Boosting STM replication via speculation

Paolo Romano, R. Palmieri, F. Quaglia, L. Rodrigues

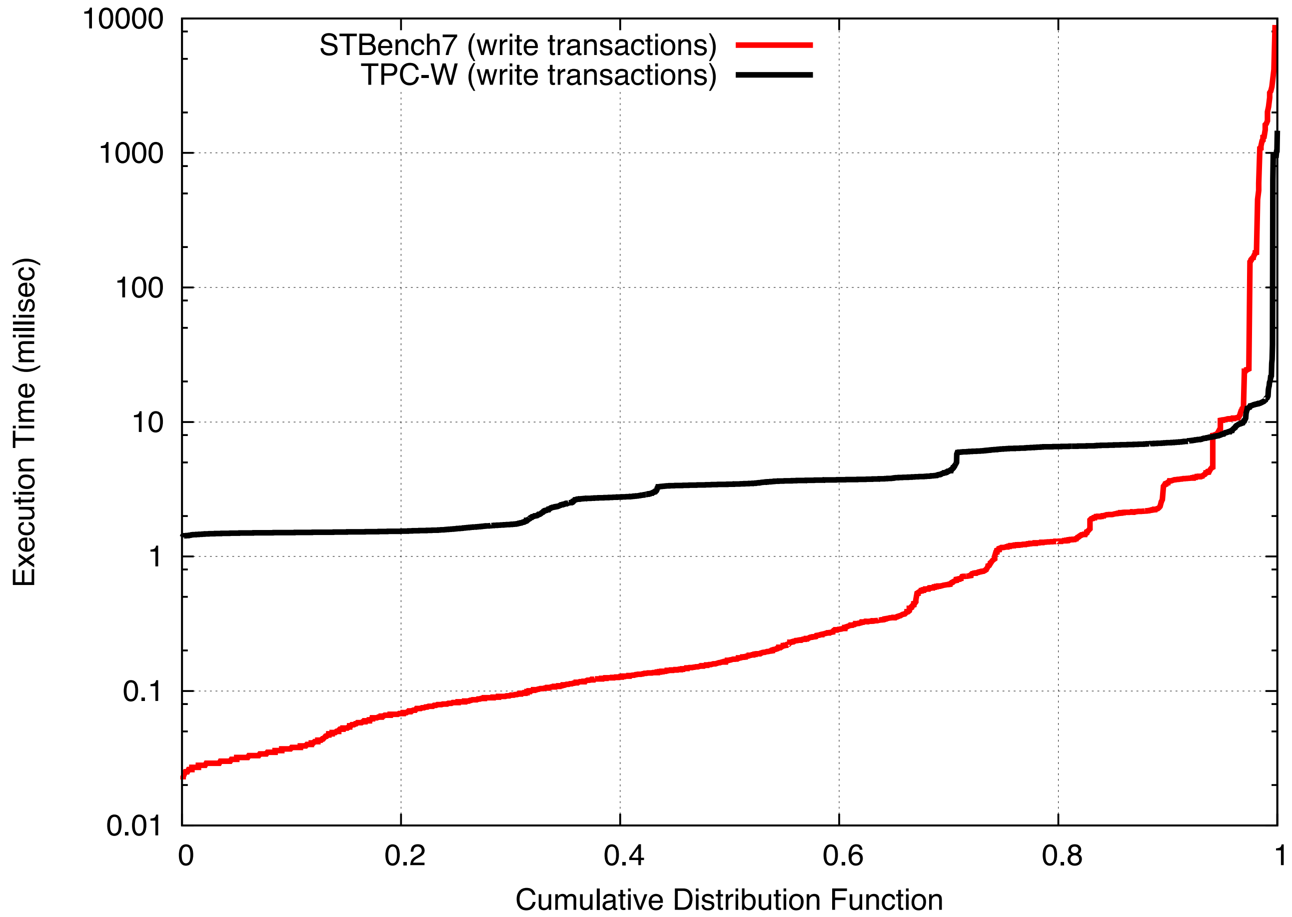
Replication and STMs

- STMs are being employed in new scenarios:
  - database caches in three-tier web apps (FenixEDU)
  - HPC programming languages (X10)
  - in-memory cloud data grids (Coherence, Infinispan)

- …and faced with new challenges:
  - scalability
  - fault-tolerance

← **REPLICATION**

Critical issue for STM replication



- >70% of transactions are 10-100 times in STMs than in DBMSs:
  ⇒ amplification of replica sync. cost when using DB replication schemes!

Active replication of transactional systems

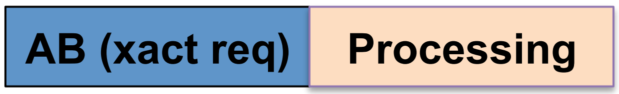
1. Agreement on request execution order
   – encapsulated by Total Order broadcast (TOB):
     + deadlock freedom
     - TOB is an expensive communication primitive

2. Deterministic request processing
   – concurrency is one of the possible sources of non-determinism:
     • concurrent execution of transactions needs to be equivalent to sequential exec. according to TOB order

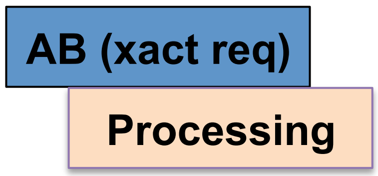
How to minimize replica synchronization costs?

