Homework 3 Report

KAI HOGAN

This report compares the performance of two sequential and two parallel SPMV algorithms. The sequential algorithms compute a sparse matrix vector product using the CSR, and COO formats and so do the parallel algorithms. I tested each implementation on the Talapas system using 4, 8, 12, 16, and 20 threads.

CCS Concepts: • **Computing methodologies** → **Parallel algorithms**; *Shared memory algorithms*; • **Theory of computation** → *Design and analysis of algorithms.*

ACM Reference Format:
Kai Hogan. 2025. Homework 3 Report. *ACM Trans. Graph.* 1, 1, Article 1 (October 2025), 1 page. https://doi.org/XXXXXXX.XXXXXXX

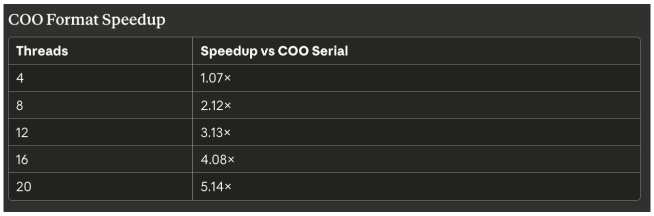
1 Findings

1.1 Sequential Algorithm vs Parallel Algorithm (COO):

Initial Performance (4 Threads):

The COO parallel implementation showed minimal improvement over the serial version when using only 4 threads, achieving just a 1.07× speedup. This marginal gain of 7

Scaling Behavior (8-20 Threads): As the thread count increased, the parallel algorithm's performance improved substantially: 8 threads: 2.12× speedup (execution time dropped from 1.45s to 0.68s), 12 threads: 3.13× speedup (execution time reduced to 0.46s), 16 threads: 4.08× speedup (execution time reduced to 0.35s), 20 threads: 5.14× speedup (execution time reduced to 0.28s). This pattern demonstrates that the COO parallel implementation benefits from increased parallelism, though the scaling is sub-linear. Ideally, doubling the thread count would double performance, but the overhead of thread management prevents this from happening.



COO Format Speedup

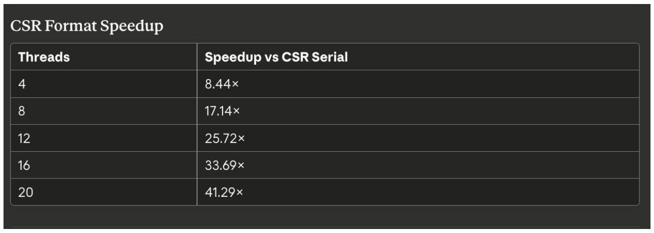
Threads	Speedup vs COO Serial
4	1.07×
8	2.12×
12	3.13×
16	4.08×
20	5.14×

Fig. 1. COO algorithm

1.2 Sequential Algorithm vs Parallel Algorithm (CSR):

The CSR parallel implementation demonstrated dramatically superior performance compared to its serial counterpart across all thread configurations. Unlike the COO format, which showed modest gains, CSR achieved remarkable speedups even at the lowest thread count tested.

Scaling Performance (4-20 Threads): The parallel CSR algorithm exhibited near-linear scaling characteristics: 4 threads: 8.44× speedup (execution time dropped from 2.28s to 0.27s), 8 threads: 17.14× speedup (execution time reduced to 0.13s), 12 threads: 25.72× speedup (execution time reduced to 0.089s), 16 threads: 33.69× speedup (execution time reduced to 0.068s), 20 threads: 41.29× speedup (execution time reduced to 0.055s). This scaling pattern is exceptional, the 20-thread configuration achieved a speedup that exceeds the theoretical maximum of 20×. This super-linear scaling suggests that the CSR format benefits from architectural advantages.



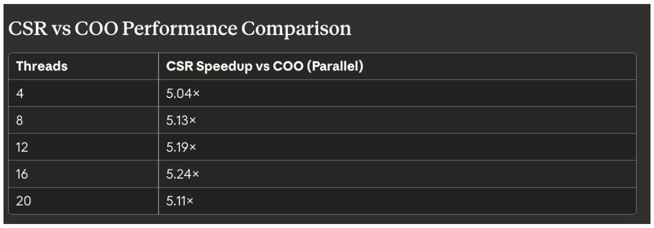
CSR Format Speedup

Threads	Speedup vs CSR Serial
4	8.44×
8	17.14×
12	25.72×
16	33.69×
20	41.29×

Fig. 2. CSR algorithm Speeds

1.3 CSR vs COO:

The CSR parallel algorithm outperformed the COO parallel algorithm for every different thread number. This disparity in performance remained consistent across 4,8,12,16, and 20 threads hovering around 5x. This significant disparity in performance is likely because the CSR parallel algorithm leads to more coalesced access and less computational overhead.



CSR vs COO Performance Comparison

Threads	CSR Speedup vs COO (Parallel)
4	5.04×
8	5.13×
12	5.19×
16	5.24×
20	5.11×

Fig. 3. CSR vs COO

Author's Contact Information: Kai Hogan, khogan@uoregon.edu.

ACM 1557-7368/2025/10-ART1
https://doi.org/XXXXXXX.XXXXXXX