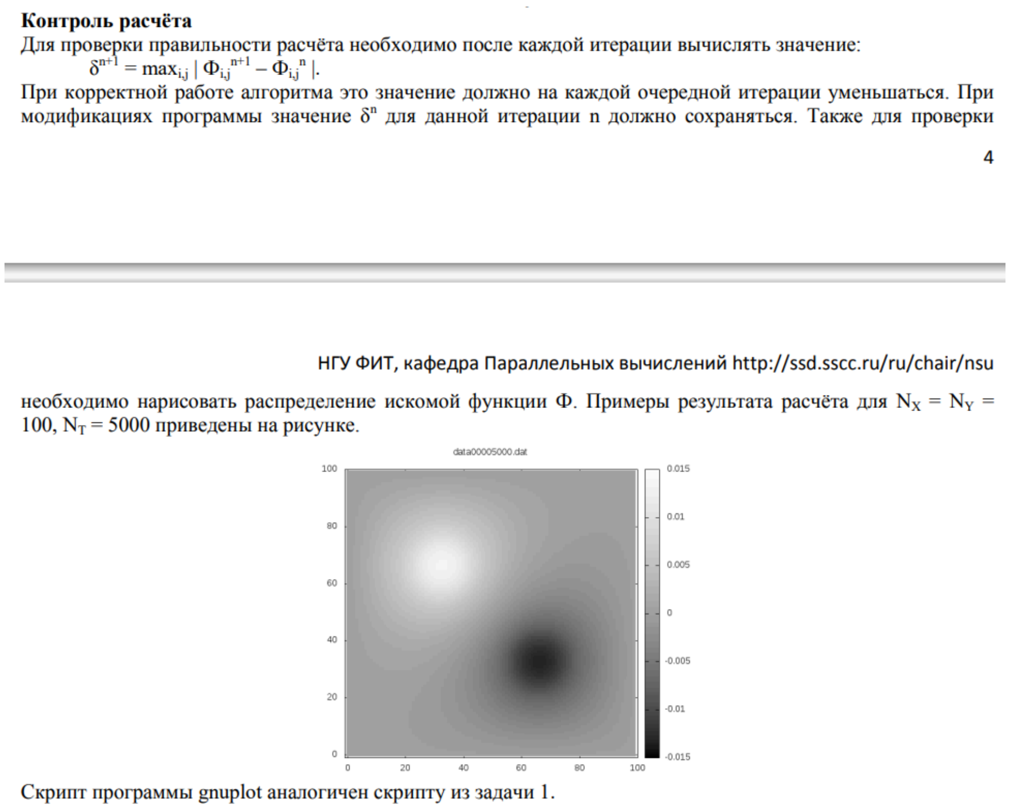
Отчет по лабораторной работе №1

Правильность работы программы



Контроль расчёта

Для проверки правильности расчёта необходимо после каждой итерации вычислять значение:

$$\delta^{n+1} = \max_{i,j} | \Phi_{i,j}^{n+1} - \Phi_{i,j}^{n} |.$$

При корректной работе алгоритма это значение должно на каждой очередной итерации уменьшаться. При модификациях программы значение δ^n для данной итерации n должно сохраняться. Также для проверки

4

НГУ ФИТ, кафедра Параллельных вычислений http://ssd.sscc.ru/ru/chair/nsu

необходимо нарисовать распределение искомой функции Φ. Примеры результата расчёта для $N_X = N_Y = 100$, $N_T = 5000$ приведены на рисунке.

Скрипт программы gnuplot аналогичен скрипту из задачи 1.

