Smart Thread Scheduling is the Key for OLTP

Pınar Tözün
EPFL

Ippokratis Pandis
IBM Almaden Research Center

ABSTRACT

Typical transactions of online transaction processing (OLTP) workloads have large instruction footprints, 128KB-1MB, that cannot fit in existing L1-I caches, 32KB [1]. Extensive capacity misses due to instructions lead to severe under-utilization of modern hardware's micro-architectural resources [3]. As opposed to the traditional way where a transaction starts and completes its execution on a single core, some recent OLTP designs employ multiple cores to execute a single transaction [1, 2], leading to performance improvements up to 70%. The aggregate size of the caches of several cores creates ample L1-I capacity for the instructions. In addition, the instructions are localized to caches to exploit instruction commonality ($\sim$90%) among different and within the same transactions.

Spreading transactions to multiple cores, however, creates the need to share the data among multiple caches even for one transaction. In turn, data locality, which is crucial when running on multi-socket multi-cores, is hindered. Most of the data misses happen when some data are accessed for the first time during a transaction; they are compulsory [1] rather than due to capacity. Memory locality combined with modern micro-architectural features can overlap the stalls caused by data misses and be sufficient to achieve high throughput. Overall, one can get the best of both worlds, instruction and shared data locality, by carefully scheduling a transaction within the boundaries of a socket, where the data required by that transaction resides.

BODY

Instructions need capacity: Use multiple L1-I.
Data misses are compulsory: Share within a socket.
Thread scheduling is the key for OLTP.

REFERENCES

[1] I. Atta, P. Tözün, A. Ailamaki, and A. Moshovos. SLICC: Self-Assembly of Instruction Cache Collectives for OLTP Workloads. In *MICRO*, 2012.

[2] I. Pandis, R. Johnson, N. Hardavellas, and A. Ailamaki. Data-Oriented Transaction Execution. *PVLDB*, 3(1), 2010.

[3] P. Tözün, I. Pandis, C. Kaynak, D. Jevdjic, and A. Ailamaki. From A to E: Analyzing TPC's OLTP Benchmarks – The obsolete, the ubiquitous, the unexplored. In *EDBT*, 2013.

Volume 2 of Tiny Transactions on Computer Science