Задание 2
Отчёт
по CUDA ADI3d

Ши Хуэй shihuicollapsor@gmail.com

1. Постановка задачи

1. Программа должна автоматически определять доступный объём памяти на GPU и выбирать максимально возможный размер сетки (L), который поместится в эту память.

2. Нужно реализовать возможность запуска на CPU и GPU, а также режим сравнения, чтобы проверить, что результаты расчётов одинаковы.

3. Редукцию (вычисление максимального значения ошибки eps) необходимо распараллелить. В программе уже используется атомарная операция на GPU, но она может быть оптимизирована.

4. Создайте Git-репозиторий с вашим кодом. В нём должен быть:

•Makefile, который позволит собрать и запустить программу на любом сервере с GPU.

•Возможность выбирать, на каком устройстве (CPU или GPU) будет выполняться программа.

•Режим проверки совпадения результатов между CPU и GPU.

5. Проверьте производительность программы:

•Сравните время выполнения программы на GPU и CPU.

•Постройте таблицу или график, показывающий ускорение программы на GPU по сравнению с последовательной версией на CPU (с максимальными опциями оптимизации).

2. Формат командной строки

nvcc adi3d_cuda.cu -o cuda2

3. Спецификация системы

- Operating system : Linux 6.8.0-45-generic
- Vendor string and code : GenuineIntel (1, 0x1)
- Model string and code : Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz (165, 0xa5)
- CPU revision : 2.000000
- CPUID : Family/Model/Stepping 6/165/2, 0x06/0xa5/0x02
- CPU Max MHz : 5000
- CPU Min MHz : 800
- Total cores : 12

- SMT threads per core : 2
- Cores per socket : 6
- Sockets : 1
- Cores per NUMA region : 12
- NUMA regions : 1
- Running in a VM : no
- Number Hardware Counters : 10
- Max Multiplex Counters : 384