- Overlapping communication and processing:
  - ***optimistic deliveries***: replicas receive messages long before their total order is established:
    - in LANs optimistic and final delivery order normally match
  - *speculatively* process transactions as soon as they are optimistically delivered

Conventional (Active) Replication Scheme



Speculative (Active) Replication Scheme

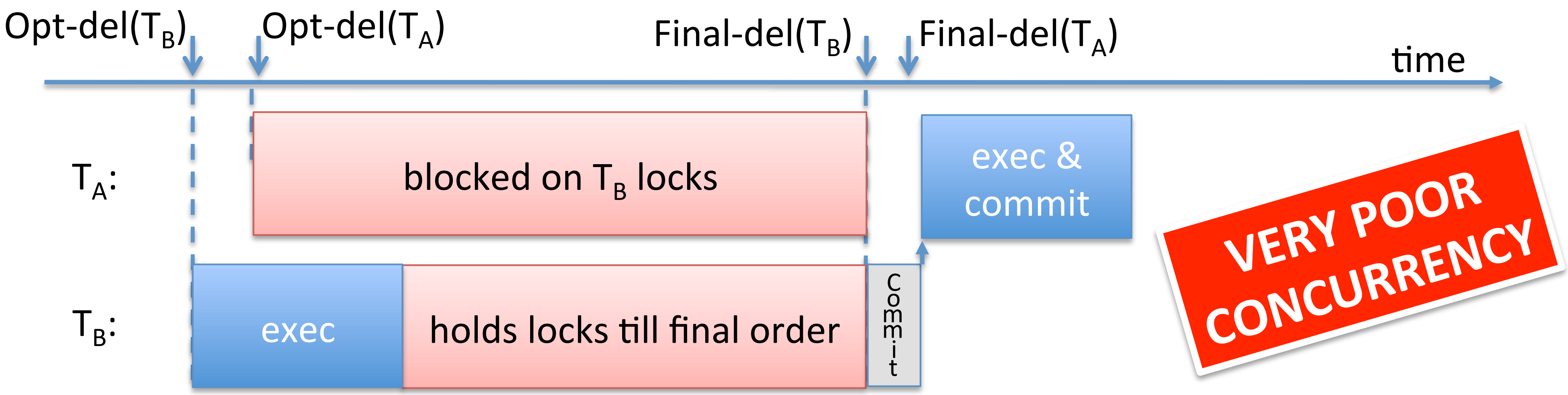


Easier said than done….

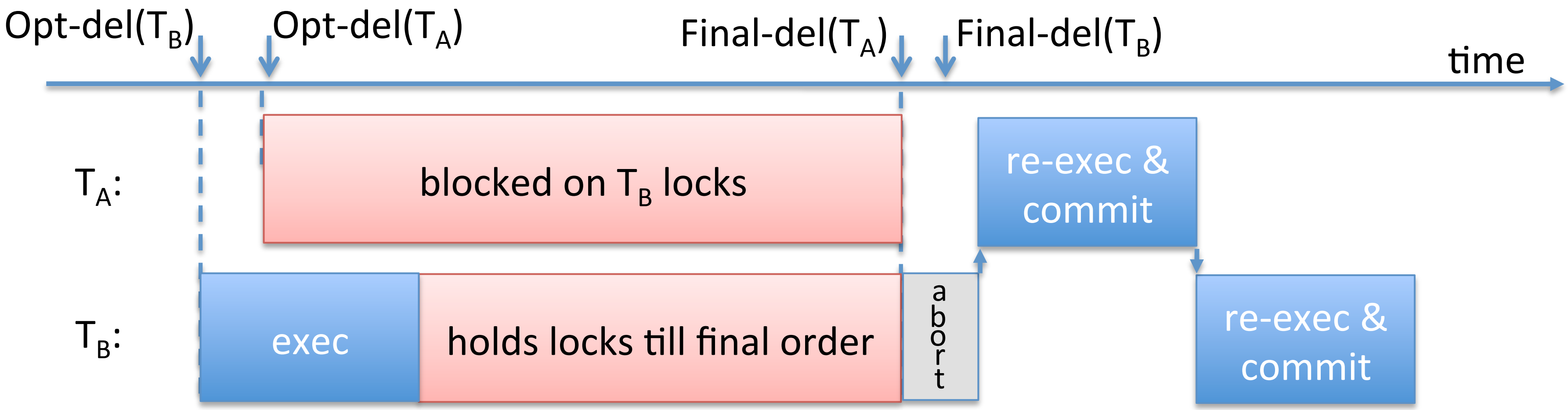
Problem 1: Deterministic transaction scheduling

Existing deterministic schedulers have significant limitations:

