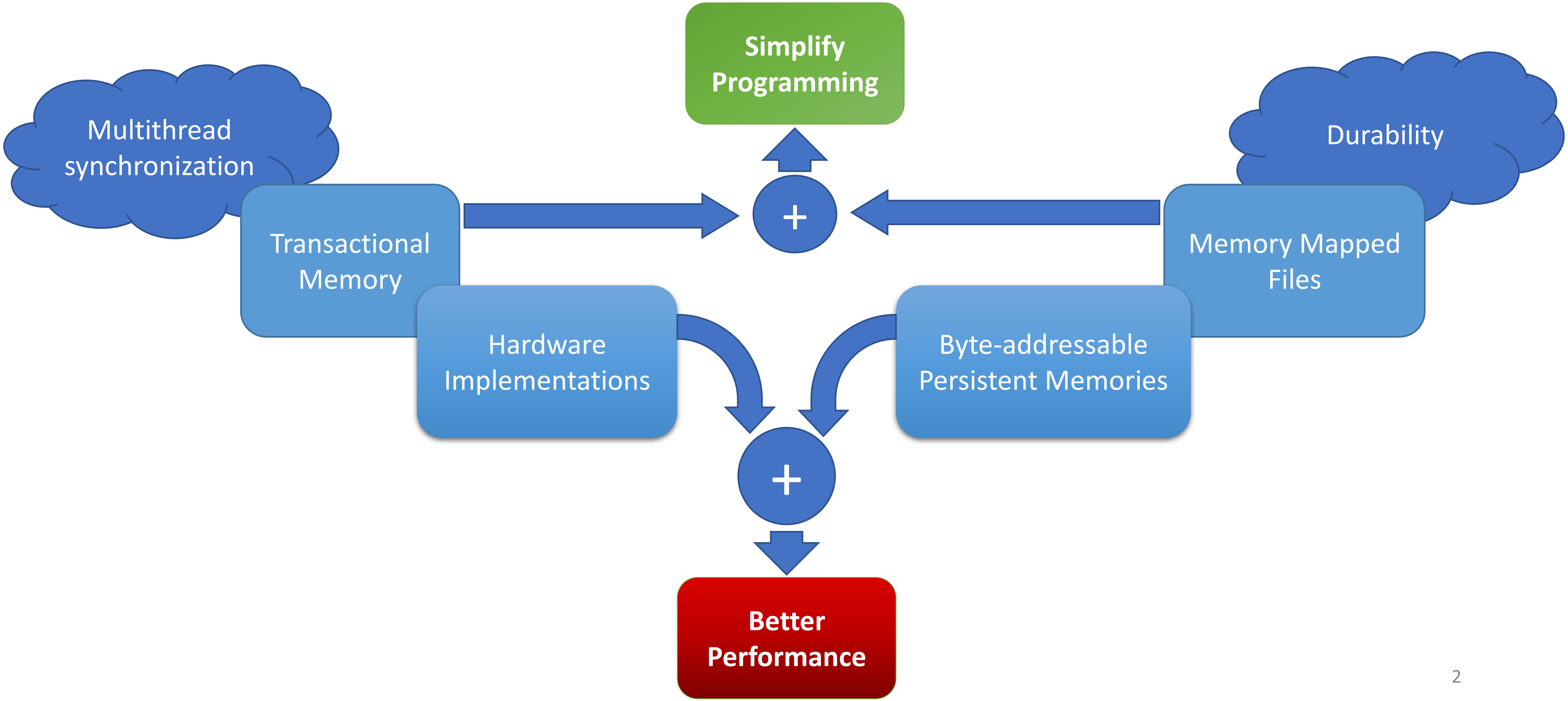
SPHT: Scalable Persistent Hardware Transactions

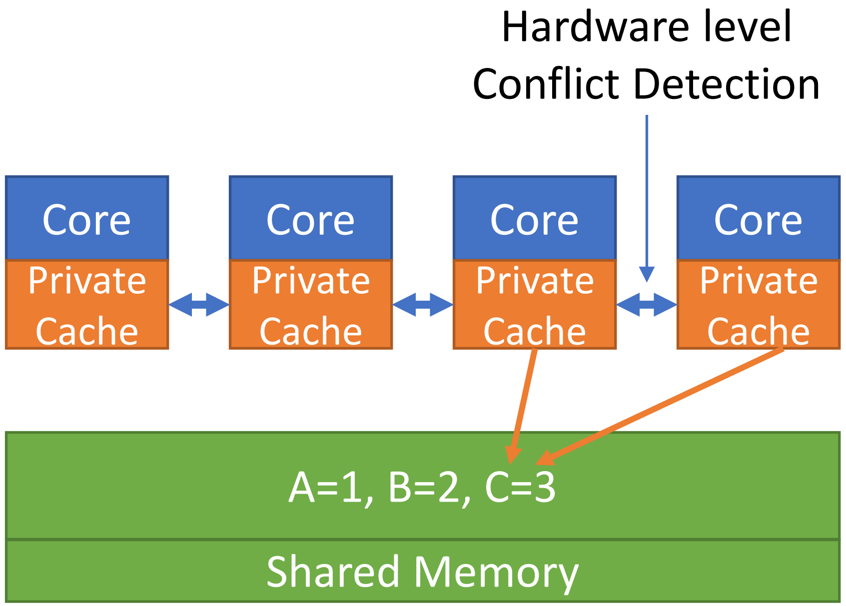
Daniel Castro[1], Alexandro Baldassin[2], Paolo Romano[1], João Barreto[1]

