Proceeding of the International Seminar and Conference 2015: The Golden Triangle (Indonesia-India-Tiongkok)
Interrelations in Religion, Science, Culture, and Economic.
University of Wahid Hasyim, Semarang, Indonesia. August 28-30, 2015
Paper No. B.6

Data Reduction Based on GPU

Khoirudin
Department of Computer Applied Technology, Nanchang University, China
999 Xuefu Road, Honggutan New District, Nanchang, Jiangxi Province, P.R. China

Telp:+86-791-83969099 Fax:+86-791-83969069
mailipunrudin@gmail.com

Abstract-Graphics Processors Units (GPU) is becoming more popular among other application developers as data parallel coprocessors. In addition, general-purpose programming for graphics processor's research is growing very fast, which is basically for the GPU graphics computation, is now used not only for the purposes of computation the graphics but also for many applications. Moreover, the GPU becomes cheaper and has high computation. Data Reduction is a key problem in the rough set theory and its application. Rough set is one many method of the most useful data mining techniques. This paper focuses on a Data Reduction (finds minimal sets of data) to process by using GPU and CUDA programming. In this paper, two major processes have to be solved, first is a data reduction process (find minimal data set) using rough set theory, and the second is a process Data Reduction on the GPU by using the CUDA programming. By using shared memory and thread block on the GPU, the data reduction process is improved faster and more efficient. The results of the experiment prove that the computing performance by using GPU is faster and more efficient than using the CPU.

Keyword: *GPU, CUDA, Rough Set, Data Reduction*

1. Introduction

Data reduction is one of the sub materials from rough set theory. Rough set is a mathematical theory. This theory was presented by Pawlak for the first time in 1982. This technique is used for solving uncertainty, imprecision, and vagueness Artificial Intelligence applications.[6] It is also an efficient theory for Knowledge Discovery Database (KDD) process on the Data mining. Generally, this theory is used in many knowledge applications, such as medicine, pharmacology, business, banking, engineering design, image processing, and decision analysis.[7]

Graphic Processing Unit (GPU) is a special process to suspend main processor of a machine in graphic computing. In early 1990, 3D games with rendering processor were appearing, since then 3D accelerator hardware was made. API Open GL was firstly from a graphic application of professional workstation and was adapted for making a graphic 3D game programming, as well as the emergence of the DirectX and Direct3D. With the encouragement of the business gaming on the PC, The GPU was becoming more affordable and more powerful than ever and the development of GPU was faster than the CPU development.[5]

Table 1. Comparison between CPU and GPU

CPU	GPU
Parallelism through time multiplexing	Parallelism through space multiplexing
Emphasis on low memory latency	Emphasis on high memory throughput
Allows wide rangeof control flows + control flow optimization	Very control flow restricted
Optimized for low latency access to caches data set	Optimized for data parallel, throughput computation
Peak computation capability low	Higher peak computation capability
Handle sequential code well	Requires massively parallel computing
CPU are great for task parallelism	GPU are great for data parallelism

ISBN 978-602-8273-53-4

2. Literature Review

David Roger, Ulf Assarsson and Nicolas Halzschuch,[21] they present a new algorithm for stream reduction, which is an essential step of many GPGPU applications. They are algorithm works by a hierarchical approach: they divide the input stream into smaller blocks, perform a fast stream reduction pass on these smaller blocks, and concatenate the results. They thus achieve a new order of asymptotic complexity (O (n) whereas previous methods where O (n log n)) and an impressive speed up. They are an algorithm even outperforms doing streams reduction using the geometry shaders.

Qing Kuichen, Li Xiao and Song Linzhuang[14] they present a new data reduction approach over the stream processor architecture. They are goal research provides a binary reduction algorithm based on three simple reduction algorithms according to the CUDA parallel computing model. It launches a large number of thread blocks to reduce data, and every thread block reduces two data. The experiments show that the performance of it is better than another simple reduction algorithm, and it has very scalability.

Durgesh srivastaka and Shweta bhalothia[15] used decision matrix for rule reduction, and they are goal research is an efficient rule reduction approach has been presented to reduce the rules by using Decision Matrix. They applied this approach to Dengue Data Set which has 25 objects. Out of 25 objects we took 20 objects as training data and 5 objects as testing data. Before applying this approach we have 11 rules, but after applying this approach they have 6 rules. They reduced 5 rules by applying this approach. So this approach becomes an effective approach.

