Analysis of Parallelized Memory Algorithms in High Performance Computing

Prathyusha M R (223CS500)
Shreyas Udaya (211CS152)
Group 09

Department of Computer Science and Engineering
National Institute Of Technology Karnataka Surathkal 575025

November 26, 2023

Overview

- Introduction
- OpenMP
- Literature Survey
- Literature Survey Summary Table
- Limitation
- Problem statement
- Approach
- References

Introduction

- Remote sensing technologies have increased geospatial data collection and resolution, which requires efficient computational algorithms to process big geographic information systems (GIS) data.
- Several algorithms are developed to support computational tasks in environmental modeling. However, with the increase in data size, calculating parameters on a single computer is not practical using serial algorithms.
- However, parallelization of flow accumulation tasks remains challenging due to spatial dependency and global computation.

OpenMP

- Parallel algorithms are used to improve computational efficiency by breaking down complex problems into manageable tasks that can be executed simultaneously using multiple processors.
- OpenMP is the API standard for parallel computing using shared memory. It provides directives that enable developers to create efficient and scalable parallel algorithms.
- In OpenMP, the program is shared among several threads, where each thread executes a portion of the code concurrently with the coordinated access to shared memory. It improves the efficiency of algorithms and applications in various fields

Literature Survey

- The flow accumulation algorithm is a crucial tool in hydrology and GIS for understanding surface water movement. This method helps identify primary flow paths within a watershed and is essential for flood prediction, watershed management, and terrain analysis.
- In existing literature, different flow accumulation algorithms are suggested to achieve fast and accurate result to calculate longest flow path and in determining how material flows.
- The existing research are summarized in a table including various approaches, evaluation methodologies, results, and the challenges.

Literature Survey Summary Table

Sl No.	Title	Authors	Approach	Results	Observation
1	High-performance parallel implementations of flow accumulation algorithms for multicore architectures	Kotyra *et al.* [1]	Two main approaches are discussed in o parallelize flow accumulation algorithms: the bottom-up approach and the top-down approach.	The result inferred that the top-down algorithm was fastest, with an average execution time of less than 30 seconds.	Compared to sequential version, the results showed a high correlation between the number of cores employed and the speedup.
2	Scalability and composability of flow accumulation algorithms based on asynchronous many-tasks	Jong *et al.* [2]	The authors developed flow accumulation algorithms to determine how the material flows downstream.	The AMT-based algorithms for flow accumulation operations perform well in terms of scalability and composability .	The algorithm function well when paired with other operations and utilize additional hardware efficiently.
3	Fast parallel algorithms for finding the longest flow paths in flow direction grids	Kotyra *et al.* [3]	Seven fast raster-based algorithms to determine the longest flow paths in flow direction grids using a linear time complexity approach.	The algorithms obtained significant speedups of up to 30 times quicker on Windows and 17 times faster on Ubuntu.	The suggested algorithm performed well in achieving fast and accurate result in determining longest flow pathways in flow direction grids.
4	A recursive algorithm for calculating the longest flow path and its iterative implementation	Cho *et al.* [4]	The longest flow path algorithm that computes a small number of rasters to enhance efficiency and decrease computation time	The algorithm's performance is affected by disk type and memory size, with solid-state drives and larger memory sizes resulting in faster computation times.	In order to speedup traversal and eliminate inferior neighbor cells, the algorithm additionally uses branching technique.

Literature Survey Summary Table Continued ...