[1]Instituto Superior Técnico & INESC-ID
[2]UNESP - Universidade Estadual Paulista

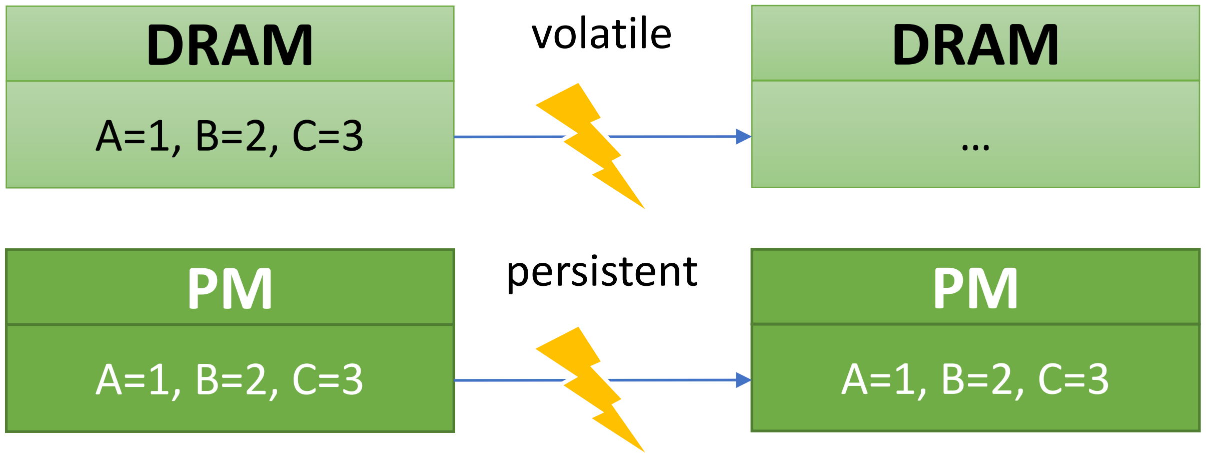
Introduction

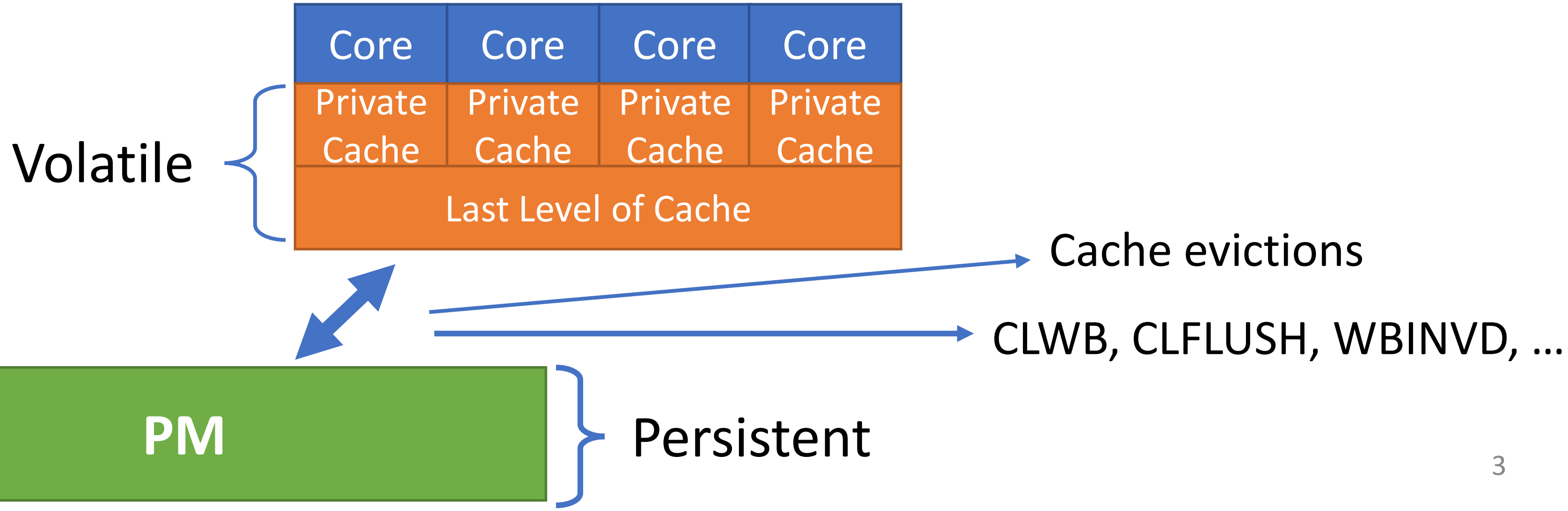


Hardware Transactional Memory



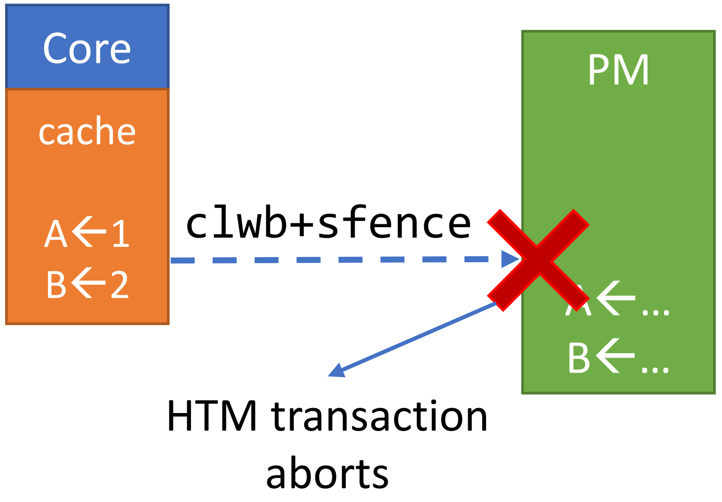
