Assignment
Gausian Kernal Nvidia cuda

Akash Sharma
UBIT-as475
as475@buffalo.edu
Department of Computer Science and Engineering
University at Buffalo
Buffalo, NY 14214

16 Dec 2021

1 Introduction

This assignment is to implement and analyze efficient Nvidia cuda program.

#include "cuda_runtime.h"
#include "device$_launch_parameters.h$"
$\#include < stdio.h >$
$\#include < chrono >$
$\#include < iostream >$
$\#include < random >$
$\#include < vector >$
$\#include < functional >$
$\#include < algorithm >$
$\#definear\_size1664$

#define A3_HPP

__global__ void gausian_kernal(float *input$_data$, $float*output\_data$, $floatsize$, $floath$)
int tid = threadIdx.x;
int gid = tid + blockIdx.x*blockDim.x;

__shared__ float arr[ar$_size$];

float sum = 0;

for (int i = 0; i < gridDim.x; i++)
arr[tid] = input_data[i*blockDim.x + tid];

$_{s}yncthreads()$;
    for (int j = 0; j < blockDim.x; j++)
int size_null = j+ i*blockDim.x;
if(size_null<size)
    sum = sum + (1/(size*h*sqrt(2*3.14159265359)))*exp(-((input$_{d}ata[gid]$ −
$arr[j])/h) * ((input_{d}ata[gid] - arr[j])/h))$;

    output_data[gid] = sum;

    void gaussian$_{k}de(intn, floath, conststd :: vector < float > x, std :: vector <$
$float > y)float * input_{d}ata, *output_{d}ata$;
    int warp_size = 128;
dim3 block_size(warp$_{s}ize$);
$dim3grid\_size(n/block_{s}ize.x + 1)$;

    cudaMalloc((float **)input_data, total_byte$_{s}ize$); $cudaMalloc((float**)output\_data, total\_byte_{s}ize)$;
    cudaMemcpy(input$_{d}ata, x.data(), total_{b}yte_{s}ize, cudaMemcpyHostToDevice)$;

    gausian_kernal«<grid$_{s}ize, block_size >>> (input_data, output_data, n, h)$;
$cudaMemcpy(y.data(), output\_data, total\_byte\_size, cudaMemcpyDeviceToHost)$;
$floatsum = 0$;

    cudaFree(input_data);
cudaFree(output_data);
// gaussian_kde

2 GPU Configuration

gcc/6.3.0
cuda/9.2
gpu:tesla$_{v}100 - pcie - 16gb$:
$node = 1$

3 Result

./a3 10000 .001 Tp: 0.245654s
./a3 1000000 .001 Tp: 5.98764s
./a3 10000000 .001 Tp: 525.193s

    With increase input value time is increasing.