Shuang Wang present algorithms for solving the reducts problem in rough sets, and they are goal is Finding minimal reducts for the decision table is an NP-hard problem. The Boolean reasoning function, an accurate method, can find the optimal solution. Therefore, its difficulty is non polynomial. The genetic algorithm can be used as a heuristic for finding minimal reducts, the computation of the number of covered rows of the distinction table is time consuming.[17]

3. GPU

GPU refer to Graphics Processing Unit and is a single chip processor used for 3D application. GPU functionality has traditionally been very limited. In fact, since long time ago the GPU just used to accelerate certain parts of the graphics pipeline.[4,8] The GPU is limited to independent vertices and fragments on the processing capacity. However, this processing can be improved in parallel using the multiple cores which is away now to available to the GPU. This is more effective when the programmer wants to process a lot of vertices or fragments in the similar way [9].

4. CUDA

At November 2006, NVDIA introduce CUDA, a general purpose computing platform and programming model that leverages the parallel compute engine in NVDIA GPUs to solve many complex computational problems in a more efficient way than on a CPU[3,5].Driven by the insatiable market demand for real-time, high-definition 3D graphics, the programmable Graphic Processing Unit or GPU has evolved into a highly parallel, multithread, many core processors with tremendous computational horsepower and very high memory bandwidth [3,4], as illustration on figure 1.

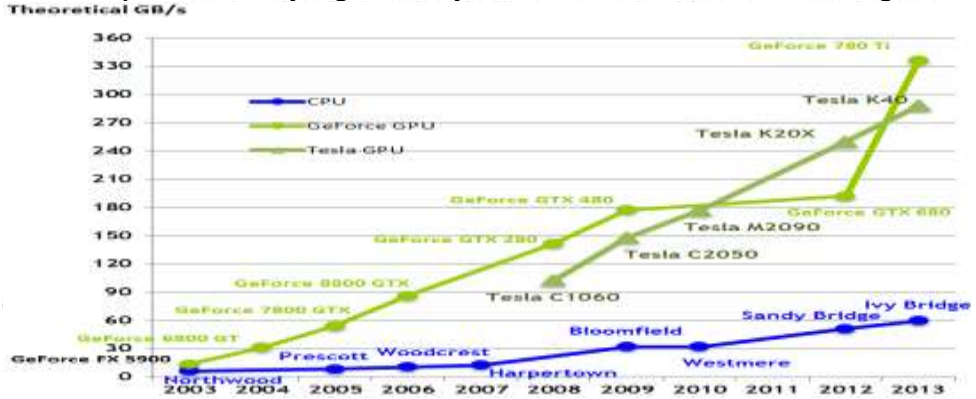

Figure 1. Memory Bandwidth of GPU and CPU

4.1 Memory Hierarchy

CUDA threads may access data from multiple memory spaces during their execution as illustrated by the figure below. Each thread has private local memory. Each thread block has shared memory visible to every thread block and with the same lifetime as the block. All threads have access to the same global memory.[13]

There are also two additional read-only memory spaces accessible by all threads; the texture and constant memory spaces. The global, constant and texture memory spaces optimize for different memory usage. Texture memory also different also offers different addressing modes, as well as data filtering, for some specific data formats. The constant memory, global, and texture memory space is persistent across kernel launches by the same application.

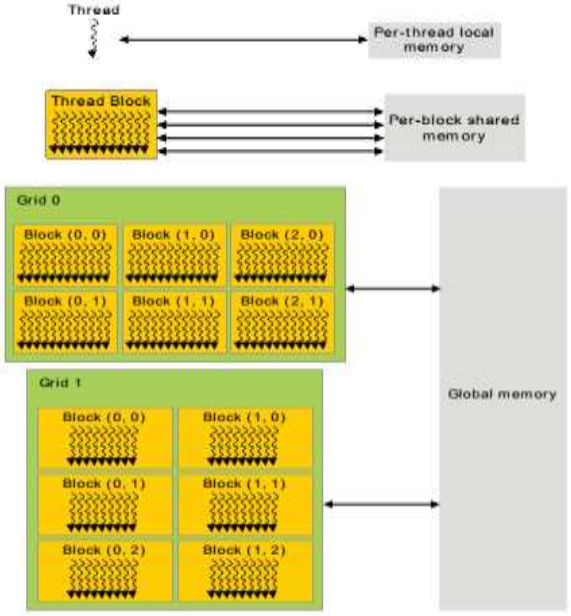

Figure 2 Memory Hierarchy

3.2 CUDA Memory

CUDA support several kinds of memory that could be used by programmers to reach high CGMA (Compute to Global Memory Access) ratios and thus great execution speeds in their kernels. Figure 3 shows this CUDA device memory. In the section of the figure 3, we can see the constant memory and global memory. These many types of memory could be read (R) and written (W) by the host, that using API functions. Register and shared memory are GPU on-chip memories. This Variable that resides in these memory types that access able at very high speed in the high parallel manner.[4,13]

