# VDBSCAN*: An efficient and effective spatial data mining algorithm using GPU

Guilherme Priólli Daniel
São Paulo State University
Department of Computer Science and Statistics
São José do Rio Preto, São Paulo, Brazil
gui.computacao@yahoo.com.br

Carlos Roberto Valêncio
São Paulo State University
Department of Computer Science and Statistics
São José do Rio Preto, São Paulo, Brazil
valencio@ibilce.unesp.br

Rodrigo Cleir Castellon Rodrigues
São Paulo State University
Department of Computer Science and Statistics
São José do Rio Preto, São Paulo, Brazil
rodrigocleir@hotmail.com

***Abstract*—Spatial Data Mining techniques enables the extraction of implicit knowledge in spatial databases. This information must be reliable because the techniques help in a decision-making process. However, sometimes the results are inadequate. Moreover, the computational cost to execute these algorithms is high. Thus, this paper proposes a new spatial clustering algorithm called VDBSCAN* that implementing new approaches to allow semantic aggregation in clustering and uses the GPU (Graphics Processing Unit) computing to optimize performance. Based on the results achieved it was found that the algorithm obtained an improvement in the clusters quality and 95% higher performance compared to the algorithm executed in CPU (Central Processing Unit).**

***Keywords*—GPU, Spatial Data Mining, Clustering, VDBSCAN*.**

## I. Introduction

There is an increasing trend of applications collecting spatial data, which are essential for decision-making process. Nevertheless, spatial data have greater complexity compared to conventional data and thus the traditional data mining techniques becomes inadequate. Consequently, spatial data mining techniques have emerged in order minimize such difficulties [1], [2], [3].

Spatial Data Mining is the process of discovering interesting and useful patterns previously unknown for large spatial datasets. The techniques of spatial data mining are used in different solutions, but the application of the algorithms in large databases is still a challenge [4], [5]. Thus, new algorithms have been proposed in order to make the knowledge-discovery from spatial databases more efficient and effective [3], [6].

Moreover, GPUs have been used in various fields for performance optimization, since modern GPUs offer a very large memory bandwidth and high computational power at low cost in comparison with other high-level systems. The graphics hardware allows implementing many parallel programs, making the GPU computing an attractive alternative to overcome problems performance of the Spatial Data Mining algorithms [1], [7], [8].

The objective of this work is to present the VDBSCAN* algorithm, a spatial clustering algorithm that considers the semantic characteristics of points, implements an approach of refinement and uses GPU computing to solve performance problem.

This paper is arranged as follows: in section 2 the bibliographical survey, in section 3 are presented the related work, in section 4 the developed work is explained, in section 5 the experiments and results are shown, and in section 6 the conclusions are presented.

## II. Bibliographical Survey

In this section, we present the basic concepts of spatial data and Spatial Data Mining techniques, the basis algorithms for VDBSCAN* and the concepts of GPU programming.

*1) Spatial Data Mining:* Spatial data refer to the spatial location expressing geographic entities from the real world and can be represented by points, lines, polygons and other geometric types describing spatial phenomena. Thus, the spatial objects are more complex than conventional objects (e.g., string, integer and real), and in relation to other data types contains temporal, multi-dimensional and volume characteristics [9], [10], [11].

The spatial data manipulation is extremely important for the knowledge discovery that is not possible to be found by conventional computing techniques. In this way, the concept of Spatial Data Mining has emerged that is the process of extracting implicit knowledge, interesting patterns, useful information and spatial relationships and non-spatial in large spatial databases [2], [12].

There are various spatial data mining techniques. One of them is clustering technique, which is a useful tool for the distribution patterns discovery in data sets. This technique aims to group the points so that the objects within the same group have similar characteristics, while objects in different groups are dissimilar [1], [2], [5], [13].

The cluster analysis is one of the first methods of knowledge discovery in spatial database; its advantage is that the clusters are obtained without the need for prior knowledge. There are five categories of clustering algorithms: partitioning, hierarchical, density, grid and model algorithms [13], [14], [15].