4. Описание алгоритмов

```
#include <math.h>
#include <stdlib.h>
#include <stdio.h>
#include <cuda_runtime.h>
#include <time.h>

#define Max(a, b) ((a) > (b) ? (a) : (b))

void init(double *a,int L);
void runOnCPU(double *a, int itmax, double maxeps, double *elapsedTime,int L);
void runOnGPU(double *a, int itmax, double maxeps, double *elapsedTime,int L);

__global__ void kernelOptimized(double *a, double *eps_d, int L);

int main(int argc, char *argv[])
{
   double maxeps = 0.01;
   int itmax = 100;
   double *a;
   double elapsedTime = 0.0;

   // Allocate memory for 3D array
   size_t free_mem, total_mem;
   cudaMemGetInfo(&free_mem, &total_mem);
   printf("Free memory: %zu bytes\n", free_mem);
   printf("Total memory: %zu bytes\n", total_mem);
```

```
int L = (int)pow((free_mem * 0.9 / (2.0 * sizeof(double))), 1.0 / 3.0);

printf("Dynamic grid size set to : %d x %d x %d \n",L,L,L);

cudaMallocHost((void **)&a, L * L * L * sizeof(double));
init(a,L);

printf("Choose execution mode: 1 - CPU, 2 - GPU\n");
int mode;
scanf("%d", &mode);

if (mode == 1)
{
   printf("Running on CPU...\n");
   runOnCPU(a, itmax, maxeps, &elapsedTime, L);
}
else if (mode == 2)
{
   printf("Running on GPU...\n");
   runOnGPU(a, itmax, maxeps, &elapsedTime, L);
}
else
{
   printf("Invalid mode. Exiting...\n");
   cudaFreeHost(a);
   return 0;
}

// Print benchmark results
printf(" ADI Benchmark Completed.\n");
printf(" Size            = %4d x %4d x %4d\n", L, L, L);
printf(" Iterations      =       %12d\n", itmax);
printf(" Time in seconds =       %12.2lf\n", elapsedTime);
printf(" Operation type  =   double precision\n");
printf(" END OF ADI Benchmark\n");
```

```
    cudaFreeHost(a);
    return 0;
}

void init(double *a,int L)
{
    for (int i = 0; i < L; i++)
        for (int j = 0; j < L; j++)
            for (int k = 0; k < L; k++)
            {
                if (k == 0 || k == L - 1 || j == 0 || j == L - 1 || i == 0 || i == L - 1)
                    a[i * L * L + j * L + k] = 10.0 * i / (L - 1) + 10.0 * j / (L - 1) + 10.0 * k / (L - 1);
                else
                    a[i * L * L + j * L + k] = 0;
            }
}

void runOnCPU(double *a, int itmax, double maxeps, double *elapsedTime,int L)
{
    double eps;
    clock_t start = clock();
    for (int it = 1; it <= itmax; it++)
    {
        eps = 0;
        for (int i = 1; i < L - 1; i++)
            for (int j = 1; j < L - 1; j++)
                for (int k = 1; k < L - 1; k++)
                    a[i * L * L + j * L + k] = (a[(i - 1) * L * L + j * L + k] + a[(i + 1) * L * L + j * L
+ k]) / 2;

        for (int i = 1; i < L - 1; i++)
            for (int j = 1; j < L - 1; j++)
                for (int k = 1; k < L - 1; k++)
                    a[i * L * L + j * L + k] = (a[i * L * L + (j - 1) * L + k] + a[i * L * L + (j + 1) * L
+ k]) / 2;

        for (int i = 1; i < L - 1; i++)
```

```
            for (int j = 1; j < L - 1; j++)
                for (int k = 1; k < L - 1; k++)
                {
                    double tmp = (a[i * L * L + j * L + (k - 1)] + a[i * L * L + j * L + (k + 1)]) / 2;
                    eps = Max(eps, fabs(a[i * L * L + j * L + k] - tmp));
                    a[i * L * L + j * L + k] = tmp;
                }

        printf(" CPU IT = %4d   EPS = %14.7E\n", it, eps);
        if (eps < maxeps)
            break;
    }
    clock_t end = clock();
    *elapsedTime = (double)(end - start) / CLOCKS_PER_SEC;
}

void runOnGPU(double *a, int itmax, double maxeps, double *elapsedTime, int L)
{
    double *a_d, *eps_d, eps;
    cudaMalloc((void **)&a_d, L * L * L * sizeof(double));
    cudaMemcpy(a_d, a, L * L * L * sizeof(double), cudaMemcpyHostToDevice);
    cudaMalloc((void **)&eps_d, sizeof(double));

    dim3 threadsPerBlock(8, 8, 8);
    dim3 numBlocks((L + threadsPerBlock.x - 1) / threadsPerBlock.x,
                   (L + threadsPerBlock.y - 1) / threadsPerBlock.y,
                   (L + threadsPerBlock.z - 1) / threadsPerBlock.z);

    cudaEvent_t start, stop;
    cudaEventCreate(&start);
    cudaEventCreate(&stop);

    cudaEventRecord(start);
    for (int it = 1; it <= itmax; it++)
    {
        eps = 0;
```

```
        cudaMemcpy(eps_d, &eps, sizeof(double), cudaMemcpyHostToDevice);

        kernelOptimized<<<numBlocks, threadsPerBlock>>>(a_d, eps_d, L);
        cudaDeviceSynchronize();

        cudaMemcpy(&eps, eps_d, sizeof(double), cudaMemcpyDeviceToHost);
        printf(" GPU IT = %4d   EPS = %14.7E\n", it, eps);

        if (eps < maxeps)
            break;
    }
    cudaEventRecord(stop);
    cudaEventSynchronize(stop);

    float ElapsedTime;
    cudaMemcpy(a, a_d, L * L * L * sizeof(double), cudaMemcpyDeviceToHost);
    cudaEventElapsedTime(&ElapsedTime, start, stop);

    *elapsedTime = ElapsedTime / 1000.0;

    cudaFree(a_d);
    cudaFree(eps_d);
    cudaEventDestroy(start);
    cudaEventDestroy(stop);
}

__global__ void kernelOptimized(double *a, double *eps_d, int L)
{
    int i = blockIdx.x * blockDim.x + threadIdx.x;
    int j = blockIdx.y * blockDim.y + threadIdx.y;
    int k = blockIdx.z * blockDim.z + threadIdx.z;

    if (i > 0 && i < L - 1 && j > 0 && j < L - 1 && k > 0 && k < L - 1)
    {
        double oldVal = a[i * L * L + j * L + k];
        double newVal = (a[(i - 1) * L * L + j * L + k] + a[(i + 1) * L * L + j * L + k]) / 2.0;
```

```
            atomicMax((unsigned long long *)eps_d, __double_as_longlong(fabs(oldVal - newVal)));
            a[i * L * L + j * L + k] = newVal;
        }
    }
```

