Parallel DBSCAN Clustering on LiDAR Point Cloud with OpenMP and CUDA

Ethan Ky (etky), Nicholas Beach (nbeach)

April 16, 2024

1 Summary

Over the past few weeks, we've done more research on the DBSCAN algorithm. For our OpenMP implementation, we've identified a parallel algorithm based on a disjoint-set data structure, which would have good workload balancing. We've also determined how to download and process raw Velodyne point cloud data from the KITTI Vision Benchmark Suite. These point clouds are loaded from text files and then formatted into vectors of n four-dimensional points. Each of the points contains its 3D euclidean coordinate as well as a reflectance value. We've identified public C++ libraries which provide implementations of KD-Trees, from nanoflann, and disjoint sets, from Boost. We expect the former to provide the ability to perform fast neighbor radius search for each of our points. In our research into parallel optimizations for DBSCAN, we identified speed-up benchmarks that we can attempt to replicate in our implementations. We have begun implementing sequential and OpenMP versions of the algorithm, which we hope to have prepared and tested later this week.

2 Plans

We are falling about one-week behind on the goals and deliverables stated in our proposal. However, we still believe that we will be able to produce all of the deliverables. We have a better idea of how to implement the OpenMP

version, and have identified possible approachs for the CUDA implementation. It should just be a matter of writing the code. Below is a detailed schedule for the coming weeks:

- Week 3 (April 14 - April 20) - Ethan: Finalize disjoint-set implementation with OpenMP. Collect baseline results for speedup. Conduct more literature review for CUDA parallelization
- Week 3 (April 14 - April 20) - Nicholas: Finalize Sequential Implementation for baseline speed. Start on rendering development.
- Week 4 (April 21 - April 27) - Ethan: Write CUDA implementation. Collect speedup results.
- Week 4 (April 21 - April 27) - Nicholas: Develop renderer for project visualization.
- Week 5 (April 28 - May 4) - Ethan: Collect and analyze performance results. Determine bottlenecks and implement possible optimizations.
- Week 5 (April 28 - May 4) - Nicholas: Wrap up remaining implementation (if any) and work on poster board and presentation.

3 Poster Session

For the poster session, we are still planning to render a visualization of the point cloud clusters. However, to better align our project with the course concepts, we will be placing greater emphasis on the analysis of our parallel performance. We should be including more plots pertaining to speedup in different data distributions, workload balance, and bottleneck sources.

4 Concerns

We do still have concerns for whether our project would have many interesting 15-418 concepts. One area we are considering exploring is how dynamic clustering can be optimized for parallel computing. We feel that there may be some interesting design choices to make when we need to be able to account for points being added or removed, especially when we are maintaining several data structures in shared memory. However, based on the papers we've

looked at, this seems to be a challenging problem that we probably won't have time to adequately explore.

5 Resources

Below are papers we have looked at for possible approaches:

- A New Scalable Parallel DBSCAN Algorithm Using the Disjoint-Set Data Structure
- CUDA-DClust+: Revisiting Early GPU-Accelerated DBSCAN Clustering Designs
- Design and optimization of DBSCAN Algorithm based on CUDA
- HY-DBSCAN: A hybrid parallel DBSCAN clustering algorithm scalable on distributed-memory computers
- STRP-DBSCAN: A Parallel DBSCAN Algorithm Based on Spatial-Temporal Random Partitioning for Clustering Trajectory Data

Below are C++ libraries we will be using:

- nanoflann
- KITTI Vision Benchmark Suite