Computing a Spatial Distance Histogram on a Quad-tree using the GPU

Brian Pinson, Karshan Arjun, Mark Tushemereirwe
University of South Florida, USA

Abstract. In this paper we explain our objectives in creating a Spatial Distance Histogram (SDH) query on a existing quad-tree based data structure by using Compute Unified Device Architecture (CUDA) techniques and methodologies, which utilizes the usage of the GPU instead of the CPU for the calculation of data in a molecular system (MS). The existing quad-tree is created with a random generator of points that were then placed in the nodes in the tree. The SDH that we created would gather the points inside of the quad-tree. Once the points were found, we put them in an atom and used a quadratic distance calculation formula to place the atoms into buckets that would output the SDH. We will discuss the objectives that were done on implementing the SDH query, the tree traversal process on the quad-tree and the testing of the quad-tree with the SDH query to find the satisfying results that would entail that the use of a SDH query would be beneficial when dealing with data that are in a particle based format.

Introduction. The ability to calculate the analytics of a particle base data systems can be extremely difficult if the correct methodologies were not applied. These systems called molecular systems are incredibly vast, systems such as stars, galaxies, or atoms are highly populated with points [2]. This data can be overwhelming to gather information from and the amount of data particles that are obtained through a MS must be done so by an efficient and well optimized approach in order to obtain efficient data analysis. The use of a SDH query will help ease the pain of the complexity of using a large number of particles in a data structure such as the quad-tree [1].
Quad-trees are an excellent data structure for holding large amount of points into nodes which have a root to child leaf structure to keep track of the data that is being processed through the data structure. The quad-tree being able to contain data on such a large scale as most MS are, can use a SDH query to obtain the optimal results [1]. Given the structure of a SDH query, the use of atoms are to be used for the calculation of the point to point distance of the vast number of particles and then is filled into buckets which will display a histogram of the analysis of the data. This implementation gives the SDH query the ability to create an organized way of observing data in a MS format.
Using the parallelism features of the GPU, the use of the CUDA architecture will greatly decrease the computation time on the calculations that are needed by the SDH. With utilizing the GPU's usage of 100s of cores, compared to the usage of the CPU's 4 to 8 cores makes utilizing the GPU with MS data a simple decision [4]. In this paper, we will discuss our objectives with utilizing the CUDA architecture to create the SDH query to compute the values given from a quad-tree with tree traversal, the testing of the SDH with different number of points, then share the results on how this algorithm is well used for the analysis of large based particle systems.

Implementation.
Our process to implement the SDH query on the quad-tree traversal was to first create a SDH query that would run on the CPU before we tackled the process on the GPU. We created the placeholders of a histogram object and an atom object. The atom object will hold the points which are then calculated by the point to point distance quadratic formula [2]. We then created a function that would perform the point to point calculation and then output the SDH query. We gathered the points that were created randomly and then store the points into the atom list object that will be used in the calculation of the distance

between the points. Once the distance calculation of the points were made, the points were then placed into the histogram object to be displayed. This was our implementation of the SDH query being created by the CPU, which had mediocre benchmarks calculation times that we knew would we could improve with the performance of using the GPU. So with this knowledge of knowing how to create the SDH query with the CPU, we started on the traversal process of the quad-tree to gather the points in the tree.
The process of learning how to traverse the quad-tree was to research ways that tree traversals were conducted on smaller trees such as binary trees and then apply the methods that were used. As we acquired this knowledge, we began to deconstruct the quad-tree to see where the insertion of the points were being placed in the nodes of the tree. Once we found out where the insertion of the points were gathered and how the quad-tree itself was created, the traversal process was made simpler to produce. We then found the data structures that stored the points of the quad-tree and the indexes of those points. The x, y coordinates of the points and the indexes of those points were the main data that was needed for the execution of the traversal.
A function called CUDA_Calculate_Histogram was created with using CUDA methods such threadIdx and blockIdx to improve the indexing computation time of accessing the data structures of the quad-tree [4]. In this function the tree traversal was performed, with using a depth first search to get all the nodes greater than the node that a given point resides in. The algorithm will then scan all the nodes, find the next node and the child nodes, keep searching right to find the remaining nodes. Once the index of the node is found, it is then inserted into the CUDA_distance_calculation, which uses the quadratic formula of the point to point distance. The distance value is then inserted into the histogram, with using the a CUDA function, atomicAdd. The end result is a SDH query that mimics the other SDH query that was made through the use of the CPU.

Testing and results.

Number of Points	Bucket Size	Number of Levels	Time Taken(ms)
10000	1000	10	40.20115
20000	2000	10	145.84467
30000	3000	10	300.42596
40000	4000	10	521.66589
50000	5000	10	823.43323

Conclusion.

With a large number of points of 50,000 can be easily be calculated under a minute with a GPU powered SDH query. The benchmark times for this new SDH query that was created is forty times faster than the one that was created by the CPU. Our results show that the SDH query created by the GPU is simply the better choice for calculating molecular systems.

References.

[1] A. Kumar, Y. Yuan, V. Grupcev, Y.-C. Tu and G. Shen, Distance Histogram Computation Based on Spatiotemporal Uniformity in Scientific Data URL: https://openproceedings.org/2012/conf/edbt/KumarGYTS12.pdf
[2] Y.-C. Tu, S. Chen and S.Pandit, Computing Spatial Distance Histograms Efficiently in Scientific Databases URL: https://pdfs.semanticscholar.org/61a7/a1b3bb0cade247ec9592118dc2bcc73c5ceb.pdf
[3] H. Haverkort, M. McGranaghan and L. Toma, An Edge Quadtree for External Memory URL: http://www.bowdoin.edu/~ltoma/papers/2013-sea-quad.pdf
[4] J. Ghorpade, J. Parande, M. Kulkarni, A. Bawaskar, GPGPU PROCESSING IN CUDA ARCHITECTURE, URL: https://arxiv.org/pdf/1202.4347.pdf