The algorithm presented in this paper uses the density-based method. This technique analyzes the density of objects in a particular region to find clusters of points and their neighbors [14], [15].

*2) GPU (Graphics Processing Unit):* GPU (Graphics Processing Unit) is a high-performance architecture specializing in image processing that, in recent years, has become an integral part of high performance systems. Modern GPUs implements many parallel algorithms for use directly in graphics hardware. Thus, algorithms that require a high computational cost, can utilize GPUs to achieve a speed increase [16], [17], [18].

This performance is attainable, because the GPU is capable of performing more floating-point calculations due to its architectural design that has more transistors devoted to data processing [17], as illustrated in Figure 1.

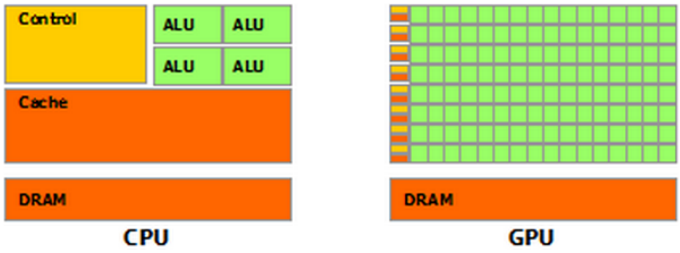


Fig. 1. Transistors in CPU and GPU [17].

This parallelism strategy is called GPGPU (General-Purpose Computation on Graphics Processing Unit) or GPU computing. The main solutions for GPU computing is OpenCL (Open Computing Language) and CUDA (Compute Unified Device Architecture) [16]. The algorithm proposed was developed in CUDA, because this platform is stable and widely used by researchers. Thus, CUDA is detailed below.

CUDA is a platform of data structure and parallel processing on GPU which was designed by NVIDIA, the communication is made through API (Application Programming Interface) that support graphics functions, mathematical functions, many libraries, runtime and driver [17], [19].

The CUDA programming model is composed of threads, blocks and grids. A thread is a program that can run independently or concurrently, and is the smallest unit of process. Groups of threads form a block and groups of blocks form a grid [17], [20]. The Figure 2 shows a grid example.

The functions in CUDA C defined by the programmer are called kernel. In contrast to common functions in C, these functions can be executed multiple times in parallel by different threads [17].

*3) Base Algorithms for VDBSCAN*:* DBSCAN (Density Based Spatial Clustering of Applications with Noise) [21] is a major density-based clustering algorithm and has always been

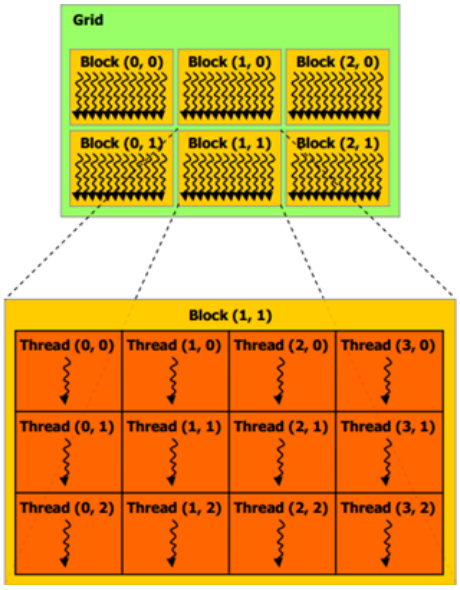


Fig. 2. Hierarchy of the CUDA threads [17].

the focus of research by scientific community to improve the clusters quality and the performance of the algorithm.

Thus, the algorithm VDBSCAN emerged in 2007 with the goal of removing the input parameter of DBSCAN, which is defined search radius, called Eps, in addition to the possibility of finding clusters of varying densities [15].

