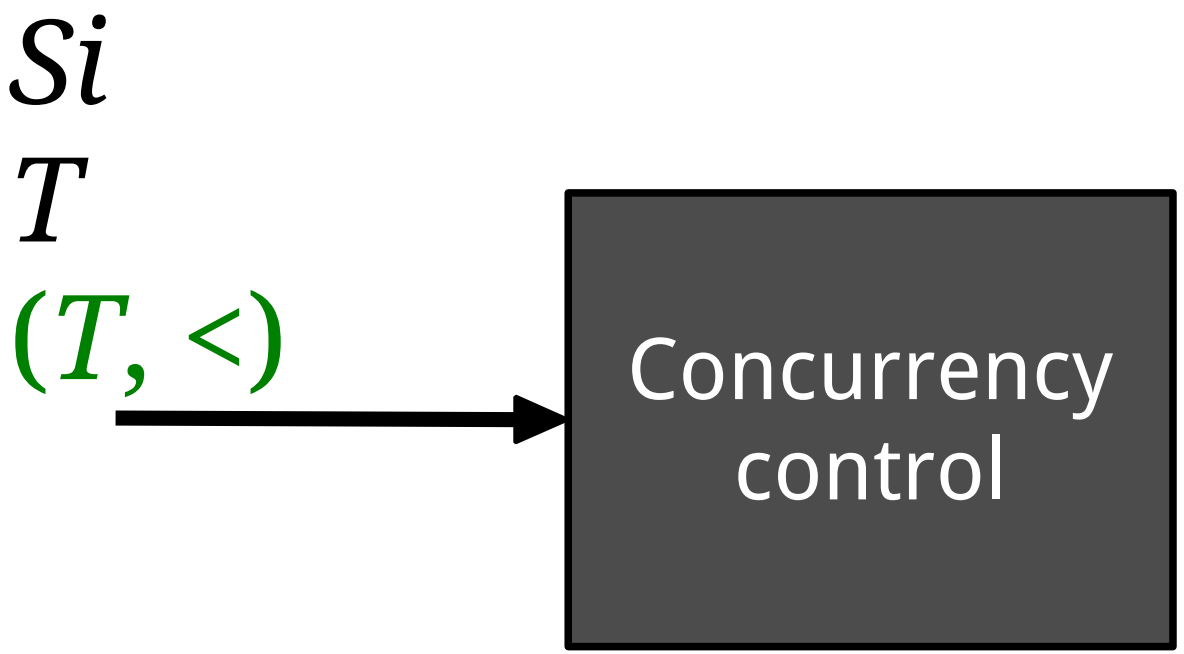
We propose to **separate** them!