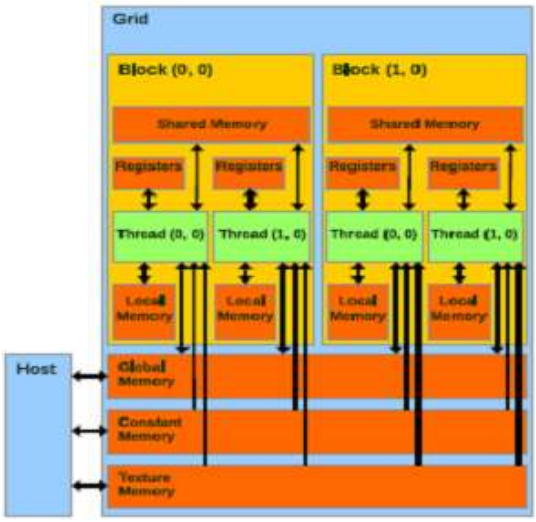

Figure 3 CUDA memory model

3.3 CUDA Architecture

GPU is a massively parallel architecture; many problems and the task can be efficiently solved by using GPU computing. The GPU has been big among of arithmetic capability. They increase the amount of programmability in the pipeline. [11]

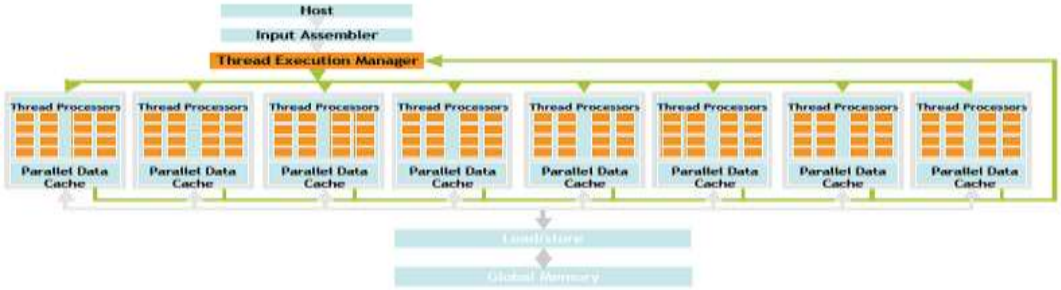

Figure 4 GPU architecture

5. Data Reduction Process

In this chapter, provided some data of patients with dengue indication. By using rough set analysis to perform data reduction process to reduce redundant data and assist the doctor in the diagnosis by developing some rules. Table 2 below, shows some data of patients with dengue indication.

Table 2 Data Patients

Patient	Conditional Attributes			Decision Attribute
	Muscular pain articulation	Blotched red skin	Temperature	Dengue
U1	No	No	Normal	No
U2	Yes	Yes	Very High	Yes
U3	No	No	Very High	Yes
U4	No	Yes	Normal	No
U5	Yes	No	Normal	No
U6	Yes	No	High	Yes
U7	Yes	No	High	Yes

U8	Yes	No	Very High	Yes
U9	Yes	Yes	High	Yes
U10	Yes	Yes	Very High	Yes
U11	No	No	High	No
U12	No	Yes	Very High	Yes
U13	No	Yes	High	No
U14	No	Yes	Normal	No
U15	Yes	No	Very High	No
U16	No	Yes	Normal	No
U17	Yes	No	Normal	No
U18	Yes	Yes	Normal	No
U19	No	No	High	No
U20	No	Yes	High	No

5.1 Data Reduction of Data Patients

A reducer is a set of significant minimum data, begin the basic proprietors of information table are maintained. Consequently, the reduce should have the capacity to classify the information, without change the form of representing the information.

