Parallelizing Dinic's Algorithm

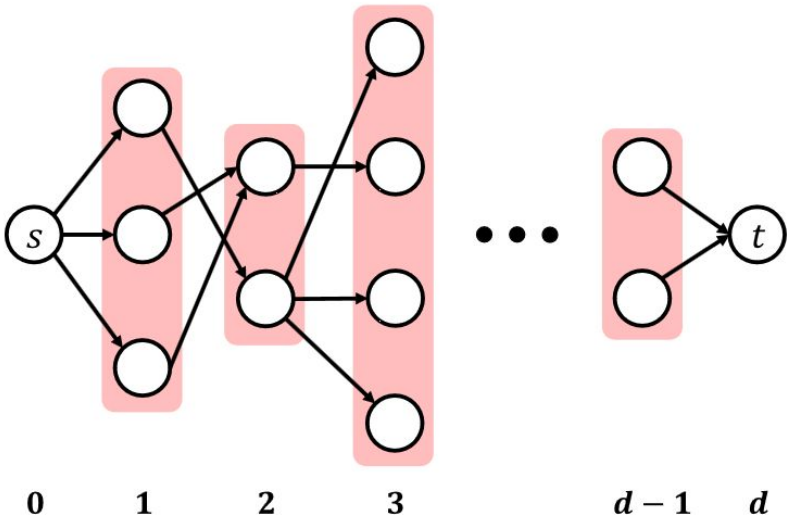
Connor Mowry, Wenqi Deng
Parallel Computer Architecture and Programming
Fall 2024
Carnegie Mellon University

Summary

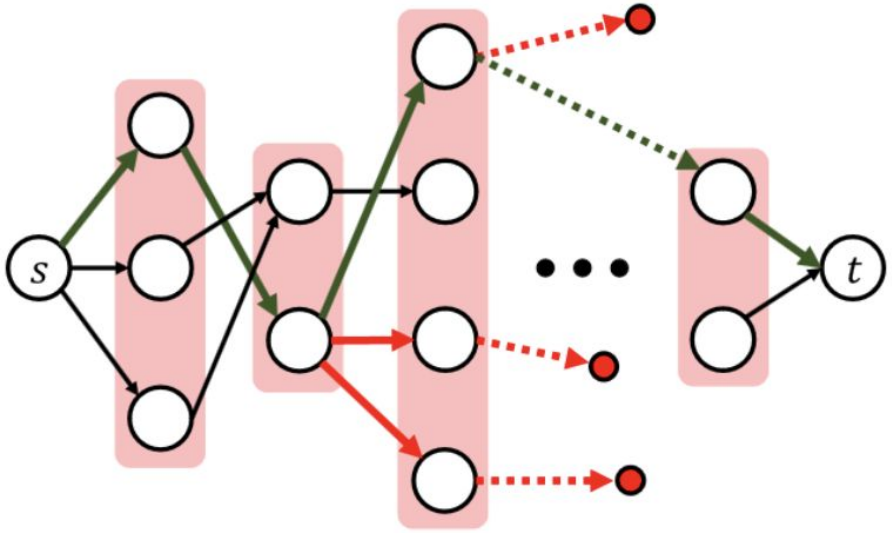
We parallelized Dinic's max-flow algorithm, optimizing the BFS and DFS phases using OpenMP and MPI.

Dinic's Algorithm Background

- **Max-flow** algorithm; alternates between:
  - **BFS phase**: Construct level graph
  - **DFS phase**: Push flow on level graph until no paths remain
- **Data Dependencies**:
  - **BFS**: Frontier updates
  - **DFS**: Dynamically changing residual capacities



