Department of Computer Science & Engineering
University of Dhaka

CSE-4209: Parallel and Distributed Systems Lab

Assignment 4

Floyd's All-Pairs Shortest Path Algorithm

Submitted by:
Md. Musfikur Hasan Oli
Roll: CSE-FH-062
Reg No: 2018-725-357

Submitted to:
Dr. Upama Kabir
Professor
Dept. of CSE
University of Dhaka

Speedup: Speedup is typically defined as the ratio of the Sequential execution time to the Parallel/Pipeline execution time. The formula for speedup (S) is:

S = T(s) / T(p)

Where:

T(s) is the execution time for the Sequential Version.

T(p) is the execution time for the Parallel/Pipeline version with p number of processors.

The speedup of a parallel computation, denoted as S, is calculated as the ratio of the execution time of the task when performed sequentially (Ts) to the execution time when performed in parallel using multiple processors or cores (Tp). A speedup value greater than 1 signifies an enhancement in performance. Superlinear speedup refers to a situation where the actual performance surpasses the anticipated speedup theoretically achievable with the given number of processors. Conversely, sublinear speedup occurs when the performance improvement attained through parallel processing falls short of the expected linear speedup predicted by models like Amdahl's Law.

1 - Parallel Speedup:

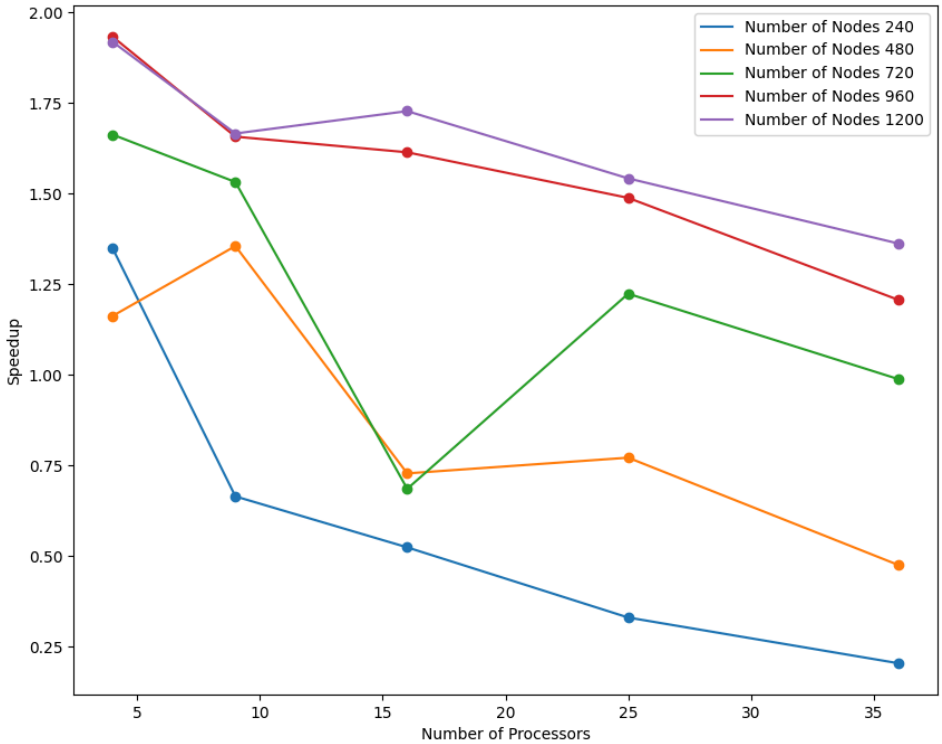

Figure 1: Parallel Speedup

This graph represents the speedup achieved by parallelizing a computational task as the number of processors increases. Here's an interpretation:

- **X-axis (Number of processors):** This represents the number of processors used in the parallel computation.
- **Y-axis (Speedup):** This represents the speedup achieved compared to the serial execution of the same task. Speedup is calculated as the ratio of the serial execution time to the parallel execution time. Higher speedup values indicate better performance.
- **Lines:** Each line in the graph corresponds to a different number of nodes in the computation. In this case, each line represents a different level of granularity or parallelism in the task.
- **Points:** The circle dots on each line represent the speedup achieved for a specific combination of the number of nodes and the number of processors. These points indicate the effectiveness of parallelization at different levels of computational complexity.
- **Trend:** Generally, as the number of processors increases, the speedup initially increases rapidly but may start to level off eventually. This leveling off can happen due to factors such as diminishing returns from parallelization overhead or limitations in the algorithm's parallel efficiency.

1.1 - Parallel Speedup Analysis:

- **Initial Rapid Increase:** In the parallel speedup graph, we observe an initial rapid increase in speedup as the number of processors increases. This is a common trend indicating the benefits of parallelism in distributing the workload among multiple processors. As more processors are added, tasks can be executed concurrently, leading to reduced overall execution time.
- **Diminishing Returns:** However, beyond a certain point, we start to observe diminishing returns in speedup. This phenomenon occurs due to factors such as communication overhead between processors, contention for shared resources, and limitations in the parallelization strategy or algorithm itself. As the number of processors increases, these overheads may outweigh the benefits of parallelism, resulting in diminishing improvements in speedup.
- **Saturation Point:** The graph may eventually reach a saturation point where the speedup levels off or even decreases slightly. This indicates that adding more processors does not significantly improve performance further. Instead, additional resources may be underutilized or even introduce inefficiencies due to increased communication and coordination overhead.

