ATLANTIS PRESS

Advances in Computer Science Research, volume 70

2nd International Conference on Mechatronics Engineering and Information Technology (ICMEIT 2017)

A Scheduling Strategy of Naive Bayesian image classification on HSA Platform

Nan Xiao

School of Beijing University of Technology, Beijing 100124, China

S201407090@emails.bjut.edu.cn

Keywords: Heterogeneous computing; HSA; Naive Bayesian image classification; Scheduling Strategy

Abstract. As the heterogeneous system based on CPU-GPU architecture has a strong performance advantages, so that it has been widely used in many fields especially image recognition, in order to enhance the transmission efficiency on HSA (Heterogeneous System Architecture) Platform. In this paper, we design a naive Bayesian image classification algorithm on the APU, which supports the HSA standard, and design a set of resource dynamic scheduling strategy for the eigenvalue extraction of the algorithm. The experimental results show that this strategy can save 29.56% execution time on average.

1. Introduction

With the development and innovation of GPU(Graphics Processing Unit)'s manufacturing process and related technologies, some of applications that are executed on the single CPU (Central Process Unit) [1] are gradually ported to the heterogeneous computing platform of CPU-GPU architecture. This collaborative approach has shown many better execution performance in some computing fields, such as the high-level quantity of floating-point operation, the image processing algorithm and so on. And this model has gradually become the focus of research in the field of heterogeneous computing.

APU architecture is an innovation of the heterogeneous computing model of CPU-GPU architecture, which integrates CPU with GPU on a single chip eliminating the need for interaction between the transmission tools such as PCIe bus and so on. [2] This structure also led to the new technology innovation. HSA is one of the key technologies, HSA is a heterogeneous system standard proposed by the HSA Foundation, which is under the advocacy of AMD in 2012[3]. The purpose of the HSA framework is to optimize the data exchange between CPU and other computing devices such as GPU, so as to improve the performance of the algorithm and to facilitate the use of developers. In addition, HSA covers two aspects of hardware and software, providing a heterogeneous storage mechanism for the APU platform, reducing the interaction hinder between various types of computing devices. Currently, the HSA framework has been used in APU Kaveri and Carrizo. HSA contains two core technologies, hUMA and hQ. The naive Bayesian image classification algorithm [4] is one of the typical image classification algorithms, which has the universality and stability which makes it have good fit on the HSA platform. However, the study of the performance of HSA is not comprehensive, the research field is not deep enough. The algorithm does not show good execution efficiency on the platform. It still has great development potential. Therefore, we research the algorithm on the HSA platform, and we design a dynamic scheduling strategy about the parallel operation of different devices. The experimental results show that the algorithm can play better performance in the HSA structure. The related results will be shown in detail later.

This paper is organized as follows. In Section 2, we introduce the HSA structure and key technologies. We give the specific process of the Bayesian image classification algorithm and the optimization scheme in Section 3. Next, experiment results and analysis are presented in Section 4. Finally, we draw some relevant conclusions in Section 5.

2. Related Works

The HSA standard was proposed by the HSA Foundation in 2012. And the first set of HSA technical specifications for practice was also proposed in 2015. The purpose of the HSA standard is to better cooperate with CPU and GPU. It has a very good fit between HSA design structure and AMD's APU product concept. In order to improve the overall performance of loads, HSA performs the most rational allocation of computing resources. HSA implements the true chip-level integration from bottom to top. The HSA has two key technologies, hUMA and hQ.

hUMA changes the mode of data isolation between CPU and GPU, achieves the shared storage between devices, one of the key features is the Shared Coherent Virtual Memory(SCVM) architecture. The SCVM includes a single virtual memory accessed by all the compute units, allowing them to use the same data through the same addresses. It means that when CPU assign tasks to GPU, it only needs to pass the relevant pointer without the need for a large amount of data transmission, at the same time, CPU can directly access the results of GPU processing in SCVM. This mechanism can reduce unnecessary overhead significantly. Not only that, but hUMA calls for cache coherency between all the compute units in order to solve the problem that each compute unit or group of compute units may still have independent cache structures.