The reduction information process is shown in the table 2;.

a. Verification of Information Data
b. Verification of equivalent information

Table 3 Reduction information of table 2

Patient	Conditional Attributes			Decision Attribute
	Muscular pain articulation	Blotched red skin	Temperature	Dengue
U1	No	No	Normal	No
U2	Yes	Yes	Very High	Yes
U3	No	No	Very High	Yes
U4	No	Yes	Normal	No
U5	Yes	No	Normal	No
U6	Yes	No	High	Yes
U8	Yes	No	Very High	Yes
U9	Yes	Yes	High	Yes
U11	No	No	High	No
U12	No	Yes	Very High	Yes
U13	No	Yes	High	No
U15	Yes	No	Very High	No
U18	Yes	Yes	Normal	No

c. The analysis of condition attribute in table 3, it can be observe that the similar data exists in measure tables.

Table 4, Table result of information reduce of table 3

Patient	Conditional Attributes			Decision Attribute
	Muscular pain articulation	Blotched red skin	Temperature	Dengue
U1	No	No	Normal	No
U3	No	No	Very High	Yes
U5	Yes	Yes	Normal	No
U6	Yes	Yes	High	Yes
U11	No	No	High	No
U15	Yes	Yes	Very High	No

6. Reduction on GPU

6.1 The Shuffle Down

Efficient reductions are the changing data between threads in the similar thread block. On the GPU this means using by shared memory, which entangle input the data to the shared memory, synchronizing and then send back the data again from the shared memory. On the shuffle instruction made able a thread to straight read the register from other thread within similar warp (64 Threads).

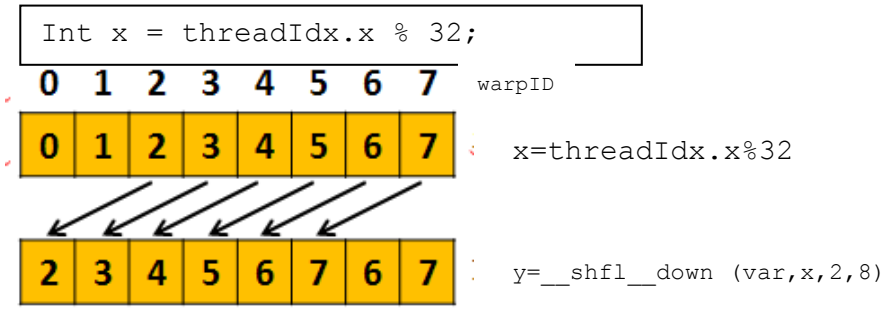

Figure 5 The instruction of shuffle down

6.2 Reduction of Shuffle Warp

Figure 6 here provide we used shuffle down to create a reduction tree. We have just only included the marks for the threads that are contributing to the last reduction. In fact every thread will be shifting values although there's no need in the reduction.

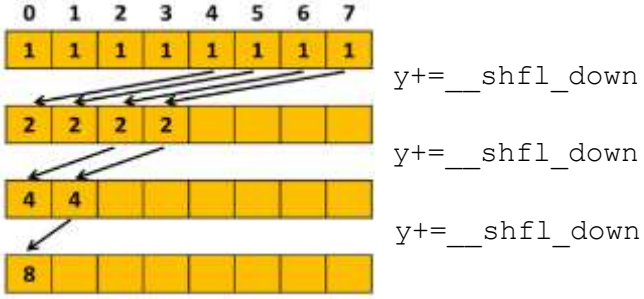

Figure 6 The algorithm reduction by using shuffle down

7. Experiment Evaluations

The experimental results for Data reduction using GPU using CUDA architecture are present here. In this section, we experiment tries to perform the reduction process of patient data in Table 2on the chapter 3, the authors conducted an experiment 5 data with different volumes, namely (160, 320, 640, 1280 and 2560) and the results are as shown figure below.

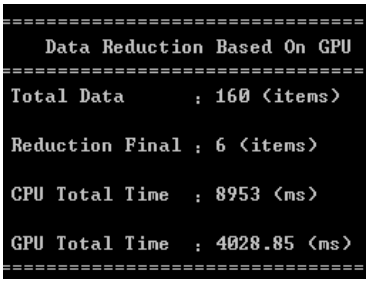

Figure 7 Reduction 160 data.

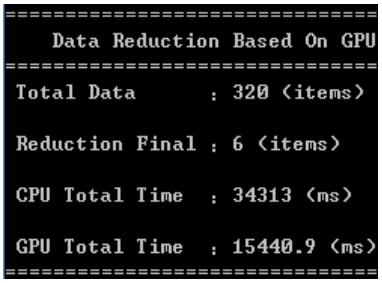

Figure 8 Reduction 320 data.