Thus, the VDBSCAN process is divided into two main steps: finding Eps values and finding clusters. In the first step, the algorithm set k-dist plot, which is a graph that contains the distance of the k-th nearest neighbor of all objects to be analyzed in ascending order, where k is the minimum number of points that a cluster must have. In each jump of values in the graph is defined as a search radius (Eps), as can be seen in Figure 3 [15].

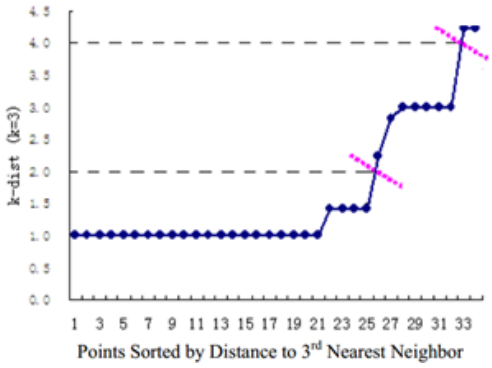


Fig. 3. Example chart k-dist [15].

In the second step of the algorithm, DBSCAN algorithm is executed for each value of Eps, which makes it possible clusters with different density [15].

To facilitate understanding, in Figure 4 we illustrate the operation of DBSCAN in clusters search, where the k is 3. The purple point is marked as core because within a circle of radius Eps it reached the number of objects satisfying the

minimum points required to form the cluster. The green point is marked as border, because a core point achieves it, but within a radius Eps, it does not have enough neighbors to be a core point. Finally, the red point is marked as noise because any point does not reach it [1], [15], [21].

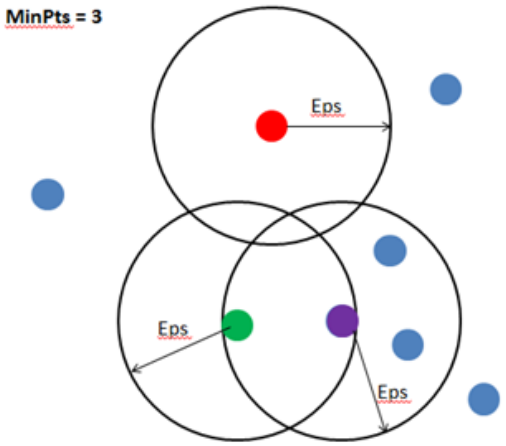


Fig. 4. DBSCAN behavior in the classification of points [1].

Although VDBSCAN return better results than the DBSCAN, it is not as efficient when applied to large databases. Thus, in 2013 VDBSCAN+ algorithm emerged, which is a version of VDBSCAN adapted for GPU computing. In this algorithm, the computation of k-dist plot, the marking of core and border points and the clusters formation and are made in parallel on GPU [1].

## III. Related Work

In the literature, several works are proposed in order to solve the performance problem of spatial data mining algorithms. Some papers that use GPU computing to optimize the runtime of the algorithms are presented.

In 2013, the G-DBSCAN algorithm presented a parallel implementation of DBSCAN on GPU. While there are other parallel versions of this algorithm, the G-DBSCAN is distinguished by simplicity of indexing the data through graphs and G-DBSCAN is up to 100 times faster than DBSCAN [22].

In the same year, the k-Means algorithm was implemented on GPU. This implementation defined two strategies: the first explores the GPU registers to reduce access latency to data sets of low dimension and the other simulated matrix multiplication and explored the GPU shared memory to achieve high rate calculation for memory access to data sets of high dimension. Thus, the results of this study was 3-8 times faster than the others K-means algorithms on GPU [23].

In 2011, another important algorithm was implemented on GPU, called WaveCluster [14]. This algorithm is divided into two sub-algorithms: the first is designated for calculating the extraction of low-frequency signal component by wavelet transform and the second is referred to the calculation of connected component labeling. The modified algorithm was 107 times faster in the first stage and 6 times faster in the second stage, compared to the original algorithm runs in CPU [4].