$N_x = N_y = 100, N_t = 5000$

Сходимость дельта:

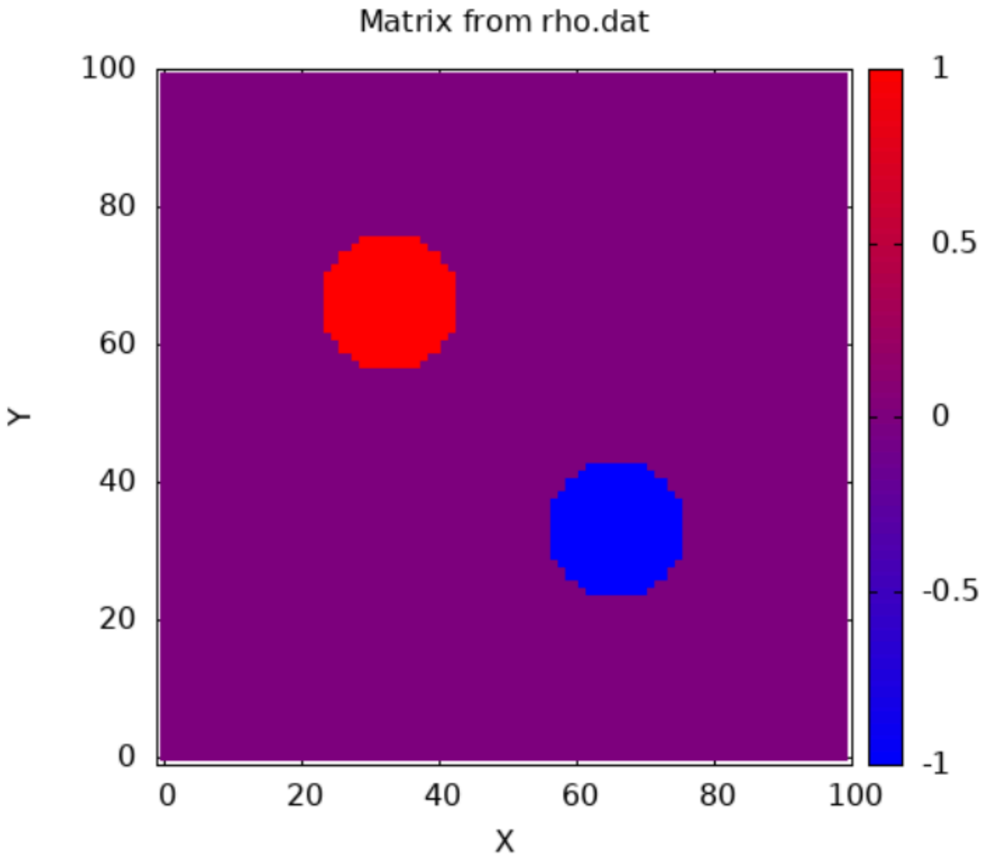
```
[0]globalDelta = 0.000490
[1]globalDelta = 0.000490
```

```
[2]globalDelta = 0.000490
[3]globalDelta = 0.000490
[4]globalDelta = 0.000490
[5]globalDelta = 0.000490
[6]globalDelta = 0.000490
[7]globalDelta = 0.000490
[8]globalDelta = 0.000490
[9]globalDelta = 0.000490
[10]globalDelta = 0.000490
[11]globalDelta = 0.000490
[12]globalDelta = 0.000489
[13]globalDelta = 0.000489
[14]globalDelta = 0.000489
[15]globalDelta = 0.000488
[16]globalDelta = 0.000487
[17]globalDelta = 0.000486
[18]globalDelta = 0.000485
[19]globalDelta = 0.000483
[20]globalDelta = 0.000481
[21]globalDelta = 0.000480
[22]globalDelta = 0.000478
[23]globalDelta = 0.000475
[24]globalDelta = 0.000473
[25]globalDelta = 0.000470
[26]globalDelta = 0.000468
[27]globalDelta = 0.000465
[28]globalDelta = 0.000462
[29]globalDelta = 0.000459
[30]globalDelta = 0.000456
...
[4990]globalDelta = 0.000000
[4991]globalDelta = 0.000000
[4992]globalDelta = 0.000000
[4993]globalDelta = 0.000000
```