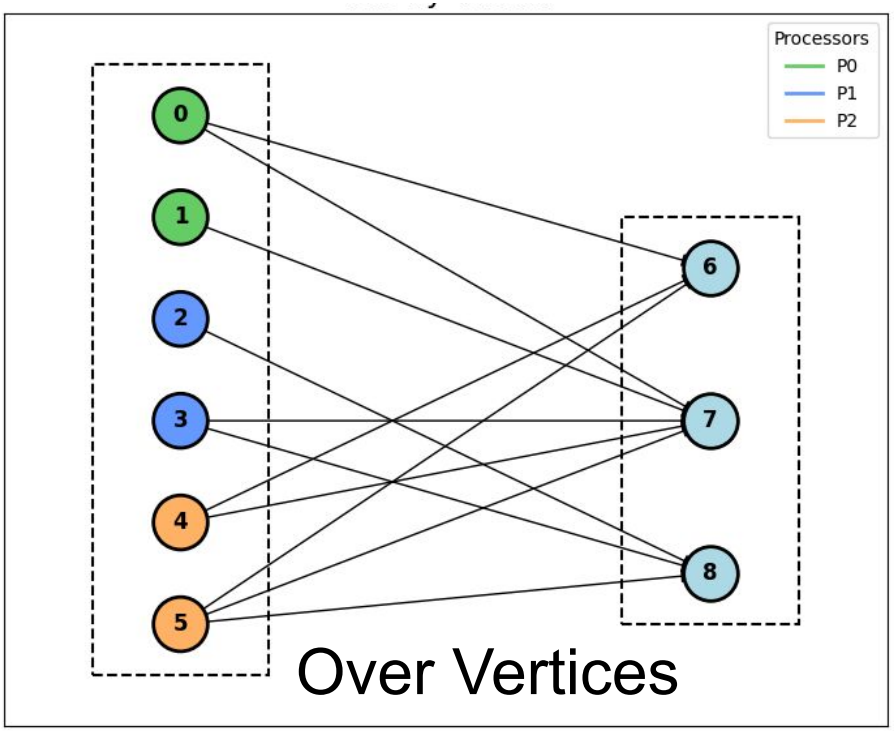
BFS phase

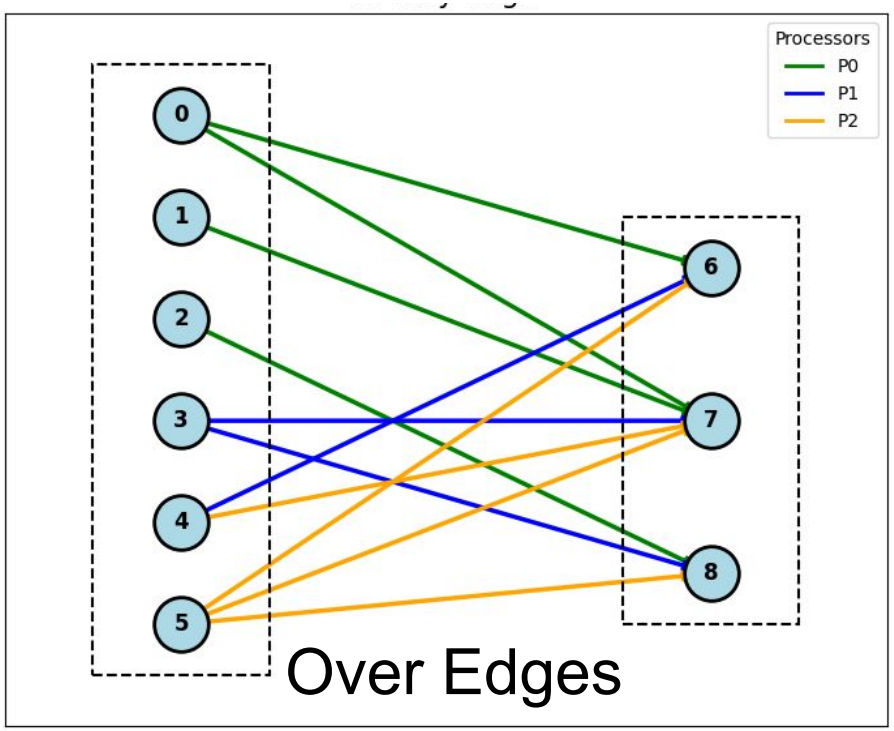


DFS phase

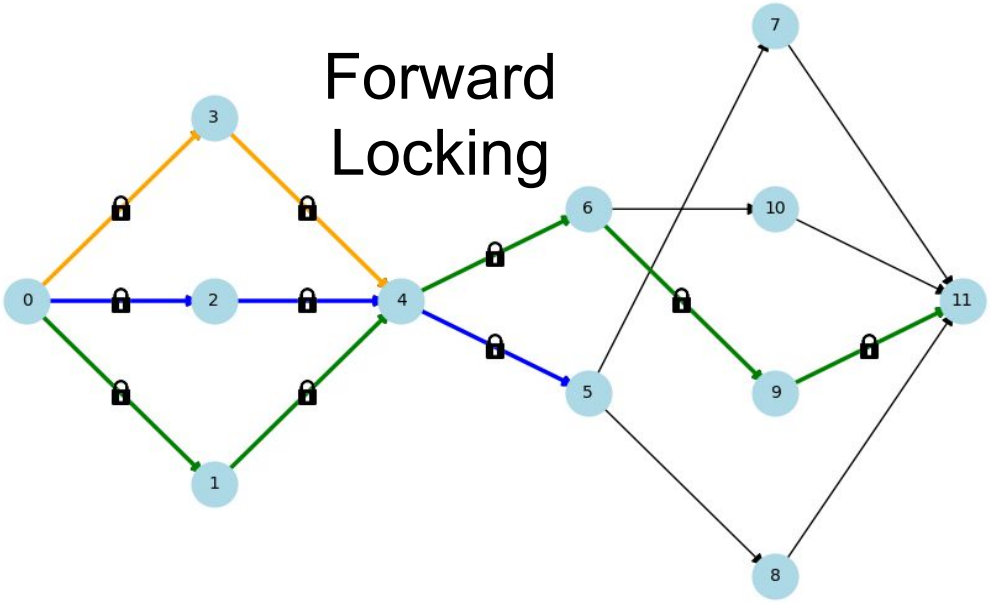