Finally, there is the VDBSCAN+ [1] that implements the VDBSCAN [15] on GPU in 2013. The algorithm divides the VDBSCAN+ in four parallel stages: the first one calculates the radius, the second one finds and marks the core points, the third one finds and marks the border points and the last one generates the clusters. Thus, the VDBSCAN+ can be 95% faster than VDBSCAN.

## IV. VDBSCAN*

VDBSCAN* is a spatial clustering algorithm which is based on VDBSCAN and was implemented in CUDA. To achieve best results were inserted new steps in the algorithm as well as an input parameter that includes a non-spatial feature in the cluster formation, the refinement technique to remove the noises points and the use of GPU computing to improve the runtime.

In Figure 5, shows the algorithm flowchart of the steps performed in the CPU and GPU.

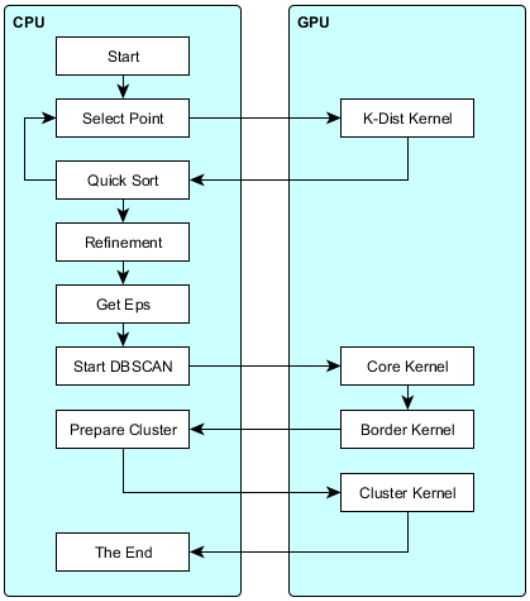


Fig. 5. VDBSCAN* flowchart.

In the initial phase, the points are loaded into memory and the analyst defines the similarity level that the algorithm will use to form clusters.

In the second step, a point is selected for distances comparison with other points. From there, K-Dist Kernel runs on GPU. In this method are measured the distances between the selected point and the others points and the result is stored in the distance vector.

The Quick Sort method sorts the distance vector and the k-th position value is added to the K-dist vector. In contrast, Methods Selects Point, K-Dist Kernel and Quick Sort are invoked for each data point. After these steps, the Quick Sort will sort K-dist vector and therefore the refinement will be performed.

In the refinement step is recalculated the density levels and performed the elimination of outliers, which can cause

distortion in the clusters. In Figure 6 shows the noises found in K-dist plot removed by refinement.

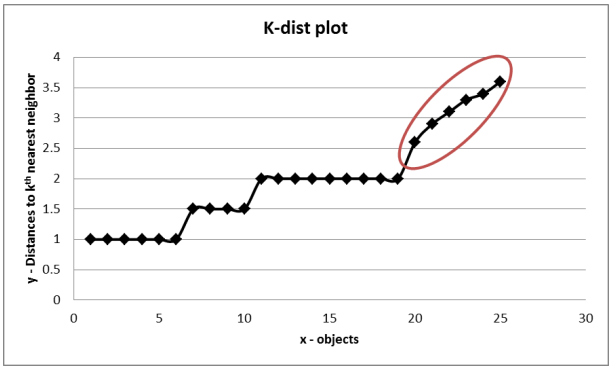


Fig. 6. Noises found in refinement step.

After the refinement stage, Eps are selected and DBSCAN runs for each value of Eps found, which allows finding the clusters of varying density.

For CUDA implementation, DBSCAN is programmed differently from the traditional. For this, for each step was added to the comparing technique for non-spatial similarity between the points.

The Core Kernel calculates the distance and the similarity of one point for each neighbor. If the neighbor is within the radius Eps and the non-spatial characteristic meet the similarity level, the number of neighbors is incremented. If the number of neighbors is greater than the minimum number of points required to form a cluster, is defined as the center point.

The Border Kernel finds the border points from a core point. If the neighbor point is not a core, is in the radius Eps and the observed characteristic satisfy the similarity conditions, the neighbor is marked as border point.

