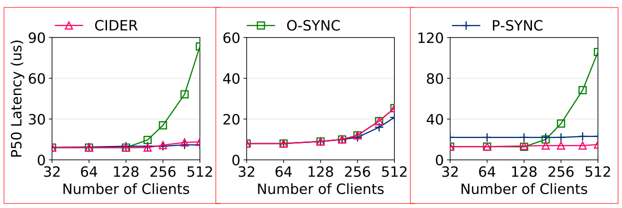
APPENDIX

A ADDTIONAL METRICS

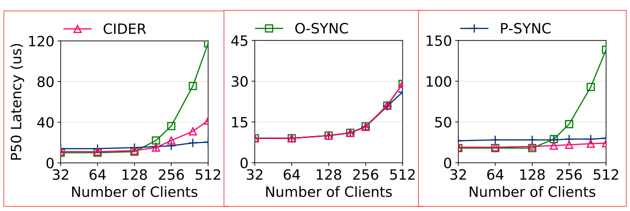
We present supplementary results for the P50 latency in this section. All experimental settings remain consistent with those described in Section 5.2 and Section 5.3.

In most cases, CIDER achieves near-optimal P50 latency. An exception occurs in Figure 22a, where CIDER exhibits slightly inferior P50 latency under a write-intensive workload on RACE, particularly as the number of threads increases. This occurs because the P50 latency in CIDER primarily reflects the latency of optimistically synchronized requests, which typically complete after a small number of retries. This retry overhead results in marginally higher latency compared to pessimistic synchronization.



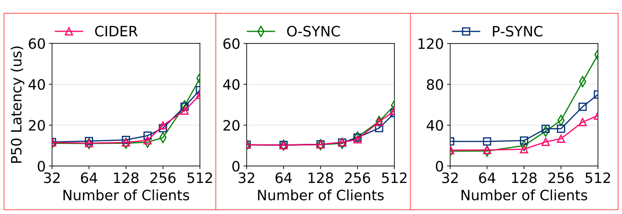
(a) Write-intensive **(b) Read-intensive** **(c) Write-only**

Figure 21: The P50 latency on a pointer array.



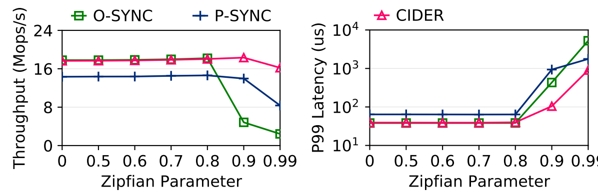
(a) Write-intensive **(b) Read-intensive** **(c) Write-only**

Figure 22: The P50 latency on *RACE*.



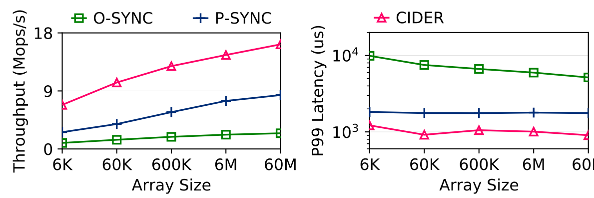
(a) Write-intensive **(b) Read-intensive** **(c) Write-only**

Figure 23: The P50 latency on *SMART*.



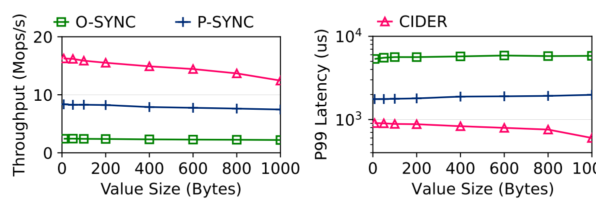
(a) The throughput. **(b) The tail latency.**

Figure 24: The performance comparison as a function of the skew factor (larger values mean more skew).



(a) The throughput. **(b) The tail latency.**

Figure 25: The performance comparison as a function of the array size.



(a) The throughput. **(b) The tail latency.**

Figure 26: The performance comparison as a function of the value size.

B SUPPLEMENTARY SENSITIVITY ANALYSIS

CIDER consistently maintains its optimization effectiveness across various scenarios (including different skewness levels, data scales, and key-value sizes). We conduct the following sensitivity tests, highlighting its capability to enhance efficiency in diverse production environments.

The impact of workload skewness. As shown in Figure 24, we evaluate how workload skewness affects the performance of the pointer array under the write-intensive workload. As the Zipfian skewness grows from 0 to 0.99, the throughputs of O-SYNC, P-SYNC, and CIDER decrease by 7.4×, 1.7×, and 1.1×, while their P99 latencies increase by 138×, 28×, and 23×, respectively. CIDER performs best under both the uniform workload and highly skewed workloads. O-SYNC shows a good performance in the uniform workload while having the poorest performance in highly skewed workloads, *i.e.*, when the skewness is larger than 0.8. This is because the I/O redundancy issue becomes more severe under highly skewed workloads. P-SYNC performs better than O-SYNC in highly skewed workloads, since the pessimistic synchronization avoids I/O retries. However, it performs worse in the uniform workload due to the overhead of lock operations.

The impact of array sizes. As shown in Figure 25, we evaluate the impact of array size on a pointer array under the write-intensive workload. When the array size is relatively small, limited workload concurrency becomes a severe performance bottleneck. This is because a majority of clients are contending for the same entry, synchronizing either by the MCS wait queue or the atomic RDMA_CAS.

As the array size increases, this contention is alleviated, leading to improved throughput across all methods.

The impact of value sizes. As shown in Figure 26, we evaluate the impact of different value sizes on a pointer array under the write-intensive workload. As the size of the value increases, the throughput of all baselines remains stable. This is because CIDER and all baselines are IOPS-bound rather than bandwidth-bound, as they employ either local or global WC techniques to eliminate redundant `update` operations, alleviating bandwidth pressure on the memory-side RNICs.