Теория параллелизма

Отчет

Лабораторные работы номер 6,7,8

Выполнил 23932, Тишкин А.А.

18.04.2025

Цель работы: реализовать решение уравнение теплопроводности (разностная схема – пятиточечный шаблон) в двумерной области на равномерных сетках (128^2, 256^2, 512^2, 1024^2). Провести профилированию и оптимизацию кода, выполнить сравнительный анализ скорости выполнения на CPU в одном и нескольких потоках и GPU.

Используемый компилятор: nvc++

Используемый профилировщик: Nsight Systems

Как производили замер времени работы: библиотека <chrono>

Выполнение на CPU

CPU-onecore

Размер сетки	Время выполнения	Точность	Кол-во итераций
128*128	1,05932	0,000001	34542
256*256	14,2217	0,000001	116257
512*512	190,29	0,000001	374821

CPU-multicore

Размер сетки	Время выполнения	Точность	Кол-во итераций
128*128	1,49251	0,000001	34542
256*256	4,24443	0,000001	116257
512*512	43,3714	0,000001	374821
1024*1024	293,913	0,000001	1000000

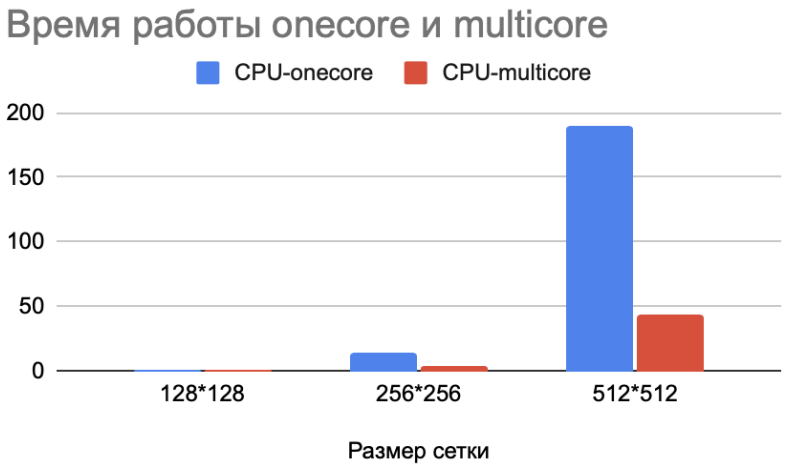


Диаграмма сравнения время работы CPU-one и CPU-multi

Выполнение на GPU

Этапы оптимизации на сетке 512*512

Этап	Время выполнения	Точность	Максимальное количество итераций	Комментарий
1	24,0808	0,000001	1_000_000	Неоптимизированный код
2	5,3986	0,000001	1_000_000	1. ручной swap указателей 2. расчет ошибки раз в 1000 шагов 3. ускоренное вычисление индексов в шаге

Размер сетки при профилировании Nsight Systems 512
Итераций <= 410000

```
 Time (%)  Total Time (ns)  Count   Avg (ns)  Med (ns)  Min (ns)  Max (ns)  StdDev (ns)          Operation
 --------  ---------------  ------  --------  --------  --------  --------  -----------  ----------------------------
     54.0        602539628  374822    1607.5    1599.0      1535     12192        137.5  [CUDA memcpy Device-to-Host]
     45.9        512422124  374821    1367.1    1376.0      1343      1792         14.6  [CUDA memset]
      0.1           642840       3  214280.0  320413.0      1407    321020     184353.7  [CUDA memcpy Host-to-Device]

[8/8] Executing 'cuda_gpu_mem_size_sum' stats report

 Total (MB)  Count   Avg (MB)  Med (MB)  Min (MB)  Max (MB)  StdDev (MB)          Operation
 ----------  ------  --------  --------  --------  --------  -----------  ----------------------------
      4.194       3     1.398     2.097     0.000     2.097        1.211  [CUDA memcpy Host-to-Device]
      2.999  374822     0.000     0.000     0.000     0.000        0.000  [CUDA memcpy Device-to-Host]
      2.999  374821     0.000     0.000     0.000     0.000        0.000  [CUDA memset]
```

Time (%)	Total Time (ns)	Instances	Avg (ns)	Med (ns)	Min (ns)	Max (ns)	StdDev (ns)	Name
64.9	3019401227	749642	4027.8	5600.0	2239	15136	1754.6	CalculateNext_80(const double *, double *, unsigned long)
35.1	1633451879	374821	4358.0	4352.0	4128	14559	120.2	StupidSwap_105(double *, const double *, unsigned long)
0.0	6592	1	6592.0	6592.0	6592	6592	0.0	InitializeGrid_57(double *, double *, unsigned long, double, double, double, double)