Approaches

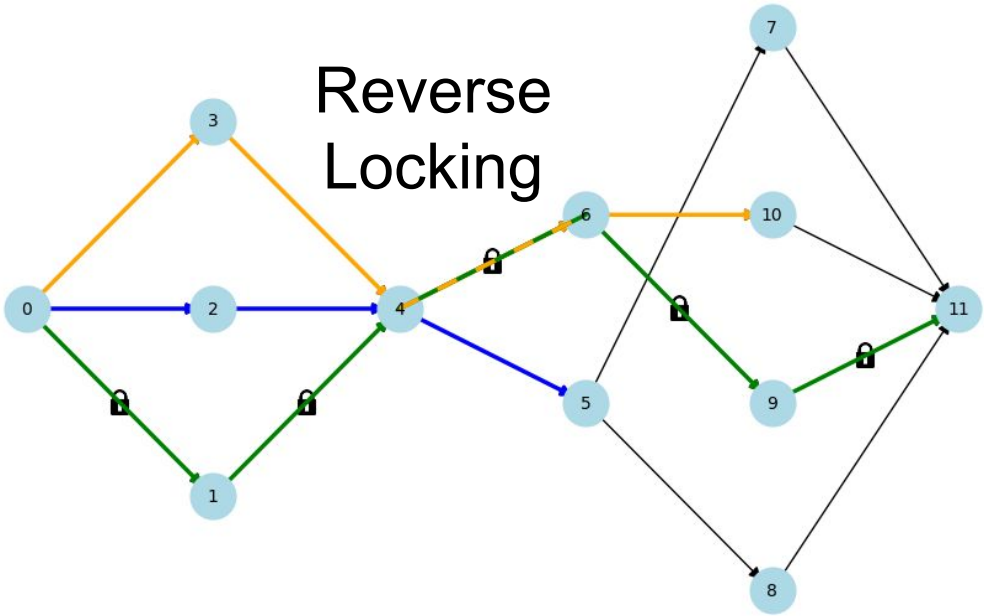
- OpenMP **BFS**:
  - Parallelize over **vertices**
  - Parallelize over **edges**