5.Заключение

collapsor@collapsor-G5-5500:~/Desktop/CUDA$ nvcc adi3d_cuda.cu -o cuda2

collapsor@collapsor-G5-5500:~/Desktop/CUDA$ ./cuda2

Free memory: 5925502976 bytes

Total memory: 6020661248 bytes

Dynamic grid size set to : 693 x 693 x 693

Choose execution mode: 1 - CPU, 2 - GPU

2

Running on GPU...	Running on CPU...
GPU IT = 1 EPS = 1.4985549E+01	CPU IT = 1 EPS = 1.4971098E+01
GPU IT = 2 EPS = 7.4927746E+00	CPU IT = 2 EPS = 7.4783237E+00
GPU IT = 3 EPS = 3.7463873E+00	CPU IT = 3 EPS = 3.7355491E+00
GPU IT = 4 EPS = 3.7463873E+00	CPU IT = 4 EPS = 2.7989523E+00
GPU IT = 5 EPS = 2.8097905E+00	CPU IT = 5 EPS = 2.0971821E+00
GPU IT = 6 EPS = 2.3414921E+00	CPU IT = 6 EPS = 1.6295611E+00
GPU IT = 7 EPS = 2.1073428E+00	CPU IT = 7 EPS = 1.3954118E+00
GPU IT = 8 EPS = 1.6390444E+00	CPU IT = 8 EPS = 1.1980178E+00
GPU IT = 9 EPS = 1.6390444E+00	CPU IT = 9 EPS = 1.0372731E+00
GPU IT = 10 EPS = 1.4048952E+00	CPU IT = 10 EPS = 9.0673105E-01
GPU IT = 11 EPS = 1.3170893E+00	CPU IT = 11 EPS = 8.2349832E-01
GPU IT = 12 EPS = 1.2050033E+00	CPU IT = 12 EPS = 7.4960339E-01
GPU IT = 13 EPS = 1.0845030E+00	CPU IT = 13 EPS = 6.8423444E-01
GPU IT = 14 EPS = 1.0443362E+00	CPU IT = 14 EPS = 6.2671412E-01
GPU IT = 15 EPS = 9.1379419E-01	CPU IT = 15 EPS = 5.7629754E-01
GPU IT = 16 EPS = 9.1379419E-01	CPU IT = 16 EPS = 5.3989227E-01
GPU IT = 17 EPS = 8.3072199E-01	CPU IT = 17 EPS = 5.0642748E-01