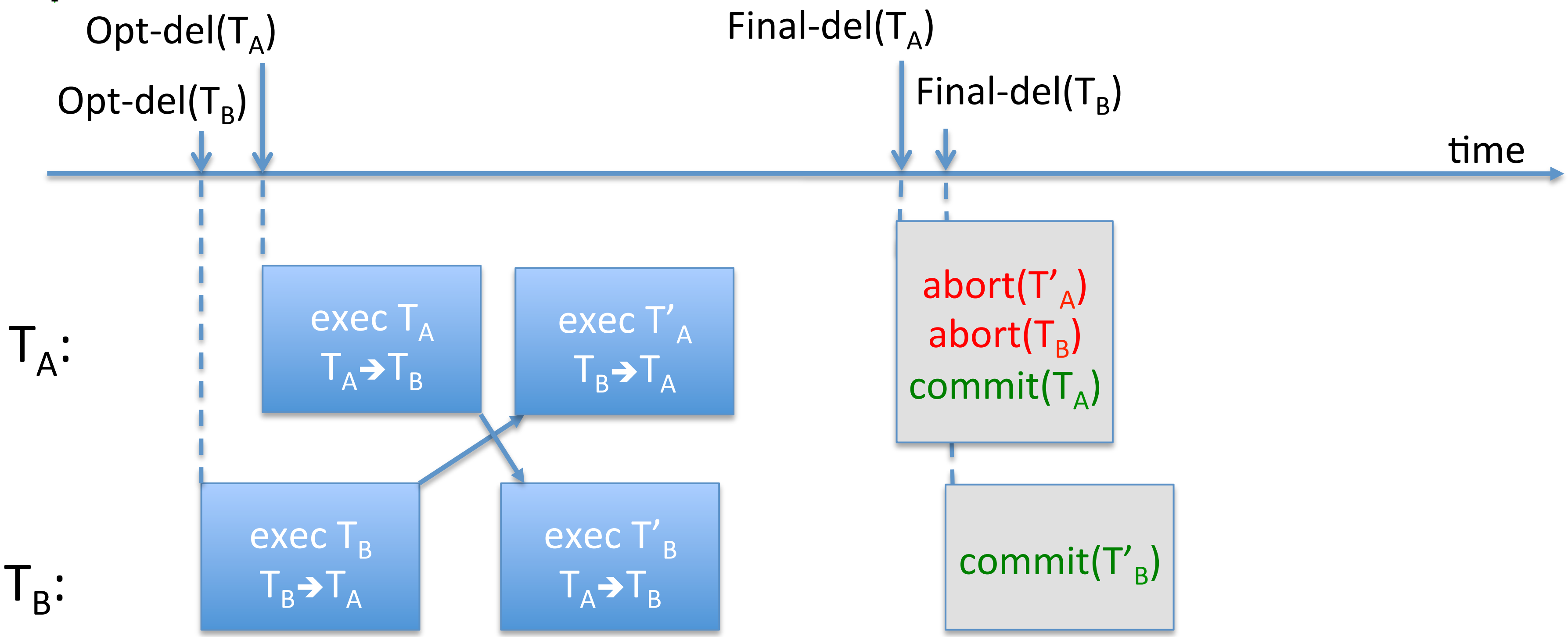
- a-priori knowledge of readsets/writesets:
  - may lead to large conflict over-estimation
- acquire **ALL** locks as xact begins, release when it's final-del.
  - way more pessimistic than classic 2PL



Problem 2: mismatches between optimistic and final delivery orders



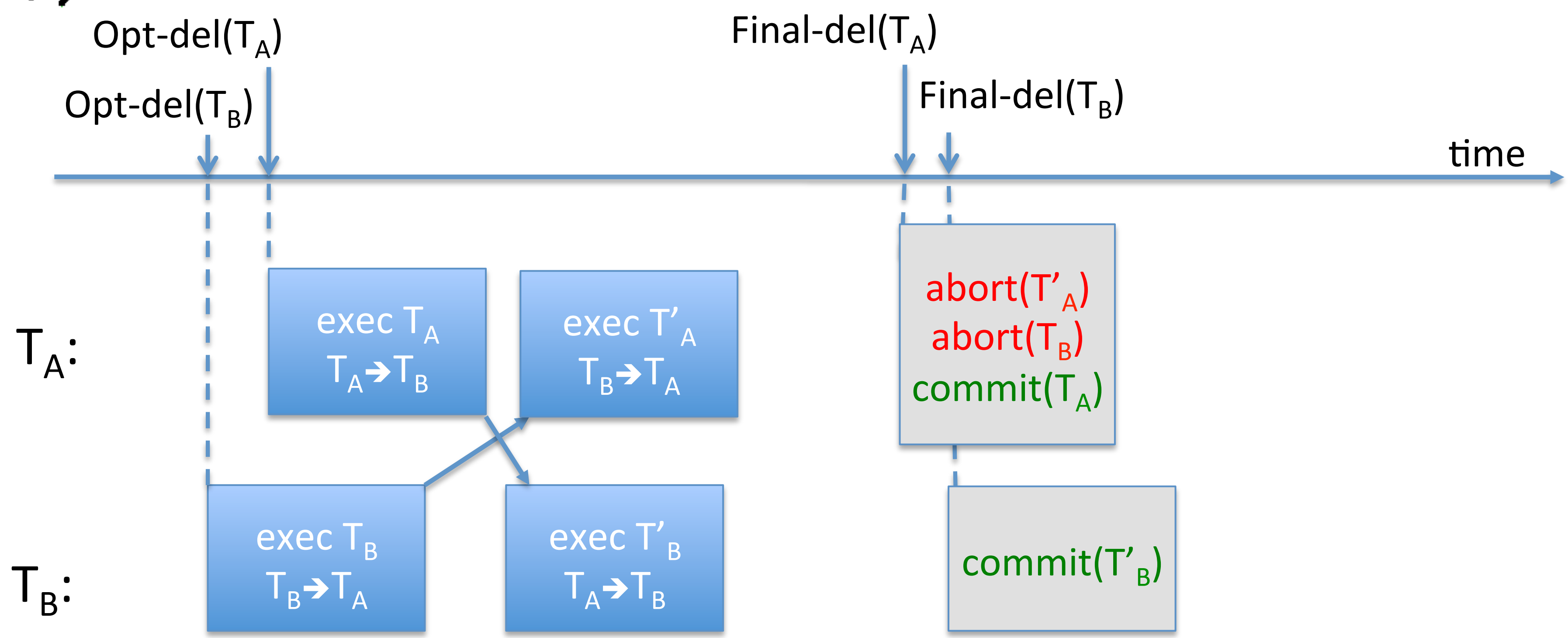
Don’t be pessimistic…be speculative!