SI No.	Title	Authors	Approach	Results	Observation
5	Identifying challenges and opportunities of in-memory computing on large HPC systems	Huang *et al.* [5]	The author presented comprehensive study of in-memory computing. They discussed portability, robustness, usability, and performance of software	The results suggested that in-memory computing offers much higher scalability and performance than the traditional post-processing.	Most of the commits were towards performance maintenance, suggesting it has a significant role towards computation.
6	High-performance watershed delineation algorithm for GPU using CUDA and OpenMP	Kotyra *et al.* [6]	The author proposed a fast watershed delineation algorithm for GPU. that uses OpenMP and CUDA.	The results showed that the algorithm outperformed traditional GIS software packages in terms of speed and efficiency.	The algorithm's performance is affected by the choice of hardware and software platforms.
7	Accelerating Multiple Flow Accumulation Algorithm Using MPI on a Cluster of Computers	Stojanovic *et al.* [7]	The author suggested accelerating the flow distribution phase using MPI on a cluster.	The experimental evaluation is conducted on several large DEM datasets and varying numbers of computers in the cluster.	The approach overlaps process computing and communication achieves the best results.
8	A Quantitative Study of Locality in GPU Caches for Memory-Divergent Workloads	Lal *et al.* [8]	The author presented a quantitative analysis on the caches for memory divergent workloads simulated by gpgpu-sim.	Higher inter-warp hits (46\%) at the L1 cache for memory-divergent workloads compared to the state-of-the-art.	Data over-fetch wastes around 50\% of cache capacity and other limited resources.

Limitation

Some of the limitation in the existing literature includes:

- The flow accumulation algorithms suggested in [1, 2, 3, 5, 6, 7] has been parallelized using OpenMP to support computational tasks in environmental modeling. But the weighted flow accumulation algorithm suggested in [4] does not support parallelization.
- The study in [7] does not cover other parallel and distributed computing methods and technologies that can be used for geospatial data processing and analysis.
- The algorithm suggested in [6] may not be suitable for all types of GIS-related problems. Also, the performance of the algorithm in [2, 4] was assessed on a limited set of datasets.
- The approach in [3] might not be applicable unsteady flow conditions since it is based on raster data and a steady-state flow assumption.

Problem statement

To parallelize the weighted flow accumulation algorithm to calculate the longest flow path using OpenMP and analyze its performance.

Approach

- The flow accumulation algorithm presented in [9] supports parallel computation using OpenMP. The source code can be found in [10].
- However, the algorithm used for calculating weighted flow accumulation and longest flow path [4] does not support parallelization. The source code can be found in [11].
- In this work, we propose to parallelize the weighted flow accumulation algorithm

Reference I

- B. Kotyra, Łukasz Chabudziński, and P. Stpiczyński, “High-performance parallel implementations of flow accumulation algorithms for multicore architectures,” *Computers and Geosciences*, vol. 151, p. 104741, 2021.
- K. de Jong, D. Panja, D. Karssenberg, and M. van Kreveld, “Scalability and composability of flow accumulation algorithms based on asynchronous many-tasks,” *Computers and Geosciences*, vol. 162, p. 105083, 2022.
- B. Kotyra and Łukasz Chabudziński, “Fast parallel algorithms for finding the longest flow paths in flow direction grids,” *Environmental Modelling and Software*, vol. 167, p. 105728, 2023.

Reference II

H. Cho, “A recursive algorithm for calculating the longest flow path and its iterative implementation,” *Environmental Modelling and Software*, vol. 131, p. 104774, 2020.

D. Huang, Z. Qin, Q. Liu, N. Podhorszki, and S. Klasky, “Identifying challenges and opportunities of in-memory computing on large hpc systems,” *Journal of Parallel and Distributed Computing*, vol. 164, pp. 106–122, 2022.

B. Kotyra, “High-performance watershed delineation algorithm for gpu using cuda and openmp,” *Environmental Modelling and Software*, vol. 160, p. 105613, 2023.

N. Stojanovic and D. Stojanovic, “Accelerating multiple flow accumulation algorithm using mpi on a cluster of computers,” *Studies in Informatics and Control*, vol. 29, no. 3, pp. 307–316, 2020.

Reference III

S. Lal and B. Juurlink, “A quantitative study of locality in gpu caches,” in *Embedded Computer Systems: Architectures, Modeling, and Simulation: 20th International Conference, SAMOS 2020, Samos, Greece, July 5–9, 2020, Proceedings*, (Berlin, Heidelberg), p. 228–242, Springer-Verlag, 2020.

H. Cho, “Memory-efficient flow accumulation using a look-around approach and its openmp parallelization,” *Environmental Modelling and Software*, vol. 167, p. 105771, 2023.

H. Cho, “Grass gis 8.3 addons manual pages - r.flowaccumulation.”

H. Cho, “Grass gis 8.3 addons manual pages - r.accumulate.”

THANK YOU