не оптимизированная версия

[7/8] Executing 'cuda_gpu_mem_time_sum' stats report

Time (%)	Total Time (ns)	Count	Avg (ns)	Med (ns)	Min (ns)	Max (ns)	StdDev (ns)	Operation
42.8	914294	3	304764.7	444635.0	1376	468283	263008.2	[CUDA memcpy Host-to-Device]
30.9	661141	412	1604.7	1568.0	1535	2304	75.9	[CUDA memcpy Device-to-Host]
26.2	560728	411	1364.3	1376.0	1343	1377	15.4	[CUDA memset]

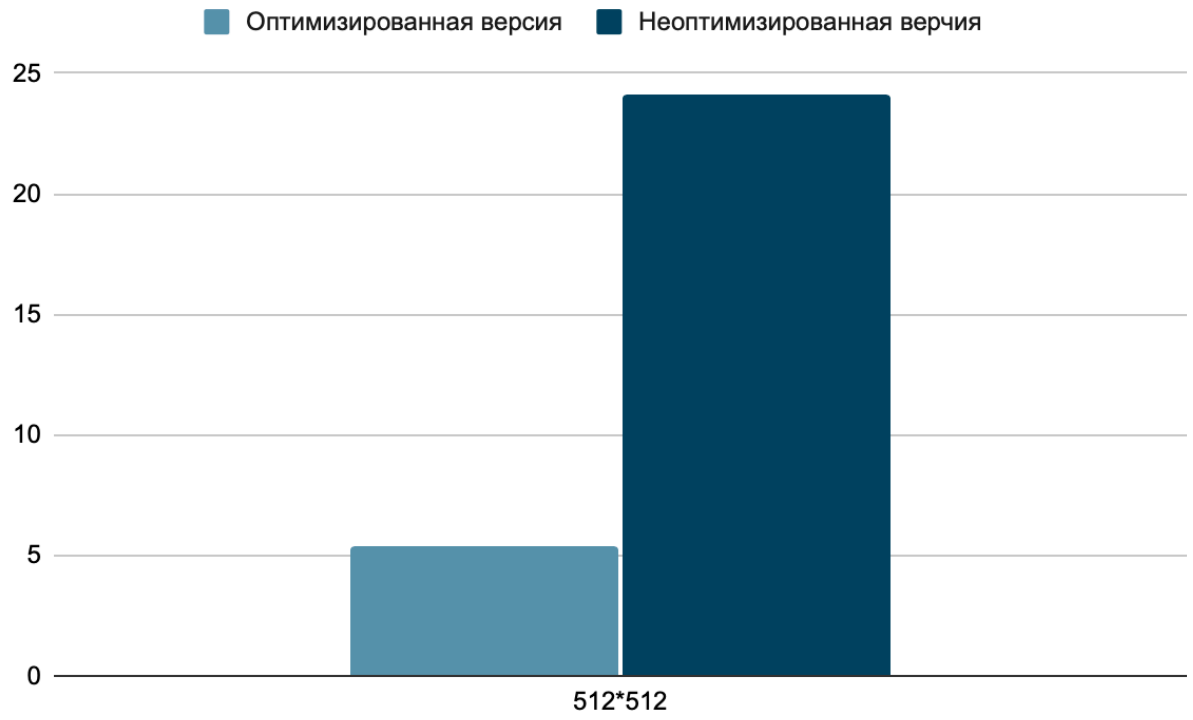
[8/8] Executing 'cuda_gpu_mem_size_sum' stats report

Total (MB)	Count	Avg (MB)	Med (MB)	Min (MB)	Max (MB)	StdDev (MB)	Operation
4.194	3	1.398	2.097	0.000	2.097	1.211	[CUDA memcpy Host-to-Device]
0.003	412	0.000	0.000	0.000	0.000	0.000	[CUDA memcpy Device-to-Host]
0.003	411	0.000	0.000	0.000	0.000	0.000	[CUDA memset]

Time (%)	Total Time (ns)	Instances	Avg (ns)	Med (ns)	Min (ns)	Max (ns)	StdDev (ns)	Name
99.8	2295948582	410001	5599.9	5600.0	5535	14784	145.9	UpdateGrid_71(const double *, double *, unsigned long)
0.2	4177242	822	5081.8	5599.5	3615	6912	1426.1	ComputeError_88(const double *, const double *, unsigned long)
0.0	5504	1	5504.0	5504.0	5504	5504	0.0	InitializeGrid_48(double *, double *, unsigned long, double, double, double, double)