Speculatively explore multiple Serialization Orders (SO)

+ **take maximum benefit from modern multi-core architectures**
+ **shelter from network reordering**
+ **avoid lock convoying**

Don't be pessimistic…be speculative!



Speculatively explore multiple Serialization Orders (SO)

– **#SOs can grow factorially with #msgs not yet finally delivered**
  - true in worst case: every xact conflicts with every other, hardly the case in practice

\+ **#SOs in which a xact observes distinct snapshots depends on actual conflict graph**

THE SPECULATIVE TRANSACTIONAL REPLICATION (STR) PROBLEM [SPAA10]

STR Problem: **Model**

- Each replica holds a full copy of the STM
- Application generates a transactional request (or simply transaction), T_i, and propagates it via TOB
- TOB delivers two events:
  - opt-delivery(T_i): early guess of final order
  - final-delivery(T_i): agreed total order
- For each T_i, one or more speculative transactions, denoted as T_i^j, are locally executed by each replica

STR Problem: **Specification Overview**

In addition to classic 1-copy serializability, the STR problem is specified by the following three properties:

1. **consistency**
2. **non-redundancy**
3. **completeness**

STR problem - **Consistency**

Consistency: *opacity*.

Prevents transactions from observing inconsistent (non-serializable) snapshots

- important for safety in non-sandboxed environments
- avoids wasting computational resources in "useless" transactions:
  - not associated with any possible final AB-delivery order

STR problem - **Non-redundancy**

Non-redundancy: *no two speculative instances of the same transaction observe the same snapshot*.

Filters out trivial solutions that blindly enumerate all permutations of Opt-delivered transactions:

- force solutions to reason on conflict relationships among transactions before exploring new serialization orders

STR problem - **Completeness**

Completeness: *Let Σ be the set of Opt-delivered, but not yet TO-delivered, transactions.*
If the system stops Opt- and TO-delivering messages, eventually every permutation of Σ that produces a ***distinct snapshot*** *is explored.*

- Shelters from any possible mismatch between optimistic and final delivery orders

An STR Protocol [ISPA2010]
Key idea
Speculative Polygraph (SP)

- inspired by Papadimitriou's Polygraph [JACM79]:
  - originally used to identify view-serializable schedules
  - polygraphs embed a family of digraphs, each associated with a different equivalent serial history

- SPs support:
  - on-line identification of *all and only* non-equivalent serialization orders of a speculative transaction
  - tolerate the coexistence of speculative transactions in the same execution history

WARNING
This is just a teaser,
details in the paper!

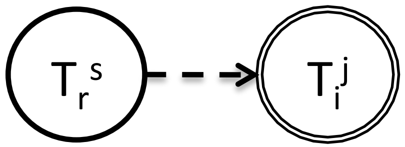
Speculative Polygraphs

SP(T_i^j)=(N,A,B) where:

N: set of vertexes, associated with (speculative) transactions

A: set of ***merging edges*** ***(T_r^s ⊙➔ T_i^j)*** which merges SP(T_r^s) and SP(T_i^j)

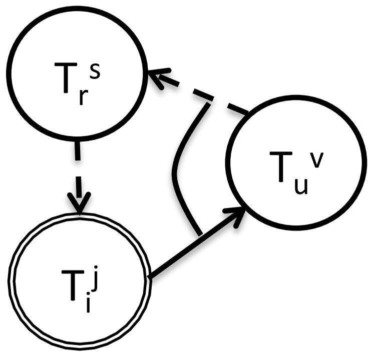
T_r^s: w(x_r^s)
T_j^j: r(x_r^s)



