Computer Architecture

Gpu Simulation

Three matrix multiplication

Name : Aya Ashraf Saber

ID : 02

Introduction

3 Matrix multiplication using Cuda Code apply on google colab

Code

```
%%cu
#include "cuda_runtime.h"
#include "device_launch_parameters.h"

#include <stdio.h>
#include <time.h>

#define N (1024*1024)
#define M (10000)
#define THREADS_PER_BLOCK 1024

__global__ void vector_add(double *a, double *b, double *c, double * d)
{
    int index = blockIdx.x * blockDim.x + threadIdx.x;
        for(int j=0;j<M;j++)
        {
            c[index] += a[index]*a[index] + b[index]*b[index]+ d[index]*d[index];
        }
}

int main()
{
    clock_t start,end;
    double *a, *b, *c, *d;
    int size = N * sizeof( double );

    a = (double *)malloc( size );
    b = (double *)malloc( size );
    c = (double *)malloc( size );
    d = (double *)malloc( size );

    for( int i = 0; i < N; i++ )
    {
        a[i] = b[i] = 2;
        c[i] = 0; // first multiplication
        d[i] = 2;
```

```
start = clock();
double *d_a, *d_b, *d_c, *d_d;

cudaMalloc( (void **) &d_a, size );
cudaMalloc( (void **) &d_b, size );
cudaMalloc( (void **) &d_c, size );
cudaMalloc( (void **) &d_d, size );


cudaMemcpy( d_a, a, size, cudaMemcpyHostToDevice );
cudaMemcpy( d_b, b, size, cudaMemcpyHostToDevice );
cudaMemcpy( d_d, d, size, cudaMemcpyHostToDevice );


vector_add<<< N/THREADS_PER_BLOCK, THREADS_PER_BLOCK >>>( d_a, d_b, d_c,d_d);

cudaDeviceSynchronize();
cudaMemcpy( c, d_c, size, cudaMemcpyDeviceToHost );
end = clock();

cudaFree( d_a );
cudaFree( d_b );
cudaFree( d_c );
cudaFree( d_d );

float time2 = ((float)(end-start))/CLOCKS_PER_SEC;

// Verify integrity
int errors = 0;
for (int i = 0; i < N; i++) {
    if (c[i] != 80000.0 ) errors++;
}
```

```
    printf("Errors: %d\n", errors);
    printf("CUDA: %f seconds\n",time2);

    return 0;
}
```

References

https://gist.github.com/k-alkiek/2b41c1489299cd7441f38d3b0ef06aa5?fbclid=IwAR2jI-2T0fgznRBUiDEHY47YtX6eSkpK_Uy9na4Sd1cvUb4rLQJkz7d8-P0

https://www.wikihow.com/Run-CUDA-C-or-C%2B%2B-on-Jupyter-(Google-Colab)

https://www.quantstart.com/articles/Matrix-Matrix-Multiplication-on-the-GPU-with-Nvidia-CUDA