The Cluster Kernel runs on GPU to generate clusters. As in previous methods, this method checks the distances and the level of similarity between objects, and if the conditions are satisfied, the point is added to the cluster.

Finally, clusters found are saved in the database and are displayed for the analyst to check the clusters quality.

## V. Experiments and Results

For the experiments we used a real database of occupational accidents occurring in the region of São José do Rio Preto, São Paulo. The occupational accidents were collected by SIVAT - Sistema de Informação e Vigilância de Acidentes do Trabalho (Work Accident Vigilance System), developed by GBD - Grupo de Banco de Dados (Database Group) in partnership with CEREST - Centro de Referência de Saúde do Trabalhador (Worker's Health Reference Center) of São José do Rio Preto.

For the clusters formation the spatial attribute that corresponds to the geographical location of the accident was used and as non-spatial characteristic was used the attribute occupation, which is the occupation of the injured person. Thus, the cluster will be defined by people of the same occupation that injured nearby. The algorithms were run on a machine with CPU Intel(R) Core(TM) i5-4200U and a NVIDIA GeForce GTX 660 Ti.

### A. Experiment 1

The first experiment aimed to demonstrate the gain in semantic aggregation in clustering. Thus, initially was executed VDBSCAN* algorithm and after the VDBSCAN+ algorithm.

From the results obtained, it was possible to verify that the VDBSCAN* had better clusters than VDBSCAN+, because it used the geographic location of the object and a non-spatial feature to form the cluster. Thus, the objects that are close to the cluster and have the different non-spatial characteristic are discarded. The Figure 7 shows the clusters generated for both algorithms.

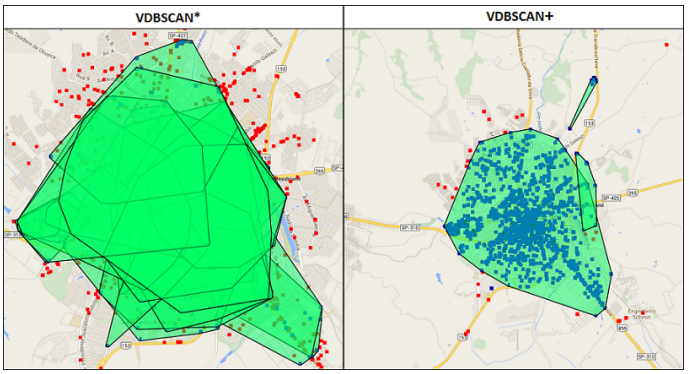


Fig. 7. Clusters generated by the algorithm VDBSCAN* and VDBSCAN+.

The numbers that summarize the information obtained in the first experiment shown in Figure 7 are as follows in Table I. Thereby, it is possible to note that the VDBSCAN* has filtered better the points that were part of the cluster, as just over 35% of the objects were part of a cluster. However, in the VDBSCAN+ about 99% of objects were part of a cluster. Moreover, the VDBSCAN+ has found fewer clusters, one of which occupied almost the entire city of São José do Rio Preto, which is not good to the analysis.

TABLE I. Results of VDBSCAN and VDBSCAN*.

| | VDBSCAN* | VDBSCAN+ |
|---|---|---|
| Analyzed points | 4589 | 4589 |
| Grouped points | 1620 | 4547 |
| Found clusters | 30 | 20 |

### B. Experiment 2

The experiment 2 was focused on proving that the refinement approach improves the clusters quality. Thus, initially VDBSCAN* algorithm was executed with the refinement technique enabled and subsequently with refinement technique disabled.