read-from relationships

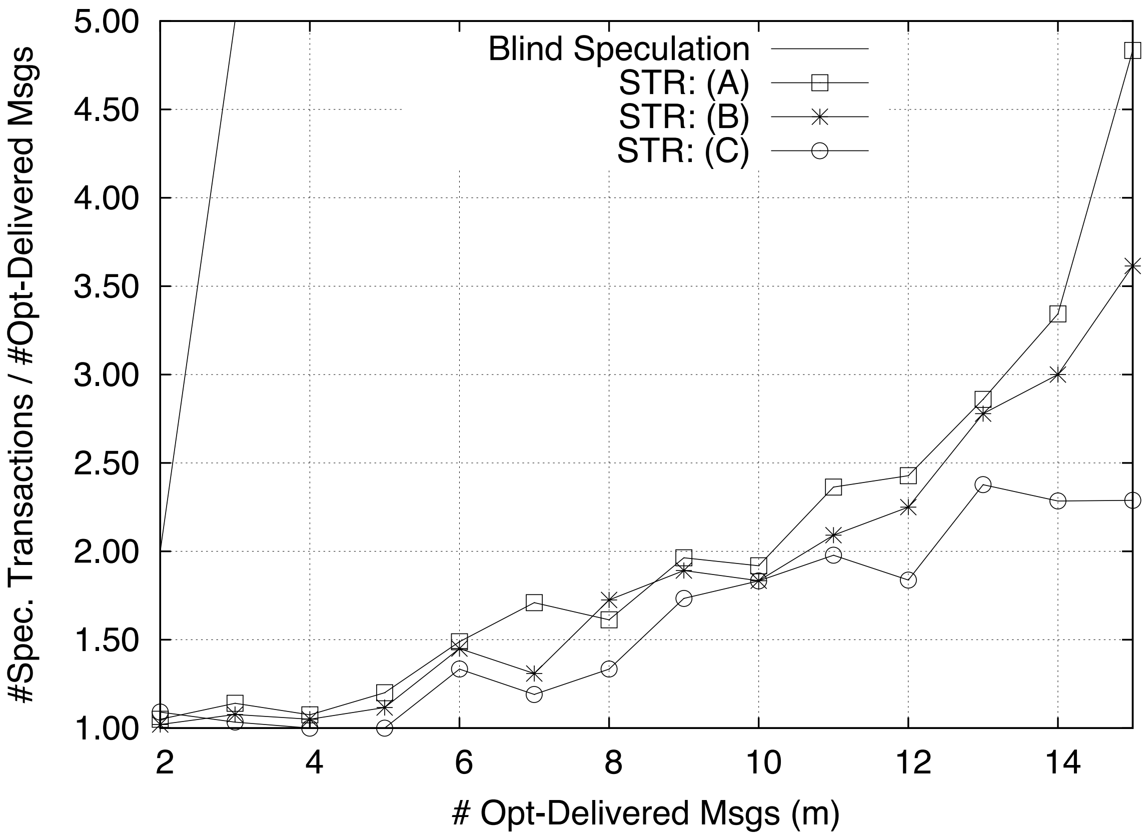
B: set of ***asymmetric bipaths*** denoted as <***(T_u^v ⊙➔ T_i^j), (T_i^j ➔ T_u^v)***>

T_r^s: w(x_r^s)
T_j^j: r(x_r^s)
T_u^v: w(x_u^v)



not read-from relationships

The importance of being … non-redundant



Simulation study based on real (STM) workloads:

Optimal STR scheme: #SOs≈[2.5-5] with 15 opt-delivered xacts

Blind enumeration: #SOs≈1,000,000 with 10 opt-delivered xacts

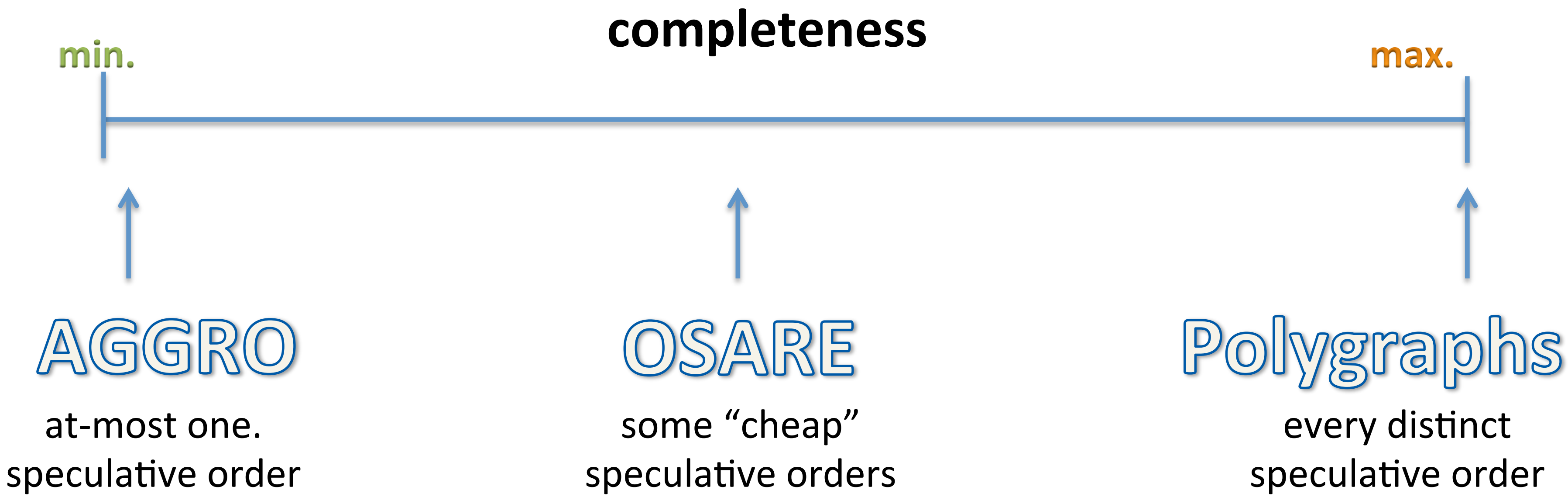
To be or not be… complete?

- Completeness can have a considerable price:
  1. querying Speculative Polygraphs has an exponential cost in the number of bi-paths
  2. the number of serialization orders in which a transaction T needs to be re-executed grows factorially with the number of transaction T conflicts with

What about relaxing completeness?