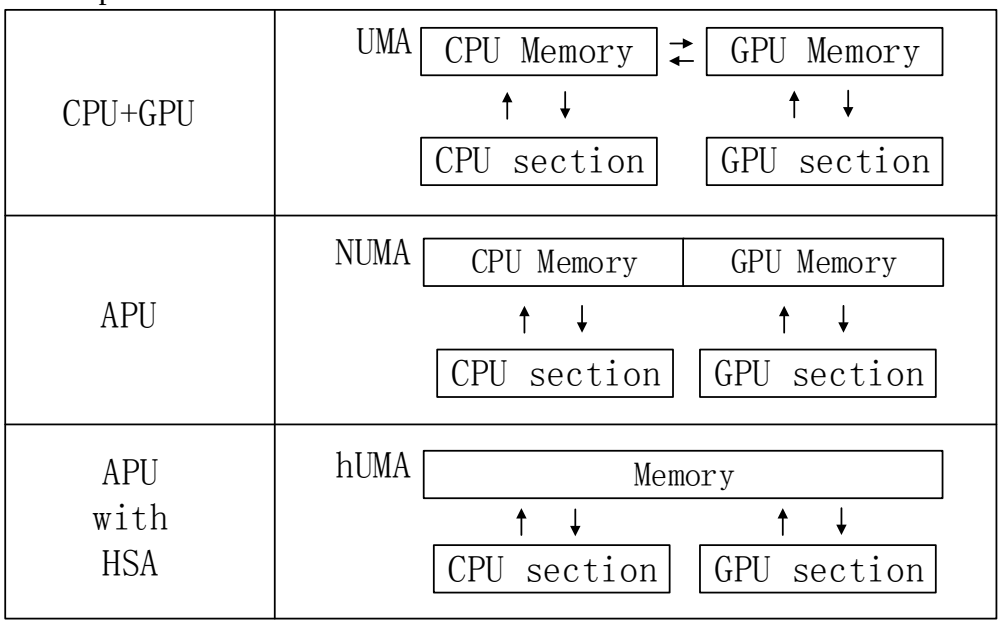

Fig. 1 Three different access memory modes

hQ is another key technology defined by HSA. It implements a special stack that is support for User-Mode Scheduling (UMS) to allow applications to arrange the desired compute unit directly.[5] This avoids the use of a kernel scheduler that intermixes threads from different applications. hQ changes the CPU-centric model that requires all arrangements and tasks to be scheduled through CPU.

In addition, HSA can support task preemption and context switching by the operating system for time slicing, and it also provides the middle-level programming languages named the Heterogeneous System Architecture Intermediate Layer (HSAIL) to all for utilization of heterogeneous computing by all programmers. HSAIL provides a standard target for higher-level language compilers or tools allowing programs to run on the HSA platform.

3. Proposed Approach

3.1 Naive Bayesian Classification Algorithm

Bayesian classification algorithm is the generic term for a class of classification algorithms, which is implemented based on the Bayesian theorem as the theoretical basis. In order to analyze the advantages and disadvantages of HSA in image processing, we use the naive Bayesian classification algorithm, which is the most commonly used Bayesian classification algorithm.

3.1.1 Naive Bayes Classifier

Naive Bayes classification algorithm is usually used to solve the problem of region feature recognition, which is widely used in satellite remote sensing and geographical survey. The algorithm can solve the problem of which category the sample to be assigned belongs to.

In order to achieve Naive Bayesian image classification algorithm, we use the category of this image as a category attribute, each piece of image is divided, and we use the color that the maximum number of pixels per block as the feature attribute. The specific algorithm steps are as follows:

a. The set X = {x1, x2,…,xm} is the set of a sample's feature attributes, where xi is an attribute of the image set:

b. The set D = {d1, d2,…,dn} is the set of training samples, di is the number of each category in the training sample set. That is the prior probability P (yi):

$$P(y_i) = \frac{d_i}{\sum_{i=1}^{n} d_i} \tag{1}$$

c. The set Y = {y1, y2,…,yn} is the set of category attributes, and calculate the mean yi and the standard deviation yi of the training sample by each of element yi:

d. Calculate the conditional probability of each feature attribute of the sample that should be classified under each kind of type condition based on the calculated mean and standard deviation of the training sample:

$$P(x_j|y_i) = \frac{1}{\sqrt{2\pi}\sigma_{yi}} e^{-\frac{(x_j-\mu_{yi})^2}{2\sigma_{yi}^2}} \tag{2}$$

e. Because the individual color characteristics of the pixels are independent of each other, through the Bayesian theorem:

$$P(y_i|x_j) = \frac{P(x_j|y_i)P(y_i)}{P(x_j)} \tag{3}$$

The value of the denominator P(X) in Equation 3 is:

$$P(x_j) = \sum_{i=1}^{n} P(x_j|y_i)P(y_i) \tag{4}$$

For all feature attributes that belong to the category set X, the denominator P(X) is a constant, so only the molecular part needs to be considered and optimized for it:

$$P(X|y_i)P(y_i) = P(x_1|y_i)P(x_2|y_i)\dots P(x_m|y_i)P(y_i) = P(y_i)\prod_{j=1}^{m} P(x_j|y_i) \tag{5}$$

f. Based on the above steps, the posterior probability P(yi|X) is obtained. If the posterior probability is:

$$P(y_k|X) = \max\{P(y_1|X), P(y_2|X), \dots, P(y_n|X)\} \tag{6}$$

The maximum probability category attribute yk belonging to the sample can be obtained as the category attribute of the sample.

3.1.2 Algorithm Correlation Analysis

The main process in the above steps is divided into two aspects. The first part is conducting the training of the known category attribute set through the method of statistics and recognition. The next section is the classification of the samples to be tested. In these processes, a lot of image preprocessing and eigenvalue extraction operations and a large number of mathematical statistical operations are used. As the process of a large number of consistent pixel data collecting and calculating, when the image size and split density increases, it will make the entire calculation of the workload doubled.

From the complexity of time, it is mainly reflected in the extraction of the characteristic data of the image. If the number of pixels to be detected is N*M, and there are K attributes of the feature to be judged, the number of known category features set is P, then the time complexity of the whole algorithm will be as shown in the following figure.

Table 1. The time complexity required for each processing session

Processing session	Time complexity
Conditional Probability	O(N*M*P*K)
Posterior Probability	O(N*M*P*K)
Data Extraction	O(N*M*P)

Not only that, although the high parallelization of GPU can solve the computational efficiency to a certain extent, but because of the use of various computing devices is not sufficient, the algorithm still has optimized space.

It can be seen that when the size of the image increases or the image segmentation density increases, the workload of the algorithm will multiply, which will greatly affect the performance of the algorithm.

3.2 Algorithm Eigenvalue Processing Optimization Scheme Based on HSA

The kernel function of GPU is often distributed by the CPU in the traditional heterogeneous platform, GPU can't complete the task scheduling independently. And as a result of the impact of programming model in the traditional CPU isomorphic platform, the design of the heterogeneous algorithm structure is often phasic. For this study of the eigenvalue processing link, each device presents a "serial" synchronous work model.

In order to solve this problem, we use the optimal scheduling program of the parallel operation between CPU and GPU. There are three main features of the scheduling: Asynchronous equipment processing, it is that each device access to the memory of the image block without waiting for each other. As long as one of them complete the processing of the current image block, it can perform the next block data operation directly; Image block is accessed only, it is that the same image block must be accessed by CPU or GPU. We can obtain the corresponding identification variables before getting image block and update the corresponding identification variables after getting image block to avoid the performance degradation of the algorithm that the same module is processed by different devices repeatedly; coincident time of equipment implementation. The consistent equipment implementation time, we can balance the implement time of different equipment, and this process is the parallel processing of each device in the macro point of view.

The scheduling algorithm id divided into three processing stages, the specific steps are as follows.

First of all, initializing parameter. Because of the image segmentation before the eigenvalue processing of the image block in the Naïve Bayesian image classification, each block will be stored in a memory space. We can set Parameters that represent the current number of blocks and the number of images, and set them to zero. Each processor must rely on them before performing data block eigenvalue operations.

Second, the learning phase of the equipment. The main task of this phase is to find out the rate at which each computing device handles the image block exactly, so the corresponding amount of data blocks at this stage is less. A) When the device deal with image block first, the number of block doesn't change next time, and record processing speed. B) It isn't the first time, it will be analyzed. We can record the processing time of the image blocks, and calculate the corresponding rate, compare the two before and after the rate value. C) If the difference between the two is greater than or equal to the minimum rate of change that proves the computing device execution rate is not optimal, the processing number of the image blocks plus one. D) Otherwise, it proves the processing rate is stabilizing. E) When the processing rate change of all computing devices reaches a specified threshold (current value of 5%) or the number of blocks currently in progress has reached the maximum learning upper limit (current value is 20%), the learning phase is completed.

Finally, the full implementation phase of the equipment, obtain the speed of the processing image block of each current device, performing the remaining image block according to the ratio of the final rate between CPU and GPU.

