Dynamic Graph Network Optimization

(Implementing SOSP And MOSP)

Usman Nadeem I212985

Muhammad Nehal i212998

Problem Statement

Challenges

Dynamic Graph Challenge: In large-scale networks, frequent updates (like edge insertions) make full re-computation of multi-objective shortest paths (MOSP) inefficient.

Scalability Limitation: Traditional sequential approaches fail to meet the performance demands of real-time, high-volume data in large graphs.

Our Parallel Solution

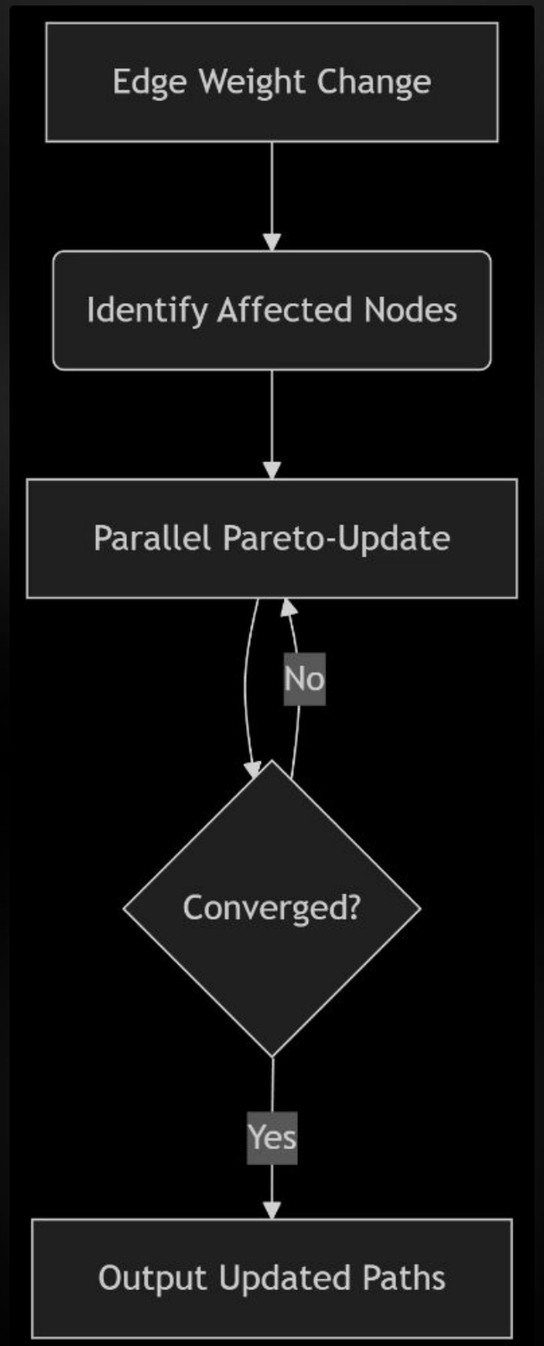
- To handle real-time updates in massive dynamic graphs efficiently, we propose a parallel solution that combines MPI, OpenMP/OpenCL, and METIS for scalable and incremental MOSP updates.

Paper Summary

SOSP (Single-Objective Shortest Path): A shortest path problem that optimizes a single objective in a graph.

MOSP (Multi-Objective Shortest Path): An extension of SOSP that finds optimal paths considering multiple objectives in a graph.

Paper Summary: The paper presents a parallel SOSP update algorithm, a heuristic MOSP update strategy for dynamic networks, and shared-memory parallel implementations optimized for scalable computation of SOSPs and MOSPs.



Key Contributions Of The Paper

Parallel SOSP Update Algorithm

The paper introduces an efficient parallel SOSP update algorithm using···

1 Parallel SOSP Update Algorithm

The paper introduces an efficient parallel SOSP update algorithm using grouping techniques to reduce the total iteration count, improving scalability for large dynamic networks.

2 Heuristic Approach for MOSP

A heuristic algorithm is proposed to quickly update a single MOSP in large networks under time-varying dynamics, providing a practical solution for real-time applications.

3 Shared-Memory Parallel Implementation

The paper develops shared-memory parallel implementations that optimize SOSP and MOSP computations, leveraging multi-core architectures for scalable performance.