Running on GPU...	Running on CPU...
GPU IT = 18 EPS = 8.0698708E-01	CPU IT = 18 EPS = 4.7565636E-01
GPU IT = 19 EPS = 7.5655039E-01	CPU IT = 19 EPS = 4.4749221E-01
GPU IT = 20 EPS = 7.1872287E-01	CPU IT = 20 EPS = 4.2168775E-01
GPU IT = 21 EPS = 6.9107968E-01	CPU IT = 21 EPS = 3.9805472E-01
GPU IT = 22 EPS = 6.4500770E-01	CPU IT = 22 EPS = 3.7975971E-01
GPU IT = 23 EPS = 6.3348971E-01	CPU IT = 23 EPS = 3.6256015E-01
GPU IT = 24 EPS = 5.8281053E-01	CPU IT = 24 EPS = 3.4637959E-01
GPU IT = 25 EPS = 5.8281053E-01	CPU IT = 25 EPS = 3.3122921E-01
GPU IT = 26 EPS = 5.4638487E-01	CPU IT = 26 EPS = 3.1697231E-01
GPU IT = 27 EPS = 5.3810631E-01	CPU IT = 27 EPS = 3.0365210E-01
GPU IT = 28 EPS = 5.1278366E-01	CPU IT = 28 EPS = 2.9111417E-01
GPU IT = 29 EPS = 4.9853967E-01	CPU IT = 29 EPS = 2.8095897E-01
GPU IT = 30 EPS = 4.8192168E-01	CPU IT = 30 EPS = 2.7126696E-01
GPU IT = 31 EPS = 4.6338623E-01	CPU IT = 31 EPS = 2.6200979E-01
GPU IT = 32 EPS = 4.5363073E-01	CPU IT = 32 EPS = 2.5322047E-01
GPU IT = 33 EPS = 4.3202927E-01	CPU IT = 33 EPS = 2.4482527E-01
GPU IT = 34 EPS = 4.2770898E-01	CPU IT = 34 EPS = 2.3683700E-01
GPU IT = 35 EPS = 4.0394737E-01	CPU IT = 35 EPS = 2.2923406E-01
GPU IT = 36 EPS = 4.0394737E-01	CPU IT = 36 EPS = 2.2197209E-01
GPU IT = 37 EPS = 3.8558612E-01	CPU IT = 37 EPS = 2.1589762E-01
GPU IT = 38 EPS = 3.8195878E-01	CPU IT = 38 EPS = 2.1004102E-01
GPU IT = 39 EPS = 3.6811969E-01	CPU IT = 39 EPS = 2.0438742E-01
GPU IT = 40 EPS = 3.6193281E-01	CPU IT = 40 EPS = 1.9896713E-01
GPU IT = 41 EPS = 3.5187912E-01	CPU IT = 41 EPS = 1.9374178E-01
GPU IT = 42 EPS = 3.4350105E-01	CPU IT = 42 EPS = 1.8870139E-01
GPU IT = 43 EPS = 3.3663102E-01	CPU IT = 43 EPS = 1.8386918E-01
GPU IT = 44 EPS = 3.2650678E-01	CPU IT = 44 EPS = 1.7921135E-01
GPU IT = 45 EPS = 3.2232080E-01	CPU IT = 45 EPS = 1.7471833E-01
GPU IT = 46 EPS = 3.1080934E-01	CPU IT = 46 EPS = 1.7086674E-01
GPU IT = 47 EPS = 3.0889077E-01	CPU IT = 47 EPS = 1.6712848E-01

Running on GPU...	Running on CPU...
GPU IT = 48 EPS = 2.9628298E-01	CPU IT = 48 EPS = 1.6349568E-01
GPU IT = 49 EPS = 2.9628298E-01	CPU IT = 49 EPS = 1.5997820E-01
GPU IT = 50 EPS = 2.8606574E-01	CPU IT = 50 EPS = 1.5657178E-01
GPU IT = 51 EPS = 2.8402852E-01	CPU IT = 51 EPS = 1.5326314E-01
GPU IT = 52 EPS = 2.7591341E-01	CPU IT = 52 EPS = 1.5005239E-01
GPU IT = 53 EPS = 2.7291436E-01	CPU IT = 53 EPS = 1.4695006E-01
GPU IT = 54 EPS = 2.6662601E-01	CPU IT = 54 EPS = 1.4393701E-01
GPU IT = 55 EPS = 2.6245998E-01	CPU IT = 55 EPS = 1.4101025E-01
GPU IT = 56 EPS = 2.5777319E-01	CPU IT = 56 EPS = 1.3845119E-01
GPU IT = 57 EPS = 2.5261773E-01	CPU IT = 57 EPS = 1.3595751E-01
GPU IT = 58 EPS = 2.4933698E-01	CPU IT = 58 EPS = 1.3352218E-01
GPU IT = 59 EPS = 2.4334330E-01	CPU IT = 59 EPS = 1.3114395E-01
GPU IT = 60 EPS = 2.4129840E-01	CPU IT = 60 EPS = 1.2883591E-01
GPU IT = 61 EPS = 2.3459566E-01	CPU IT = 61 EPS = 1.2658268E-01
GPU IT = 62 EPS = 2.3363813E-01	CPU IT = 62 EPS = 1.2438287E-01
GPU IT = 63 EPS = 2.2633694E-01	CPU IT = 63 EPS = 1.2223776E-01
GPU IT = 64 EPS = 2.2620696E-01	CPU IT = 64 EPS = 1.2015311E-01
GPU IT = 65 EPS = 2.2007913E-01	CPU IT = 65 EPS = 1.1811797E-01
GPU IT = 66 EPS = 2.1924582E-01	CPU IT = 66 EPS = 1.1613097E-01
GPU IT = 67 EPS = 2.1418212E-01	CPU IT = 67 EPS = 1.1436414E-01
GPU IT = 68 EPS = 2.1260631E-01	CPU IT = 68 EPS = 1.1263931E-01
GPU IT = 69 EPS = 2.0850159E-01	CPU IT = 69 EPS = 1.1094849E-01
GPU IT = 70 EPS = 2.0627130E-01	CPU IT = 70 EPS = 1.0929106E-01
GPU IT = 71 EPS = 2.0303141E-01	CPU IT = 71 EPS = 1.0766868E-01
GPU IT = 72 EPS = 2.0022365E-01	CPU IT = 72 EPS = 1.0608546E-01
GPU IT = 73 EPS = 1.9776457E-01	CPU IT = 73 EPS = 1.0453381E-01
GPU IT = 74 EPS = 1.9444765E-01	CPU IT = 74 EPS = 1.0301305E-01
GPU IT = 75 EPS = 1.9269766E-01	CPU IT = 75 EPS = 1.0152297E-01
GPU IT = 76 EPS = 1.8893016E-01	CPU IT = 76 EPS = 1.0007012E-01
GPU IT = 77 EPS = 1.8781779E-01	CPU IT = 77 EPS = 9.8646224E-02