Relaxing completeness

- The relevance of the completeness property depends on the likelihood of mismatches between final and optimistic delivery orders
- This led us to design two additional protocols:



AGGRO [NCA10] – Main Idea

BASE ASSUMPTION: "*optimistic and final delivery order coincide with high probability*"

- Transactions are speculatively started immediately after their optimistic delivery…
- and ***try*** to execute in a serialization order compliant with the optimistic delivery order:
  - speculative snapshot are aggressively propagated along chains of speculative conflicting transactions

AGGRO – Key Problem

- To maximize parallelism, transactions are activated without waiting for previously opt-delivered ones to be committed

- Two consequences:
  - writes of prev. xacts may be observed too soon:
    - non-opaque schedules
  - writes of prev. xacts may be missed:
    - different serialization order **(snapshot miss event)**

AGGRO – algorithm in a nutshell (i)

- If T_i writes X:
  - mark X as Work in Progress (WIP) by T_i
  - kill all T_j that:
    1. already read X, and
    2. follow T_i in opt-delivery order

    ⇒ ensure T_j is aligned with opt-delivery order

snapshot miss

- If T_i reads X:
  - if X is marked as WIP by a xact that precede T_i in the opt-delivery order: wait till T_i unmarks X as WIP
  - return the version created by the most recent xact preceding T_i

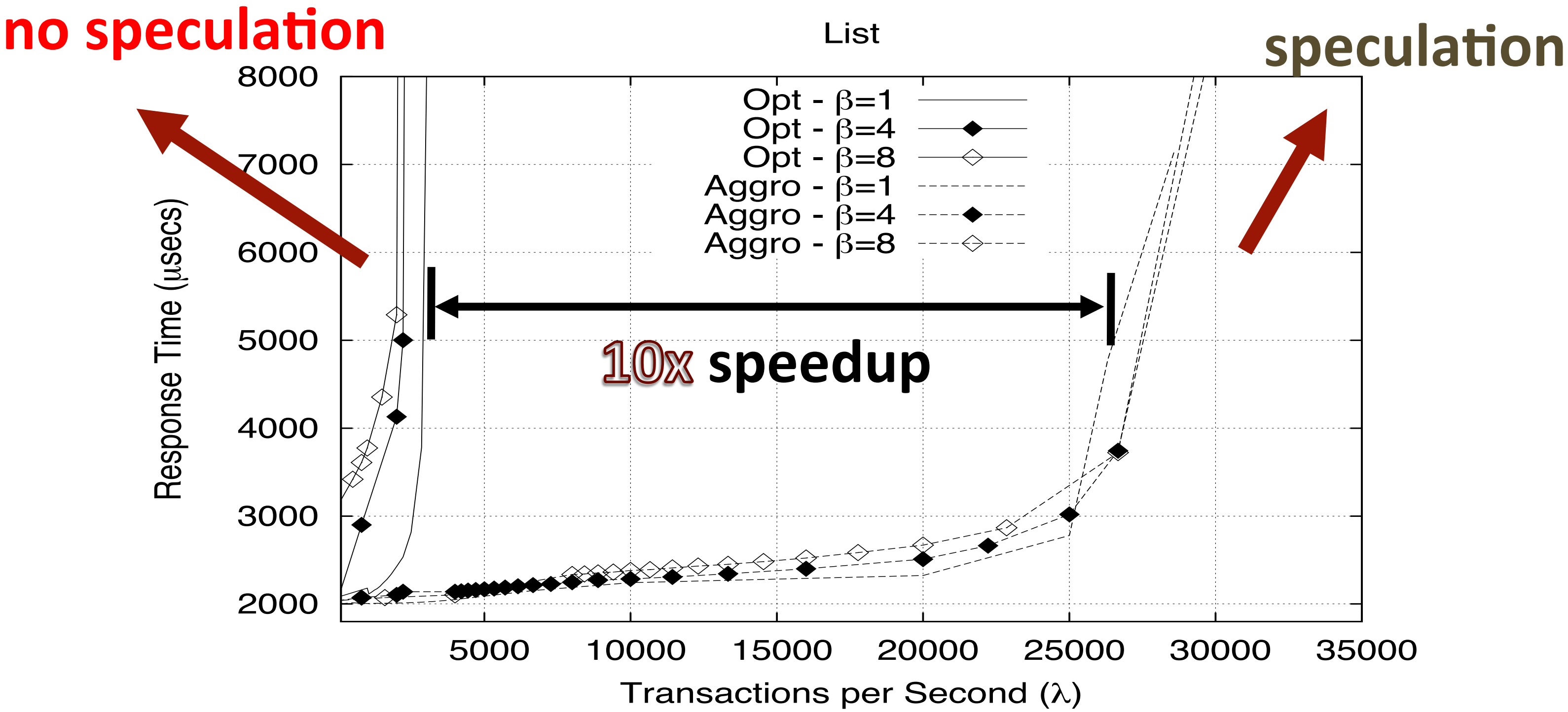
AGGRO – algorithm in a nutshell (ii)

- Upon completion of transact. execution:
  - release locks on writeset:
    - this makes write-set readable only when it's stable
    - avoiding leakage of intermediate snapshots

- Upon final delivery of T_i
  - wait until all xacts preceding T_i in final order commit
  - validate(T_i) and accordingly commit/abort it

AGGRO - What speedups?

- no mismatch between optimistic and final delivery
- baseline uses opt-deliveries but does not propagate snapshots



OSARE [SRDS11] – Main Idea

- As in AGGRO, attempt to serialize xacts according to opt-delivery order.