Tools Used

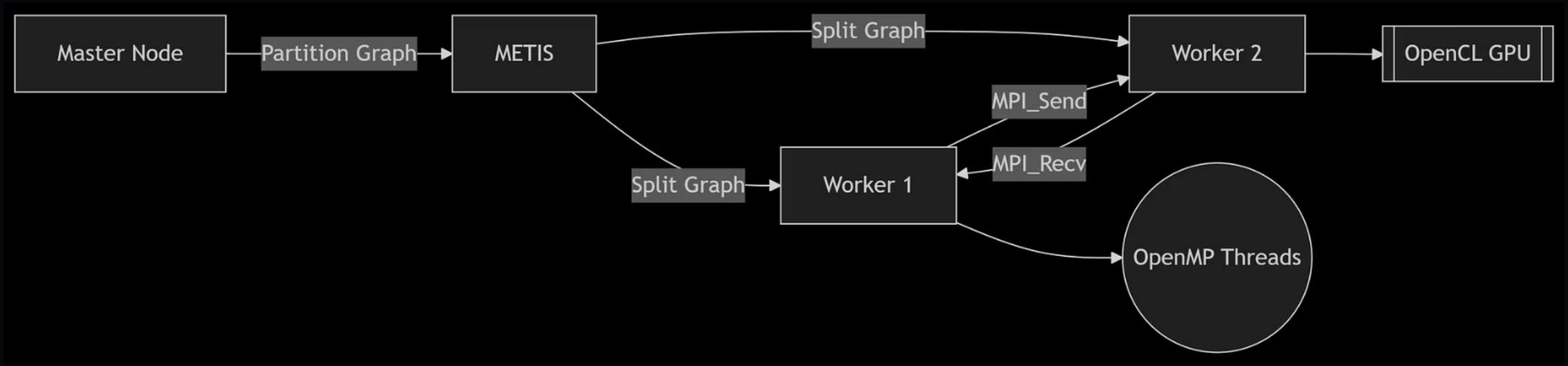
Tools

1. METIS
2. **MPI**
3. OpenMP
4. OpenCL

Role

1. Partitions graph to minimize edge cuts to reduce communication.
2. Distributes partitions and handles inter-process communication.
3. Parallelizes path computations within each node using multithreading.
4. Offers GPU acceleration for intensive Pareto-frontier updates.

System Architecture



Implementation Plan And Workflow

1

Graph Partitioning

METIS partitions the large graph into smaller subgraphs minimizing edge cuts.

2

Task distribution (MPI)

Each partition is assigned to an MPI process for parallel execution with minimal communication.

3

Intra-Node Computation (OpenMP/OpenCL)

Each node updates SOSP/MOSP locally using OpenMP threads or GPU via OpenCL.

4

Pareto-Optimal Update & Synchronization

Update MOSP solutions and synchronize affected nodes across partitions.

Proposed Parallel Strategy

The proposed strategy combines graph partitioning, distributed computing, and multithreading. Below is the pseudo code:

```
G = load_graph()
Parts = METIS_Partition(G, P)

MPI_Init()
rank = MPI_Comm_rank()
LocalG = Parts[rank]

#pragma omp parallel for
for node in LocalG:
    update_SOSP(node)

MPI_Allgather(updates)

if GPU_enabled:
    launch_OpenCL(update_Pareto)
```

Expected Results

Strong Thread Scaling

OpenMP scales well from 1 to 64 threads. Speed increases for different ΔE sizes.

Decreased Execution Time

Sparse graphs improve with increased threads. Shows consistent improvements.

Road-USA Speedup

Up to 15× speedup due to size and sparsity. This network is the most performant.

SOSP Update Dominance

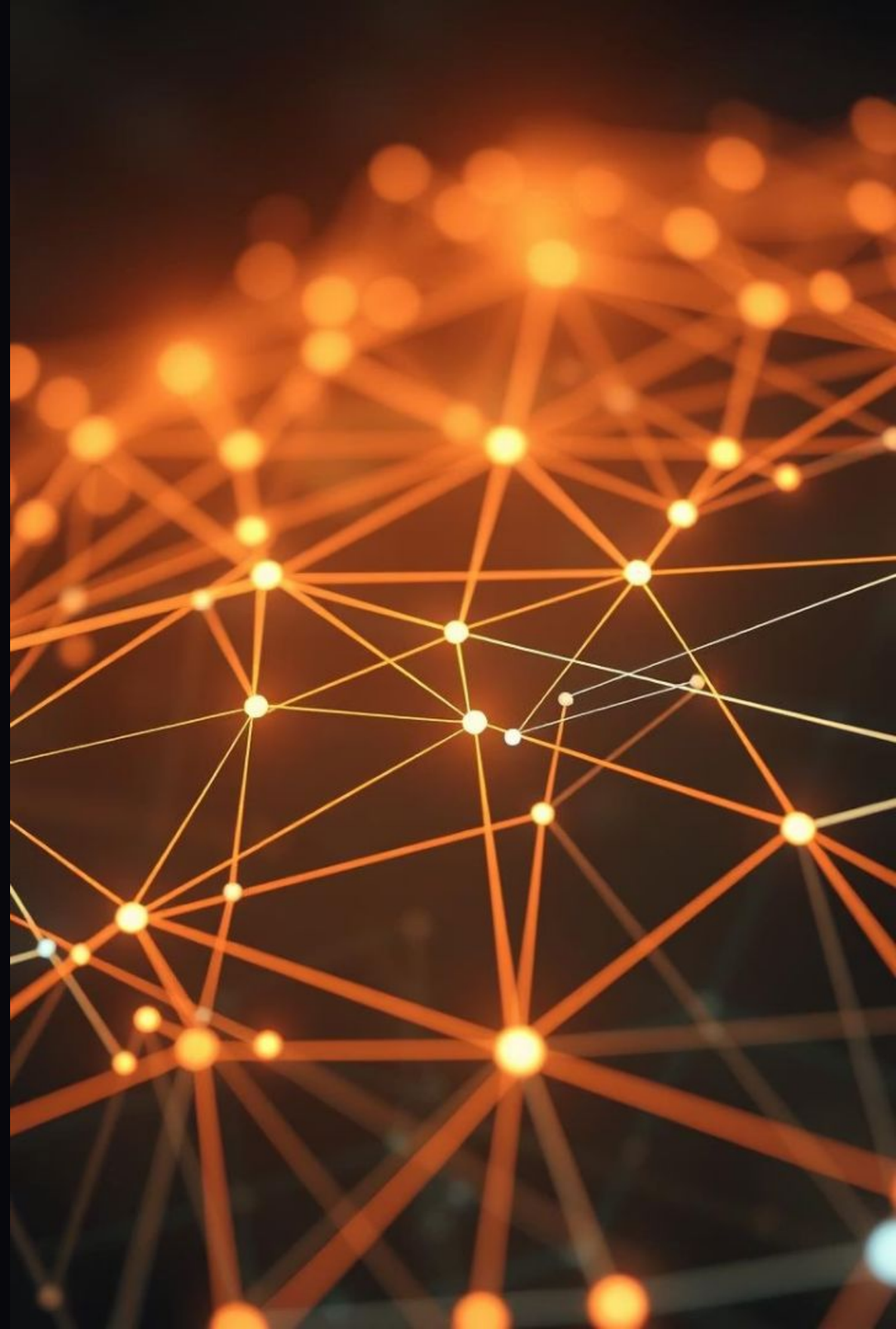
SOSP_update accounts for 90% of runtime. Key contributor to parallel performance.