After analyzing the results, it was observed that the clusters found with refinement technique active had better quality since it eliminated the objects that are located at a greater distance from the dense region, that is, the noises points were removed. In Figure 8 displays an example of cluster benefited by refinement technique, where the left image is the

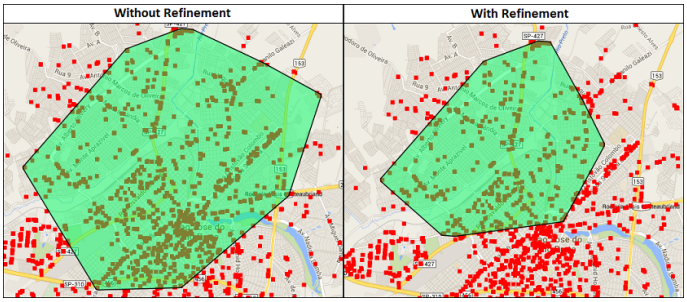


Fig. 8. Cluster found without refinement technique and cluster found with refinement technique.

cluster generated without refinement technique and right with refinement technique.

This is beneficial to the quality of spatial data mining results, since the location is a relevant factor and the cluster of a very distant object do not represent geographical similarity.

### C. Experiment 3

Finally, the last experiment the focus was the performance of the algorithm on GPU. For this, VDBSCAN* algorithm has been run for 4589 data points for both GPU and CPU, and the runtimes (in *ms*) were measured.

The results showed that the VDBSCAN* executed on CPU in 677155 ms and the algorithm executed on the GPU in 30477 ms, achieved an efficiency greater than 95%. In Figure 9, is presented the graph of runtimes of VDBSCAN* on GPU and CPU, where the performance improved by GPU is evident.

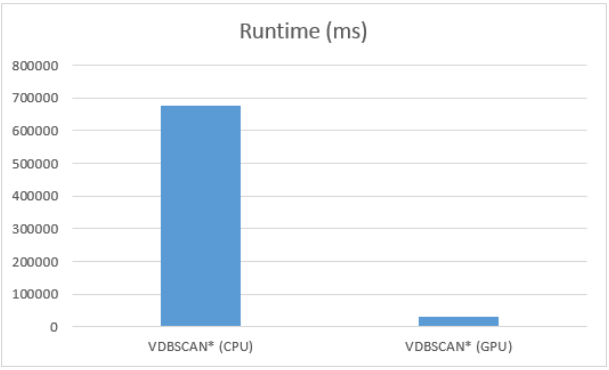


Fig. 9. Runtimes of VDBSCAN* in CPU and GPU.

## VI. Conclusions

The proposed algorithm was VDBSCAN-based and introduced a new approach to improving results in analyzing of large spatial databases, focusing in runtime optimization using GPU computing.

Through the experiments, it was confirmed that the approaches that were implemented enabled avoided the semantic loss in the clusters information and eliminated the noises points.

The resource use of GPU to optimize performance also represented a significant contribution, because spatial data mining using conventional techniques do not present results in adequate time.

## References

[1] C. R. Valencio, G. P. Daniel, C. A. D. Medeiros, A. M. Cansian, L. C. Baida, and F. Ferrari, "Vdbscan+: Performance optimization based on gpu parallelism," in *Parallel and Distributed Computing, Applications and Technologies (PDCAT), 2013 International Conference on*, Dec 2013, pp. 23–28.

[2] C. R. Valêncio, T. Kawabata, C. A. de Medeiros, R. C. G. de Souza, and J. M. Machado, "3d geovisualisation techniques applied in spatial data mining," in *Machine Learning and Data Mining in Pattern Recognition*. Springer, 2013, pp. 57–68.

[3] W. Tang and W. Feng, "Parallel map projection of vector-based big spatial data: Coupling cloud computing with graphics processing units," *Comput. Environ. Urban Syst.*, Feb. 2014.

[4] A. A. Yıldırım and C. Özdoğan, "Parallel wavelet-based clustering algorithm on gpus using cuda," *Procedia Computer Science*, vol. 3, pp. 396–400, 2011.

[5] C. R. Valêncio, C. de Medeiros, F. Ichiba, and R. C. G. de Souza, "Spatial clustering applied to health area," in *Parallel and Distributed Computing, Applications and Technologies (PDCAT), 2011 12th International Conference on*. IEEE, 2011, pp. 427–432.