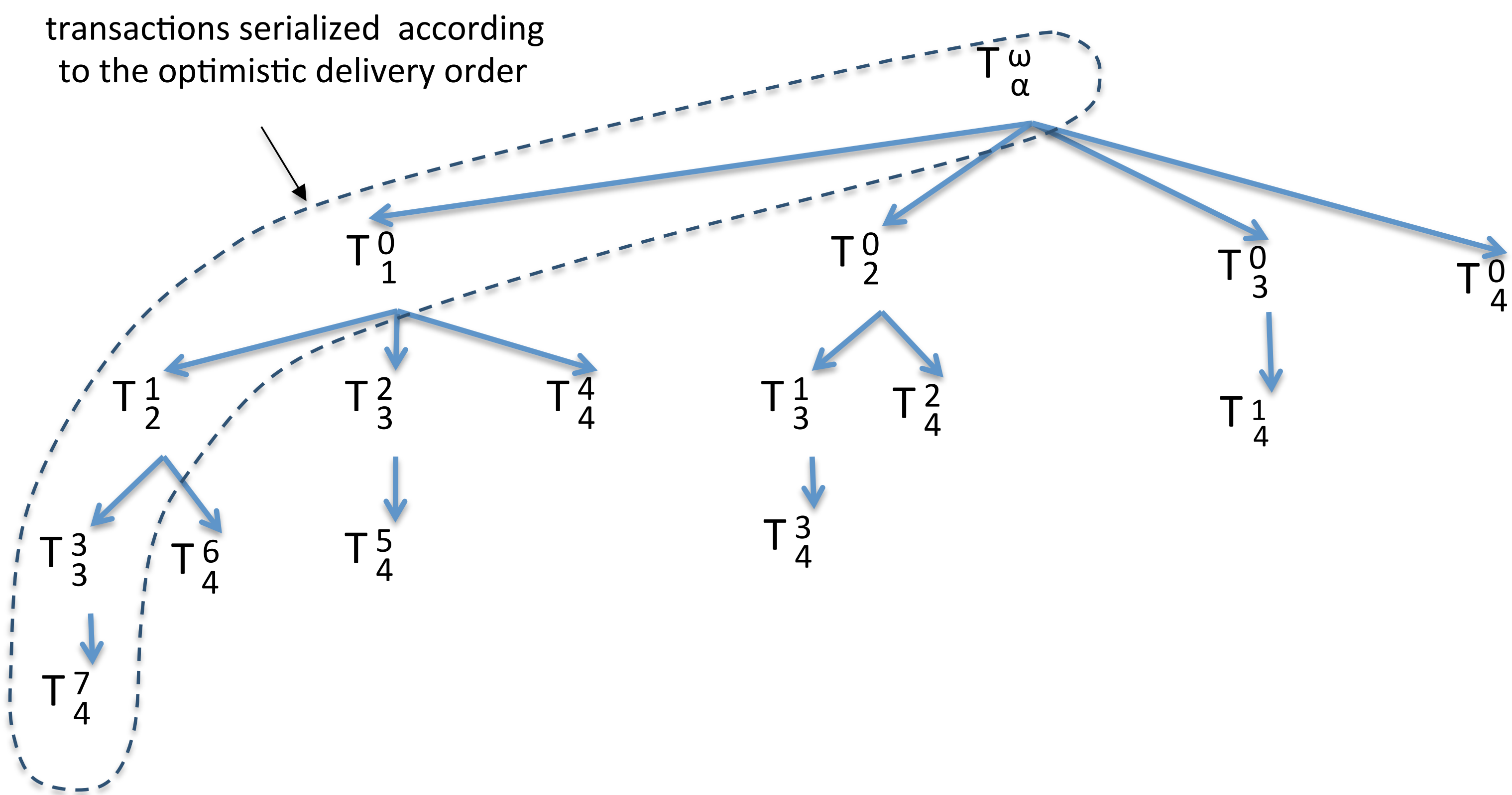
- Unlike AGGRO, if a xact T undergoes a snapshot miss, don't abort it, but
  - explore new speculative serialization orders in an ***opportunistic*** fashion:
    - avoiding expensive polygraphs' manipulations
  - activate a new instance of T realigned with optimistic delivery order

