Concurrency optimization for data processing

Performance comparison

Range scan speedups

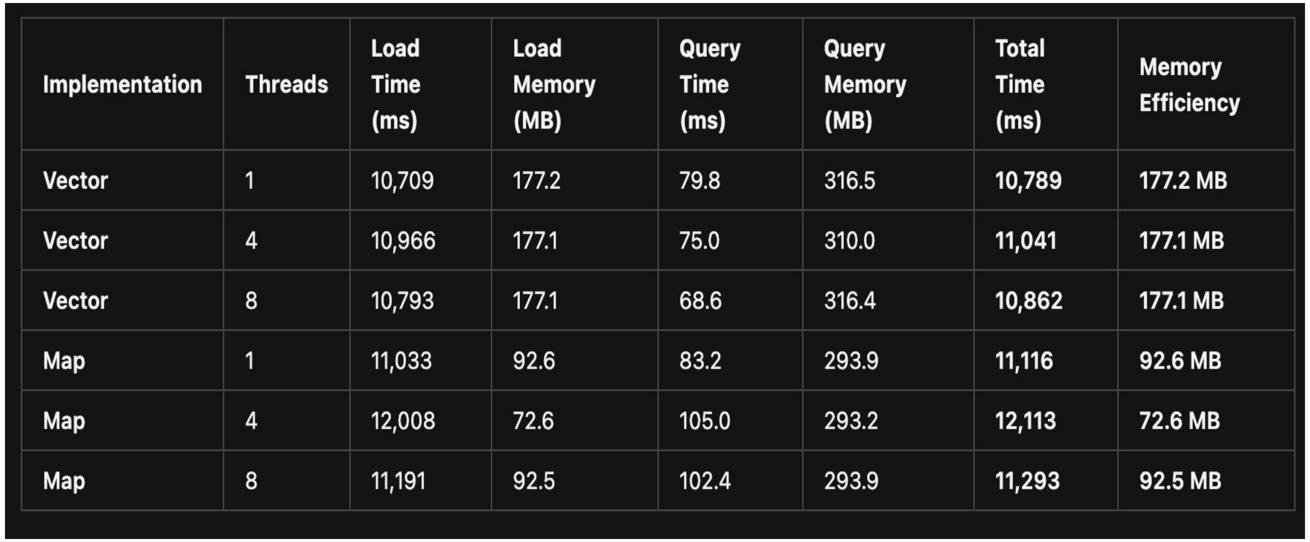
WorldBank: vector 8T vs vector 1T ≈ 3.5×; vector 8T vs map 1T ≈ 2.7×

AirNow: vector 8T vs vector 1T ≈ 2.7×; vector 8T vs map 1T ≈ 2.1×

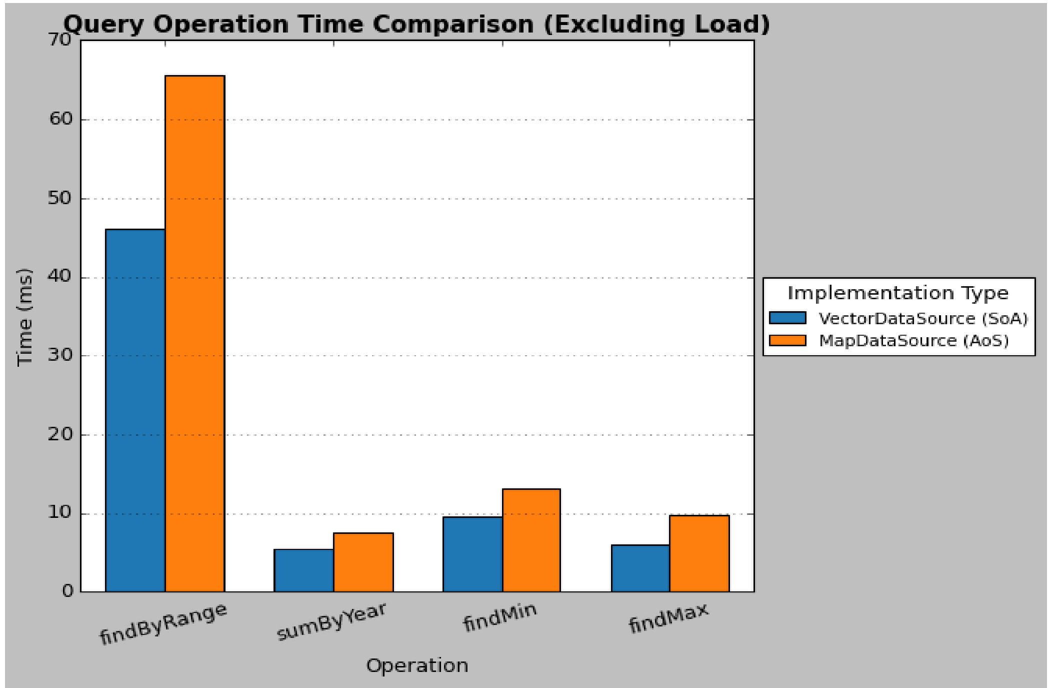
Aggregates (sum/min/max) are already sub-millisecond to low-millisecond; parallelism provides little benefit there due to short critical sections and memory-bound behavior.

Summary of Experiment

- vectorized layout while allocating only dataset-relevant fields reduced memory waste and improved throughput.
- vectorized backend delivered 2.6–3.5× faster range scans on WorldBank and ~2.7× on AirNow under parallel execution, with competitive serial performance and minimal overhead for global aggregates (min/max/yearly sums).



Implementation	Threads	Load Time (ms)	Load Memory (MB)	Query Time (ms)	Query Memory (MB)	Total Time (ms)	Memory Efficiency
Vector	1	10,709	177.2	79.8	316.5	10,789	177.2 MB
Vector	4	10,966	177.1	75.0	310.0	11,041	177.1 MB
Vector	8	10,793	177.1	68.6	316.4	10,862	177.1 MB
Map	1	11,033	92.6	83.2	293.9	11,116	92.6 MB
Map	4	12,008	72.6	105.0	293.2	12,113	72.6 MB
Map	8	11,191	92.5	102.4	293.9	11,293	92.5 MB


Query Operation Time Comparison (Excluding Load)
Time (ms)
findByRange
sumByYear
findMin
findMax
Operation
Implementation Type
VectorDataSource (SoA)
MapDataSource (AoS)