# Deterministic concurrency control

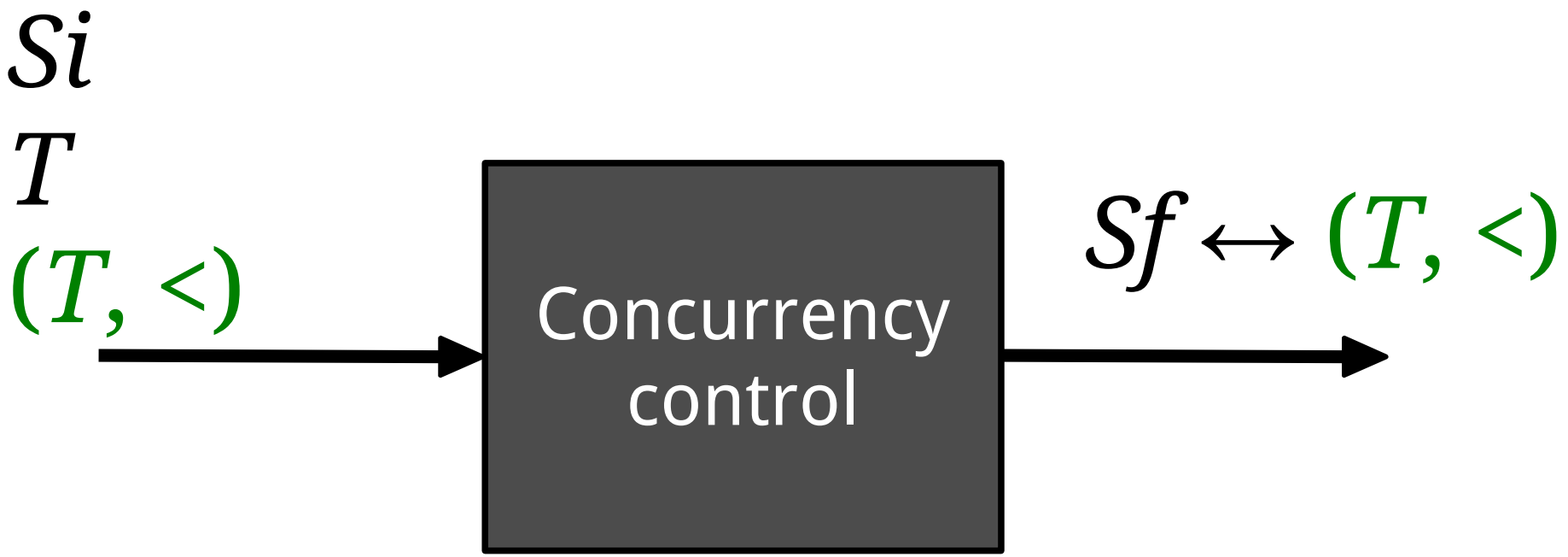
$Si$
$T$



# Deterministic concurrency control

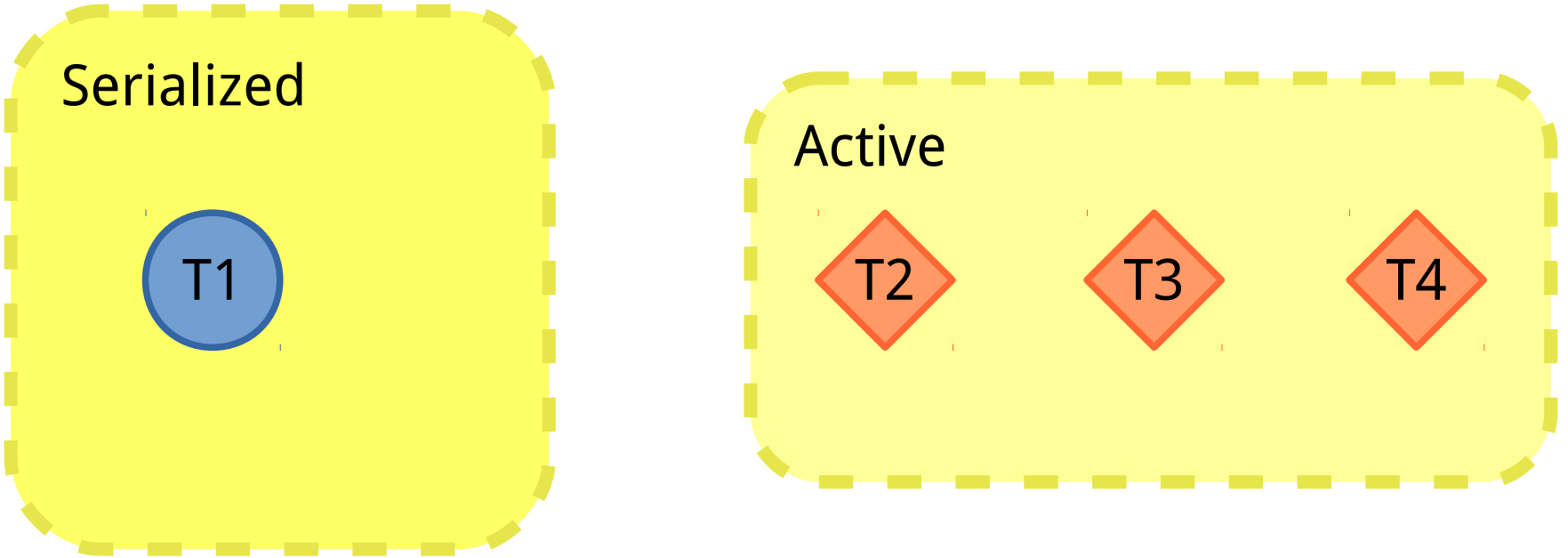


# Deterministic concurrency control

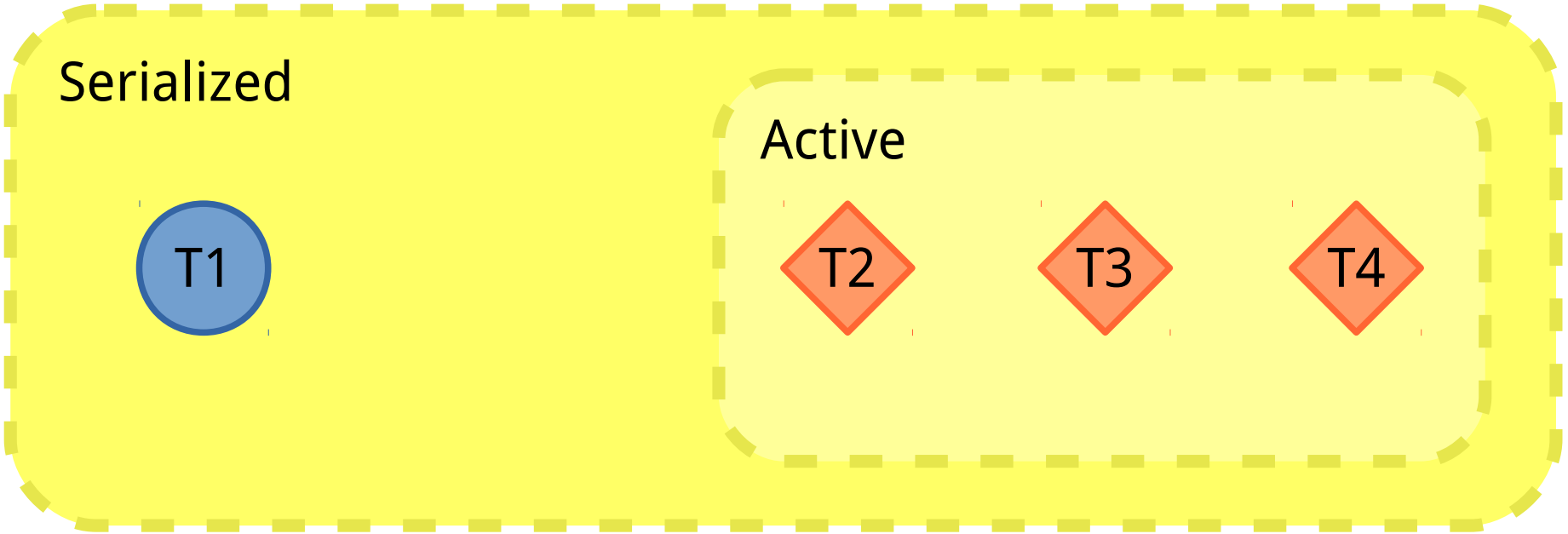


# Optimistic Concurrency Control (OCC)
# **+ Ordered commit**

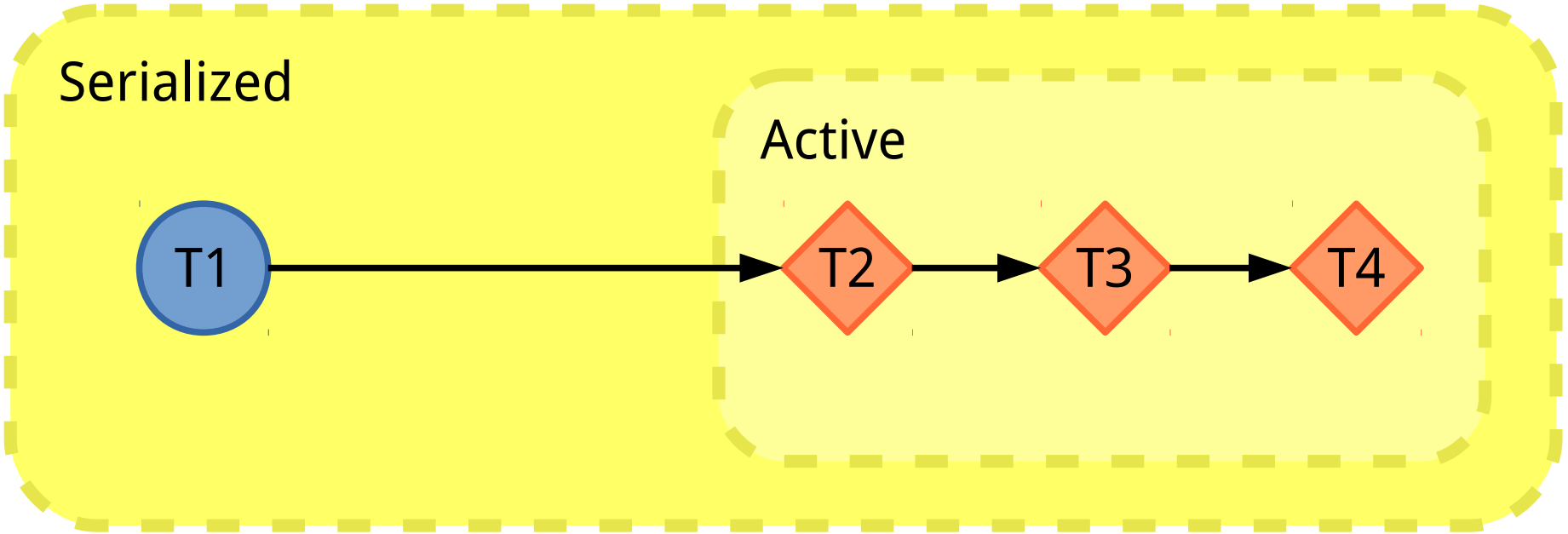