- **Optimal Parallelism:** The optimal number of processors for achieving maximum speedup depends on various factors such as the nature of the computation, the characteristics of the parallelization strategy, and the underlying hardware architecture. Identifying this optimal point is crucial for achieving efficient parallel execution.

2 - Pipeline Speedup:

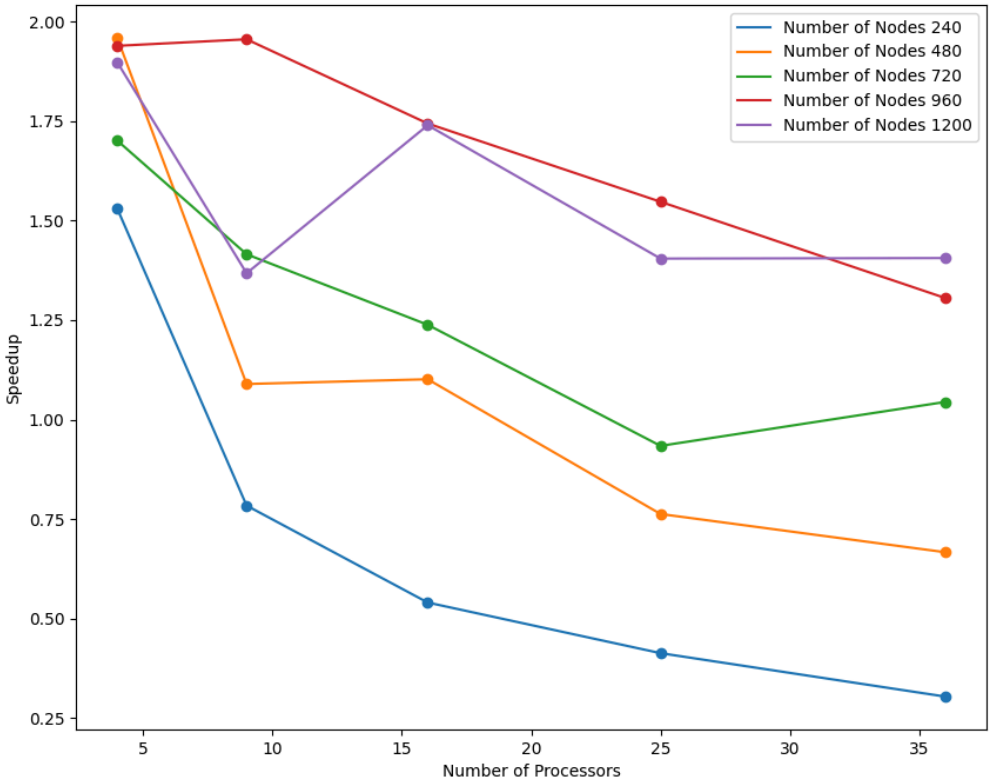

Figure 2: Pipeline Speedup

This graph represents the speedup achieved by a pipelined computational approach as the number of nodes increases. Here's an interpretation:

- **X-axis (Number of nodes):** This represents the number of nodes (or stages) in the pipeline.
- **Y-axis (Speedup):** Similar to the Parallel.png graph, this represents the speedup achieved compared to the serial execution of the task.

- **Lines:** Each line in the graph corresponds to a different number of processors used within each node.
- **Points:** The circle dots on each line represent the speedup achieved for a specific combination of the number of nodes and the number of processors. These points indicate the effectiveness of pipelining at different levels of computational complexity.
- **Trend:** Similar to the parallel graph, the speedup initially increases rapidly but may start to level off eventually. However, the leveling off can occur at different rates depending on the specific characteristics of the pipelined computation and the scalability of the algorithm.

2.2 - Pipeline Speedup Analysis:

- **Incremental Improvement:** In the pipeline speedup graph, we observe a similar trend of incremental improvement in speedup with an increase in the number of nodes (or stages) in the pipeline. Each additional node allows for overlapping of computation and can potentially reduce the overall execution time by exploiting parallelism at different stages of the pipeline.
- **Balancing Workload:** However, the effectiveness of pipelining depends on balancing the workload across different stages of the pipeline. If the workload is unevenly distributed or if there are bottlenecks in certain stages, the speedup may be limited by the slowest stage, leading to sublinear scaling.
- **Scalability Challenges:** As with parallel computing, scalability challenges such as communication overhead, synchronization issues, and resource contention can impact the scalability of pipelined systems. Ensuring efficient communication and coordination between pipeline stages is crucial for achieving optimal speedup.
- **Optimizing Pipeline Depth:** The optimal depth of the pipeline, i.e., the number of stages, depends on the nature of the computation, the granularity of tasks, and the characteristics of the input data. Too few stages may not fully exploit parallelism, while too many stages may introduce excessive overhead and diminish speedup.

Thank You…