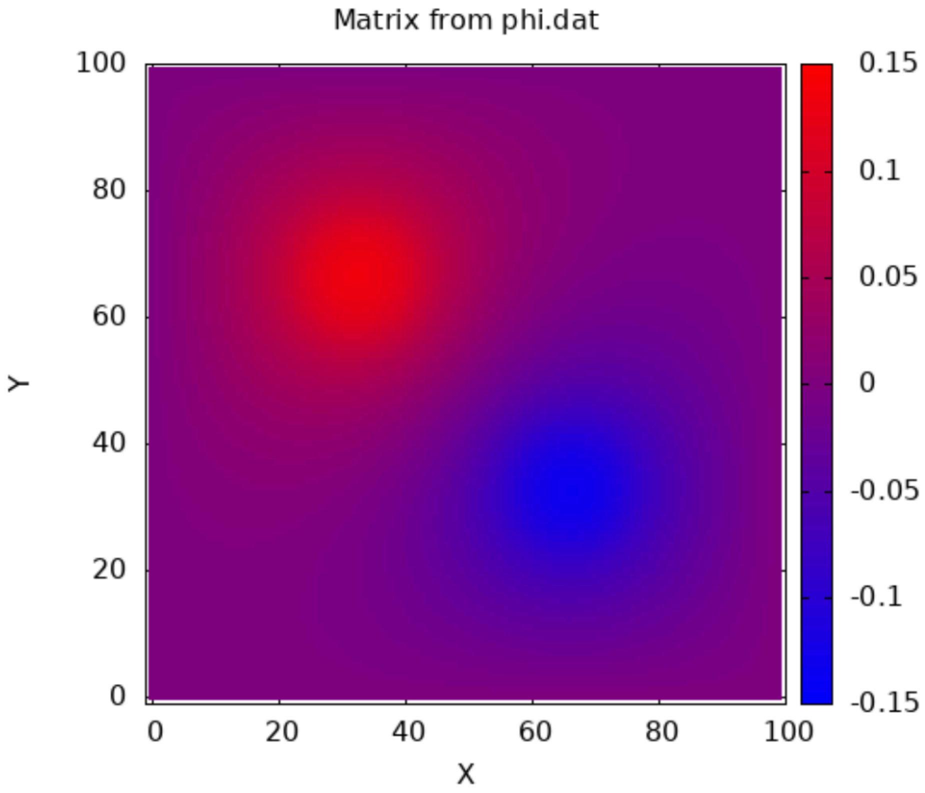
```
[4994]globalDelta = 0.000000
[4995]globalDelta = 0.000000
[4996]globalDelta = 0.000000
[4997]globalDelta = 0.000000
[4998]globalDelta = 0.000000
[4999]globalDelta = 0.000000
[5000]globalDelta = 0.000000
```

Получившийся рисунок:

Начальное распределение тепла на пластине



Результат



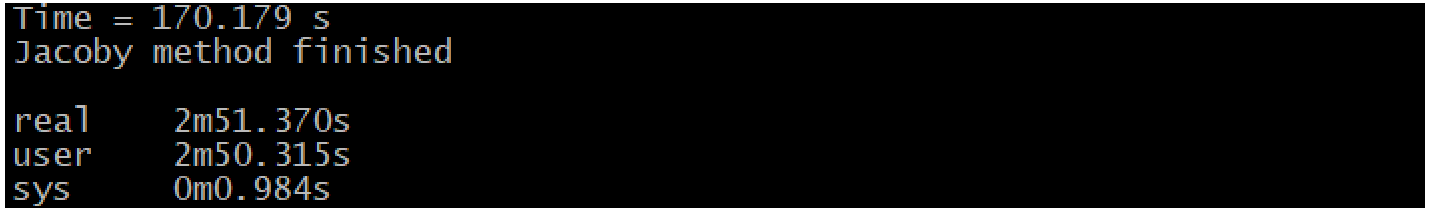
Оптимизации

- Хранение матрицы в виде одномерного массива, а не двумерного
- Swap указателей при подсчета итерации матрицы
- Оптимизации компилятора (`-Ofast`)
- Убрал "лишние вычисления" - оптимизация формулы подсчета
    - вместо деления на 4 - умножение на 0,25 и т. п.
    - подсчет коэффициентов итерационной формулы
- Использование глобальных переменных