оптимизированная версия

Диаграмма оптимизации



GPU – оптимизированный вариант

Размер сетки	Время выполнения	Точность	Количество итераций
128*128	0.39060	0,000001	37000
256*256	1.50662	0,000001	125000
512*512	5.35846	0,000001	410000
1024*1024	33.9098	0,000001	1000000

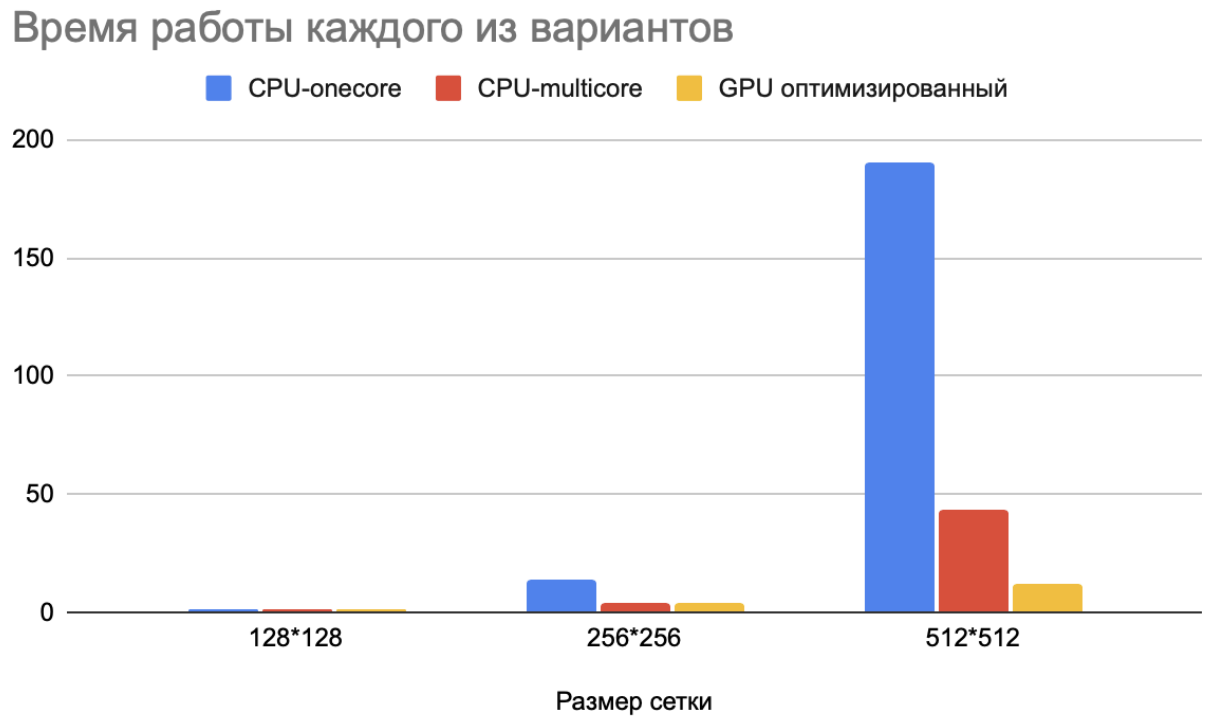


Диаграмма сравнения времени работы CPU-one, CPU-multi, GPU(оптимизированный вариант) для разных размеров сеток

```
(.env) a.tishkin1@a72c6cfa2589:~/Theory_of_parallelism/lab6$ ./main6
10 10.8333 11.6667 12.5 13.3333 14.1667 15 15.8333 16.6667 17.5 18.3333 19.1667 20
10.8333 11.6667 12.5 13.3333 14.1667 15 15.8333 16.6667 17.5 18.3333 19.1667 20 20.8333
11.6667 12.5 13.3333 14.1667 15 15.8333 16.6667 17.5 18.3333 19.1667 20 20.8333 21.6667
12.5 13.3333 14.1667 15 15.8333 16.6667 17.5 18.3333 19.1667 20 20.8333 21.6667 22.5
13.3333 14.1667 15 15.8333 16.6667 17.5 18.3333 19.1667 20 20.8333 21.6667 22.5 23.3333
14.1667 15 15.8333 16.6667 17.5 18.3333 19.1667 20 20.8333 21.6667 22.5 23.3333 24.1667
15 15.8333 16.6667 17.5 18.3333 19.1667 20 20.8333 21.6667 22.5 23.3333 24.1667 25
15.8333 16.6667 17.5 18.3333 19.1667 20 20.8333 21.6667 22.5 23.3333 24.1667 25 25.8333
16.6667 17.5 18.3333 19.1667 20 20.8333 21.6667 22.5 23.3333 24.1667 25 25.8333 26.6667
17.5 18.3333 19.1667 20 20.8333 21.6667 22.5 23.3333 24.1667 25 25.8333 26.6667 27.5
18.3333 19.1667 20 20.8333 21.6667 22.5 23.3333 24.1667 25 25.8333 26.6667 27.5 28.3333
19.1667 20 20.8333 21.6667 22.5 23.3333 24.1667 25 25.8333 26.6667 27.5 28.3333 29.1667
20 20.8333 21.6667 22.5 23.3333 24.1667 25 25.8333 26.6667 27.5 28.3333 29.1667 30
Steps: 1000
Time : 0.0107311s
```