Persistent Memory

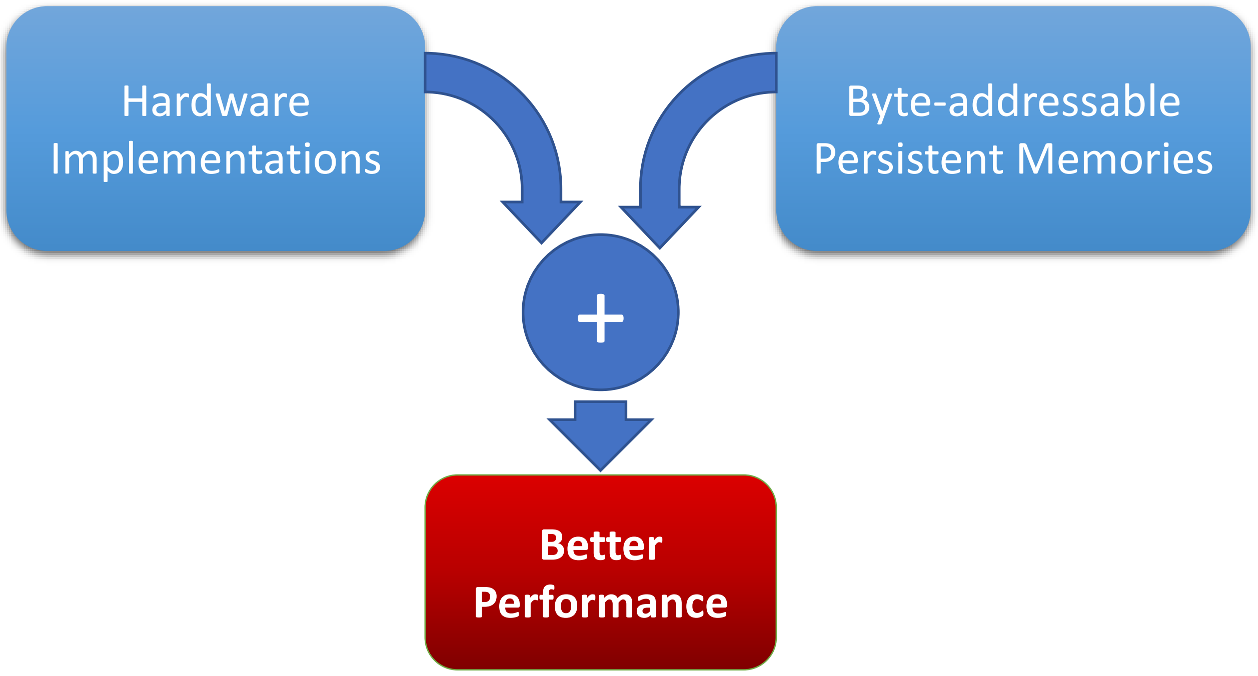




How to combine HTM and PM?

```
htm_begin
  A ← 1
  B ← 2
  persist?
htm_end
```



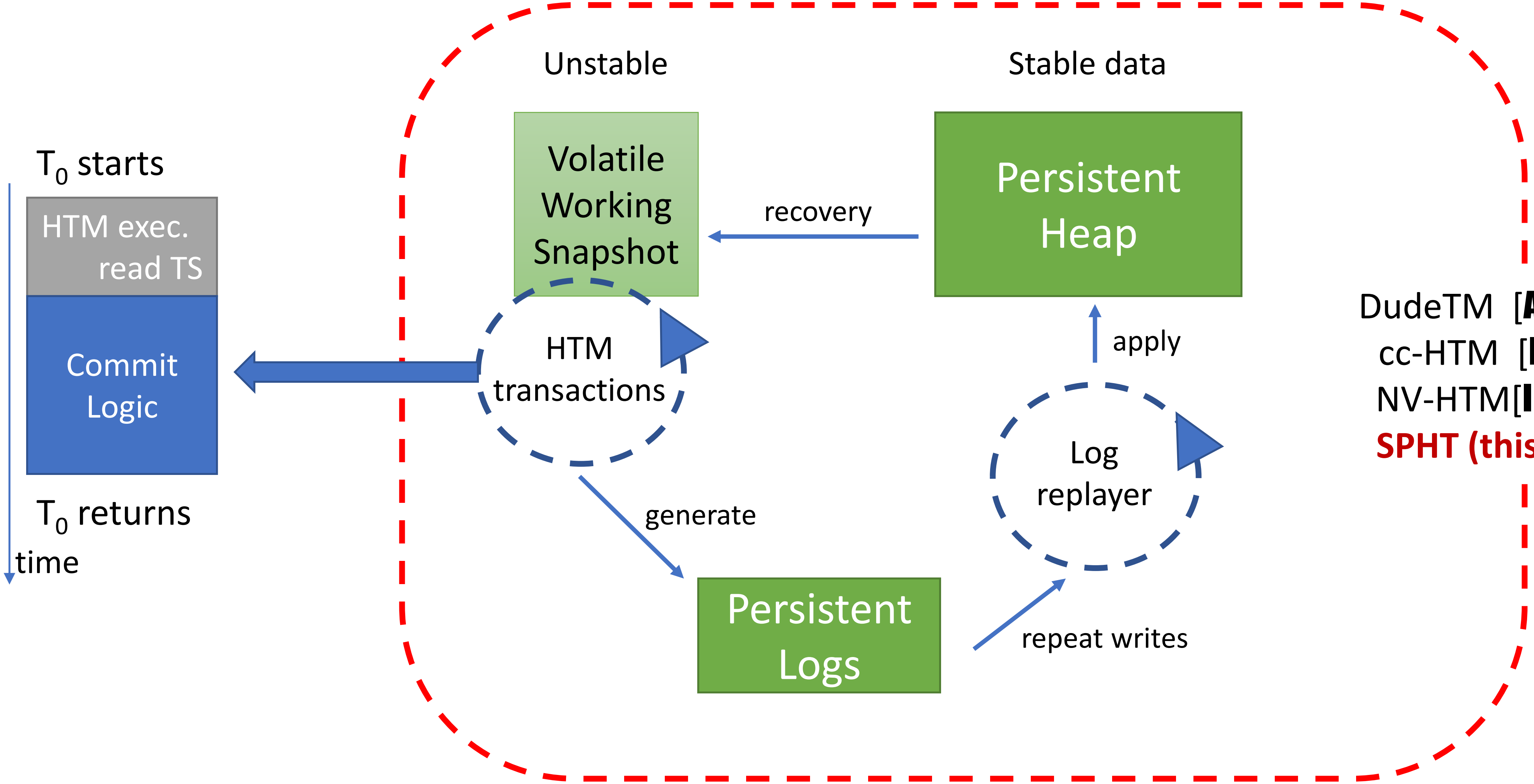


Existing solutions:

I'll focus on this ←

