Workload Interference Analysis for HTAP*

Utku Sirin
utku.sirin@epfl.ch
EPFL

Sandhya Dwarkadas
sandhya@cs.rochester.edu
University of Rochester

Anastasia Ailamaki
anastasia.ailamaki@epfl.ch
EPFL

Hybrid Transactional and Analytical Processing (HTAP) systems suffer from workload interference at the software and hardware level. We examine workload interference for HTAP systems and highlight investigation directions to mitigate the interference.

We use the popular two-copy HTAP architecture. The OLTP and OLAP sides are independent components with their own private copies of the data. The OLTP side is a row-store, whereas the OLAP side is a column-store. The OLTP and OLAP sides are connected by means of an intermediate data structure, *delta*, that keeps track of the fresh tuples that are generated by the OLTP side, but not yet transferred to the OLAP side. OLTP transactions register their modifications to delta before committing. OLAP queries first propagate fresh tuples from the OLTP side to the OLAP side and then perform query execution over the data at the OLAP side [1].

Benchmarks & hardware. The OLTP benchmark is a transaction that randomly updates one row. The OLAP benchmark is either an aggregation or a join query. The database size is 30GB. We use a commodity Intel server with two CPU sockets, 14 cores per socket, 2 hyper-threads per core, and three levels of caches. The first two levels of caches are per-core, whereas the last-level cache (LLC) is per-socket, i.e., shared among the 14 cores of each socket.

Software-level interference. OLAP execution time is composed of two parts: (i) fresh tuple propagation time and (ii) query processing time. Fresh tuple propagation time is considered software-level interference.

We fix the number of fresh tuple propagation threads to 1 and the number of OLAP query processing threads to 10 and vary the number of OLTP threads. We place OLTP and OLAP threads and data on separate CPU sockets so that they do not interfere at the hardware level. The propagation thread and delta are placed on the same CPU socket as OLAP. For the aggregation query, fresh tuple propagation time is 2% and 23% of OLAP execution time with 1 and 7 OLTP threads. At these levels of OLTP throughput, software-level interference can be considered modest.

Fresh tuple propagation time increases exponentially for 14 or more OLTP threads since the OLTP tuple generation throughput exceeds the fresh tuple propagation throughput of 1 propagation thread for 14 or more OLTP threads. In order to avoid this, software must ensure that tuple propagation throughput keeps pace with OLTP's tuple generation throughput.

Hardware-level interference. Hardware-level interference is defined as the amount of throughput drop at the OLTP or OLAP side when running OLTP and OLAP concurrently on the same CPU socket compared to when running OLTP or OLAP alone.

Table 1: Normalized OLTP & OLAP throughput. 1T+12A: 1 OLTP and 12 OLAP threads.

		1T+12A	7T+6A	12T+1A
Aggregation	OLTP	0.58	0.73	0.80
	OLAP	0.98	0.99	0.95
Join	OLTP	0.78	0.82	0.89
	OLAP	0.99	0.98	0.98

We fix the number of fresh tuple propagation threads to 1. We place the OLTP, OLAP, and propagation threads on the same CPU socket, where they share LLC and memory bandwidth. We use single CPU socket. We fix the total number of OLTP and OLAP threads at 13 (+one thread used for propagation) and measure throughput when using 1, 7, and 12 OLTP threads. We do not use hyper-threads. Table 1 shows that the OLTP throughput is significantly reduced when running with the aggregation query. The reason is that the sequential-scan-heavy aggregation query highly stresses the memory sub-system. As a result, the OLTP threads are blocked by the OLAP threads at the LLC and memory bandwidth.

The decrease in OLTP throughput is significantly less when running with the join query compared to when running with the aggregation query. This is because the join query is random-data-access-heavy, and hence stresses the memory sub-system less than the aggregation query. The OLAP throughput does not significantly drop for all the cases. This is because the OLTP component stresses the memory sub-system significantly less than the OLAP component.

Conclusions. HTAP systems suffer from interference at both the software and hardware level. Software-level interference depends on the OLTP and fresh tuple propagation throughput. In order to minimize interference, HTAP systems should ensure that fresh tuple propagation throughput is greater than the throughput of the OLTP transactions that generate the fresh tuples.

Hardware-level interference depends on the demand for shared resources such as LLC and memory bandwidth by the OLTP and OLAP workloads. HTAP systems should isolate the OLTP and OLAP workloads in the shared resources and use micro-architectural resource allocation policies that assign the optimal amount of resources to OLTP and OLAP workloads to minimize hardware-level interference.

References

[1] Darko Makreshanski, Jana Giceva, Claude Barthels, and Gustavo Alonso. 2017. BatchDB: Efficient Isolated Execution of Hybrid OLTP+OLAP Workloads for Interactive Applications. In *SIGMOD*. 37–50.

*This work was supported by US National Science Foundation (NSF) Award CNS-1900803, the European Union Seventh Framework Programme (ERC-2013-CoG), under grant agreement no 617508 (ViDa), and Swiss NSF, Project No.: 200021 146407/1 (Workload- and hardware-aware transaction processing).