Running on GPU...	Running on CPU...
GPU IT = 78 EPS = 1.8365465E-01	CPU IT = 78 EPS = 9.7250624E-02
GPU IT = 79 EPS = 1.8299728E-01	CPU IT = 79 EPS = 9.5992862E-02
GPU IT = 80 EPS = 1.7847883E-01	CPU IT = 80 EPS = 9.4761231E-02
GPU IT = 81 EPS = 1.7847883E-01	CPU IT = 81 EPS = 9.3552222E-02
GPU IT = 82 EPS = 1.7459885E-01	CPU IT = 82 EPS = 9.2363592E-02
GPU IT = 83 EPS = 1.7412697E-01	CPU IT = 83 EPS = 9.1195004E-02
GPU IT = 84 EPS = 1.7083378E-01	CPU IT = 84 EPS = 9.0047749E-02
GPU IT = 85 EPS = 1.6993466E-01	CPU IT = 85 EPS = 8.8924514E-02
GPU IT = 86 EPS = 1.6718109E-01	CPU IT = 86 EPS = 8.7820351E-02
GPU IT = 87 EPS = 1.6589508E-01	CPU IT = 87 EPS = 8.6734907E-02
GPU IT = 88 EPS = 1.6363800E-01	CPU IT = 88 EPS = 8.5667830E-02
GPU IT = 89 EPS = 1.6200162E-01	CPU IT = 89 EPS = 8.4622365E-02
GPU IT = 90 EPS = 1.6020161E-01	CPU IT = 90 EPS = 8.3596535E-02
GPU IT = 91 EPS = 1.5824793E-01	CPU IT = 91 EPS = 8.2588070E-02
GPU IT = 92 EPS = 1.5686886E-01	CPU IT = 92 EPS = 8.1670465E-02
GPU IT = 93 EPS = 1.5462788E-01	CPU IT = 93 EPS = 8.0766076E-02
GPU IT = 94 EPS = 1.5363667E-01	CPU IT = 94 EPS = 7.9879126E-02
GPU IT = 95 EPS = 1.5113561E-01	CPU IT = 95 EPS = 7.9005574E-02
GPU IT = 96 EPS = 1.5050192E-01	CPU IT = 96 EPS = 7.8144739E-02
GPU IT = 97 EPS = 1.4758838E-01	CPU IT = 97 EPS = 7.7296433E-02
GPU IT = 98 EPS = 1.4721867E-01	CPU IT = 98 EPS = 7.6460482E-02
GPU IT = 99 EPS = 1.4432571E-01	CPU IT = 99 EPS = 7.5641014E-02
GPU IT = 100 EPS = 1.4426149E-01	CPU IT = 100 EPS = 7.4833541E-02
ADI Benchmark Completed.	ADI Benchmark Completed.
Size = 693 x 693 x 693	Size = 693 x 693 x 693
Iterations = 100	Iterations = 100
Time in seconds = 20.82	Time in seconds = 570.09
Operation type = double precision	Operation type = double precision
END OF ADI Benchmark	END OF ADI Benchmark
Running on GPU...	Running on CPU...