- **Shadow memory** → DudeTM[**ASPLOS'17**], cc-HTM[**ISMM'17**], NV-HTM[**IPDPS'18**]
- **Non-destructive undo log** → Crafty[**PLDI'20**]

Combining HTM and PM



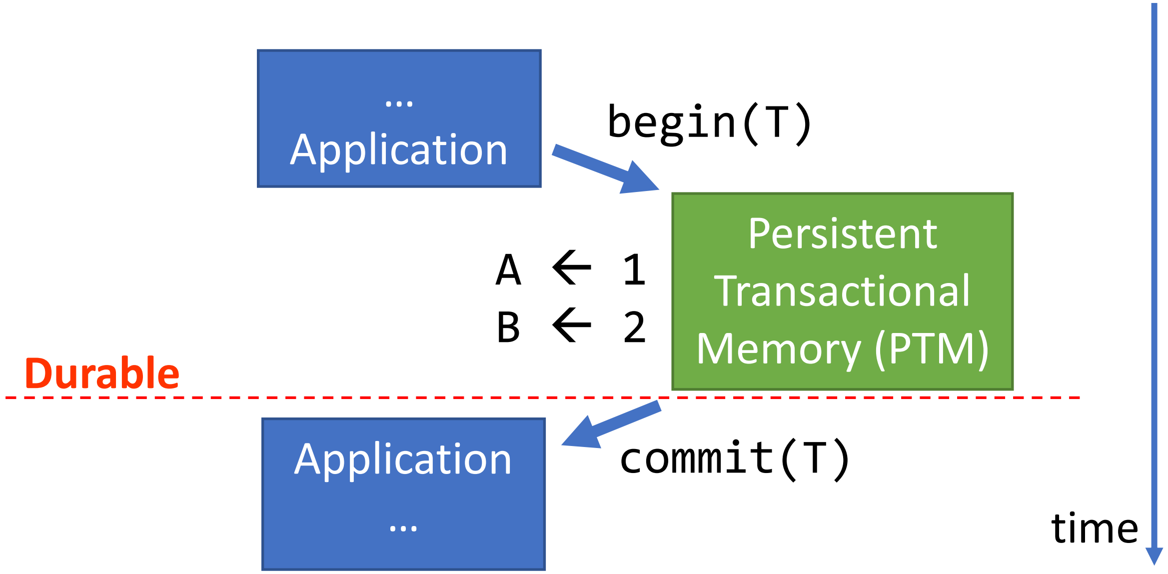
DudeTM [**ASPLOS'17**]
cc-HTM [**ISMM'17**]
NV-HTM[**IPDPS'18**]
SPHT (this paper)

