CSE 6220 / CX 4220 – Introduction to High-Performance Computing

Report 3 - SUMMA SpGEMM and APSP

Jiahao Chen

Implementation of 2D SUMMA SpGEMM

We implement sparse matrix-matrix multiplication $C = A \times B$ using a 2D SUMMA (Scalable Universal Matrix Multiplication Algorithm) approach, parallelized with MPI. Each processor holds a sub-block of the sparse input matrices A and B, stored as lists of triplets $((i, j), \text{val})$ representing nonzero entries.

Algorithm Steps

1. **Broadcast Phase (SUMMA-style):** For each step $k \in [0, \sqrt{P} - 1]$:
   - Processors in row k broadcast their local block of A to all processors in their row communicator.
   - Simultaneously, processors in column k broadcast their local block of B to all processors in their column communicator.
2. **Efficient Local Multiplication:** Each processor multiplies the received submatrices using a hash-based kernel:
   - Construct a hash map from rows of B_k to their corresponding (col, val) pairs.
   - For each nonzero A_{ik}, look up matching rows B_{kj} to compute and accumulate C_{ij}.
   - Results are accumulated in a local map to form a sparse representation of the local block of C.

   Runtime Analsysis

   - In a given row, the process with column coordinate k acts as the root and broadcasts its local A and B blocks.The broadcast cost per iteration is typically expressed as: $O(\alpha + m\beta)$, where m is the message size in bytes, α represents latency, and β is the inverse bandwidth. Consider that we have $\sqrt{p}$ nodes on a row or column, the total broadcast cost is $O(\sqrt{p}(\alpha + \beta m))$.
   - The compuation cost is $O(\sqrt{p}(d_A \times d_B))$, where d_A and d_B are average non-zero value in one block. In a sparse scenario, d_A and d_B will be much smaller than the full block sizes, so the computation cost is often dominated by the cost of matching the nonzeros and the broadcasting.

2. APSP using (min, +) Matrix Multiplication (5 pts)

The APSP algorithm computes all-pairs shortest paths by repeatedly squaring the weighted adjacency matrix using the SpGEMM routine.

Let D be the weighted adjacency matrix of a graph, where D_{ij} is the weight of the edge from i to j, or ∞ if no edge exists. The $(\min, +)$ semiring replaces standard arithmetic:

- "Multiplication" becomes addition: $a \cdot b \to a + b$
- "Addition" becomes minimization: $a + b \to \min(a, b)$

Why this work? Repetead Min-Plus Multiplication Compute $D^2, D^4, D^8, \ldots$ in the min-plus semiring. After each squaring, the matrix's (i, j) entry represents the shortest path distance from i to j using up to that many edges.

For instance, after one min-plus multiplication,

$$D^2(i, j) \;=\; \min_k \big(D(i, k) + D(k, j)\big),$$

which gives the shortest path with at most two hops. Doubling the exponent at each step eventually covers the maximum path length needed:

$$D^{(2^h)}(i, j) \quad \text{with} \quad 2^h \;\geq\; n - 1.$$

Final Result After $O(\log n)$ such multiplications, the final matrix will contain the shortest path distances between every pair of vertices (assuming there are no negative cycles).

Algorithm

1. Initialize D with edge weights (and zeros on the diagonal).
2. Compute powers of D using $(\min, +)$ multiplication:

   $$D^{(2)} = D \otimes D, \quad D^{(4)} = D^{(2)} \otimes D^{(2)}, \ldots$$

   where $\otimes$ denotes matrix multiplication in the $(\min, +)$ semiring.
3. After at most $\lceil \log_2(n) \rceil$ steps for n nodes, D converges to the shortest path matrix.

Performance/scalabiltiy graphs

As we can see from the figure, for a large data set, the complexity is proportional to $\sqrt{p}$. The efficiency for the SpGEMM is approximately 1, while for APSP it is larger than 1, because of early termination (convergence).

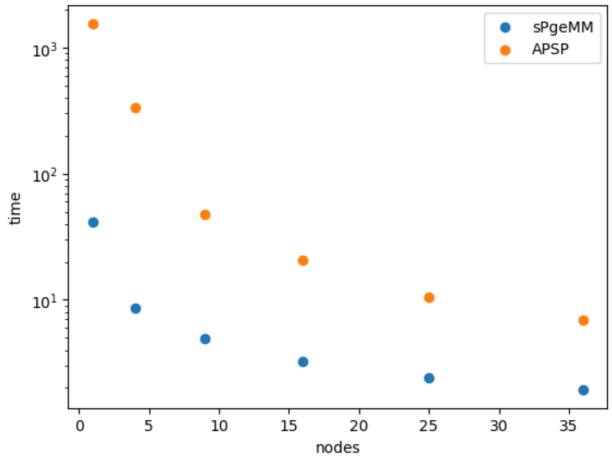

Figure 1: Performace test. Blue points (SpGEMM). Orange points (APSP). SpGEMM is tested on the provided GL7d14.dat dataset. APSP is tested on G12.dat .

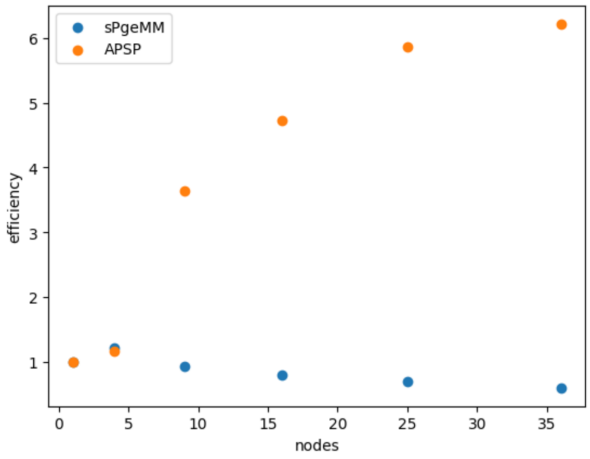

Figure 2: Efficiency $E = T(p)/T(1)$. Blue points (SpGEMM). Orange points (APSP). SpGEMM is tested on the provided GL7d14.dat dataset. APSP is tested on G12.dat .

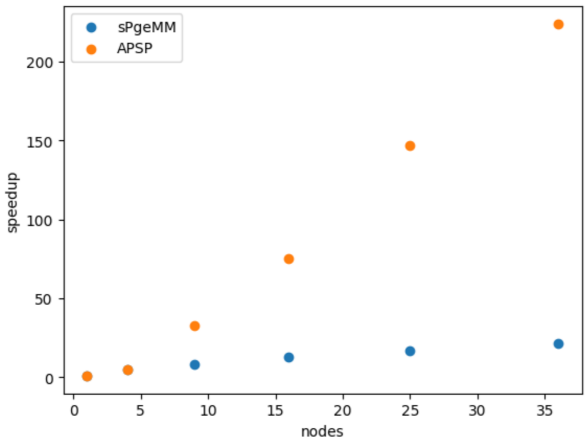

Figure 3: Speedup $T(p)/T(1)$. Blue points (SpGEMM). Orange points (APSP). SpGEMM is tested on the provided GL7d14.dat dataset. APSP is tested on G12.dat .