Serval: A Wait-free and NUMA-aware Deterministic Concurrency Control Scheme

Haowen Li
tony_li.haowen@keio.jp
Keio University

Rina Onishi
rina.onishi2951@keio.jp
Keio University

Hideyuki Kawashima
river@sfc.keio.ac.jp
Keio University

Abstract

Caracal[2] is a batch-based, multi-version, shared memory deterministic concurrency control protocol designed to handle contention. However, latch acquisition is still required when performing remote writes on the global version array using dynamically allocated buffers during the initialization phase. This paper introduces Serval, a novel method incorporating bitmap and dynamic local version arrays to address Caracal's limitations.

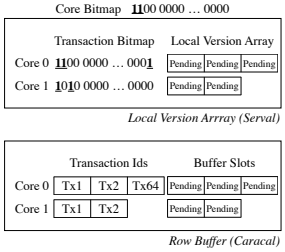

Figure 1: Caracal's data structure has dynamic buffer slots containing the pending version (lower part). We propose a bitmaps and local version array set to achieve wait-free and NUMA-aware(upper part).

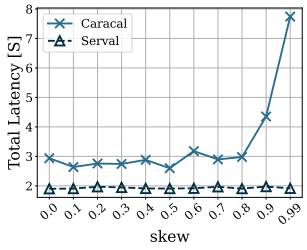

Figure 2: Total latency of Caracal and Serval executing 4,096,000 transactions under write-only workload.

1 Background

Deterministic Concurrency Control (DCC) determines the order of transaction processing before execution to eliminate concurrency control-related aborts. Caracal [2] is a multi-version DCC scheme that incorporates batch append optimization by simultaneously stashing all pending versions of transactions assigned in the same core to a per-core per-row buffer when contention occurs. However, under the high skew level, the latch of the global version array will be contended, resulting in unavoidable wait time with high-latency and low-bandwidth remote write.

2 Proposal

Inspired by concise hash table[1], we propose a new DCC scheme, incorporating two bitmaps: Core Bitmap and Transaction Bitmap, and dynamic local version arrays. As shown in Fig. 1, Core Bitmap has the length of the total number of cores and is inside each row. It indicates which core has a write operation to the corresponding row. Transaction Bitmap has the length of the number of transactions assigned to the corresponding core and is inside the local version array. It indicates which transaction performs the write operation in the corresponding core.

Instead of appending pending versions to the global version array, Serval performs append in the local version array while updating corresponding bitmaps. This method is similar to Chunked Parallel Radix partition, an optimization method used in the Parallel Radix Join algorithm, which amortizes the cost of large sequential reads from remote memory by eliminating remote write[3].

Serval performs up to 4 times better than Caracal under skew level 0.99, as shown in Fig. 2. The acceleration is due to the wait-free data structure and NUMA-aware local write operation.

Acknowledgments

This paper is based on results obtained from the project, "Research and Development Project of the Enhanced infrastructures for Post-5G Information and Communication Systems" (JPNP20017) and the project (JPNP16007) commissioned by the New Energy and Industrial Technology Development Organization (NEDO), and JSPS KAKENHI Grant Number 22H03596.

References

[1] R. Barber, G. Lohman, I. Pandis, V. Raman, R. Sidle, G. Attaluri, N. Chainani, S. Lightstone, and D. Sharpe. Memory-efficient hash joins. *PVLDB*, 8(4):353–364, 2014.

[2] D. Qin, A. D. Brown, and A. Goel. Caracal: Contention management with deterministic concurrency control. In *SOSP*, page 180–194, 2021.

[3] S. Schuh, X. Chen, and J. Dittrich. An experimental comparison of thirteen relational equi-joins in main memory. In *SIGMOD*, page 1961–1976, 2016.