# Теория параллелизма

Отчет

Задание 8

Выполнила Агапова Александра, 23932 группа

### Выполнение на СРИ

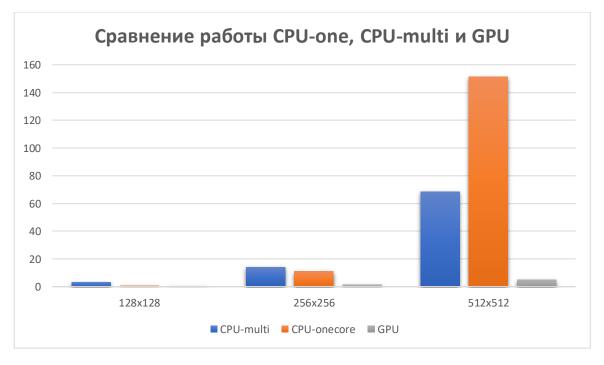
**CPU-onecore** 

Размер сетки	Время	Точность	Количество
	выполнения		итераций
128x128	0.828311	7.53014e-07	1.000.000
256x256	11.0707	9.91166e-07	1.000.000
512x512	151.382	9.92416e-07	1.000.000

### **CPU-multicore**

Размер сетки	Время	Точность	Количество
	выполнения		итераций
128x128	3.38594	7.53014e-07	1.000.000
256x256	14.1547	9.91166e-07	1.000.000
512x512	68.6179	9.92416e-07	1.000.000
1024x1024	461.941	1.37575e-06	1.000.000

# Диаграмма сравнения времени работы CPU-one и CPU-multi



# Выполнение на **GPU** (оптимизированный вариант)

Размер сетки	Время	Точность	Количество
	выполнения		итераций
128x128	0.488777	7.53014e-07	1.000.000
256x256	1.21184	9.91166e-07	1.000.000
512x512	4.74862	9.92416e-07	1.000.000
1024x1024	36.6044	1.37575e-06	1.000.000

## Этапы оптимизации на сетке 512х512

 $\it M$ аксимальное количество итераций -1.000.000

Этап	Время	Точность	Комментарии
$N_{\underline{0}}$	выполнения		
1	148.368	9.99984e-	Начальная версия
		07	_
2	13.15386	9.92416e-	Подсчет ошибки не на каждой
		07	итерации, а каждую 1000
			итерацию
3	8.97609	9.92416e-	Вложенные #pragma acc loop
		07	внутри #pragma acc parallel loop
			заменила на collapse(2)
4	4.74862	9.92416e-	Вместо копирования каждого
		07	элемента из Āпеw в А использую
			swap, который мгновенно меняет
			местами указатели A и Anew

1)

0s <b>▼</b>	⊦859,	5ms	859,966ms	+860,5ms		
CUDA HW (0000:89:00.0 - Tesla V100-SXI	Kernel Memory			l l		
▼ 51,8% Kernels			L I			
▶ 35,1% _10second_cpp_main_98_gpu						
▶ 31,7% _10second_cpp_main_81_gpu				_cpp_main_98_gpu		
▶ 24,5% _10second_cpp_main_106_gpu				Begins: 0,859964s Ends: 0,859977s (+13,344 µs)		
▶ 8,7% _10second_cpp_main_98_gpure			grid: <<<510 block: <<<12	, 1, 1>>>		

98 строка – подсчет ошибки на каждой итерации

0s ▼		+ <sup>8</sup> 82	6,639n	ns	+826	5,8m	S	4	-827m	IS	
▼ CUDA HW (0000:89:00.0 - Tesla V100-SXI	Kernel Memory			П	11	П	П	H	Ш		1
▼ 24,9% Kernels				П	П				Ш		
▶ 60,5% _10second_cpp_main_81_gpu											
➤ 37,1% _10second_cpp_main_106_gpu											
2 kernel groups hidden — +				_10second_cpp_main_106_gpu Begins: 0,826639s							
▶ 75,1% Memory			Er	Ends: 0,826643s (+4,544 µs) grid: <<<510, 1, 1>>>							
NVTX			block: <<<128, 1, 1>>>								
▼ Threads (4)	Launch Type: Regular										

 $81\ cmpoka$  — #pragma acc parallel loop present(A, Anew) цикл обновления значений в матрице Anew на основе соседних значений из A

3)

0s <b>▼</b>			987,3	354ms <sup>7</sup>	,4ms		 +	987,6	ms
▼ CUDA HW (0000:89:00.0 - Tesla V10	Kernel Memory				1			L	
▼ 23,8% Kernels									
> 53,6% _9first_cpp_main_81_gpu									
▶ 43,2% _9first_cpp_main_104_gpu				_ <b>9first_cpp_main_81_gpu</b> Begins: 0,987351s Ends: 0,987357s (+6,688 μs)					
2 kernel groups hidden — +									
▼ 76,2% Memory		grid: <<<2033, 1, 1>>> block: <<<128, 1, 1>>>							

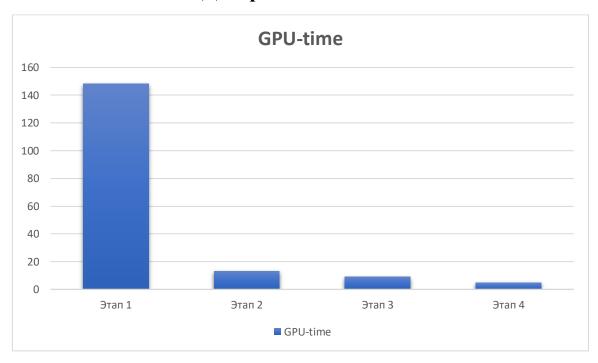
81 строка - #pragma acc parallel loop collapse(2) present(A, Anew) объединила два вложенных цикла в один двухмерный параллельный цикл