Durability semantics

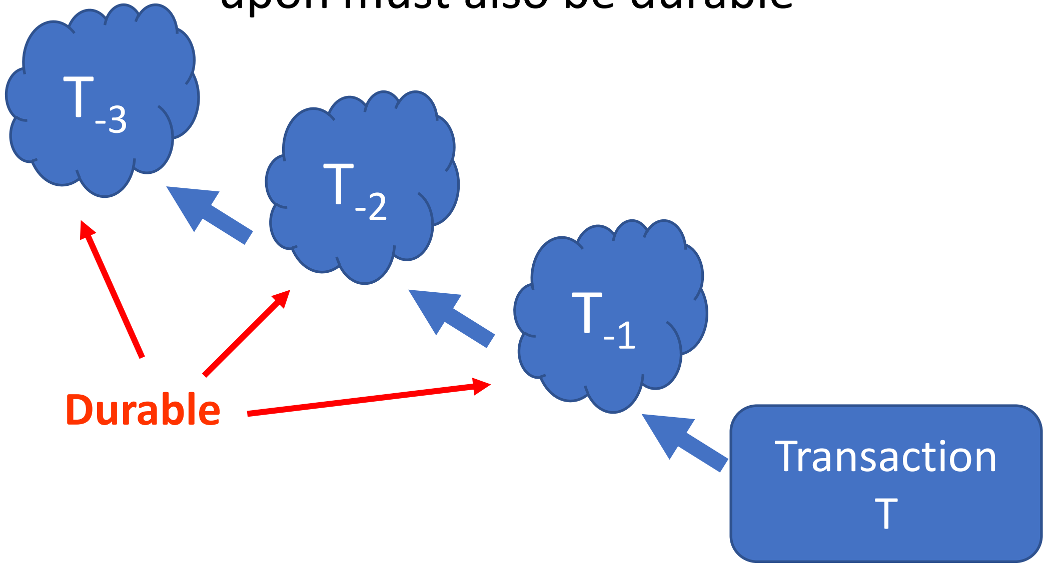
- DBMS guarantee ACID: externalized commits are durable
  - what about PTM systems?

Immediate Durability:

The application is guaranteed that a transaction T is durable after the commit logic completes



All transactions that T may depend upon must also be durable



Scalability limitations of the SoTA

	DudeTM [ASPLOS'17]	Cc-HTM [ISMM'17]	NV-HTM [IPDPS'18]	Crafty [PLDI'20]
Global Clock updated by TXs	Y	N	N	Y
Extended TX vulnerability window	N	N	N	Y
Sequential mechanism to ensure durability	N	Y	Y	Y
Sequential Log Replay	Y	Y	Y	N

Limitations at the level of log replay

Limitations at the level of transaction processing

- **Workaround:** relax durability semantics …

  … not always applicable & more complex

Scalability limitations of the SoTA

	DudeTM [ASPLOS'17]	Cc-HTM [ISMM'17]	NV-HTM [IPDPS'18]	Crafty [PLDI'20]	SPHT
Global Clock updated by TXs	Y	N	N	Y	N
Extended TX vulnerability window	N	N	N	Y	N
Sequential mechanism to ensure durability	N	Y	Y	Y	N
Sequential Log Replay	Y	Y	Y	N	N

Limitations at the level of log replay

Limitations at the level of transaction processing

- **Workaround:** relax durability semantics …

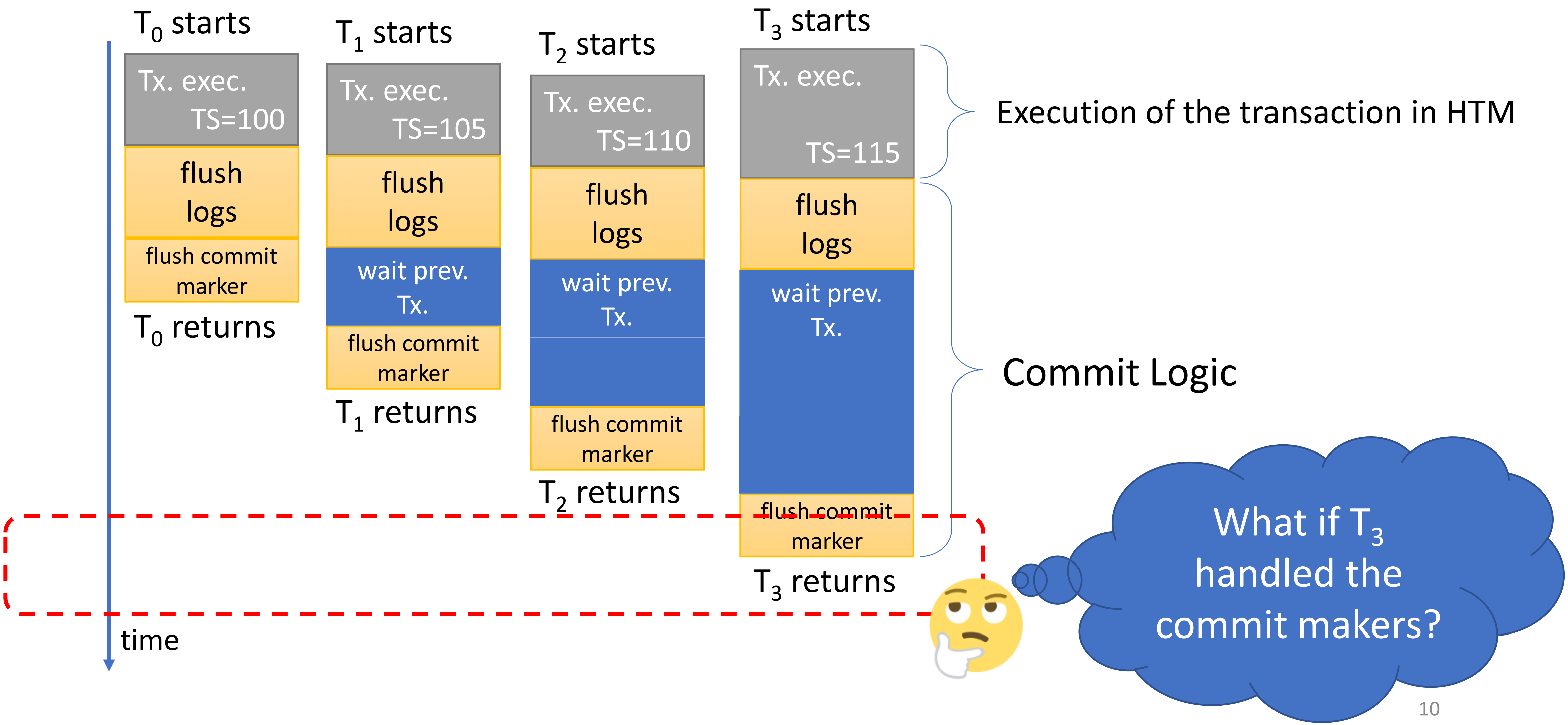
  … not always applicable & more complex

