An Efficient FPGA Implementation for odd-even sort based KNN algorithm using OpenCL

Hai Peng, Letian Huang
School of Communication and Information
University of Electronic Science and Technology of China
Chengdu, China
Corresponding Author: huanglt@uestc.edu.cn

John Chen
PSG University Program
Intel
Chengdu, China

***Abstract*—Owing to the rising demands for resources integration, a lot of research efforts has been devoted to accelerating the data classification techniques. K-Nearest Neighbor(KNN)algorithm, as one of the most important data classification algorithms, is widely used in text categorization, predictive analysis etc. Two most vital issues of improving KNN are accelerating the speed of convergence and efficiently optimizing the parallel implementation. In this paper, we propose an efficient implementation on FPGA based heterogeneous computing system using OpenCL. An odd-even sort based KNN is designed to make full use of the parallel pipeline structure of FPGA. The results has shown that the performance of the proposed algorithm is much better than traditional GPU based KNN .**

***Index Terms*—OpenCL; KNN; Odd-even Sort; FPGA; Heterogeneous Computing;**

I. Introduction

With the increasingly growing demands for data classification in recent decades, Field Programmable Gate Arrays(FPGAS), which are highly suitable for parallel implementation of a lot of classification algorithms ,have combined with CPU and formed a new kind of heterogeneous computing architecture. Open Computing Language (OpenCL) [1]is a framework which relies on advanced programming language and at the same time provides supports to FPGA targets. KNN [2] has been widely used in the filed of data classification , but the cost of KNN is extremely intensive.

In this paper, an efficient odd-even sort based KNN is presented. By using OpenCL, it makes full use of properties of odd-even sort algorithm and redesign the KNN algorithm in parallel. Thus, the efficiency of distance calculation and distance rank has been improved.

Traditional KNN algorithms firstly need to sort all of N distances and then choose the k minimum distances from N. However, due to the fact that in KNN only k minimum distances are necessary, we redesign the KNN in a parallel way using odd-even sort algorithm. For each query object, only k work-items are used instead of N. As a consequence, the efficiency of KNN has been obviously improved.

The rest of this paper is organized as follows. Section II describes the KNN algorithms and the OpenCL program framework. In Section III we propose an efficient FPGA implementation for odd-even based KNN algorithm using OpenCL. Section IV details the performance results and comparisons.

II. Background

A. K Nearest Neighbor Algorithm

KNN is used to classify objects by a majority vote of k nearest reference objects [3].Distance calculation and distance rank are the two most time-consuming processes in KNN. Firstly, N distances are calculated between query objects and reference objects. The distance rank process is the most important part in KNN. Based on the distance matrix obtained by distance calculation part, distance rank part sorts each row of the matrix and chooses the k smallest distances for each row.

B. OpenCL Architecture

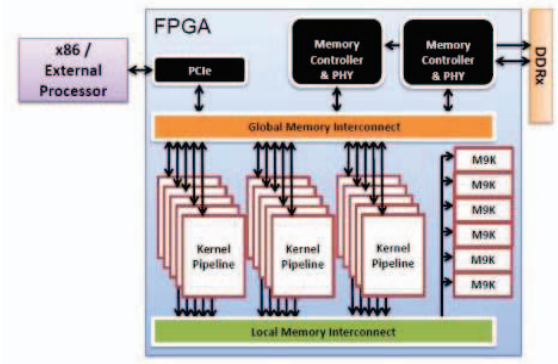

Fig. 1. OpenCL platform

OpenCL is a framework for parallel programming on heterogeneous platforms.it is based on a runtime host library and C99 extensions for device programming adapted to support vectorized data types. As shown in Fig 1, the framework consists of two main parts. The first part is the host which handles the data-flow generated by applications and transfers it to the OpenCL devices through PCIe. The second part is the FPGA which works as the OpenCL device. The work-items on a device are organized into work-groups which share a common memory region and synchronization points. The memory of the device consists of three levels: global memory, local memory and private memory.

III. Odd-even sort Based KNN Algorithm using OpenCL

A. Distance Calculation Kernel

Distances between query objects and reference objects are calculated in this kernel. As shown in Fig 2,for the reason that distance calculations are independent, this kernel can be fully parallel mapped on FPGA [4].Due to the low efficiency of access to global memory, we load the reference dataset into local memory from global memory in advance. The calculation results are stored in distances matrix.