# Ordered commit



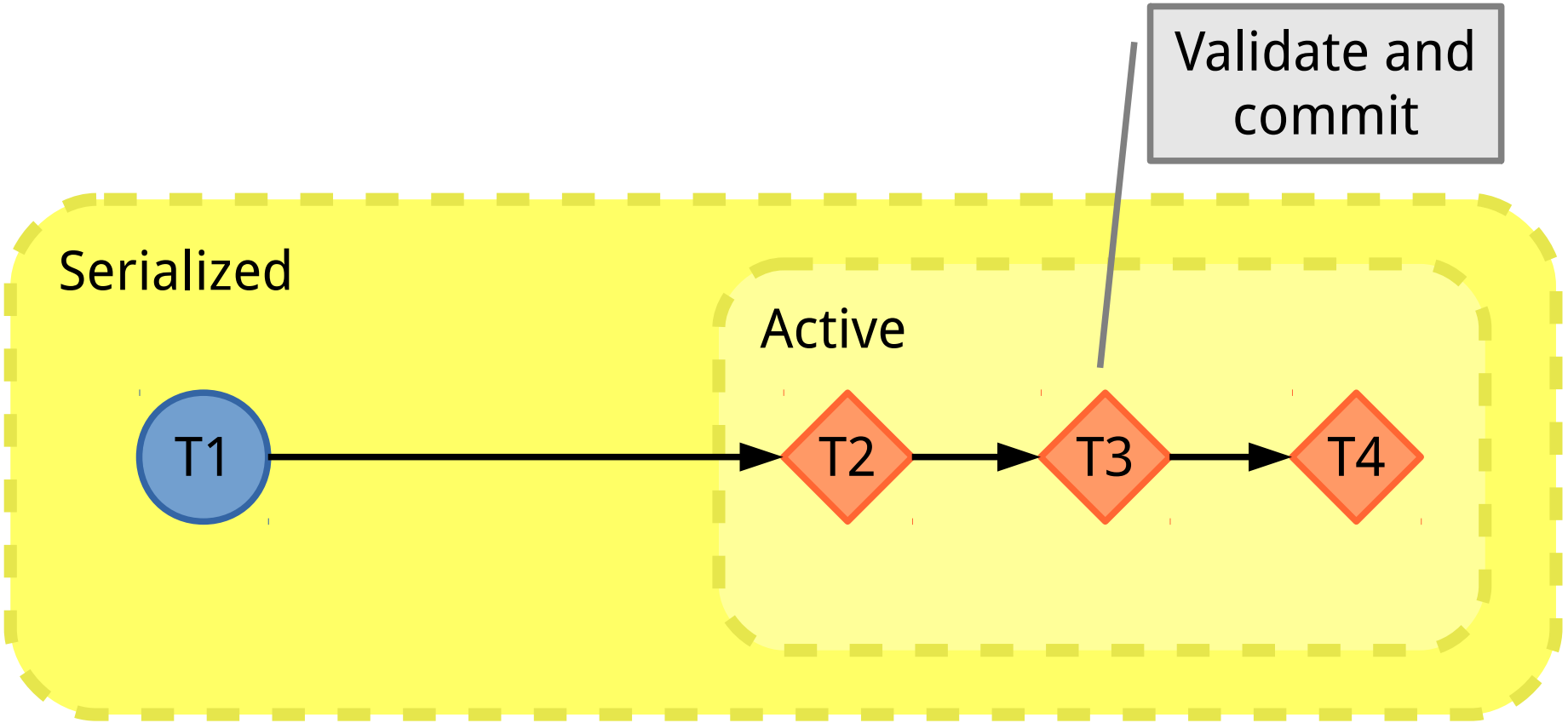
# Ordered commit



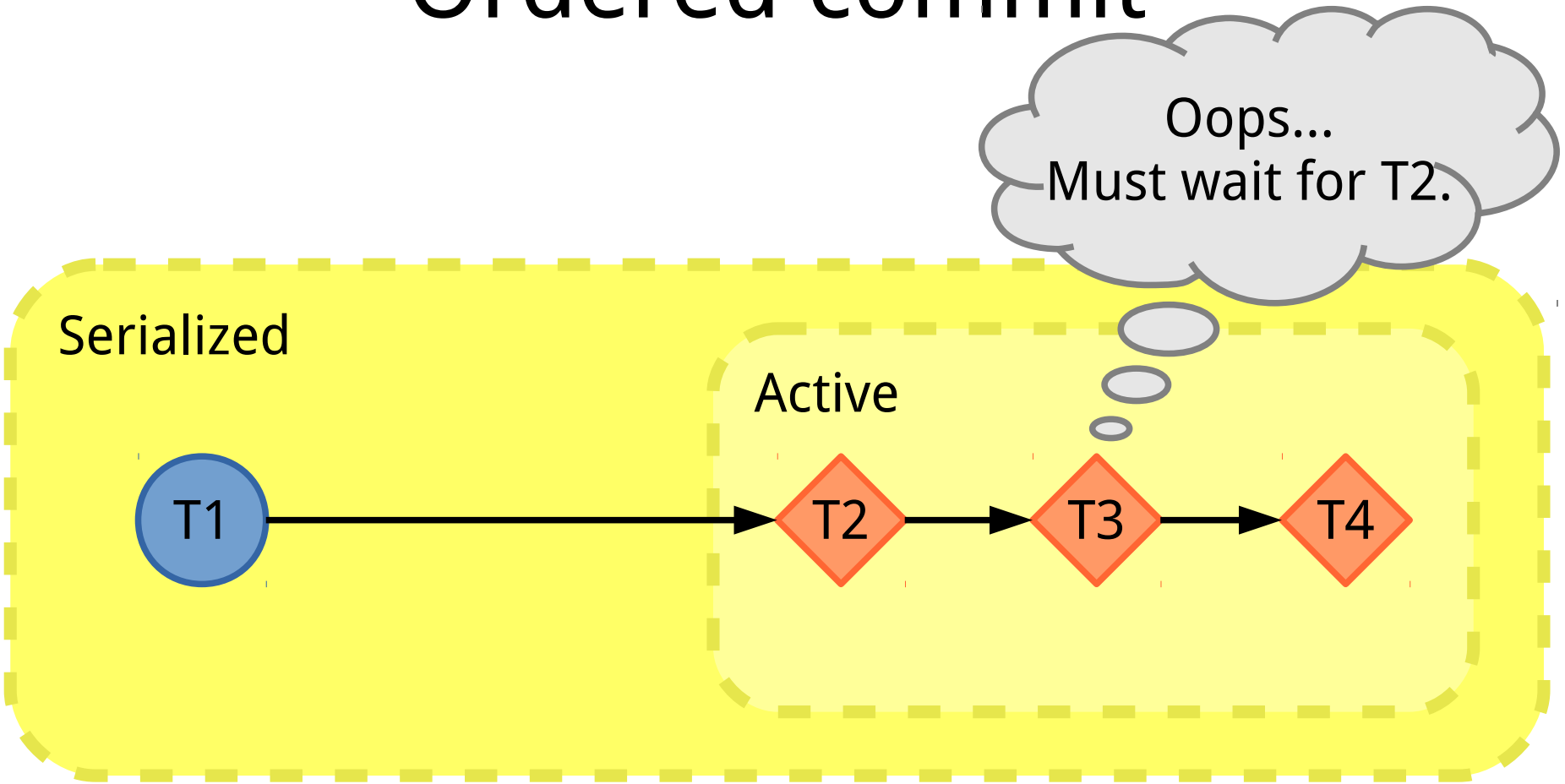
# Ordered commit



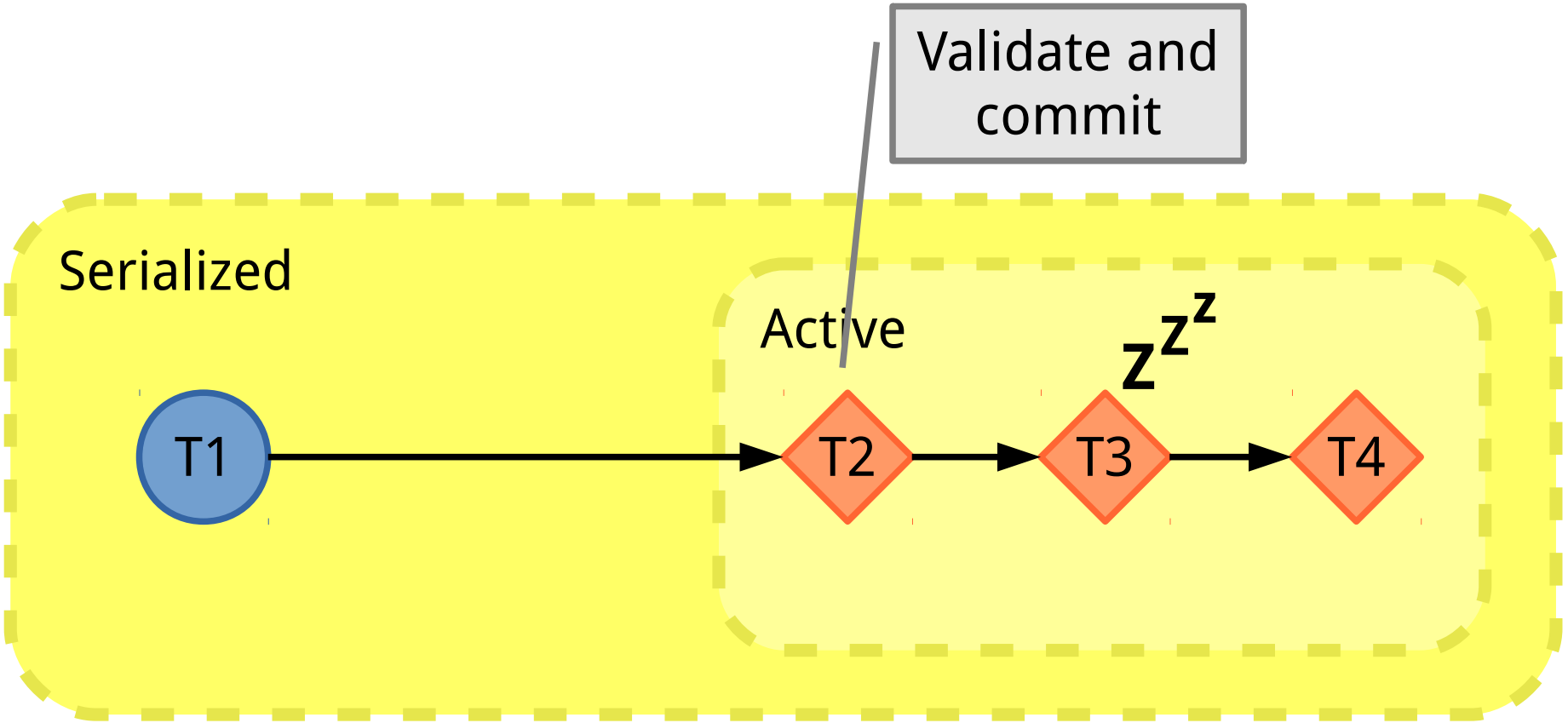
# Ordered commit



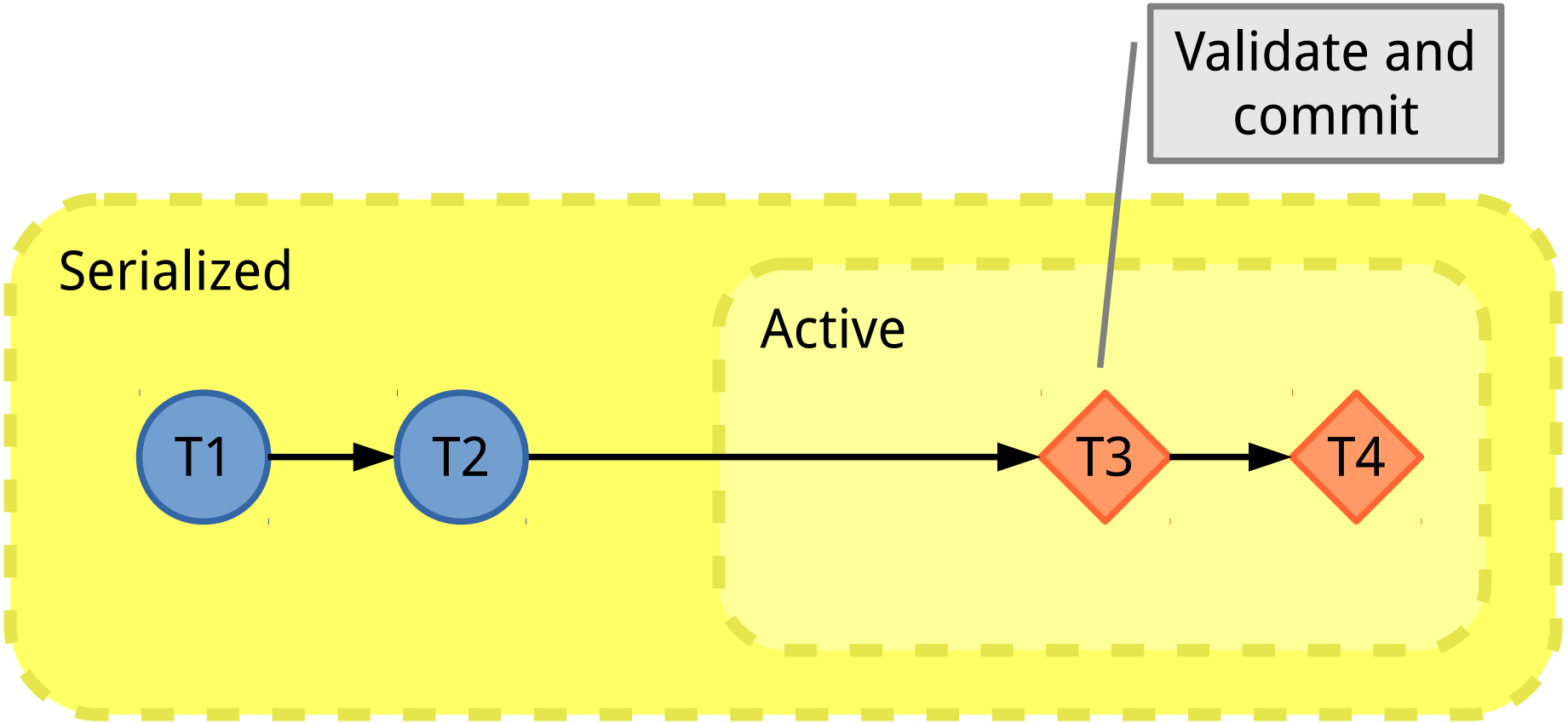
# Ordered commit



# Ordered commit

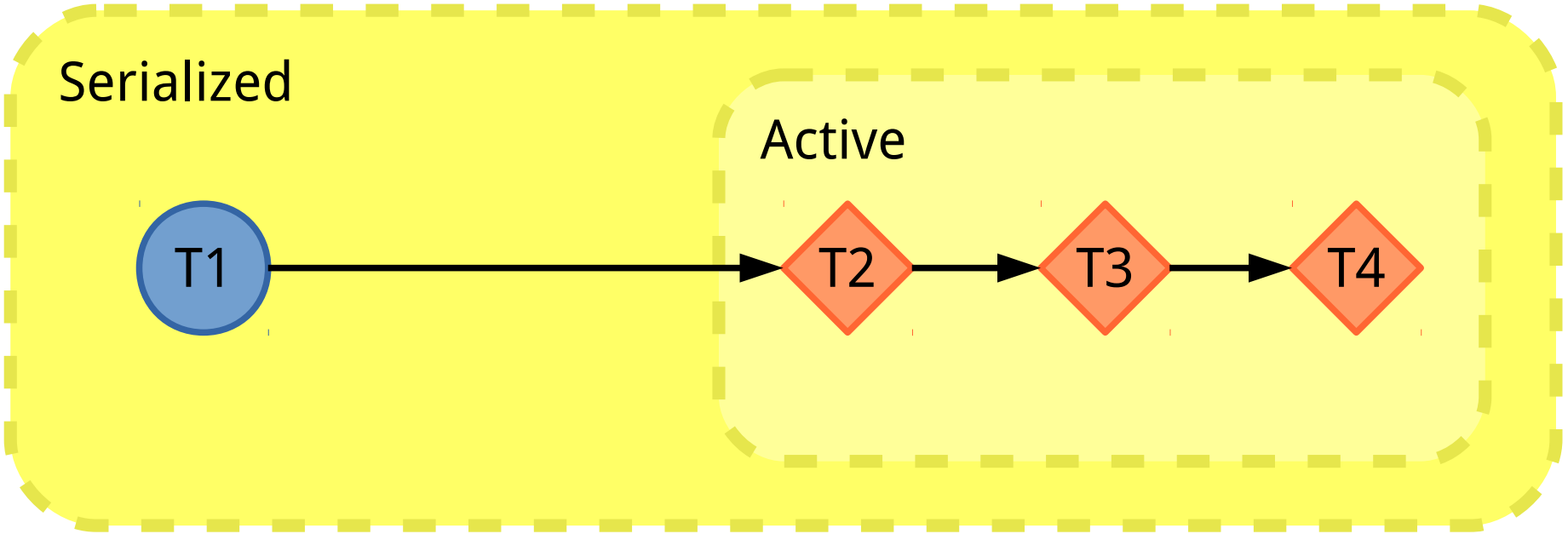