gpu. n = 13*13

Cublas

Размер сетки	Время выполнения	Точность	Количество итераций
128*128	0.48438	0,000001	37000
256*256	1.55101	0,000001	125000
512*512	5.61332	0,000001	410000
1024*1024	33.99431	0,000001	1000000

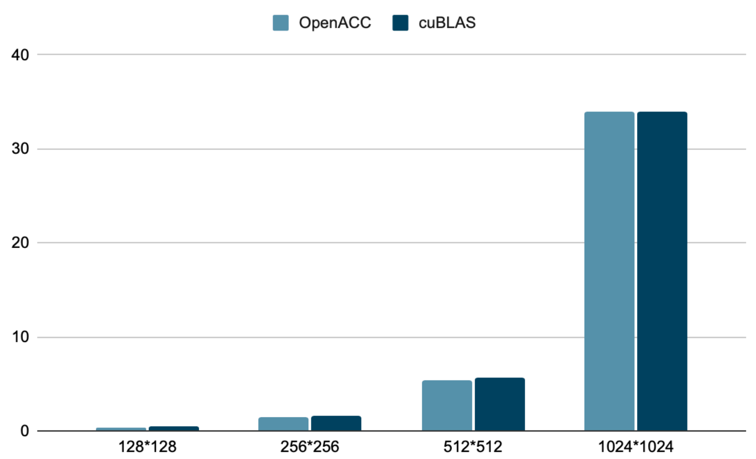
```
 Time (%)  Total Time (ns)  Count  Avg (ns)  Med (ns)  Min (ns)  Max (ns)  StdDev (ns)           Operation
 --------  ---------------  -----  --------  --------  --------  --------  -----------  ----------------------------
     56.2          1368329    823    1662.6    1600.0      1535      2272        124.0  [CUDA memcpy Device-to-Host]
     43.8          1065304      4  266326.0  326365.5      1408    411165     181089.3  [CUDA memcpy Host-to-Device]

[8/8] Executing 'cuda_gpu_mem_size_sum' stats report

 Total (MB)  Count  Avg (MB)  Med (MB)  Min (MB)  Max (MB)  StdDev (MB)           Operation
 ----------  -----  --------  --------  --------  --------  -----------  ----------------------------
      6.291      4     1.573     2.097     0.000     2.097        1.049  [CUDA memcpy Host-to-Device]
      0.005    823     0.000     0.000     0.000     0.000        0.000  [CUDA memcpy Device-to-Host]
```

```
 Time (%)  Total Time (ns)  Instances  Avg (ns)  Med (ns)  Min (ns)  Max (ns)  StdDev (ns)                                     Name
 --------  ---------------  ---------  --------  --------  --------  --------  -----------  ----------------------------------------------------------------------------------
     99.8       2294665771     410001    5596.7    5600.0      5535     14656         85.6  UpdateGrid_74(const double *, double *, unsigned long)
      0.2          3451223        822    4198.6    4304.0      3264      6080        767.1  void iamax_kernel<int, double, double, (int)256>(cublasIamaxParams<T1, T2, T3>)
      0.1          2136726        411    5198.8    5184.0      5120      7744        165.5  ComputeError_91(const double *, const double *, double *, unsigned long, cublasContext *)
      0.0             5568          1    5568.0    5568.0      5568      5568          0.0  InitializeGrid_49(double *, double *, unsigned long, double, double, double, double)
```