Speedup

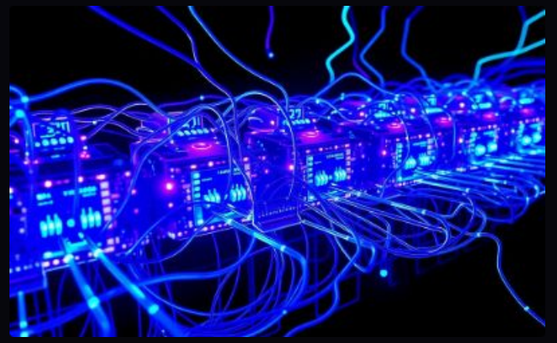
We target a 5–8x speedup on an 8-node cluster using parallel computing.

Scalability

Aim for weak scaling efficiency >70%. Double nodes, double problem size.

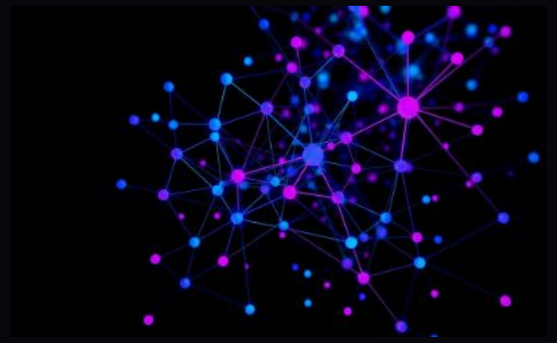


Conclusion



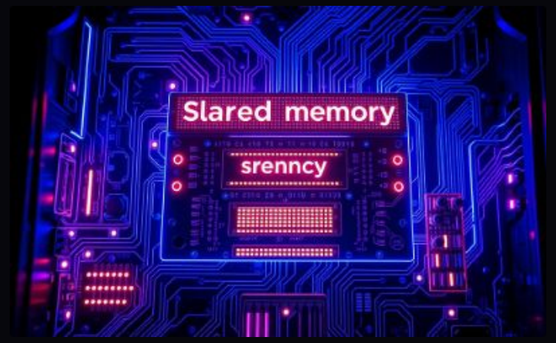
Parallel SOSP Algorithm

Enhanced the algorithm by grouping nodes to reduce iterations.



MOSP Update Approach

Developed a heuristic-based approach for large dynamic graphs.



Scalability Achieved

Significant scalability using shared-memory parallelism.

Future-Ready Implementation

Proposed MPI + OpenMP/OpenCL + METIS for hybrid parallelism.

Ending Note

Thank you for your attention. This work demonstrates efficient parallelization for multi-objective shortest path updates in dynamic networks. Future improvements with MPI, OpenMP/OpenCL, and METIS will further enhance scalability and performance.

Appendix

Dataset Sources: DIMACS, SNAP, OSMnx (listed in report).

METIS Parameters:

```
gpmetis -ptype=rb -objtype=cut input_graph.graph 8
```

MPI Commands:

```
mpirun -np 8 ./mosp_algorithm --input network.txt
```

Reference paper:

A Parallel Algorithm for Updating a Multi-objective Shortest Path in Large Dynamic Networks

dl.acm.org