Замеры времени работы программы

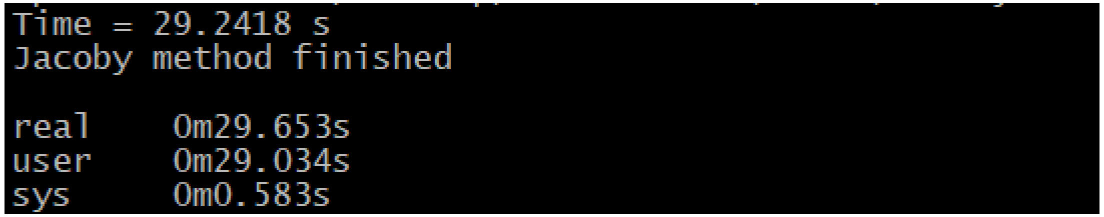
$N_x = N_y = 8000, N_t = 100$

Без оптимизаций компилятора



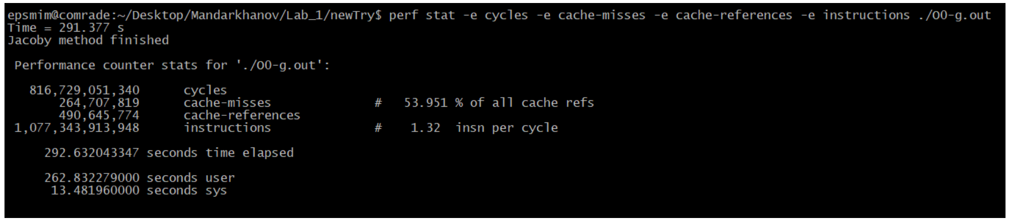
```
Time = 170.179 s
Jacoby method finished

real    2m51.370s
user    2m50.315s
sys     0m0.984s
```

Ofast



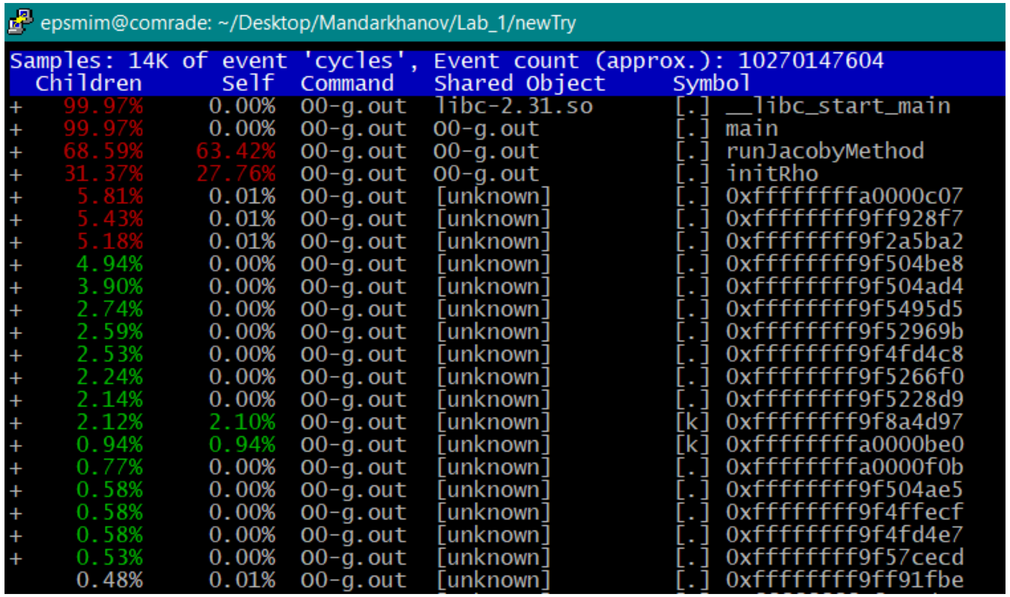
```
Time = 29.2418 s
Jacoby method finished

real    0m29.653s
user    0m29.034s
sys     0m0.583s
```