Размер сетки при профилировании Nsight Systems 512

Итераций <= 410000

Диаграмма сравнения времени выполнения



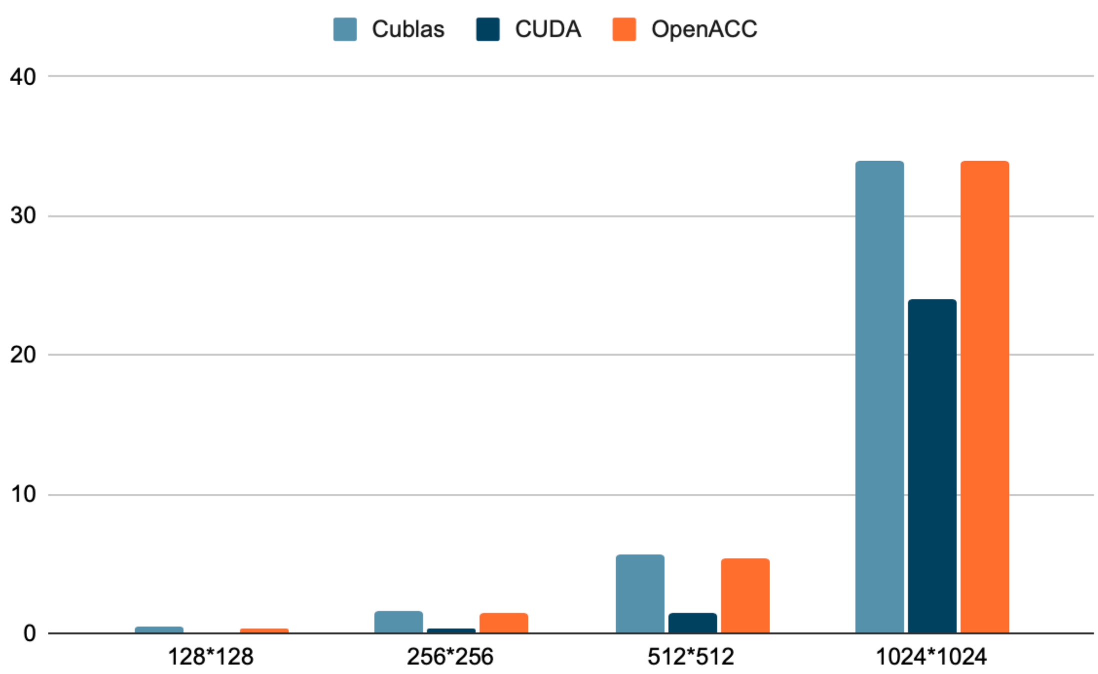
CUDA

Размер сетки	Время выполнения	Точность	Количество итераций
128*128	0.0791	0,000001	37000
256*256	0.2923	0,000001	125000
512*512	1.4933	0,000001	410000
1024*1024	24.0850	0,000001	1000000

```
Time (%)  Total Time (ns)  Instances  Avg (ns)  Med (ns)  Min (ns)  Max (ns)  StdDev (ns)                                    Name
--------  ---------------  ---------  --------  --------  --------  --------  -----------  ----------------------------------------------------------------------------------
   100.0             4224          1    4224.0    4224.0      4224      4224          0.0  InitializeEdges(double *, double *, unsigned long, double, double, double, double)

[7/8] Executing 'cuda_gpu_mem_time_sum' stats report

Time (%)  Total Time (ns)  Count  Avg (ns)  Med (ns)  Min (ns)  Max (ns)  StdDev (ns)           Operation
--------  ---------------  -----  --------  --------  --------  --------  -----------  ----------------------------
    98.2           709331    410    1730.1    1696.0      1664      2431         78.1  [CUDA memcpy Device-to-Host]
     1.8            13280      2    6640.0    6640.0      4960      8320       2375.9  [CUDA memset]

[8/8] Executing 'cuda_gpu_mem_size_sum' stats report

Total (MB)  Count  Avg (MB)  Med (MB)  Min (MB)  Max (MB)  StdDev (MB)           Operation
----------  -----  --------  --------  --------  --------  -----------  ----------------------------
     4.194      2     2.097     2.097     2.097     2.097        0.000  [CUDA memset]
     3.359    410     0.008     0.008     0.008     0.008        0.000  [CUDA memcpy Device-to-Host]
```

Размер сетки при профилировании Nsight Systems 512
Итераций <= 410000

Диаграмма сравнения времени выполнения



Вывод: Скорость выполнения на GPU значительно выше чем на cpu за большего количества профильных вычислительных ядер. Оптимизация кода важна и дает ощутимое ускорение. Профилирование позволяет найти участки, занимающие больше всего времени. CUDA показал бОльшую производительность благодаря использованию графа сокращающего количество запусков ядер.