Hilbert Curve based Flexible Dynamic Partitioning Scheme for Adaptive Scientific Computations

Milinda Fernando, Hari Sundar

School of Computing, University of Utah

ABSTRACT

Space Filling Curves (SFC) are commonly used by the HPC community for partitioning data[3, 4, 6] and for resource allocations[2, 5]. Amongst the various SFCs, Hilbert curves have been found to be in general superior to the more ubiquitous Morton or Z-Curve [3]. However, their adoption in large-scale HPC codes, especially for partitioning applications, has not been as common as Morton curves due to the computational complexity associated with the Hilbert ordering. In this work, we present an alternative algorithm for computing the Hilbert ordering that can be implemented almost as efficiently as the Morton ordering. Additionally, being based on the concept of the *nearest common ancestor*—a fundamental property of all SFCs—the algorithm can be applied to all SFCs. We also present a small modification to the standard SFC based partitioning algorithm that allows us to obtain significant improvements in part ition quality while using Hilbert ordering.

1. INTRODUCTION

Load balancing and partitioning are critical when it comes to parallel computations. Generally partitioning involves the tasks of equally dividing the work and data among the processors, reducing processor idle time and the communication costs. In this research we are focused on developing an efficient flexible dynamic partitioning scheme, based on the Hilbert curve, targeted at adaptive-mesh computations.

As we march towards exascale machines, the cost of data movement and load-imbalances therein are a major bottleneck for achieving scalability. We consider an alternative SFC-based partitioning scheme where we allow some (user-specified) flexibility in the work assignment, so as to minizmize the data-dependencies across partitions. Effectively, we allow the flexibility in minimizing the communication load-imbalance at the cost of a marginal increase in work load-imbalance. The traditional SFC-based partitioning can be recovered by setting the flexibility to zero.

One of the main advantage of SFC based partitioning is the preservation of geometric locality of objects between processors. Depending on the SFC (i.e. Morton, Moore, Hilbert) that used for partitioning, the amount of locality preserved differs [1]. Most of the SFC based partitioning use the Morton curve which is good for current range of clusters in terms of giving good load balance and the efficiency of the implementation. But as we focus on larger levels of parallelism, we show that Hilbert is more effective. Recursive computation of Hilbert ordering can be inefficient, which can lead to low performance in overall computation. In this paper we present an approach based on Nearest Common Ancestor (NCA) which can be extended to calculate any SFC ordering efficiently. Considering the results we gathered we can show that the Hilbert ordering based on NCA, is 9 times faster than the traditional recursive approach.

2. NEAREST COMMON ANCESTOR (NCA) BASED ORDERING

SFC is a surjective mapping between the one dimensional space to higher dimensional space. Due to the recursive nature SFCs have, most of the SFCs can be computed recursively. In this paper we present an algorithm to compute Hilbert ordering based on the NCA which is 9 times faster than the recursive approach. The main advantage of the NCA based Hilbert order calculation is the extensibility of the algorithm to other SFCs. The key idea is to find the NCA for the given coordinates (see Fig. 1 ,Fig.**??** & Fig.**??**). Next we need to figure out the ordering within the NCA element. This might be fixed as in the case of Morton ordering, or might require a traversal from the root of the tree to the NCA, as in the case of Hilbert.

To compare the NCA based ordering approach with other approaches we have implemented both Hilbert and Morton ordering as follows: a) baseline (recursive) Hilbert implementation, b) baseline Morton implementation [6], c) new NCA-based Hilbert, and d) NCA-based Morton implementation for comparison.

2.1 Allowing flexibility in dynamic load balancing

As mentioned earlier most of the partitioning schemes are focused on uniform distribution of tasks across all the processor. The main drawback of this approach is, the uniform-work load balancing can cause imbalance in the communication load, slowing down the overall computation. In this paper we introduce some flexibility to the load balancing in order to reduce the overall communication costs. To demonstrate the above claim 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' (at most 10% of N/p) in the amount of work that each process gets in order to reduce the communication costs.

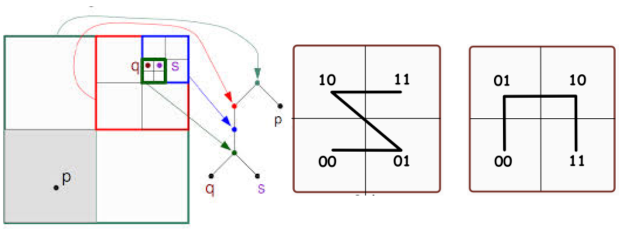

Figure 1: Nearest Common Ancestor (NCA) of an octree (In this senario, immediate parent node of nodes q and s is the NCA for nodes q and s)

3. RESULTS

Baseline and NCA based Hilbert and Morton ordering algorithms are executed with varying maximum depth for sorting one million (2D & 3D) points. Each implementation of the Hilbert and Morton ordering executed twenty times and mean executing time in milliseconds is taken as the performance measure. Fig.2 & Fig.3 depict the 2D and 3D Hilbert and Morton ordering performance for different implementations. All the SFC implementations are run in Intel Xeon E7-4820v2 CPU @2.00 GHz (32 cores) processor with 64 GB DDR3 RAM and using the g++ 4.8 compiler in the Ubuntu 14.04 LTS environment.