Профилирование



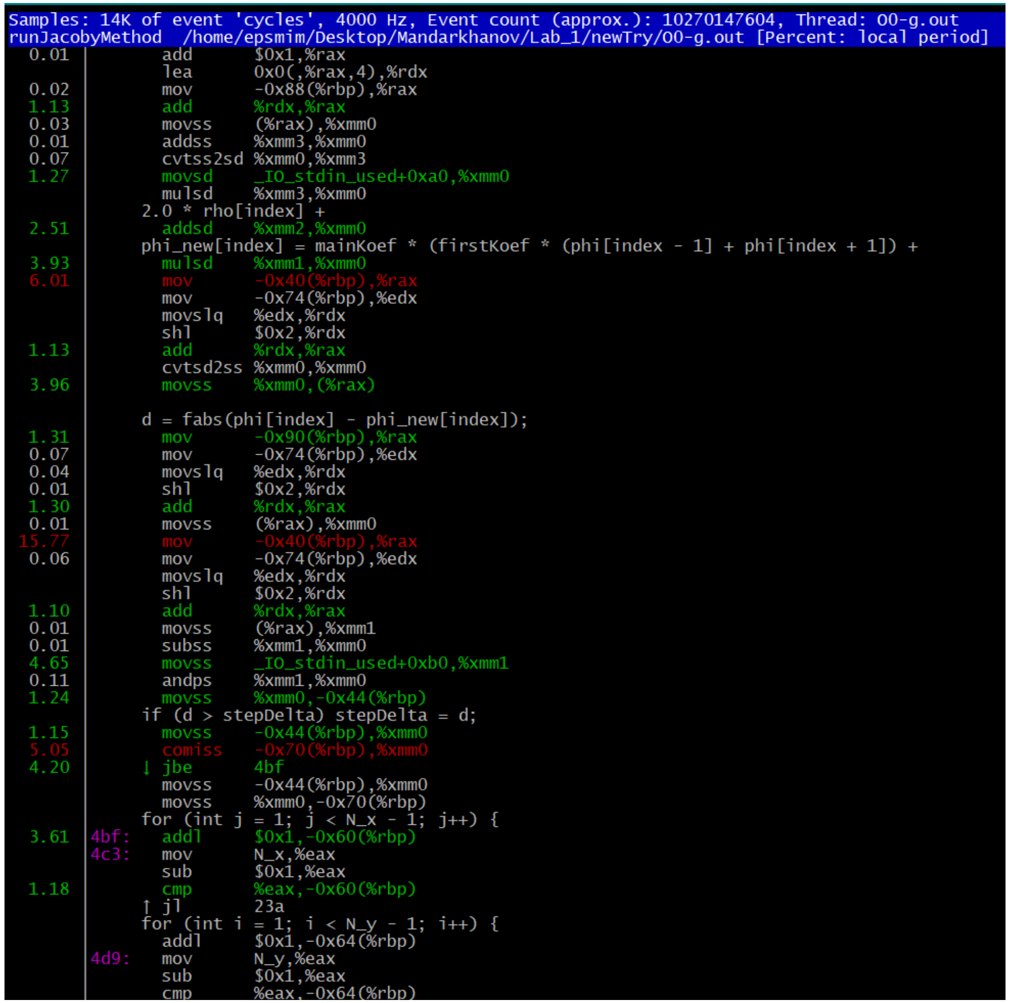
```
epsmim@comrade:~/Desktop/Mandarkhanov/Lab_1/newTry$ perf stat -e cycles -e cache-misses -e cache-references -e instructions ./O0-g.out
Time = 291.377 s
Jacoby method finished

 Performance counter stats for './O0-g.out':

   816,729,051,340      cycles
       264,707,819      cache-misses              #   53.951 % of all cache refs
       490,645,774      cache-references
 1,077,343,913,948      instructions              #    1.32  insn per cycle

     292.632043347 seconds time elapsed

     262.832279000 seconds user
      13.481960000 seconds sys
```