4. Experimental Results

4.1 Environment

The hardware environment of this test is AMD APU series A10-7850K, the product is the first hardware equipment, which supports HSA architecture. On the operating system, you can use the Ubuntu 15.04, which support the Linux kernel 3.19 version. In addition, we used the kfd v1.4 HSA driver and HSA Runtime 1.0f. In order to obtain the execution time and utilization rate of AMD GPU, we use the AMD CodeXL for testing. And through the time instruction in the Linux system, we obtain the relevant program execution time; the result is the mean by multiple measurements.

4.2 Results

By deploying the above optimization scheme, compare the result of the algorithm before and after optimization. First of all, we take the training phase of the eigenvalue selection stage as an example; we set the number of images as a variable. The size of image is 256*256. Each image is divided into 16 sub-blocks; we compare the execution time of the algorithm before and after optimization. Through the picture can be seen, on the whole, the optimization program on the algorithm has a good efficiency. As the number of blocks processed is increased, the Figure 2 shows that our optimization saves 29.56 % of execution time on average.

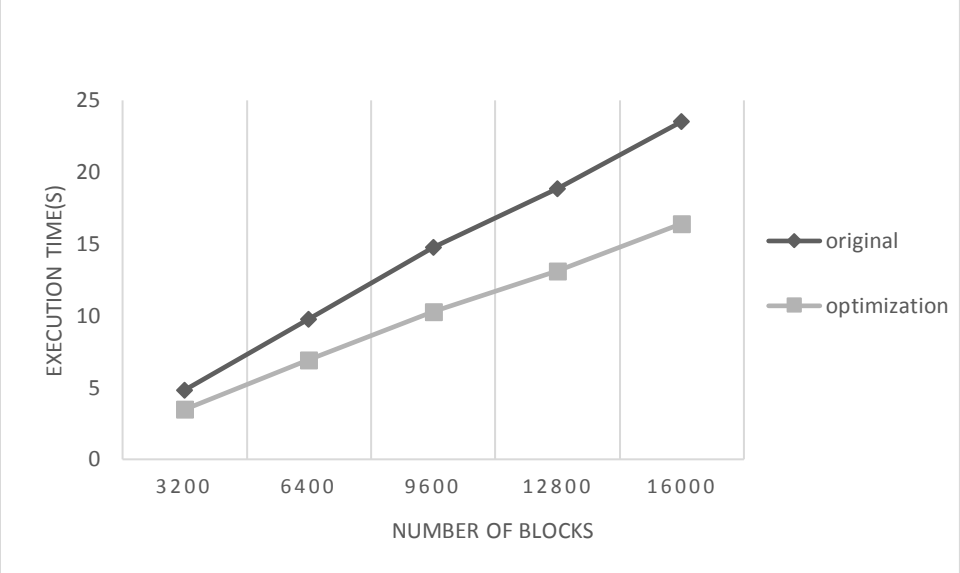

Fig. 2 The execution time before and after optimization

In order to be able to better display the optimization effect, we show the data distribution under different image blocks, and get the effect as shown in Figure 3:

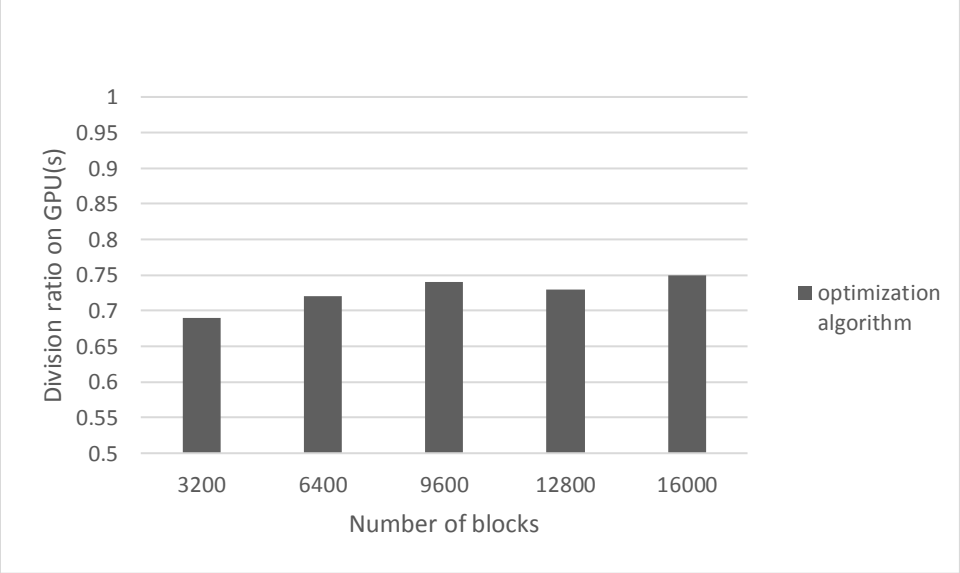

Fig. 3 The execution time before and after optimization

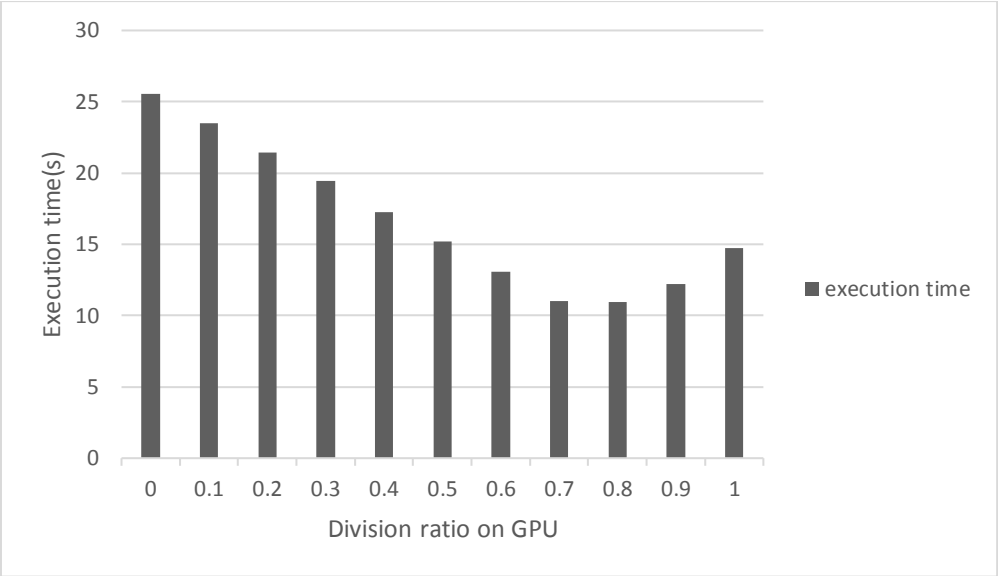

Fig. 4 The execution time before and after optimization

It can be seen that the number of blocks processed by the CPU and the GPU is roughly the same in the different number of image blocks, and the ratio is between 0.69 and 0.75. Taking 9600 image blocks as an example, the image segmentation is divided according to the proportion of the number of CPU and GPU processing in Figure 4. It can be seen that the best distribution ratio is between 70% and 80%, the algorithm optimization effect is almost in it.

5. Conclusion and Future

At present, the research on CPU and GPU architecture focuses mainly on the single computing equipment, and there is not much research on the cooperative use of equipment. Most of the links in the algorithm are still in a single computing device processing state, which also has the restriction of data transmission between CPU and GPU. In this paper, we study the Naive Bayesian image classification algorithm on HSA heterogeneous platform, and design a set of effective allocation scheme for the algorithm eigenvalue processing. The experimental results show that the optimization scheme can save the average execution time of the whole algorithm by 29.56%. This experiment not only provides a good solution for the optimization of the Naive Bayesian image classification algorithm on the HSA platform, but also provides guidance for other algorithm optimization on the HSA platform. In the future, we will study a set of data distribution strategies and image processing algorithm of HSA.

6. References

[1]. Owens J D, Houston M, Luebke D, et al. GPU Computing [J]. Proceedings of the IEEE, 2008, 96(5):879-899.

[2]. Calandra H, Dolbeau R, Fortin P, et al. Evaluation of successive CPUs/APUs/GPUs based on an OpenCL finite difference stencil[C]// Euromicro International Conference on Parallel, Distributed, and Network-Based Processing. IEEE Computer Society, 2013:405-409.

[3]. Information on http://www.hsafoundation.com/

[4]. Tang J H. Research of Vehicle Video Image Recognition Technology Based on Naive Bayesian Classification Model[C]// Information and Computing (ICIC), 2010 Third International Conference on. IEEE, 2010:17-20.

[5]. Blinzer P. The Heterogeneous System Architecture: It's beyond the GPU[C]// International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation. 2014: iii-iii.