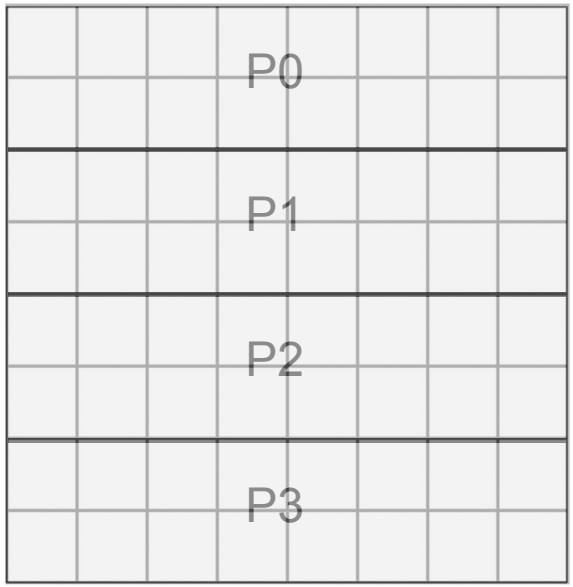


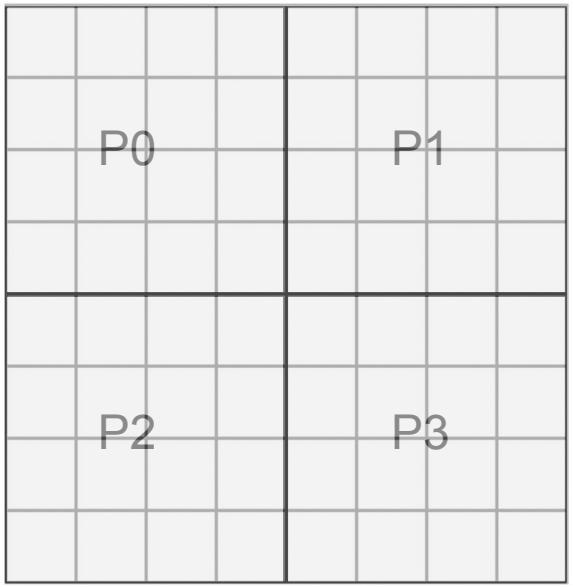
- OpenMP **DFS**:
  - Forward locking: lock during **exploration**
  - Reverse lock: lock after **completion**





- MPI **BFS**:
  - **1D** decomposition: partition **vertices**
  - **2D** decomposition: partition **vertices** and **edges**