OSARE(*): Opportunistic Speculation in Active Replication

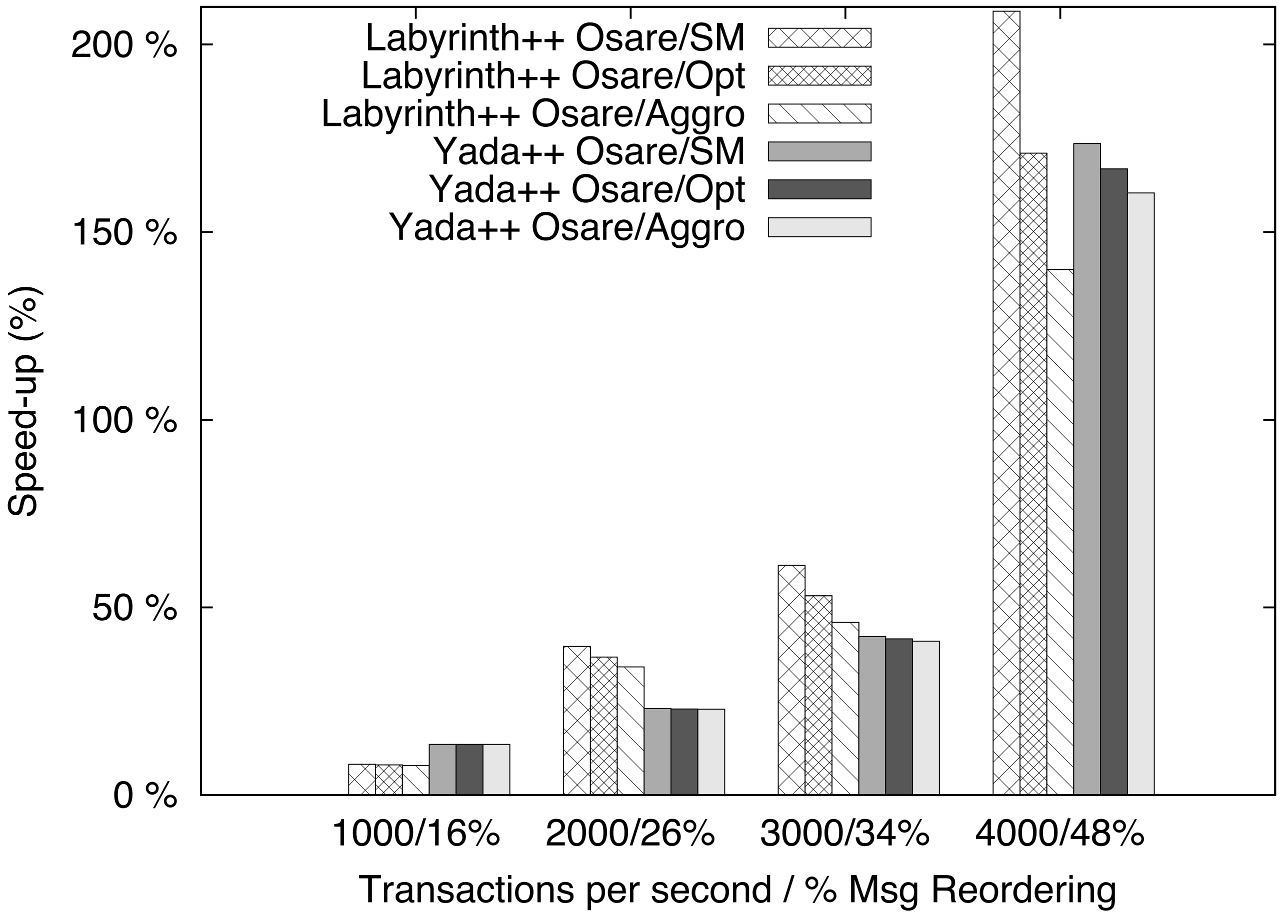
(*) OSARE means "to dare" in Italian

OSARE – how complete?

- Largest set of explored speculative serialization orders: $O(2^n)$
  - Full permutation tree is $O(n!)$



OSARE – what speedups?



Conclusions

- Bad news:
  - Replication overhead are strongly amplified in STMs
- Good news:
  - Active replication costs can be strongly reduced by speculatively overlapping processing and communication
- We formalized the Speculative Transactional Replication (STR) problem…
- …and proposed three protocols exploring different trade-offs for what concerns completeness

Open questions

- Are there other interesting **trade-offs** for what concerns **completeness**?
- How to apply speculative techniques to replication protocols **other than active replication**?
  - deferred update replication technique (a.k.a. certification or DBSM [DPD03]):
    - in [Systor11] we proposed a non-complete (à-la AGGRO) speculative protocol
    - what about different degrees of completeness?
  - lease based certification protocols [Middleware10]?
  - partial replication protocols [SRDS10,PRDC11]?
  - your favorite replication protocol!

Thanks for the attention

References

[DPD03] Fernando Pedone, Rachid Guerraoui, and Andr\&\#233; Schiper. 2003. The Database State Machine Approach. Distrib. Parallel Databases 14, 1 (July 2003), 71-98.

[ISPA10] P. Romano, R. Palmieri, F. Quaglia, N. Carvalho and L. Rodrigues, An Optimal Speculative Transactional Replication Protocol, Proc. 8th IEEE International Symposium on Parallel and Distributed Processing with Applications (ISPA), Taiwan, Taipei, IEEE Computer Society Press, September 2010

[Middleware10] N. Carvalho, P. Romano and L. Rodrigues, Asynchronous Lease-based Replication of Software Transactional Memory, Proceedings of the ACM/IFIP/USENIX 11th Middleware Conference (Middleware), 2010

[PRDC11] P. Ruivo, M. Couceiro, Paolo Romano and L. Rodrigues, Exploiting Total Order Multicast in Weakly Consistent Transactional Caches, Proc. IEEE 17th Pacific Rim International Symposium on Dependable Computing (PRDC'11), Pasadena, California, Dec. 2011

[SPAA10] P. Romano, R. Palmieri, F. Quaglia, N. Carvalho and L. Rodrigues, On Speculative Replication of Transactional Systems (Brief Announcement), Proc. 22nd ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), Santorini, Greece, ACM Press, June 2010

[SRDS10] N. Schiper, P. Sutra and F. Pedone, P-Store: Genuine Partial Replication in Wide Area Networks, 29th International Symposium on Reliable Distributed Systems (SRDS 2010)

[SRDS11] R. Palmieri, F. Quaglia and P. Romano, OSARE: Opportunistic Speculation in Actively REplicated Transactional Systems, The 30th IEEE Symposium on Reliable Distributed Systems (SRDS 2011), Madrid, Spain, Oct. 2011

[Systor11] N. Carvalho, Paolo Romano and L. Rodrigues, SCert: Speculative Certification in Replicated Software Transactional Memories, Proceedings of the 4th Annual International Systems and Storage Conference (SYSTOR 2011)