SPHT: contributions

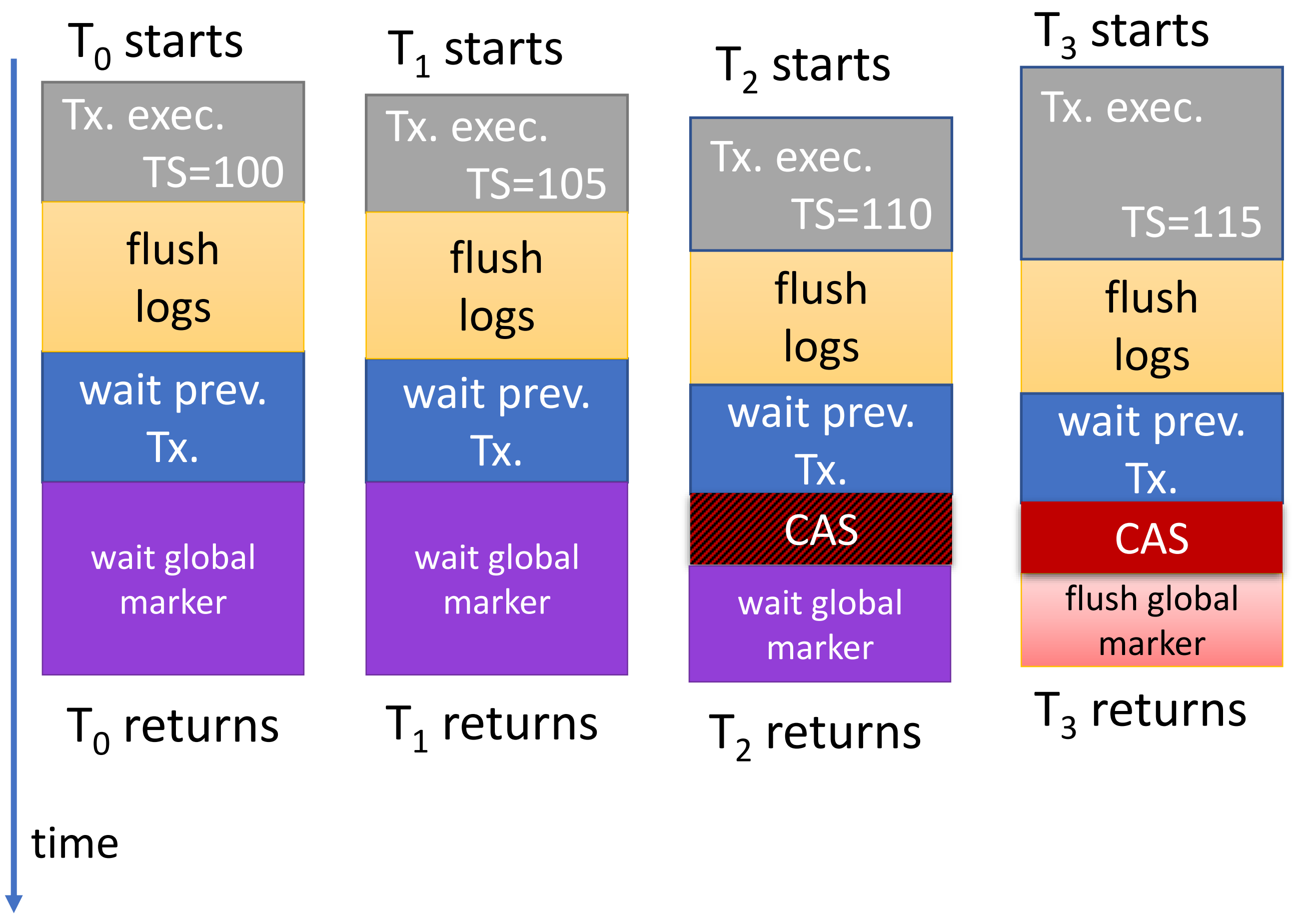
- Evaluate performance of various PTMs in **commodity hardware**:
  - DudeTM, cc-HTM, NV-HTM, Crafty, PSTM (Mnemosyne), SPHT
  - Previous solution evaluated in emulated PM
- Can Immediate Durability scale with commodity HTM+PM?
  - Yes! Using our novel SPHT design.
- Novel **commit logic**
- Novel **log replay techniques**

