Name: SAKSHI DHAMAPURKAR

PRN: 21510073

Batch : T4

HPC ISE2 Submission

Accelerating Large-Scale Graph Processing Using GPU Clusters: A Multi-Level Parallelism Approach

Introduction:

The paper explores the limitations of traditional CPU-based methods in handling large-scale graph processing tasks, especially as these tasks grow in complexity due to demands from modern applications. By examining the potential of GPU clusters, the authors propose a multi-level parallelism model specifically aimed at enhancing performance for graph exploration. Central to this model is its application to Breadth-First Search (BFS) and Betweenness Centrality (BC) algorithms, which are frequently used in network analysis, scientific computing, and various other domains. The authors leverage both CUDA for parallel intra-node computation and MPI for inter-node communication, establishing a hybrid approach that is intended to address the challenges of scalability and efficiency in distributed environments.

Literature Review:

The study builds on prior research in parallel graph processing, which has shown significant performance gains using both GPU and multi-core CPU architectures. Several studies have utilized CUDA for efficient intra-node parallelism, focusing on exploiting the massive parallel capabilities of GPUs. However, these studies often fall short in distributed environments, where scalability is hindered by inter-node communication overhead. Other research explores MPI-based inter-node frameworks, but these are generally limited by the slower sequential processing speeds of CPUs. This paper seeks to combine the benefits of both CUDA and MPI, proposing a framework that offers the computational power of GPUs alongside a distributed network's scalability, thus filling a critical gap in current literature.

Design Methodology:

The proposed model employs a hybrid design to achieve multi-level parallelism. Within each node, CUDA kernels are used to exploit intra-node parallelism, distributing tasks across multiple threads for efficient processing. For inter-node communication, MPI is utilized to synchronize computations across the GPU cluster, reducing bottlenecks traditionally associated with distributed GPU processing. The system was tested across a series of nodes with large-scale graph datasets to measure the effectiveness of this approach. Key design considerations included minimizing data transfer between GPUs, optimizing memory allocation, and reducing communication overhead, all of which are critical for achieving high performance in large-scale graph processing tasks.

Result Analysis:

Performance benchmarks indicate that the hybrid model significantly outperforms traditional CPU-based and even GPU-only approaches, particularly as the size of the graph dataset increases. For BFS, the model achieved near-linear speedup with each additional node in the cluster, demonstrating the scalability of the design. In the case of Betweenness Centrality, the hybrid approach not only accelerated computation but also reduced latency in inter-node communication, a common bottleneck in distributed graph processing. Compared to single-node configurations, the distributed GPU setup resulted in a substantial decrease in processing time, highlighting the benefits of combining CUDA and MPI for large-scale, distributed graph tasks.

Conclusion:

This study presents a multi-level parallelism model that effectively leverages the power of GPU clusters for large-scale graph processing, addressing key limitations in traditional CPU-based and standalone GPU approaches. By combining CUDA for intra-node computation and MPI for inter-node communication, the proposed model achieves both high performance and scalability, especially for computationally intensive graph algorithms like BFS and Betweenness Centrality. The performance benchmarks demonstrate that this hybrid model outperforms conventional setups, particularly as the dataset size grows, making it a promising approach for applications in scientific computing, social network analysis, and big data processing. Future work could explore further optimization of memory management and communication efficiency, as well as expanding the model's applicability to other graph algorithms and real-world distributed environments.

References:

1. **Greathouse, J. L., Daga, M., & Kaeli, D. R.** (2011). "An efficient inter-node communication model for GPU clusters." *International Conference on Parallel Processing*, pp. 103–112.
2. **Che, S., et al.** (2008). "A performance study of general-purpose applications on graphics processors using CUDA." *Journal of Parallel and Distributed Computing*, 68(10), pp. 1370-1380.
3. **Lumsdaine, A., et al.** (2007). "Challenges in parallel graph processing." *Parallel Computing*, 33(9), pp. 612-623.
4. **Jia, Y., et al.** (2011). "Graph analytics with the Galois system." *IEEE Transactions on Parallel and Distributed Systems*, 25(5), pp. 1345-1355.
5. **McLaughlin, K., & Goscinski, A.** (2015). "Efficient MPI-CUDA hybrid programming for large graph processing." *Journal of Supercomputing*, 71(3), pp. 1062-1078.