# Ordered commit

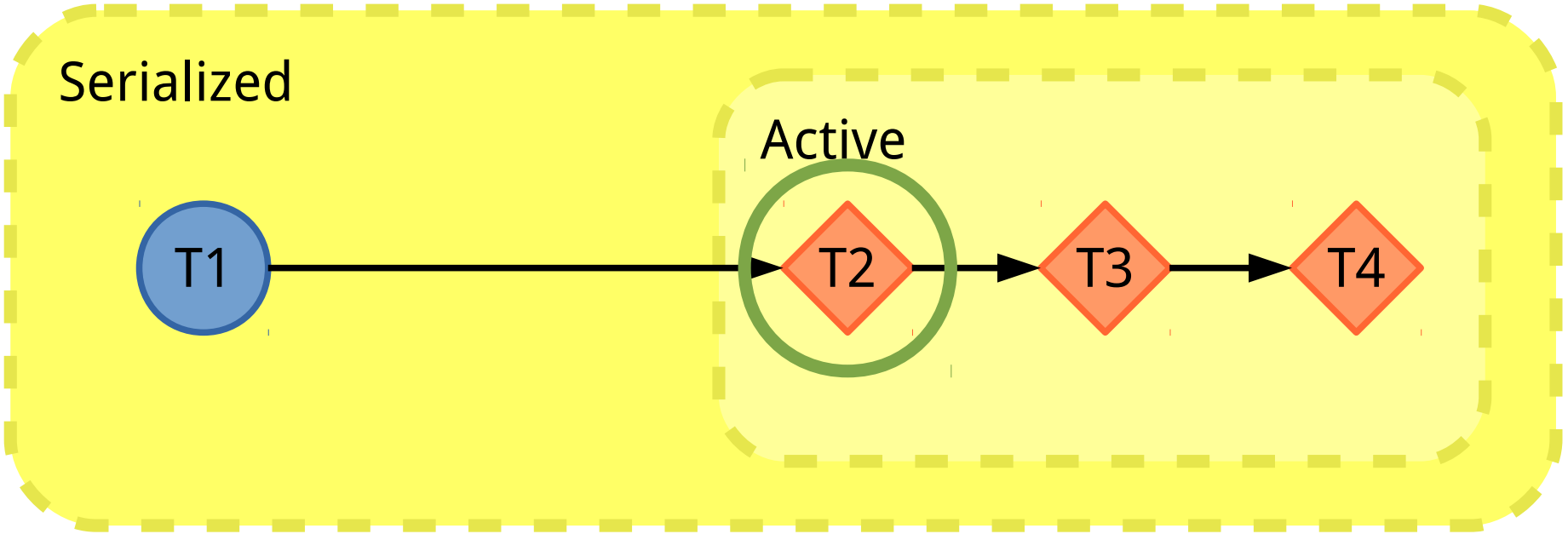


Optimistic Concurrency Control (OCC)
+ Ordered commit
**+ Fast transactions**

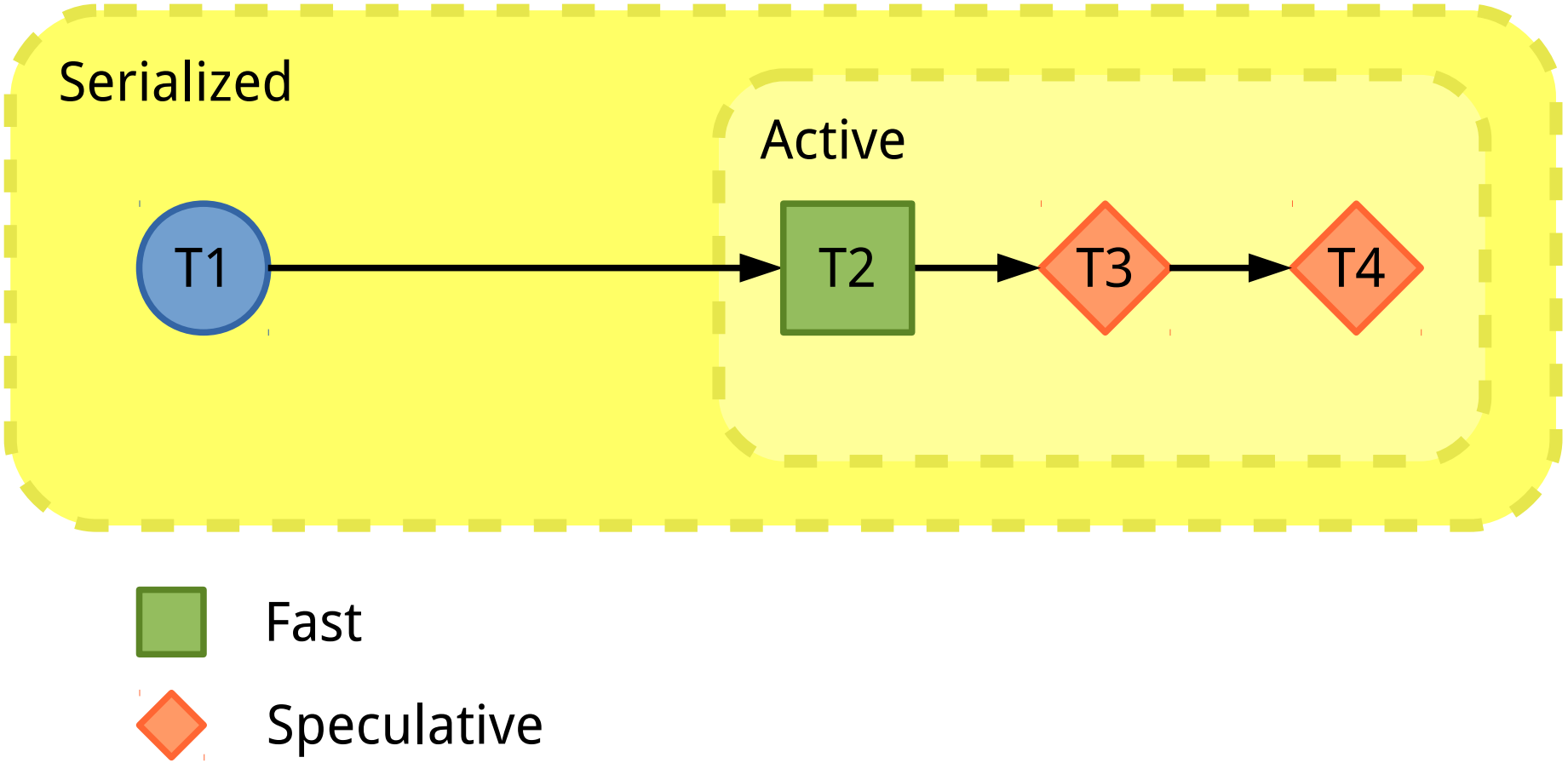
# Fast transactions



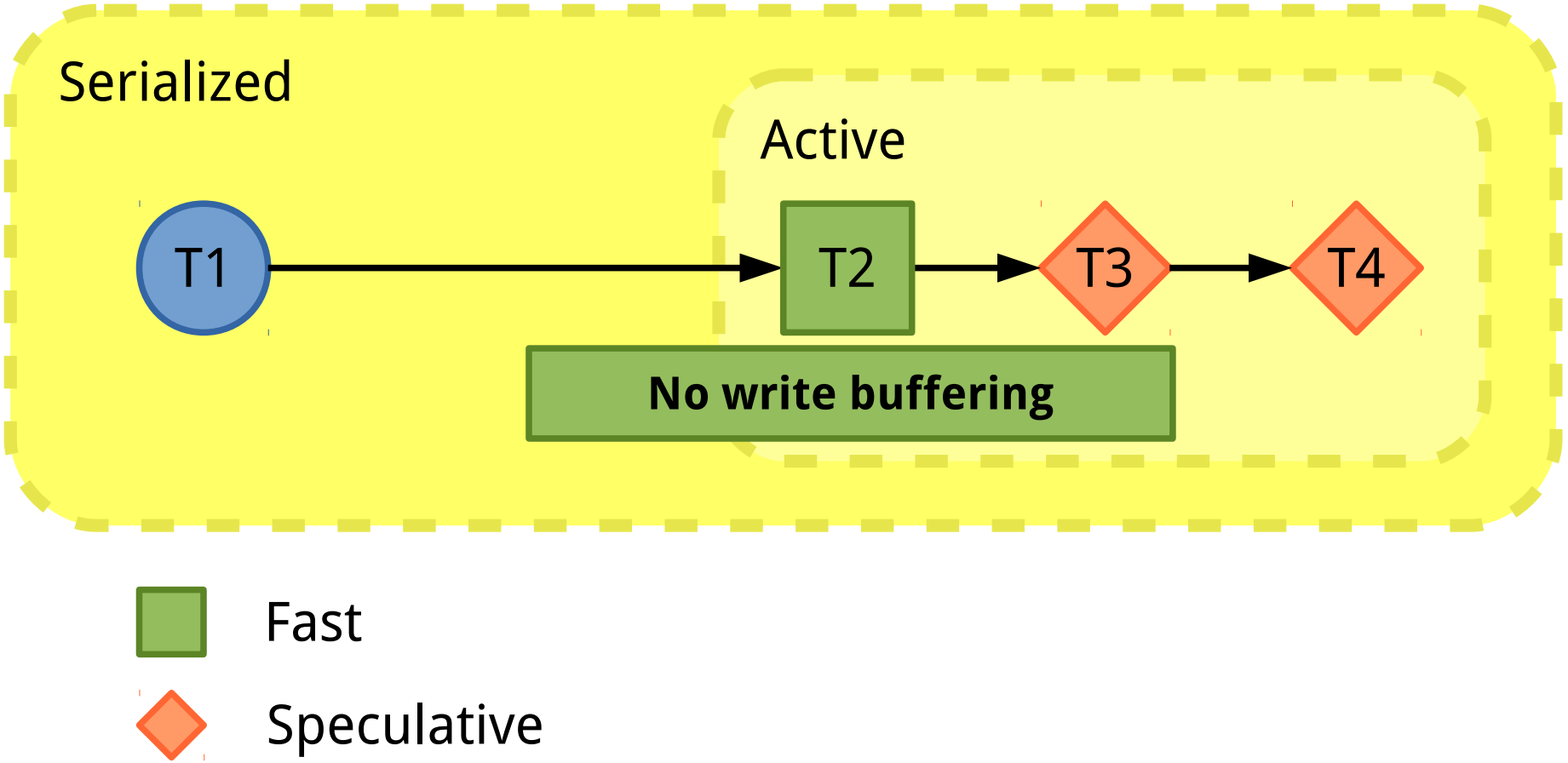
# Fast transactions



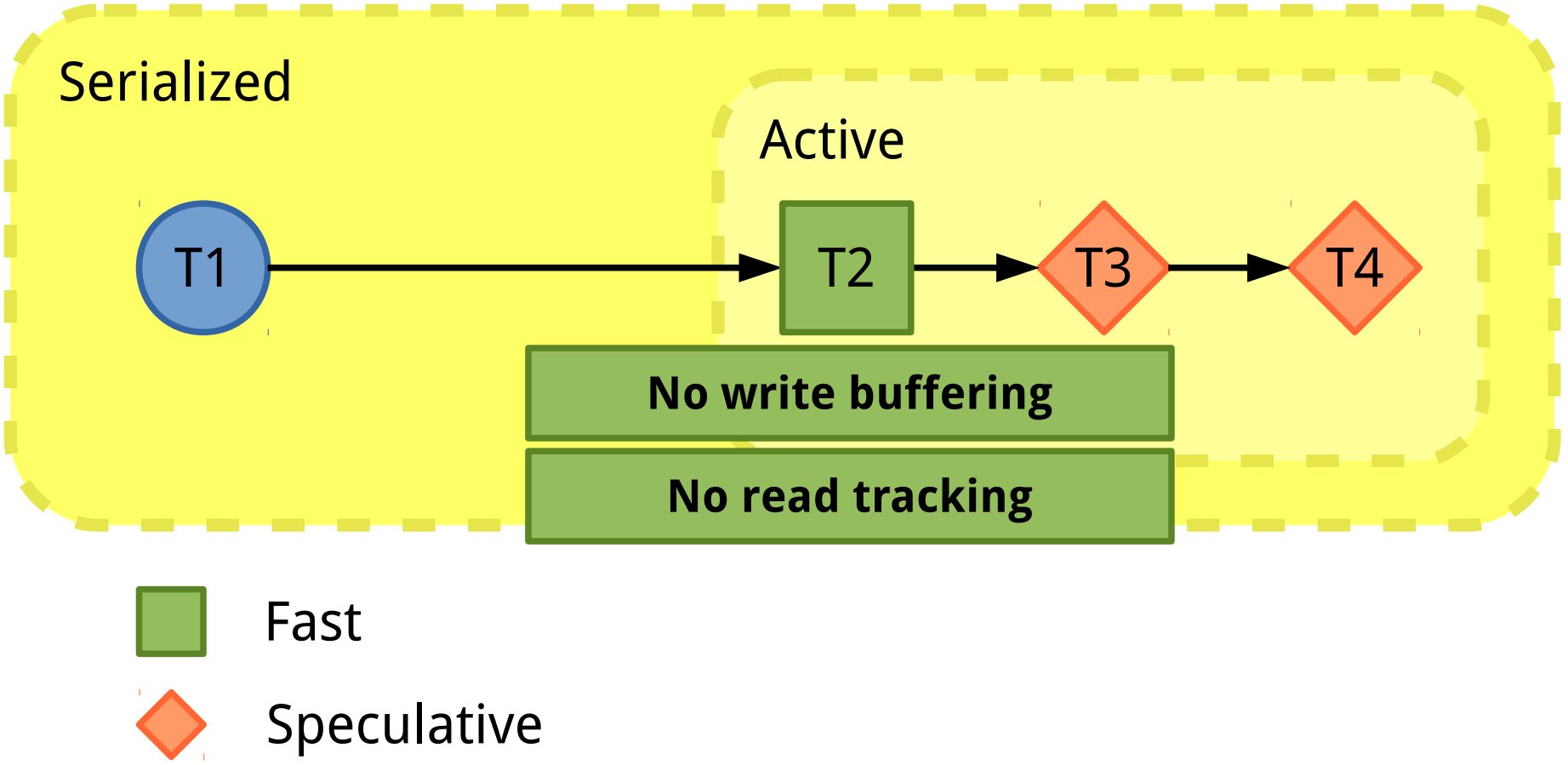