	DudeTM	Cc-HTM	NV-HTM	Crafty	SPHT
Global Clock updated by TXs	Y	N	N	Y	N
Extended TX vulnerability window	N	N	N	Y	N
Sequential mechanism to ensure durability	N	Y	Y	Y	N
Sequential Log Replay	Y	Y	Y	N	N

Immediate Durability: NV-HTM



Immediate Durability: SPHT



SPHT: log replay

- Background process replays logs in PM:
  - Prunes logs during normal execution, recovers application state on bootstrap
- Performance is critical!
  - System availability;
  - Frees log space.
- SPHT log replayer overview:
  - Linked log: avoid searching for the next transaction in the log;
  - Parallel log replay (via sharding of the persistent heap);
  - NUMA-aware;
  - Exploit WBINVD to avoid tracking which addresses to flush.

Details and optimizations in the **paper**

SPHT: linked log

- We propose SPHT and two variants using linking
- Without hints in the per-thread logs:
  - Log Replayer has to scan logs to order transaction
- **Linked log**: avoid search the next transaction in the log:
  - **Key idea**: log contains hints of where the next transaction is;
  - Pays **a relatively low cost** in transaction processing:
    - Exploit the commit logic to **pinpoint the predecessor** OR **successor**

Check pseudo-code
in the **paper**

Evaluation

- Dual-socket Intel Xeon Gold 5218 CPU (16C/32T – HTM enabled)
- 128GB DRAM / 512GB PM (4 DIMMs)
- Code available[1]:
  - SPHT-NL, SPHT-FL, SPHT-BL, NV-HTM, DudeTM, Crafty, cc-HTM and PSTM;
  - All implementations guarantee immediate durability.
- Tested in:
  - **STAMP** benchmark suit (check all benchmarks **in the paper**)
  - **TPC-C: in the paper**

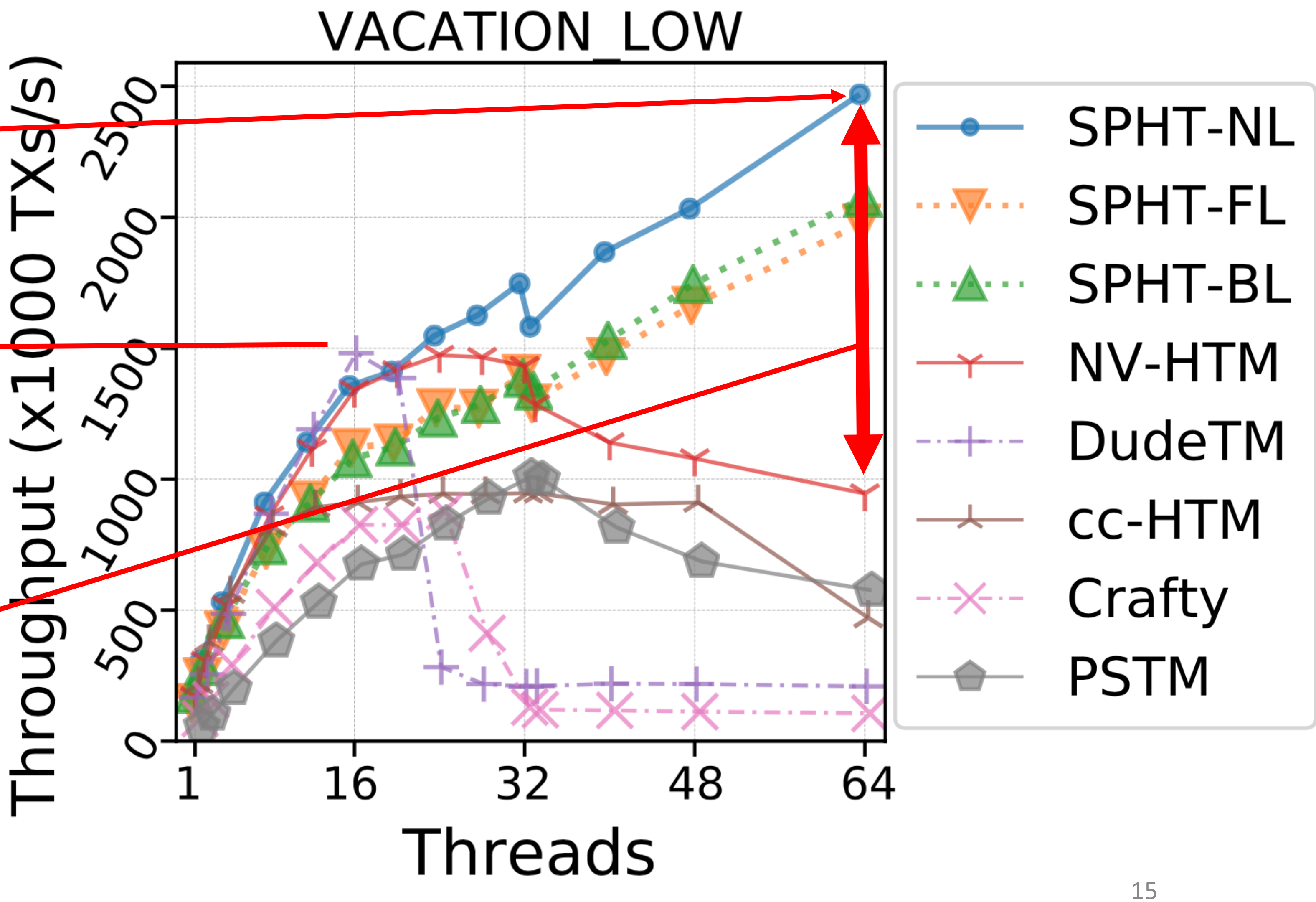
[1] - https://bitbucket.org/daniel_castro1993/spht