[6] R. Thapa, C. Trefftz, and G. Wolffe, "Memory-efficient implementation of a graphics processor-based cluster detection algorithm for large spatial databases," *Electro/Information Technol. ( ...*, vol. 49401, 2010.

[7] D. Luebke and G. Humphreys, "How gpus work," *IEEE Comput.*, 2007.

[8] D. Coleman and D. Feldman, "Porting existing radiation code for GPU acceleration," *Sel. Top. Appl. Earth ...*, vol. 6, no. 6, pp. 2486–2491, 2013.

[9] M. Hemalatha and N. Saranya, "A Recent Survey on Knowledge Discovery in Spatial Data Mining," *Int. J. Comput. Sci. ...*, vol. 8, no. 3, pp. 473–479, 2011.

[10] Y. Zhong, J. Han, T. Zhang, Z. Li, J. Fang, and G. Chen, "Towards Parallel Spatial Query Processing for Big Spatial Data," *2012 IEEE 26th Int. Parallel Distrib. Process. Symp. Work. PhD Forum*, pp. 2085–2094, May 2012.

[11] W. Shuliang, D. Gangyi, and Z. Ming, "Big spatial data mining," *2013 IEEE Int. Conf. Big Data*, pp. 13–21, Oct. 2013.

[12] W. Jinlin, C. Xi, Z. Kefa, Z. Haibo, and W. Wei, "Research of GIS-based Spatial Data Mining Model." *WKDD*, pp. 159–162, Jan. 2009.

[13] Y. Kim, K. Shim, M.-S. Kim, and J. Sup Lee, "DBCURE-MR: An efficient density-based clustering algorithm for large data using MapReduce," *Inf. Syst.*, vol. 42, pp. 15–35, Jun. 2014.

[14] G. Sheikholeslami, "Wavecluster: A multi-resolution clustering approach for very large spatial databases," *VLDB*, 1998.

[15] P. Liu, D. Zhou, and N. Wu, "VDBSCAN: varied density based spatial clustering of applications with noise," *Serv. Syst. Serv. ...*, pp. 1–4, Jun. 2007.

[16] J. Owens, M. Houston, D. Luebke, S. Green, J. Stone, and J. Phillips, "Gpu computing," *Proceedings of the IEEE*, vol. 96, no. 5, pp. 879–899, May 2008.

[17] "NVIDIA CUDA Getting Started Guide," 2014. [Online]. Available: http://docs.nvidia.com/cuda/pdf/CUDA_C_Programming_Guide.pdf

[18] R. Cook, E. Dube, I. Lee, L. Nau, C. Shereda, and F. Wang, "Survey of novel programmingmodels for parallelizing applications at exascale," Technical report, Lawrence Livermore National Laboratory, 2011. 24, Tech. Rep., 2011.

[19] Y. Zhao, Z. Huang, B. Chen, Y. Fang, M. Yan, and Z. Yang, "Local acceleration in Distributed Geographic Information Processing with CUDA," *2010 18th Int. Conf. Geoinformatics*, pp. 1–6, Jun. 2010.

[20] S. Dashora and N. Khare, "Implementation of graph algorithms over GPU: A comparative analysis," *2012 IEEE Students' Conf. Electr. Electron. Comput. Sci.*, pp. 1–8, Mar. 2012.

[21] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise." *Kdd*, 1996. [Online]. Available: http://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf

[22] G. Andrade, G. Ramos, D. Madeira, R. Sachetto, R. Ferreira, and L. Rocha, "G-DBSCAN: A GPU Accelerated Algorithm for Density-based Clustering," *Procedia Comput. Sci.*, vol. 18, pp. 369–378, Jan. 2013.

[23] Y. Li, K. Zhao, X. Chu, and J. Liu, "Speeding up k-Means algorithm by GPUs," *Journal of Computer and System Sciences*, vol. 79, no. 2, pp. 216–229, Mar. 2013.