# Fast transactions



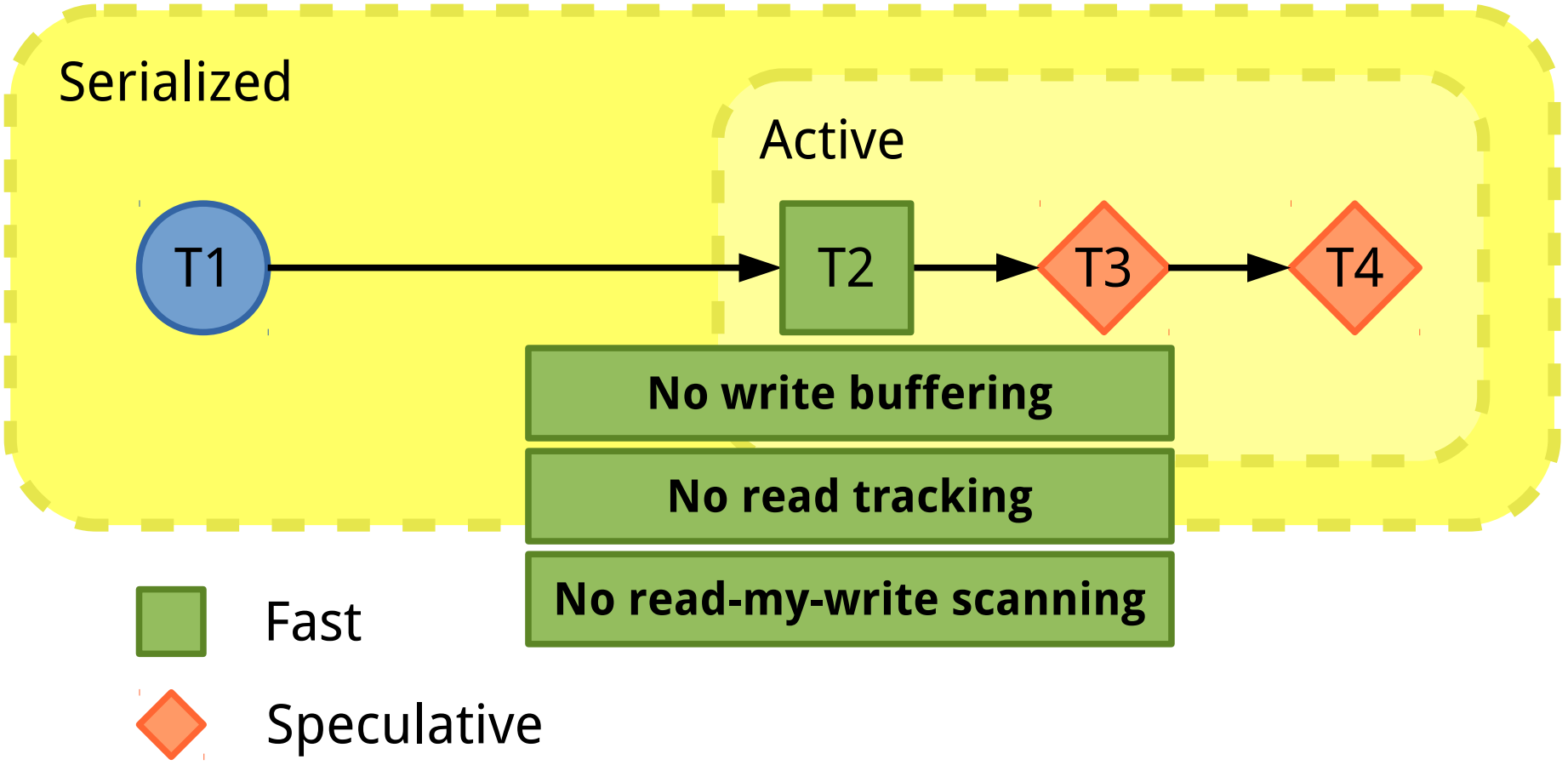
# Fast transactions



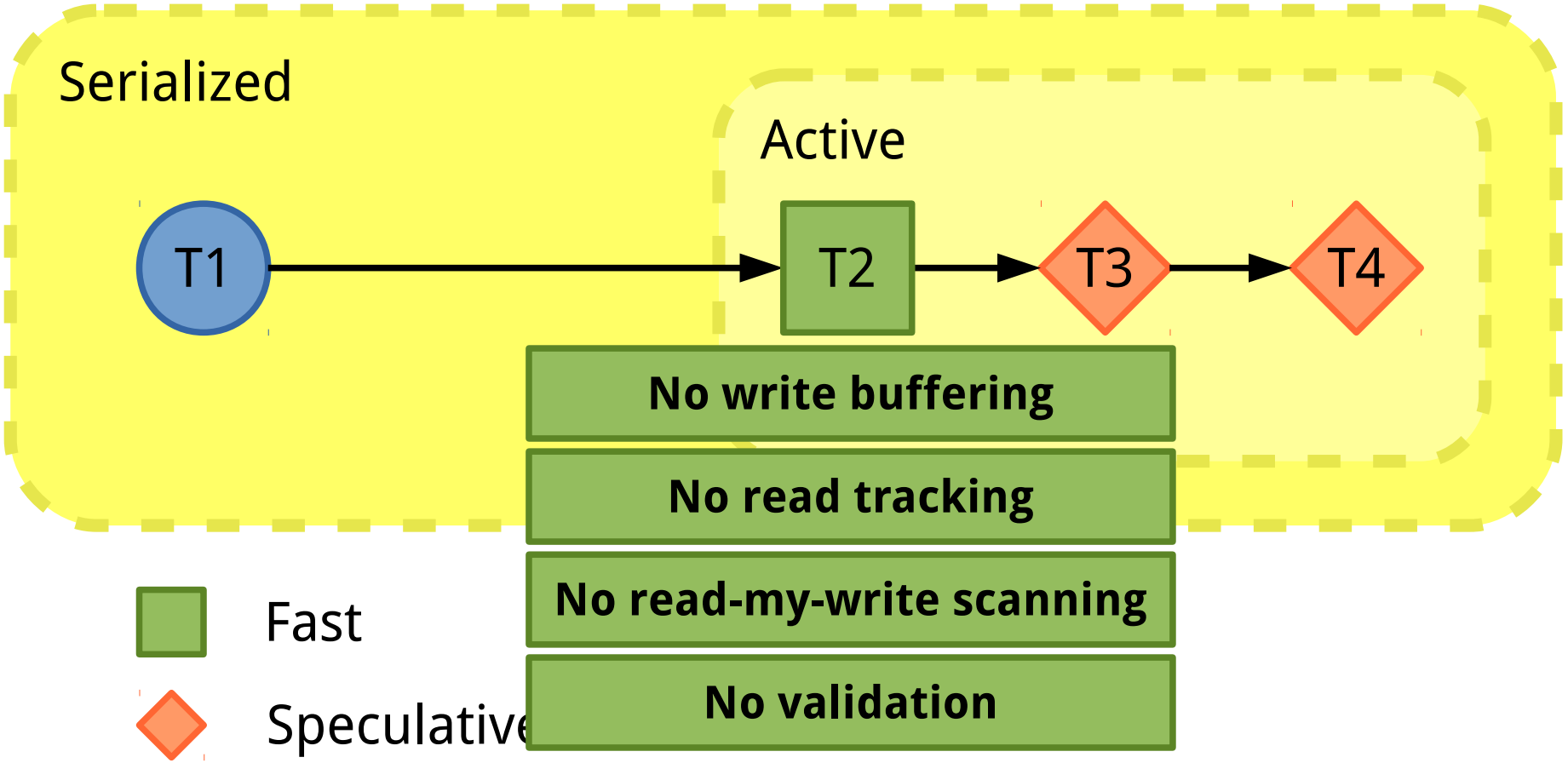
# Fast transactions



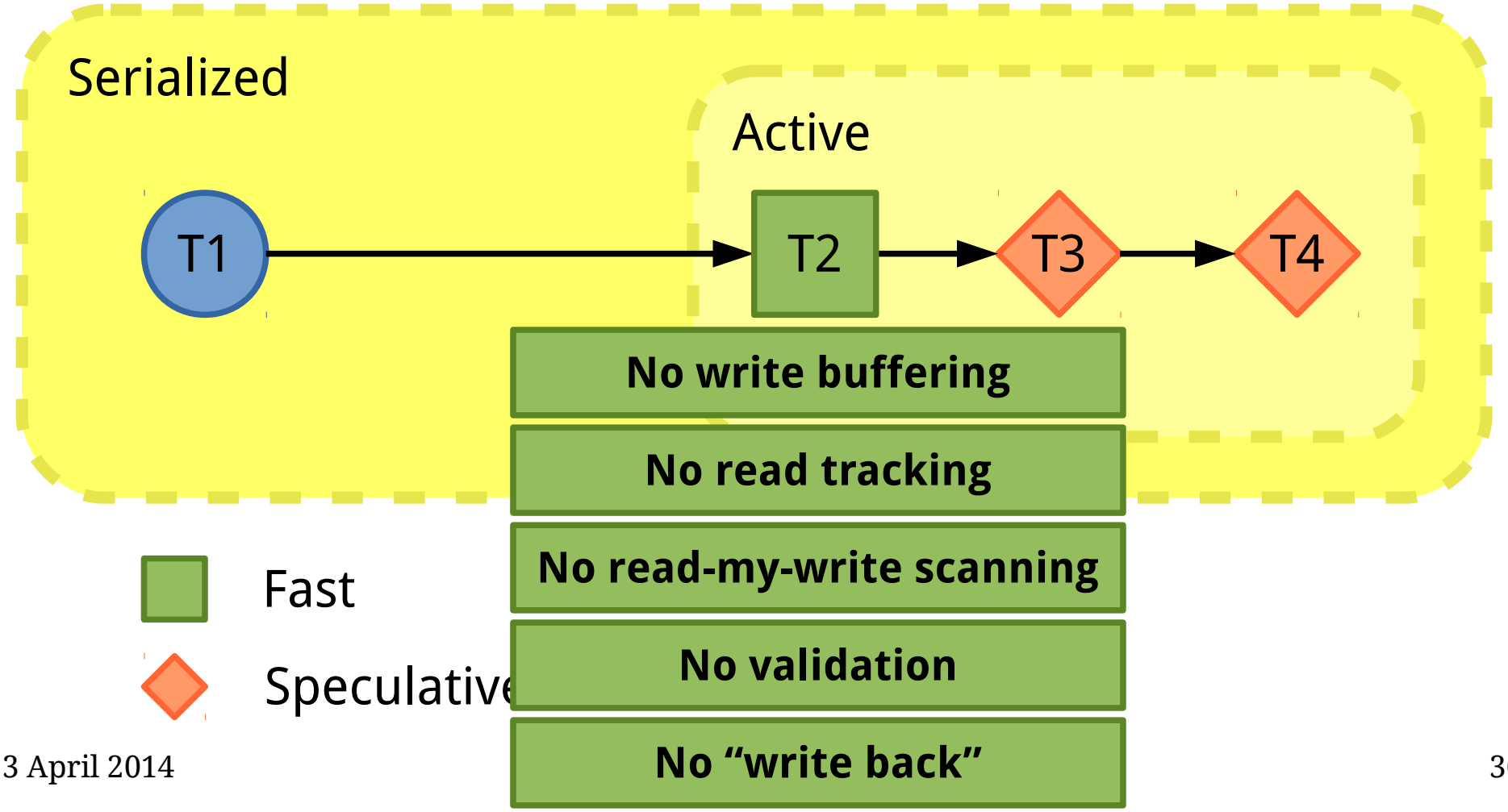
# Fast transactions



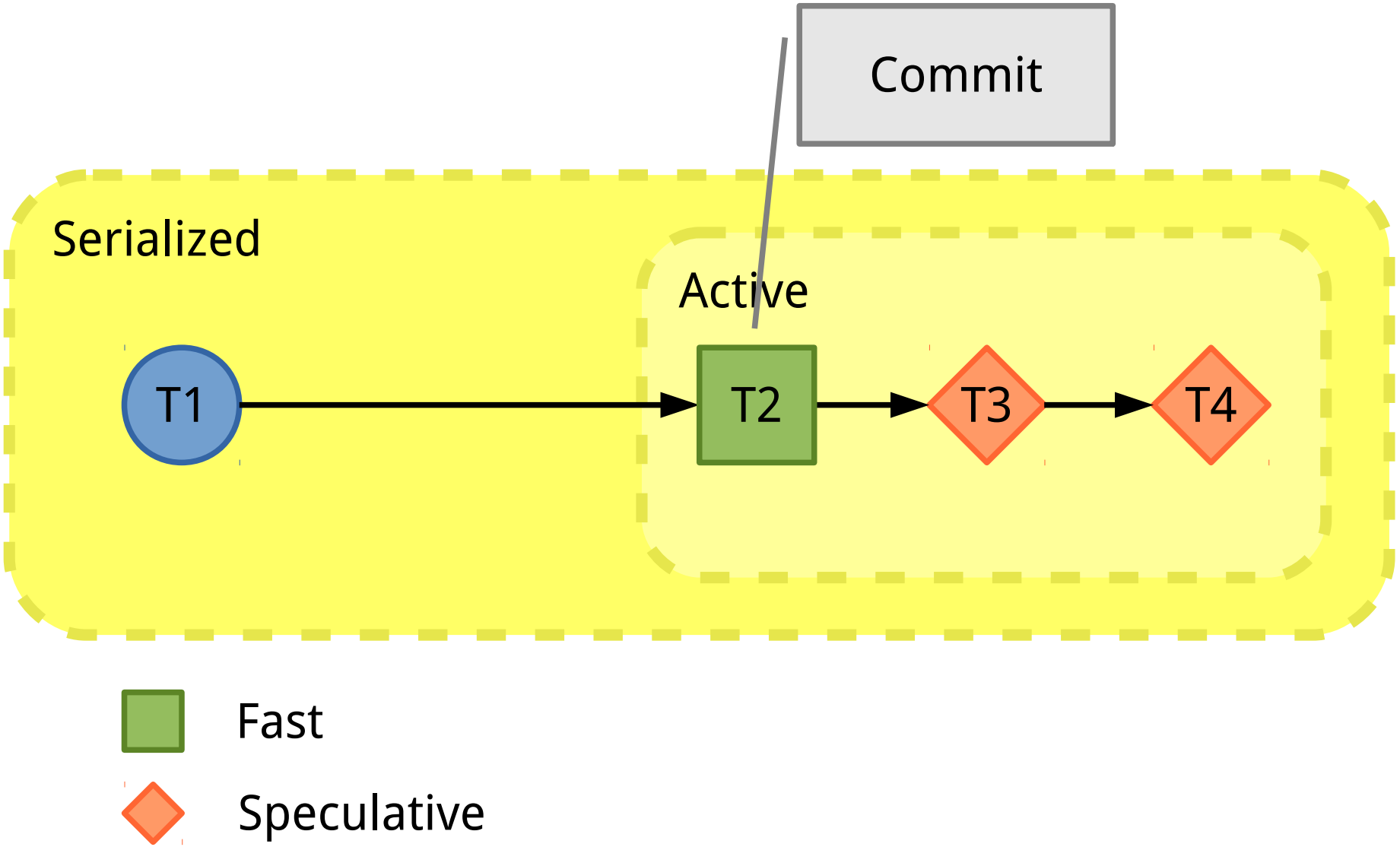
# Fast transactions



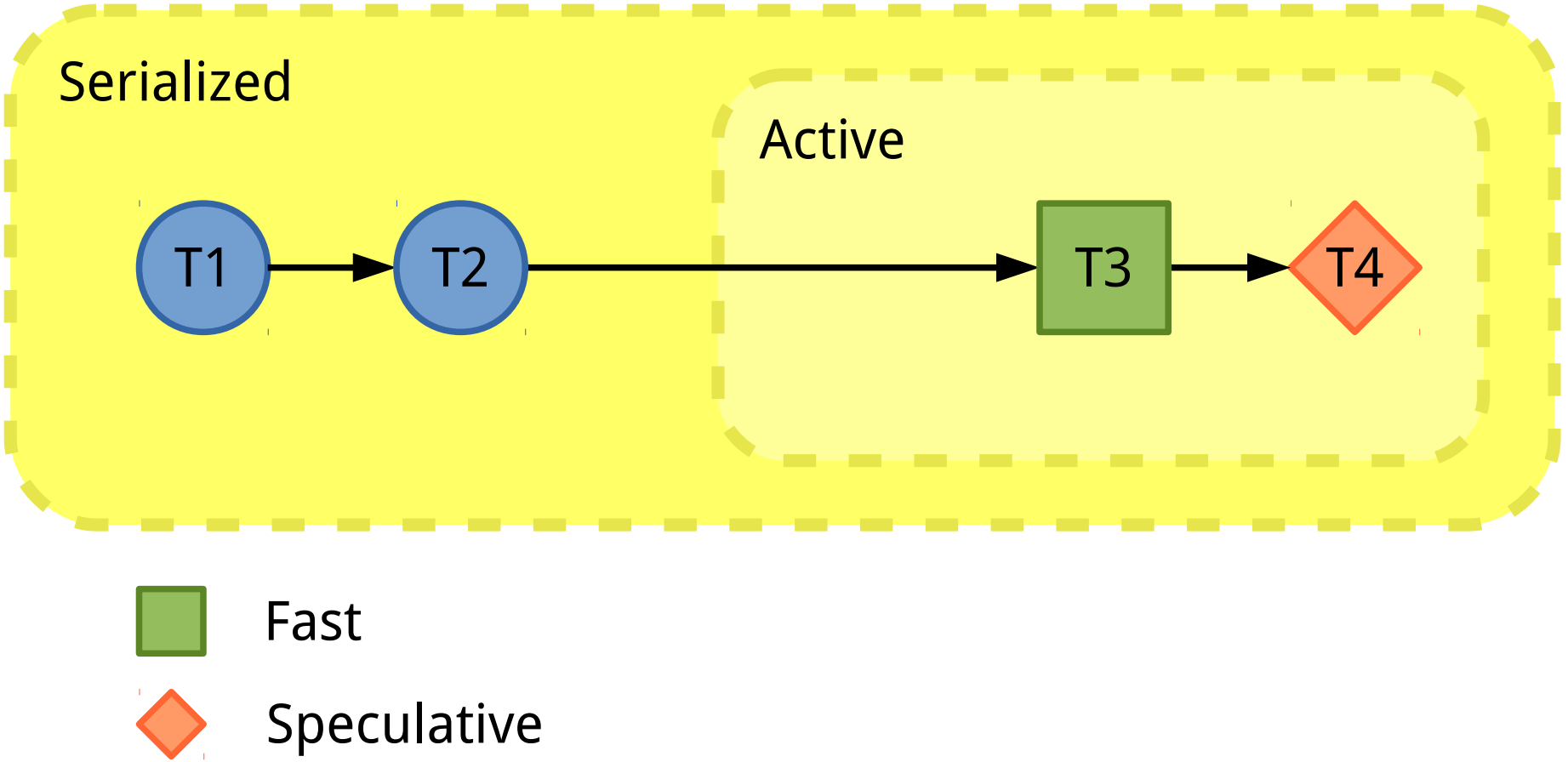
