Hilbert curve based flexible dynamic partitioning scheme for adaptive scientific computations

Nearest Common Ancestor (NCA) based approach for Hilbert order calculation

Milinda Fernando & Hari Sundar (Advisor)

School of Computing, University of Utah

Motivation

Partitioning and Load Balancing

- Partitioning : Assignment of application data/tasks to processors for parallel computation
- Load Balancing: Concerned with optimized resource utilization, maximum throughput, minimize response time and avoid overloading of any single resource.

The following images depicts the partitioning of unstructured finite element mesh.

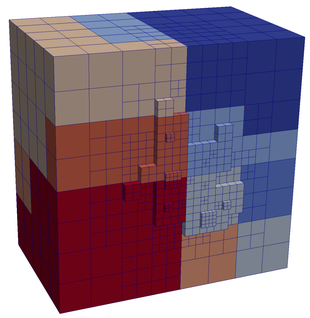

Figure 1: Octree partitioned using space filling curve

There are two main approaches to perform partitioning. Those are,

- Space Filling Curve (SFC) based partitioning
- Graph based partitioning

Graph Partitioning

- Widely used for static partitioning tasks.
- No geometric locality of objects preserved.
- No linear ordering of objects.
- Not incremental when mesh becomes finer.

SFC Partitioning

- Widely used for dynamic partitioning tasks.
- Geometric locality of objects preserved in partitioning.
- Linear ordering of objects may improve the cache performance.
- Easily incremental and adaptive approach.

Our Contributions

1. We propose a new algorithm based on NCA to compute the Hilbert ordering, 9 times faster than the recursive approach.
2. Our NCA based algorithm is easily extensible to compute the ordering of any generic SFC (i.e. Morton, Peano etc.)
3. We empirically show that by introducing some flexibility(slack) to the load balancing, we can reduce the communication costs further, improving the efficiency of overall computation.

Methodology

Space Filling Curves (SFC)

SFC is a surjective mapping between the one dimensional space to higher dimensional space. Generally almost all the SFC adhere to the recursive nature in curve generation.Because of the recursive nature SFCs have, most of the SFCs can be computed recursively.

Here are the images of Hilbert and Morton Space filling curves.

New Nearest Common Ancestor (NCA) based Hilbert Ordering

The NCA of a two given octants (or coordinates) (see Fig. 2 & 3), is the octant which encloses both given octants and the closest to the given two contants (or coordinates).

The new NCA based Hilbert ordering computed as follows.

- Compute the Nearest Common Ancestor(NCA) of the two octants.
- Traverse the octree from root to NCA once in order to compute the rotation pattern inside the considering NCA
- Use the rotation pattern inside the NCA to determine the relative ordering of the given octants.

Improving the efficiency of Rotation Pattern Calculation

We have improved the rotation pattern calculation for given NCA octant by using the following properties of the Hilbert curve.

- If we fixed upon a initial Hilbert ordering *Hilbert curve has 4 unique rotation patterns* and *3D Hilbert curve has 24 unique rotation patterns*.
- We store each rotation pattern and all it's children patterns in hard coded array which can be used to figure out the child rotation pattern for a given octant. The storage required to store the rotation patterns is fixed (*In 2D we need* 4×4 *and in 3D we need* 24×8) and independent from the Hilbert Curve order.
- By this approach we can compute the ordering between P_1 and P_2 by $O(Depth(NCA(P_1, P_1)))$

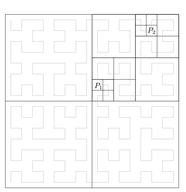

Figure 2: Refined Hilbert curve covering the points p_1 *and* p_2

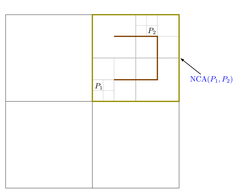

Figure 3: Ordering of points based on the Rotation pattern inside the NCA

Even though there are multiple SFCs available, we mainly focus on the Morton and Hilbert curves and their ordering computations. To compare the NCA based ordering approach with other approaches we have implemented both Hilbert and Morton as follows.

- Hilbert ordering baseline implementation (based on recursive approach)
- Morton ordering baseline implementation (based on non-recursive approach)
- Hilbert ordering NCA implementation (based on NCA approach)
- Morton ordering NCA implementation (based on NCA approach)

Allowing Flexibility in Dynamic Load Balancing

In this research we introduce some flexibility (slack) to the load balancing (i.e near uniform load balancing) in order to reduce the communication costs and reduce the overall execution time of the computation. To demonstrate that we conduct two experiments.

- Standard SFC based partition, where the work is uniform across all processes and compare the communication (using the surface area of the partition as a surrogate)
- A flexible SFC based partition, where we allow a small 'slack' (flexibility) in the amount of work that each process gets in order to reduce the communication costs.

Results

Following figures show (See Fig.5 & 4) the results of the four approaches that we implemented in order to compare the NCA based ordering Vs recursive based ordering. By considering the mean execution time (ms) along the varying depth, we can conclude that the NCA based Hilbert ordering is 9 times faster than the traditional recursive based approach. If we consider the NCA based Morton implementation mean execution time (ms) along the varying depth, we can see that it is 1.13 times slower than the Morton baseline implementation (implementation in dendro).

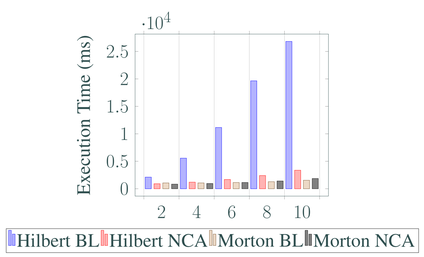

Figure 4: 2d ordering of points for Hilbert and Morton baseline, Hilbert and Morton NCA approaches.

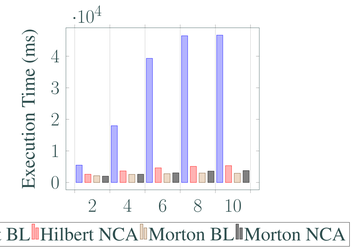

Figure 5: 3d ordering of points for Hilbert and Morton baseline, Hilbert and Morton NCA approaches.

In the second experiment we allow some flexibility to the load, and observe how contour ratio (ratio between the surface area of the mesh that each node gets with some flexibility, Vs 0 flexibility case) behaves with Hilbert and Morton ordering based partitioning. Our results show that (see Fig.6) by using the Hilbert ordering we can achieve low contour ratios compared with the Morton ordering which implies low communication costs.We can conclude that by using Hilbert ordering with some flexibility we can reduce communication costs further compared to the Morton ordering (even with some flexibility).

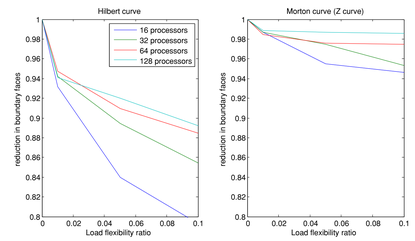

Figure 6: Flexible partitioning results using Hilbert and Morton ordering

Conclusions

- Hilbert ordering is better than Morton ordering in terms of preserving the geometric locality of objects.
- We can conclude that the NCA based Hilbert ordering algorithm is 9 times faster than the traditional recursive approach for Hilbert ordering.
- Allowing some flexibility to the load at each node can reduce the communication further (approximately in the range of 1.125-1.175 times), and reduce the overall computation time.

More information and animations can be found via this qr.