cycles



```
Samples: 14K of event 'cycles', Event count (approx.): 10270147604
  Children      Self  Command     Shared Object      Symbol
+   99.97%     0.00%  O0-g.out    libc-2.31.so       [.] __libc_start_main
+   99.97%     0.00%  O0-g.out    O0-g.out           [.] main
+   68.59%    63.42%  O0-g.out    O0-g.out           [.] runJacobyMethod
+   31.37%    27.76%  O0-g.out    O0-g.out           [.] initRho
+    5.81%     0.01%  O0-g.out    [unknown]          [.] 0xffffffffa0000c07
+    5.43%     0.01%  O0-g.out    [unknown]          [.] 0xffffffff9ff928f7
+    5.18%     0.01%  O0-g.out    [unknown]          [.] 0xffffffff9f2a5ba2
+    4.94%     0.00%  O0-g.out    [unknown]          [.] 0xffffffff9f504be8
+    3.90%     0.00%  O0-g.out    [unknown]          [.] 0xffffffff9f504ad4
+    2.74%     0.00%  O0-g.out    [unknown]          [.] 0xffffffff9f5495d5
+    2.59%     0.00%  O0-g.out    [unknown]          [.] 0xffffffff9f52969b
+    2.53%     0.00%  O0-g.out    [unknown]          [.] 0xffffffff9f4fd4c8
+    2.24%     0.00%  O0-g.out    [unknown]          [.] 0xffffffff9f5266f0
+    2.14%     0.00%  O0-g.out    [unknown]          [.] 0xffffffff9f5228d9
+    2.12%     2.10%  O0-g.out    [unknown]          [k] 0xffffffff9f8a4d97
+    0.94%     0.94%  O0-g.out    [unknown]          [k] 0xffffffffa0000be0
+    0.77%     0.00%  O0-g.out    [unknown]          [.] 0xffffffffa0000f0b
+    0.58%     0.00%  O0-g.out    [unknown]          [.] 0xffffffff9f504ae5
+    0.58%     0.00%  O0-g.out    [unknown]          [.] 0xffffffff9f4ffecf
+    0.58%     0.00%  O0-g.out    [unknown]          [.] 0xffffffff9f4fd4e7
+    0.53%     0.00%  O0-g.out    [unknown]          [.] 0xffffffff9f57cecd
     0.48%     0.01%  O0-g.out    [unknown]          [.] 0xffffffff9ff91fbe
```