# Fast transactions



# Fast transactions



# Fast transactions



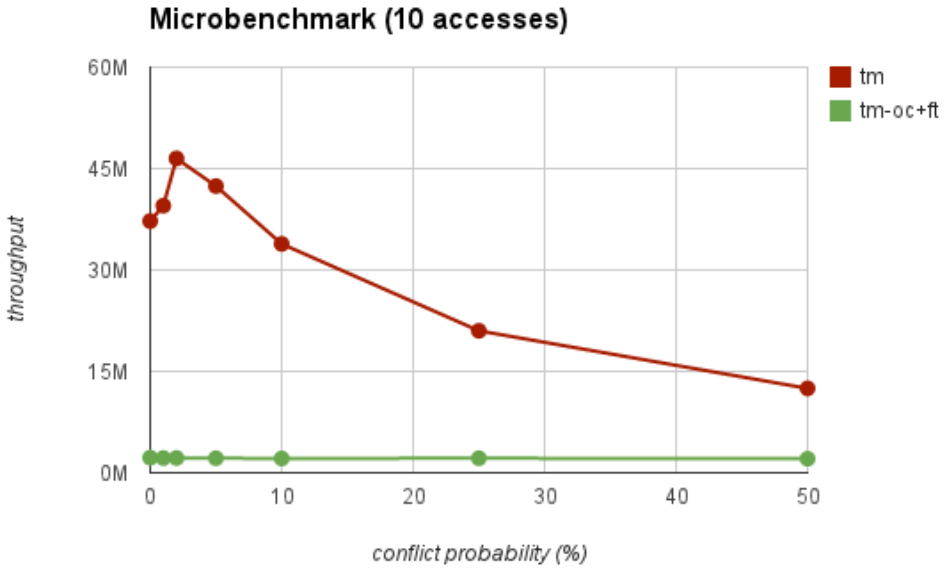
# Use cases

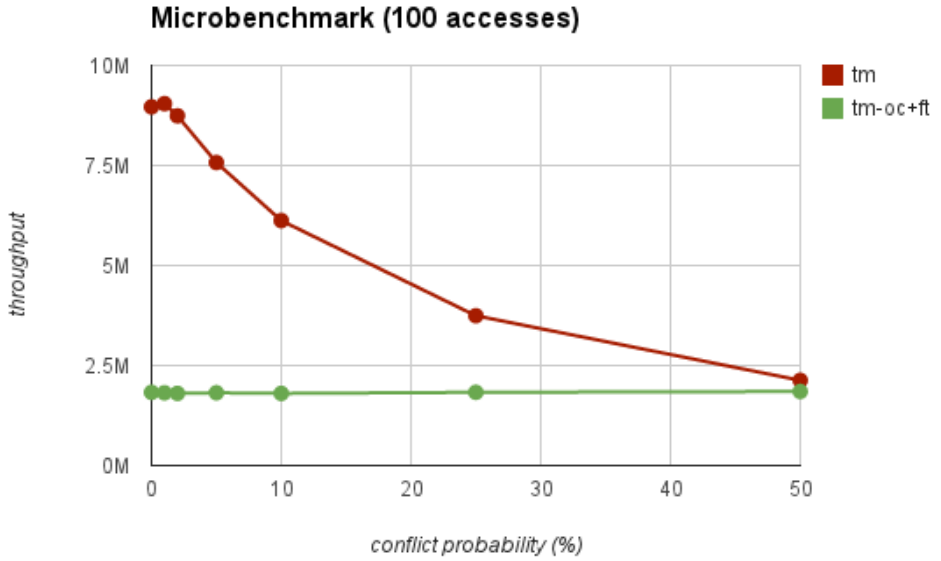
- State machine replication (SMR): agreement phase defines the total order
- Deterministic multi-threading: (DMT) threads take turns committing transactions
- Starvation-free (SF) transactions