According to the Fig.2 & 3, we can see that the execution time of recursive approach resides in the range of 10s to 40s, while the execution time of the NCA approach is less than 3s (in 2D implementation) and 4.5s (in 3D implementation). In the average case, Morton NCA and Morton baseline performances are very close to each other. Our results suggest that the NCA based Hilbert ordering is 9 times faster (mean performance) than the traditional recursive Hilbert ordering and the NCA property can be used to calculate the ordering of a generic SFC efficiently.

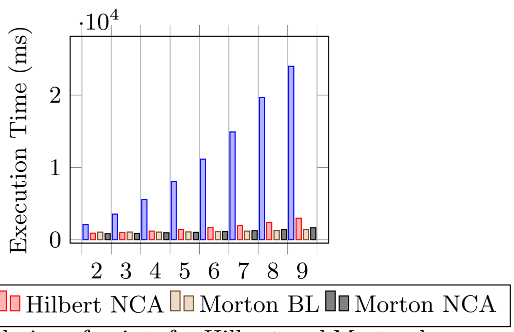

Figure 2: 2d ordering of points for Hilbert and Morton baseline, Hilbert and Morton NCA approaches.

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 flexibility, Vs

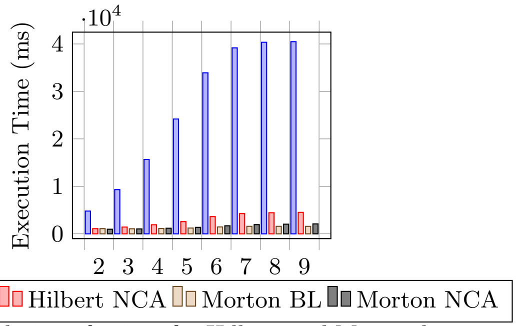

Figure 3: 3d ordering of points for Hilbert and Morton baseline, Hilbert and Morton NCA approaches.

0 flexibility case) behaves with Hilbert and Morton ordering based partitioning. Our results show that (see Fig.4) by using the Hilbert ordering we can achieve low contour ratios compared with the Morton ordering which implies low communication costs.

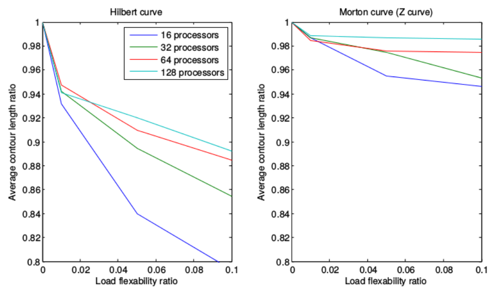

Figure 4: Flexible partitioning results using Hilbert and Morton ordering

Conclusions

In this work we presented a faster algorithm for computing the ordering of SFCs, specifically the Hilbert curve. We also demonstrated the improvement in partitioning quality by allowing for non-uniform assignment of work. Future work will focus on testing this for larger clusters and actual communication costs.

4. REFERENCES

[1] M. Bader. *Space-filling curves: an introduction with applications in scientific computing*, volume 9. Springer Science & Business Media, 2012.

[2] M. A. Bender. Compute process allocator (cpa). *Urbana*, 51:61801.

[3] P. M. Campbell, K. D. Devine, J. E. Flaherty, L. G. Gervasio, and J. D. Teresco. Dynamic octree load balancing using space-filling curves. *Williams College Department of Computer Science Technical Report CS-03*, 1:68, 2003.

[4] K. D. Devine, E. G. Boman, R. T. Heaphy, B. A. Hendrickson, J. D. Teresco, J. Faik, J. E. Flaherty, and L. G. Gervasio. New challenges in dynamic load balancing. *Applied Numerical Mathematics*, 52(2):133–152, 2005.

[5] L. Sliwko and V. Getov. Workload schedulers-genesis, algorithms and comparisons. *International Journal of*

Computer Science and Software Engineering, 4(6):141–155, 2015.

[6] H. Sundar, R. S. Sampath, S. S. Adavani, C. Davatzikos, and G. Biros. Low-constant parallel algorithms for finite element simulations using linear octrees. In *Proceedings of the 2007 ACM/IEEE Conference on Supercomputing*, SC '07, pages 25:1–25:12, New York, NY, USA, 2007. ACM.