STAMP

SPHT and its variants scale up to 64 threads

other solutions bottleneck around 1.5 MTX/s

gains over NV-HTM of 2.6x



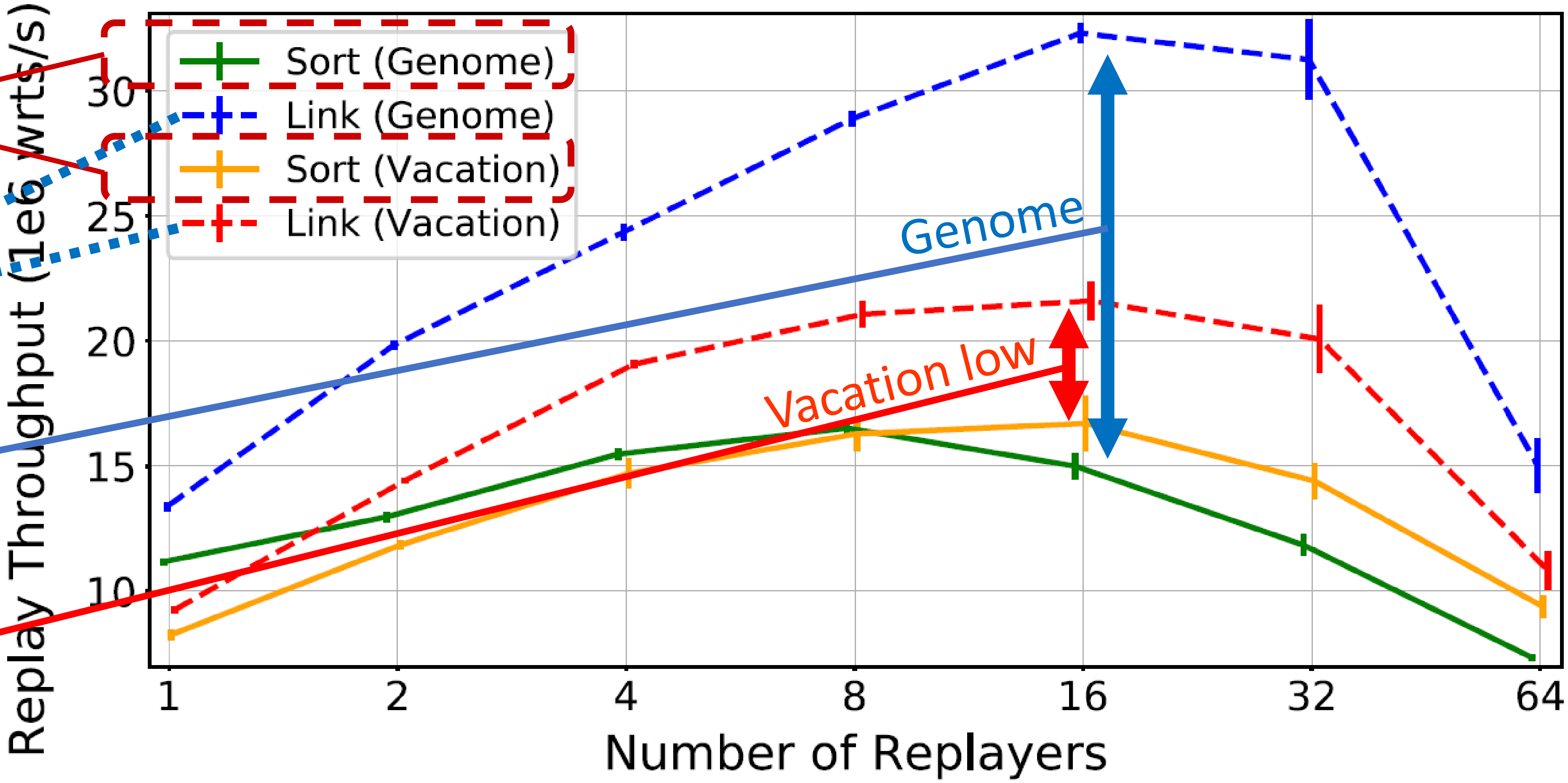
Log Replay

Logs generated by **SPHT-NL**

Linking approaches

Gains 2.1x

Gains 1.3x



Key takeaways

- HTM in CPUs since 2013 and PM available since April 2018:
  - Many HTM+PM solutions proposed:
    - DudeTM [**ASPLOS'17**], cc-HTM [**ISMM'17**], NV-HTM [**IPDPS'18**], Crafty [**PLDI'20**]
  - Little focus on immediate durability;
  - First experimental study on real hardware that combine these systems.
- Existing solutions have scalability limitations:
  - During transaction processing:
    - SPHT introduces a novel group commit approach;
    - **2.6x better throughput** at high thread count (64 threads).
  - In the replay phase:
    - Parallel log replay (scales up to **16 threads**), linked log (**up to 2.1x speedup**)

Thank you! Send follow-up questions to:

daniel.castro@ist.utl.pt