# Understanding the tradeoff

# Understanding the tradeoff

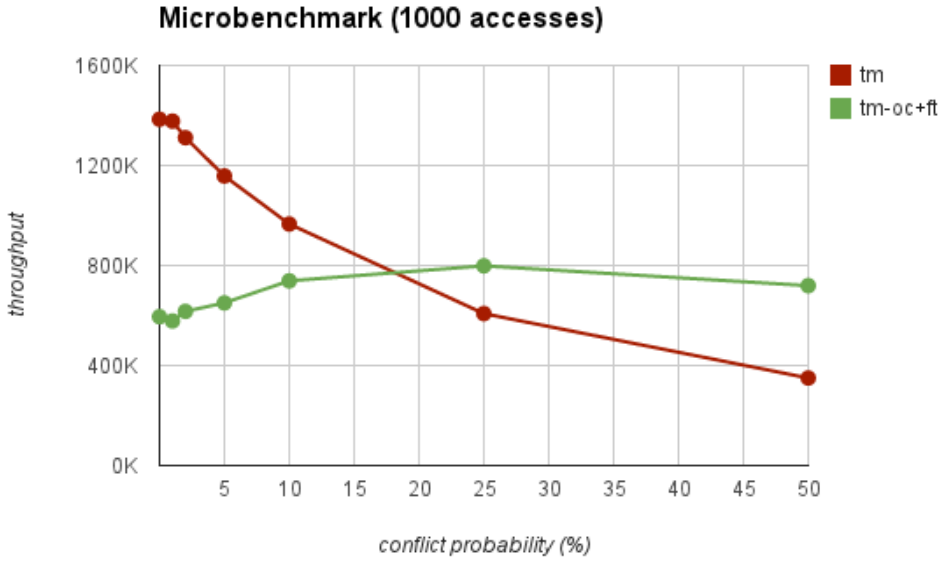
Transaction length

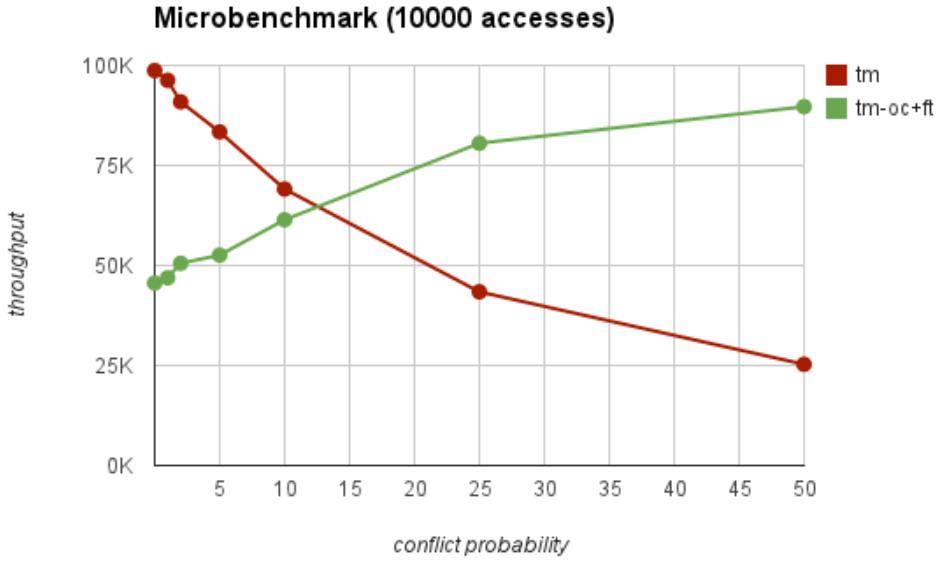


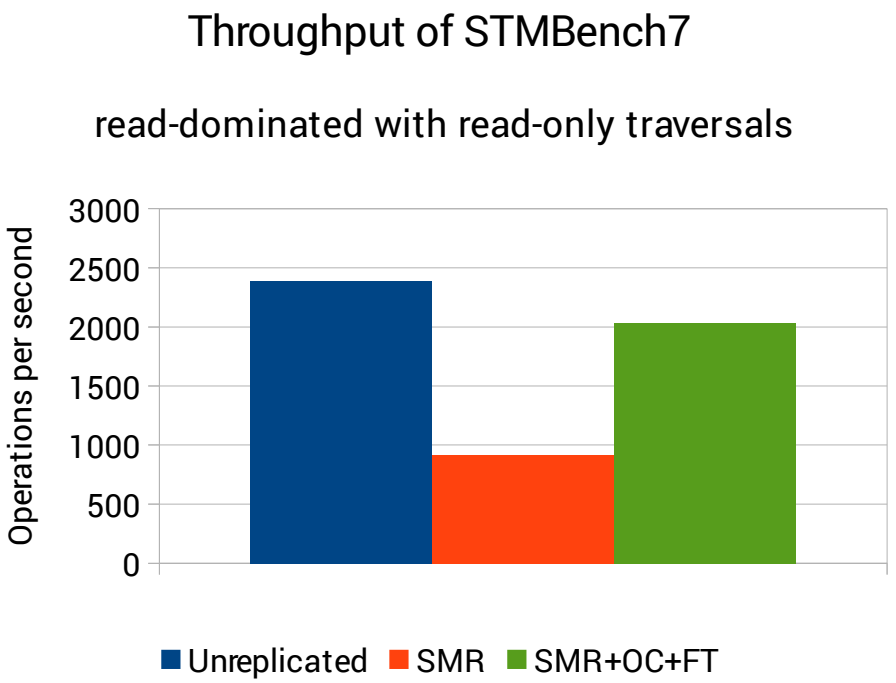
# Understanding the tradeoff

Transaction length

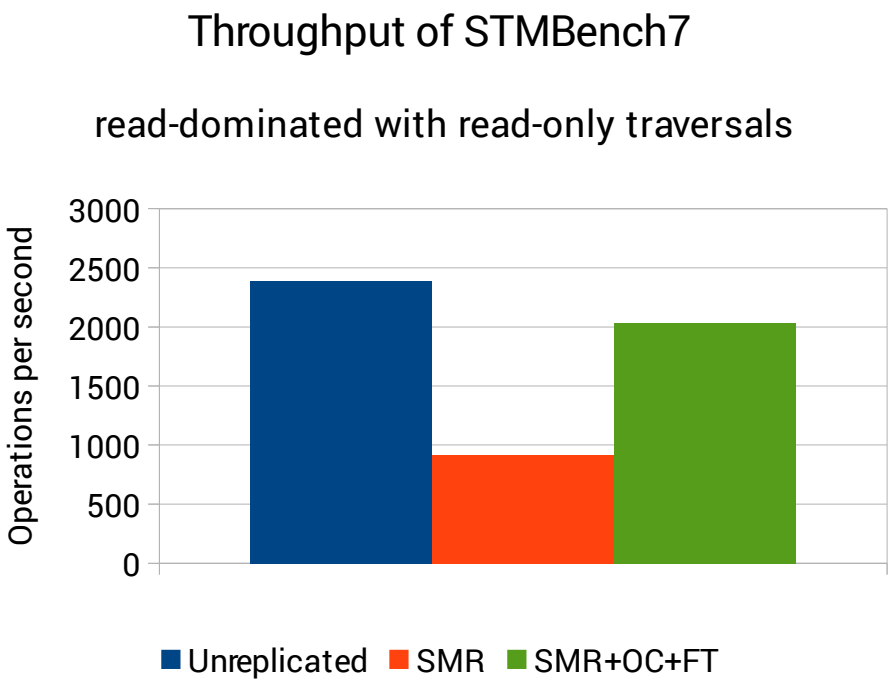


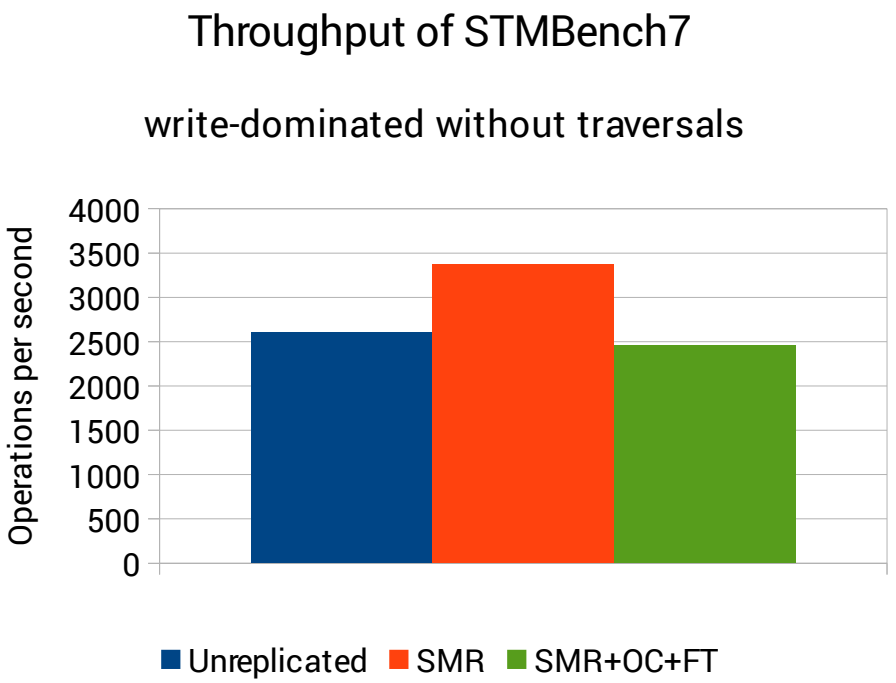



# Preliminary results: SMR

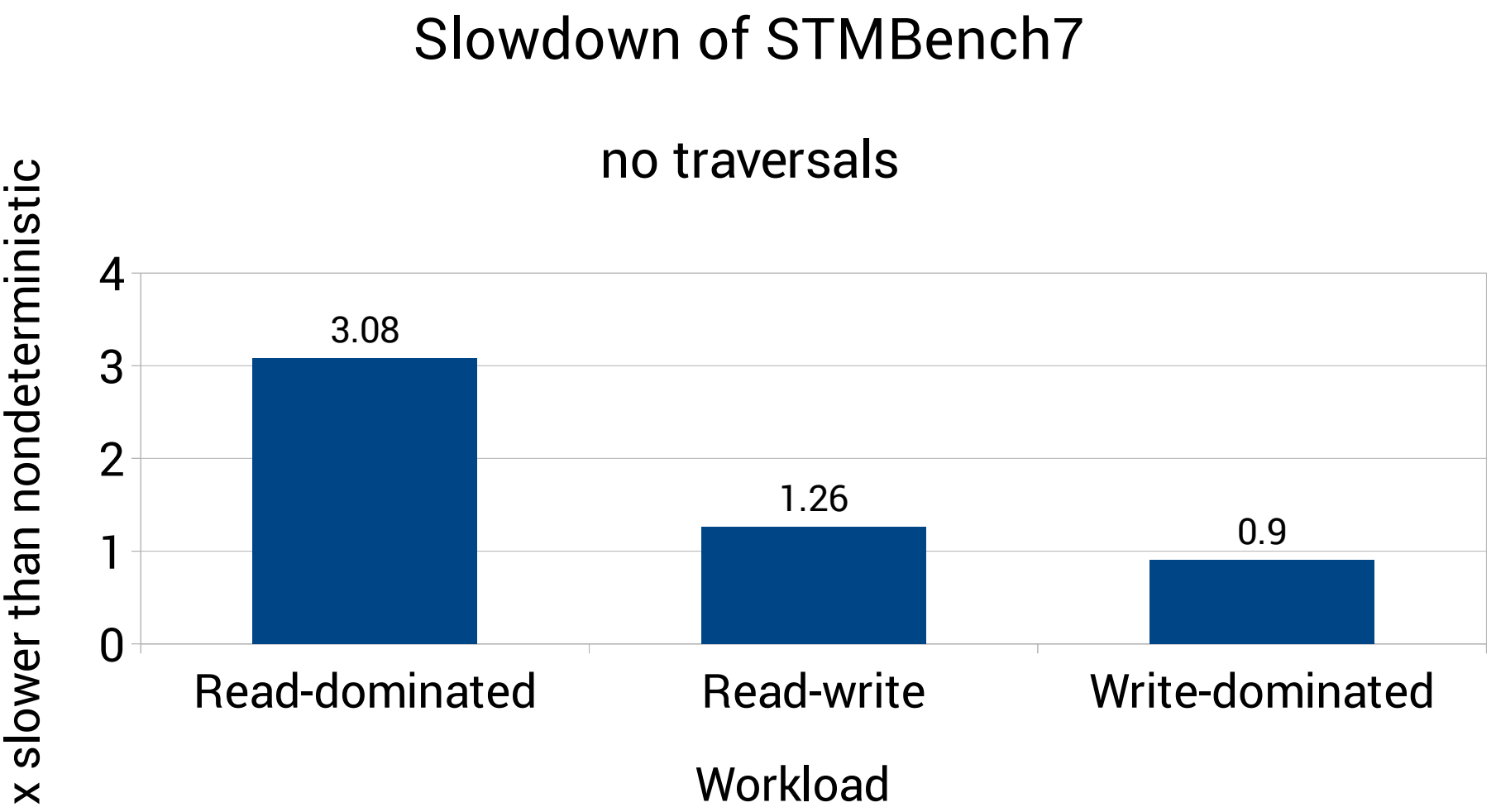


# Preliminary results: SMR

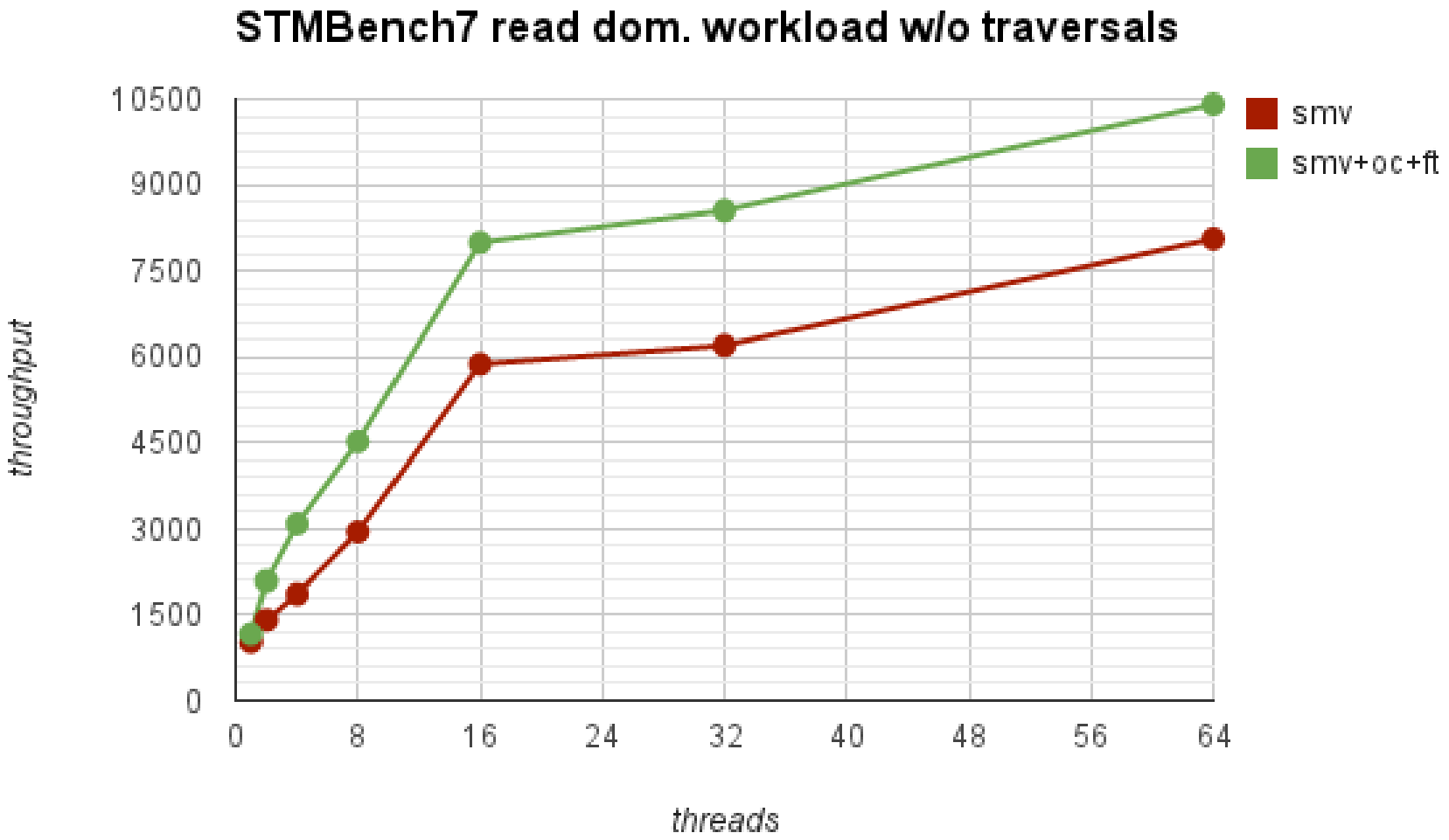


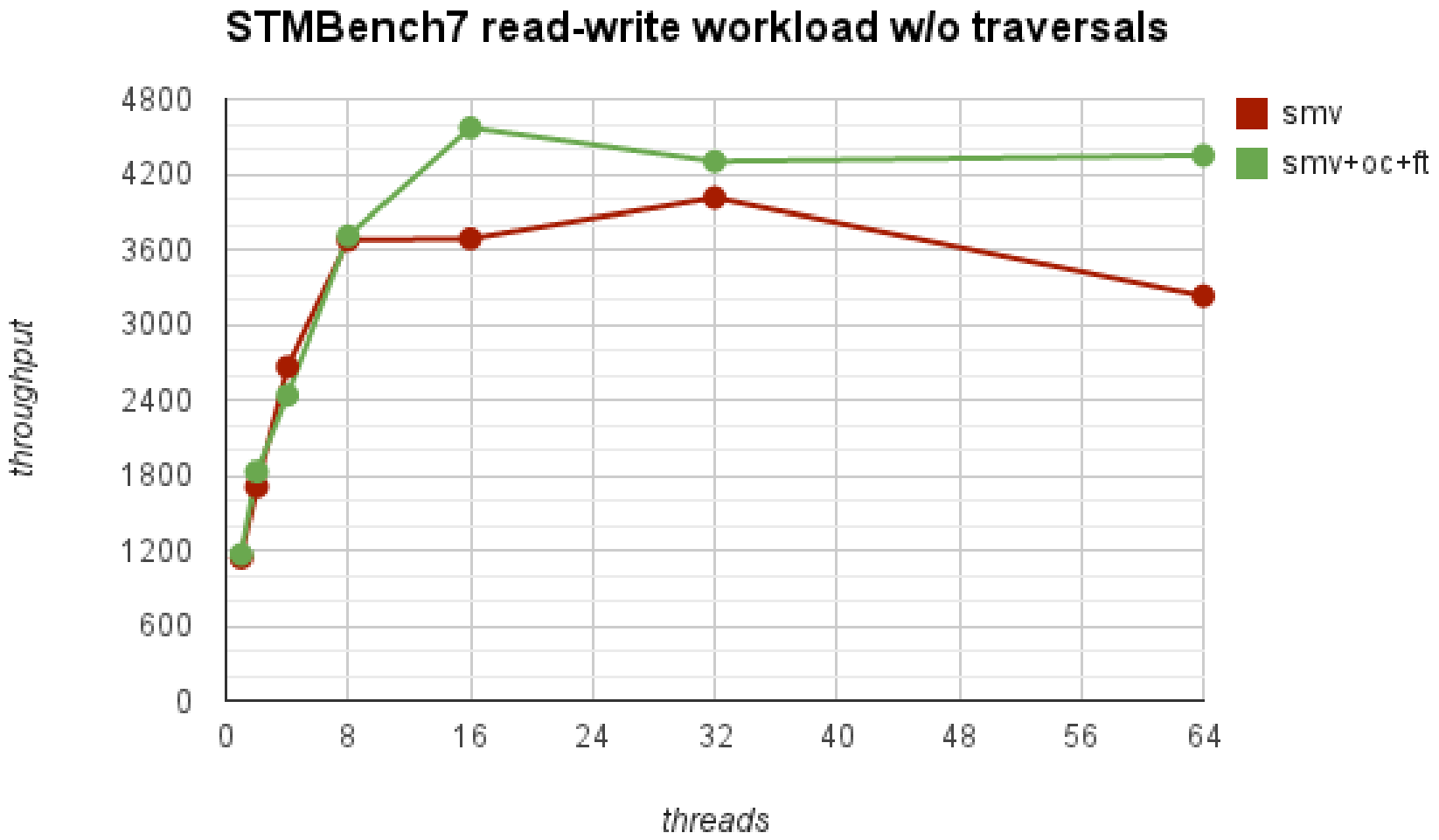