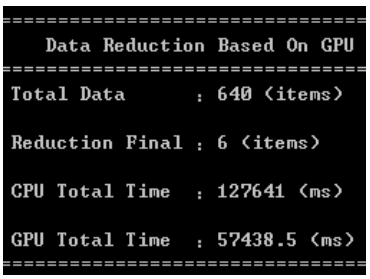

Figure 9 Reduction 640 data.

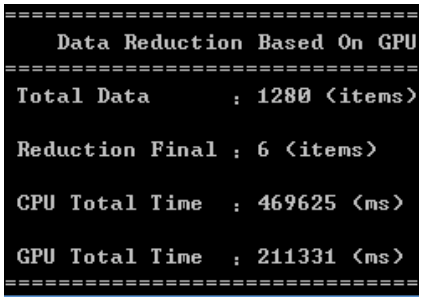

Figure 11 Reduction 2560 data.

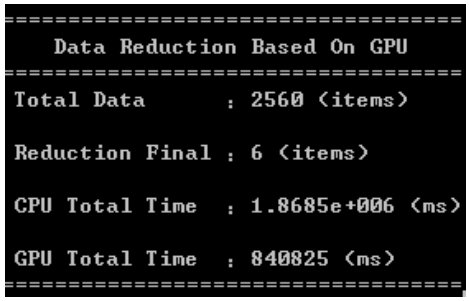

Figure 10 Reduction 1280 data.

Based on the figure 7, 8, 9, 10 and 11 on the experiment of 5 (160, 320, 640, 1280 and 2560) data reduction, we can obtain the result that the use memory space and thread block on GPU to process reduction data with the rough set algorithms is more efficient than using the CPU. For more clearly we can see the results in the table 5 and figure 12 below.

Table 5 Comparison process between CPU and GPU

Data	CPU (ms)	GPU (ms)	Comparison (ms)
160	8953	4029	4924
320	34313	15441	18872
640	127641	57438	70203
1280	469625	211331	258294
2560	1868506	840825	1027681

In Table 5 above shows, the results of experiments of 5 patient data with different amounts are (160, 320, 640, 1280 and 2560). The results showed that the reduction of data by using GPU a better than on the CPU, it can be seen also in the figure graphic 10 below;

ISBN 978-602-8273-53-4

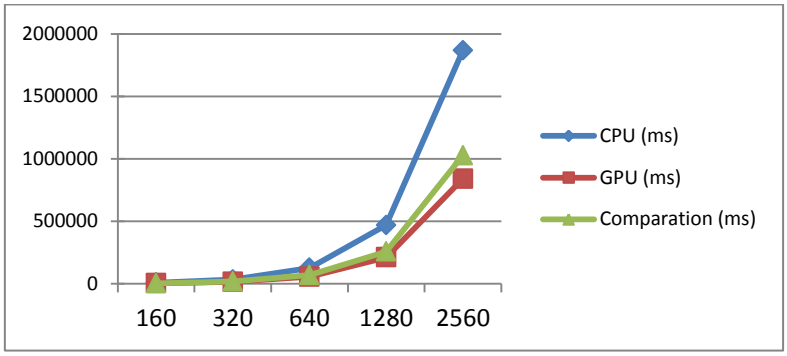

Figure 12 Graphic Comparison Data reduction between CPU and GPU

As shown in Figure 10, we can see that the data reduction using GPU and utilize rough set algorithms can improve the process of data reduction, it is because the GPU proved faster and more efficient in the process of data reduction when compared to the CPU, especially for data processing in a very large number

8. Conclusions

CUDA makes various hardware spaces available to the programmer. It is important that the CUDA programmer used the available memory space with the three orders of magnitude not similar in bandwidth between the kinds of CUDA memory types. A failure using of memory can result in lower performance. CUDA defines shared memory, register, and constant memory that can be accessed at maximum speed and in the parallel manner than global memory. Using memory effectively seems like require redesign of the algorithms, a popular technical strategy to maximize the locality of data access and enable effective use of shared memory.

By utilizing shared memory and thread block on the GPU, we can improve the data reduction process faster and more efficient. And the experiment result showed that performance using by GPU produces a faster and more efficient than using the CPU. Based on the table 5 and figure 10, it can be concluded that the process of data reduction of large size, the use of GPU proved to be much more efficient than the CPU in the data reduction.

References

[1]. Boma Anantasyaadhi, “Implementation General Purpose GPU to Process Singular Value Decompotition on Simple-O[D]” Indonesia University.2010.

[2]. Calle Ledjfors , “High Level GPU Programming[D]”, Department of Computer Science Lund University. 2008.