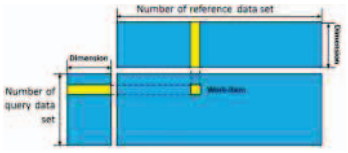

Fig. 2. Distance Calculation Kernel

B. Distance Rank Kernel

Once the distances matrix is generated from distance cal culation kernel, distance rank kernel is launched to find the k smallest distances in each row of the matrix. As shown in Fig 3, the entire sorting process is divided into three stages. In the first stage, k work-items sort the first 2k distances in a row using odd-even sort. Each work-item compare 2 elements at one time. For example, in the even turn, the first work- item compare the 1st and 2nd distances in each row. While in the odd turn, the first work-item compare the 2nd and the 3rd distances. The k smallest distances out of the 2k distances are gained after a comparison up to 2k+1 times. In the second stage, parallel bubble sort [4] is used to enable k work-items to carry the k smallest distances from the head to the tail of the row. The third stage is the same with the first stage. Odd-even sort is applied to ensure the k smallest distances of the row are chosen out. During the whole process, only k work-items are used.

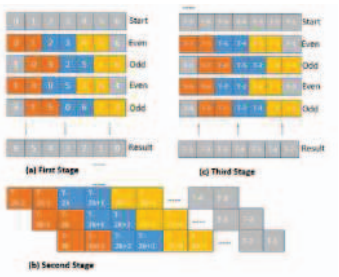

Fig. 3. Distance Rank Kernel

IV. Evaluation

A. Test Environment

Three target devices are considered: a CPU, a GPU and an FPGA. The CPU is an Intel Core i7-3770K running at 3.5GHZ with the reference software written in matlab. The GPU is an AMD Radeon HD7950 running for comparison with FPGA. The FPGA board is a Terasic DE5 with a Stratix V 5SGX.

B. Hardware Resources Management and Utilization

KDD-CUP 2004 quantum physics dataset is used to test our KNN algorithm. The number of dimensions of each record is 40.Since K is usually not large compared to the number of reference objects, we set it 64. TABLE 1 shows resources usage condition of the KNN system when implemented on the DE5 FPGA.

TABLE I
RESOURCE USAGE

Stratix V	
Logic utilization	70%
Dedicated logic registers	34%
Memory bits	45%
DSP block(18-bit)	6%
Clock Frequency	131.42MHZ

C. Comparison

TABLE 2 shows the performances for each kernel on GPU and FPGA, along with the software reference results. Through comparison between GPU and FPGA, we have verified that the new KNN based on FPGA has better performances than traditional GPU devices. By comparing the energy efficiency ratio(EER),FPGA based KNN is more than 3 times higher than traditional GPU based KNN.

TABLE II
PERFORMANCES

Platform	CPU	GPU	FPGA
Feature size/nm	22	28	40
Runtime/ms	11020.42	29.28	77.59
Objects/s	1.85	699.45	263.96
Speedup	/	378	142
Power/w	130	200	24
Objects/J	0.014	3.497	10.998
EER	/	249	786

V. Conclusion

This paper has presented an efficient FPGA implementation for an old-even sort based KNN algorithm using OpenCL.The EER of FPGA based KNN is more than 3 times higher than traditional GPU based KNN.

References

[1] M. Aaftab, "The opencl specification," *Khronos OpenCL Working Group*, vol. 1, 2011.

[2] G. Nolan, "Improving the k-nearest neighbour algorithm with cuda," *Honours Programme, The University of Western Australia*, 2009.

[3] S. Liang, C. Wang, Y. Liu, and L. Jian, "Cuknn: A parallel implementation of k-nearest neighbor on cuda-enabled gpu," in *Information, Computing and Telecommunication, 2009. YC-ICT'09. IEEE Youth Conference on*. IEEE, 2009, pp. 415–418.

[4] Y. Pu, J. Peng, L. Huang, and J. Chen, "An efficient knn algorithm implemented on fpga based heterogeneous computing system using opencl," in *Field-Programmable Custom Computing Machines (FCCM), 2015 IEEE 23rd Annual International Symposium on*. IEEE, 2015, pp. 167–170.