# Preliminary results: DMT



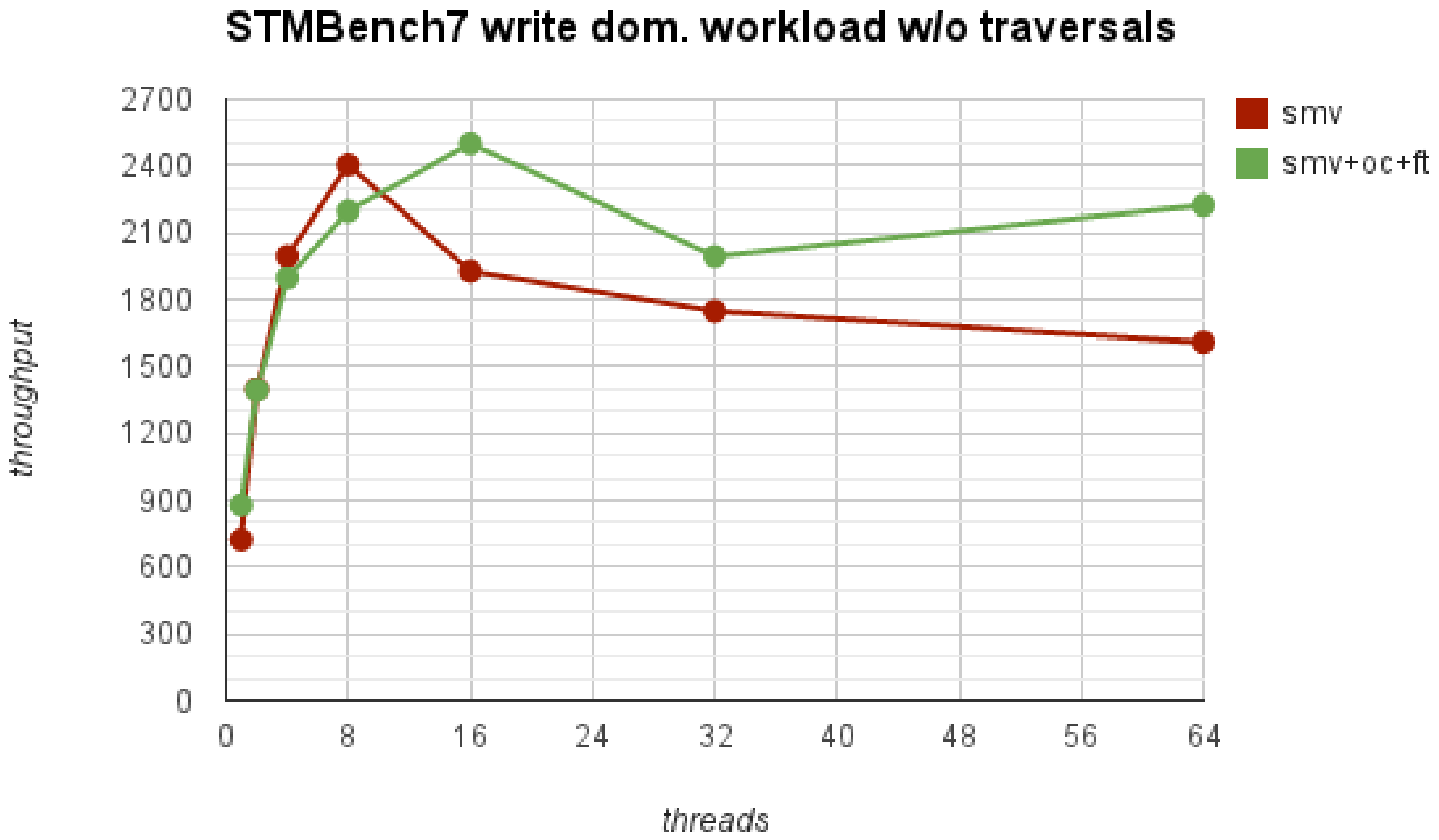
# Preliminary results: Starvation



# Preliminary results: Starvation



# Preliminary results: Starvation



# Conclusion

- Separate serialization from concurrency control
- Allows concurrent SMR, DMT, and SF transactions
- Transaction size is a key factor: unavoidable for SMR and DMT, but not SF transactions
- Require more information about transactions to relax ordered commit constraint

Thank you for your time and attention.

Criticism, comments and suggestions are welcome!