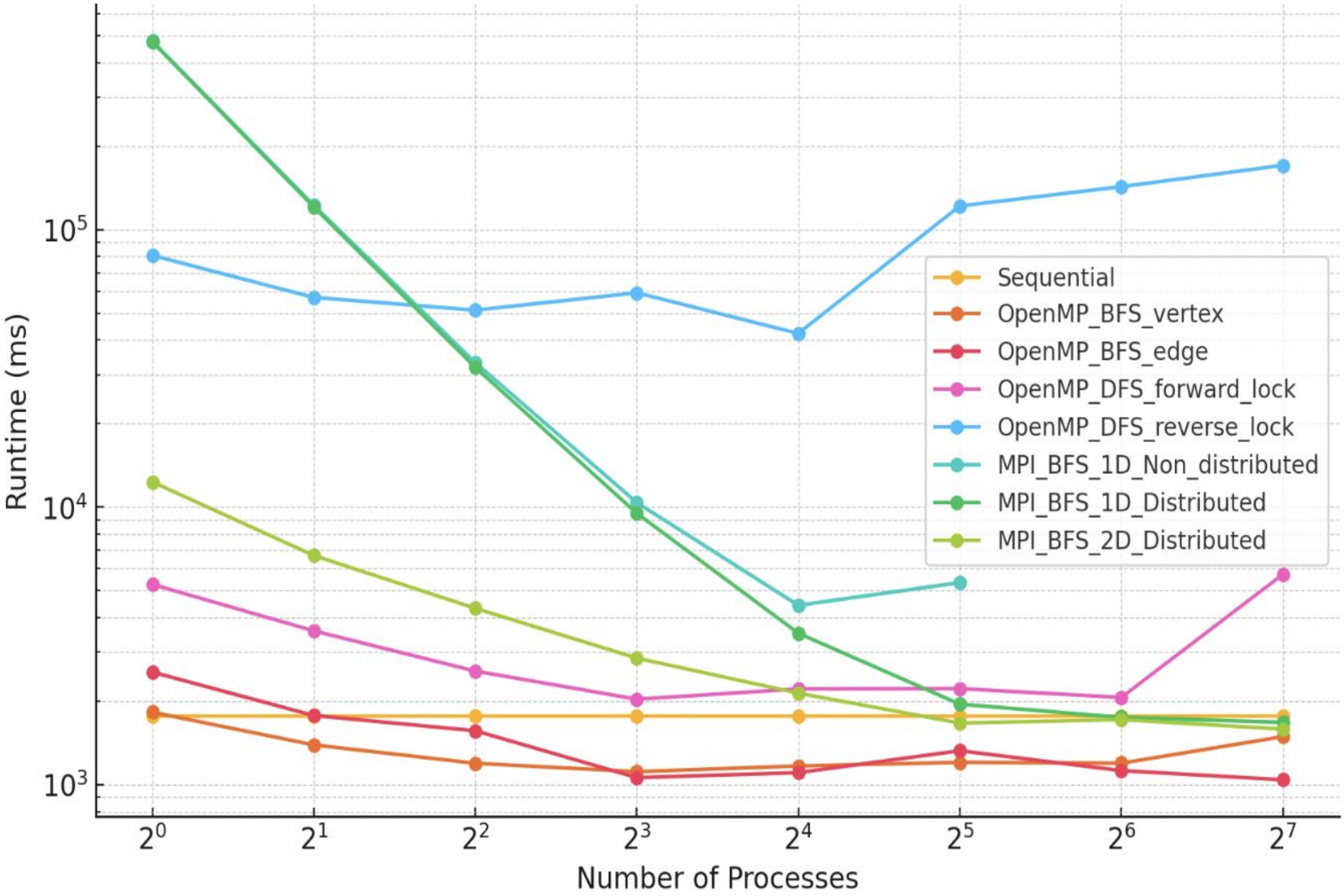




Challenges

- **BFS** Challenges:
  - **Load imbalance**: Uneven degree distribution
  - **Communication overhead** (MPI only): Synchronizing frontier updates across processes
- **DFS** Challenges:
  - **Data dependencies**: Rely on capacities that are concurrently updated
  - **Synchronization costs**: Lock mechanism

Results


Runtime (ms)
10^5
10^4
10^3
2^0
2^1
2^2
2^3
2^4
2^5
2^6
2^7
Number of Processes
Sequential
OpenMP_BFS_vertex
OpenMP_BFS_edge
OpenMP_DFS_forward_lock
OpenMP_DFS_reverse_lock
MPI_BFS_1D_Non_distributed
MPI_BFS_1D_Distributed
MPI_BFS_2D_Distributed

Discussions

- **OpenMP BFS:**
  - Outperforms sequential with ~2 threads
  - **Vertex:** generally faster
  - **Edge:** faster for **dense** graphs
- **OpenMP DFS:**
  - Slower than sequential, but exhibits speedup compared to single thread
  - **Forward lock** outperforms **reverse lock**
- **MPI BFS:**
  - Outperforms sequential with ~32 threads
  - **2D** is faster for **dense** graphs
  - **1D** is faster for **sparse** graphs

Conclusions

Approach	Setting	Effects
Sequential	Sequential	Faster in small and sparse graphs
By Vertex	OpenMP BFS	Faster than sequential in large and dense graphs
By Edge	OpenMP BFS	Faster than by vertex in very dense graphs
Forward Locking	OpenMP DFS	Slower than sequential, but exhibits speedup compared to single thread
Reverse Locking	OpenMP DFS	Slower than forward locking
1D Partition	MPI BFS	Slower than OpenMP BFS; faster than 2D in sparse graphs
2D Partition	MPI BFS	Slower than OpenMP BFS; faster than 1D in dense graphs