4)

' <i>)</i>								
0s <b>▼</b>		1ms	673,21	30ms	+673,3m	S	+673,4	lms
▼ CUDA HW (0000:89:00.0 - Tesla V100-	Kernel Memory	_	1	1				
▼ 16,7% Kernels								
▶ 94,4% _8main_cpp_main_93_gpu								
▶ 3,8% _8main_cpp_main_110_gpu								
▶ 1,8% _8main_cpp_main_110_gpu					cpp_mail	ո_110_ց	jpu	
▶ 83,3% Memory				Ends: 0,6	,673207s 73215s (+		)	
NVTX				_	2033, 1, 1> <128, 1, 1>			

После замены копирования элементов на swap пропала kernel копирования

#### Диаграмма оптимизации

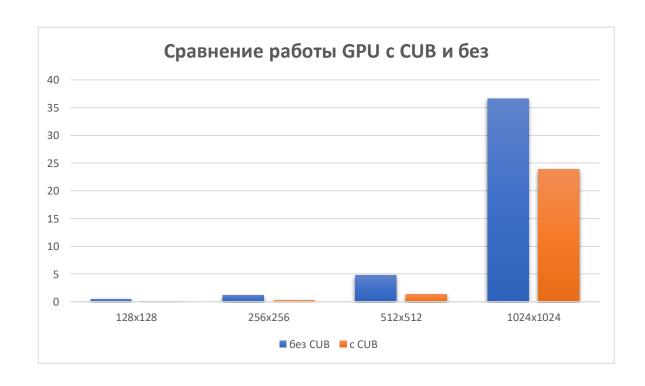


```
Посчитанная матрица 10 х 10
0, ошибка = 14.4444
10
        11.1111 12.2222 13.3333 14.4444 15.5556 16.6667 17.7778 18.8889 20
11.1111 12.2222 13.3333 14.4444 15.5556 16.6667 17.7778 18.8889 20
                                                                        21.1111
12.2222 13.3333 14.4444 15.5556 16.6667 17.7778 18.8889 20
                                                                21.1111 22.2222
13.3333 14.4444 15.5556 16.6667 17.7778 18.8889 20
                                                        21.1111 22.2222 23.3333
14.4444 15.5556 16.6667 17.7778 18.8889 20
                                                21.1111 22.2222 23.3333 24.4444
                                        21.1111 22.2222 23.3333 24.4444 25.5556
15.5556 16.6667 17.7778 18.8889 20
16.6667 17.7778 18.8889 20
                                21.1111 22.2222 23.3333 24.4444 25.5556 26.6667
17.7778 18.8889 20
                        21.1111 22.2222 23.3333 24.4444 25.5556 26.6667 27.7778
                21.1111 22.2222 23.3333 24.4444 25.5556 26.6667 27.7778 28.8889
        21.1111 22.2222 23.3333 24.4444 25.5556 26.6667 27.7778 28.8889 30
1001 итераций
Ошибка: 0
Время: 0.193013 секунд
```

```
Посчитанная матрица 13 х 13
0, ошибка = 14.5833
     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 <u>20</u>
                                                        20.8333 21.6667 22.5 23.3333 24.1667
      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
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
       20.8333 21.6667 22.5 23.3333 24.1667 25
                                                 25.8333 26.6667 27.5
                                                                      28.3333 29.1667 30
1001 итераций
Ошибка: 3.55271e-15
Время: 0.181927 секунд
```

## Реализация с библиотекой CUB

Размер сетки	Время	Точность	Количество
	выполнения		итераций
128x128	0.0727762	7.53014e-07	1.000.000
256x256	0.278514	9.91166e-07	1.000.000
512x512	1.33679	9.92416e-07	1.000.000
1024x1024	23.8688	1.36929e-06	1.000.000

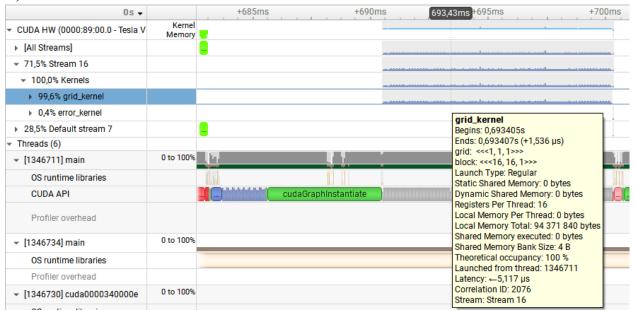


### Этапы оптимизации на сетке 512x512

 $\it M$ аксимальное количество итераций -1.000.000

Этап	Время	Точность	Комментарии
No	выполнения		
1	4.59136	9.84603e-07	Начальная версия
	1.10565	0.00405.05	
2	1.19765	9.92435e-07	Единый крупный запуск
			ядра, убрала лишние
			cudaStreamSynchronize()





Вычисления ядра grid\_kernel были раздроблены на множество мелких запусков (по одному блоку на запуск), каждый из которых синхронизировался вручную с CPU через cudaStreamSynchronize(). Даже несмотря на то, что каждый запуск занимал доли миллисекунды, общее количество привело к 99% загрузке GPU по времени