[3]. Jason Sanders, Edward Kandrot, “Cuda by Example[M]” Nvdia.

[4]. Rob Farber, “CUDA Application Design And Development[M]”, NVDIA.

[5]. vanden Boer, Dirk. "General Purpose Computing on GPU's.[D]" (2005).

[6]. Z. Pawlak: Rough sets, International Journal of Computer and Information Sciences[J], 11, 341-356, 1982.

[7]. Richart Janshen, Qiang Shen. Data Reduction with Rough Sets[J]. The University of Wales, Aberystwyth.

[8]. Rouh Set Theory-Fundamental Concepts, Principals, Data Extraction, and applications[J]. Silvio Rissiano; Germano Lambert- Torres.Federal University of Rondonia, Itajuba Federal University Brazil.

[9]. Zbigniew Suraj. An Introduction to Rough Set theory and Its Application[J]. University of Information Technology and Management, H. Sucharskiego Str. 2, 35-225 Rzeszów, Poland.

[10]. Harris, Mark. "Gpgpu: General-purpose computation on gpus." SIGGRAPH 2005 GPGPU COURSE (2005).

[11]. Ghorpade, Jayshree, et al. "GPGPU processing in CUDA architecture[J]." arXiv preprint arXiv:1202.4347 (2012).

[12]. Harris, Mark. "Optimizing parallel reduction in CUDA." NVIDIA Developer Technology 2.4 (2007).

[13]. “CUDA C programming guide version 6.5”, NVDIA Corporation, August 2014.

[14]. Chen, Qingkui, Li Xiao, and Songlin Zhuang. "A new data reduction approach over the stream processor architecture[J]." *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 IEEE 26th International*. IEEE, 2012.

[15]. Srivastava, Durgesh, and Shweta Bhalothia. "Rules Reduction Using Decision Matrix[J]."

[16]. Roger, David, Ulf Assarsson, and Nicolas Holzschuch. "Efficient stream reduction on the GPU[J]." Workshop on General Purpose Processing on Graphics Processing Units. 2007.

[17]. Wang, Shuang. "Algorithms for solving the reducts problem in rough sets[D]." (2012).

[18]. 李建明. 基于GPU加速的实时虚拟鱼系统[D]. 大连理工大学, 2007. DOI:10.7666/d.y1193428..

[19]. 傅轶娜. 基于MapReduce和遗传算法的粗糙集属性约简研究[D]. 安徽大学, 2014.

[20]. Kirk, David. "NVIDIA CUDA software and GPU parallel computing architecture."ISMM. Vol. 7. 2007.

[21]. 江顺亮,黄强强,董添文,徐少平. 基于CUDA的离散粒子系统模拟仿真及其实现. 南昌大学学报（工科版）. 33(3) 2011.9。

[22]. An Approximate Approach to Attribute Reduction [J]. XiaoPing Bei, YuanZheng Wang, Department of Computer science, Huazhong University of Science and Technology, Wuhan, Hubei, China.2006

[23]. Rough Set Data Analysis: A Road to non-invasive Knowledge discovery[J]. Ivo Duntsch, Gunther Gediga. german. 2000.

[24]. Jensen, R., & Shen, Q. (2004). Semantics-Preserving Dimensionality Reduction: Rough and Fuzzy-Rough Based Approaches. IEEE Transactions on Knowledge and Data Engineering, 16(12), pp.1457-1471.

[25]. Pawlak, Z. (1991). Rough Sets: Theoretical Aspects of Reasoning about Data[M]. Kluwer Academic Publishing, Dordrecht.

[26]. 周勇. 基于并行计算的数据流处理方法研究[D]. 大连理工大学, 2013.

[27]. Garland, Michael, et al. "Parallel Computing in CUDA[C]." IEEE micro 28.4 (2008): 13-27.

[28]. 黄强强,江顺亮,徐少平,董添文. The Implement of Discrete Particle Systems based on CUDA[C]. The 2011 International Conference on Business Computing and Global Informatization. 2011.7
[29]. 刘磊，冯前进，王文辉等. 基于GPU的医学图像快速面绘制[J]. 计算机工程与应用，2007, 43(32):189-191. DOI:10.3321/j.issn:1002-8331.2007.32.056.
[30]. 江顺亮, 项阳刚, 徐少平. 粒子系统模拟的单双GPU加速技术及其实现[J]. 南昌大学学报：工科版, 2014, 36(1):90-96.