asm runJacobyMethod



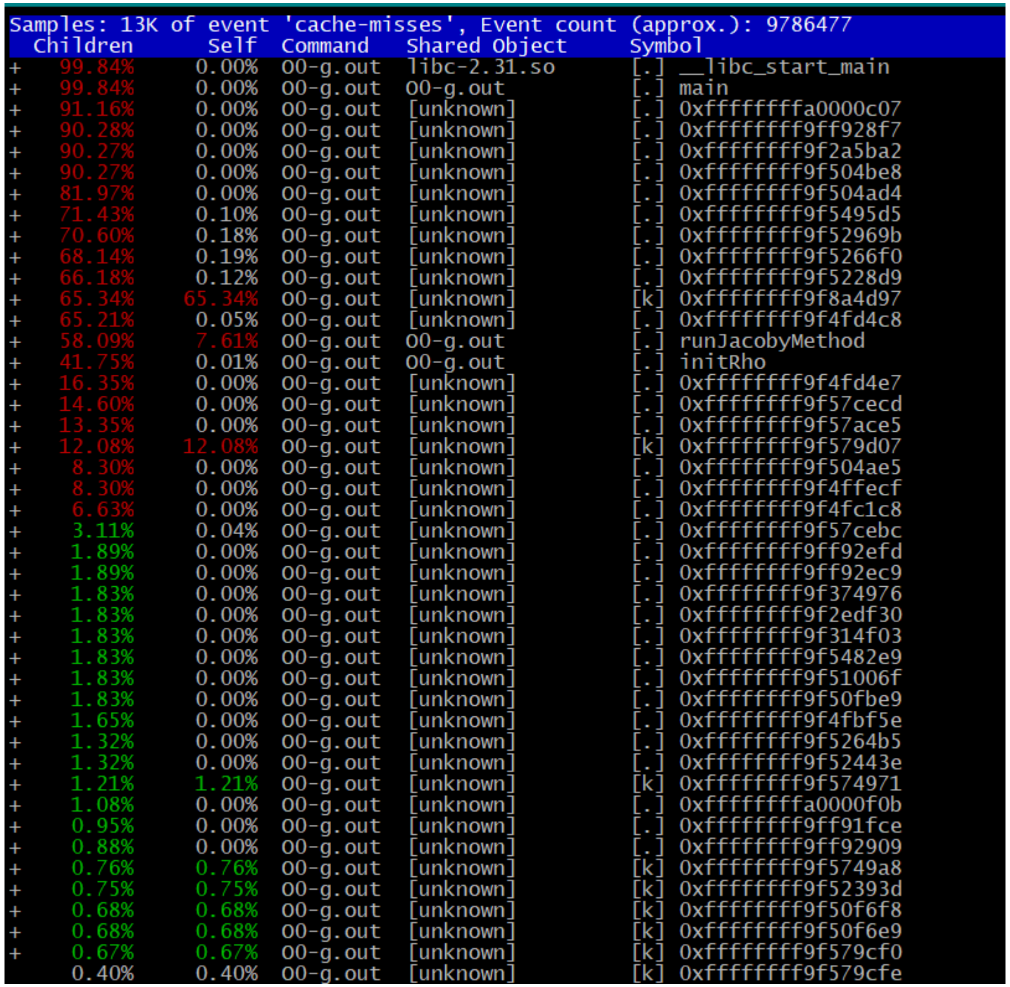
```
Samples: 14K of event 'cycles', 4000 Hz, Event count (approx.): 10270147604, Thread: O0-g.out
runJacobyMethod  /home/epsmim/Desktop/Mandarkhanov/Lab_1/newTry/O0-g.out [Percent: local period]
  0.01 │           add      $0x1,%rax
       │           lea      0x0(,%rax,4),%rdx
  0.02 │           mov      -0x88(%rbp),%rax
  1.13 │           add      %rdx,%rax
  0.03 │           movss    (%rax),%xmm0
  0.01 │           addss    %xmm3,%xmm0
  0.07 │           cvtss2sd %xmm0,%xmm3
  1.27 │           movsd    _IO_stdin_used+0xa0,%xmm0
       │           mulsd    %xmm3,%xmm0
       │         2.0 * rho[index] +
  2.51 │           addsd    %xmm2,%xmm0
       │         phi_new[index] = mainKoef * (firstKoef * (phi[index - 1] + phi[index + 1]) +
  3.93 │           mulsd    %xmm1,%xmm0
  6.01 │           mov      -0x40(%rbp),%rax
       │           mov      -0x74(%rbp),%edx
       │           movslq   %edx,%rdx
       │           shl      $0x2,%rdx
  1.13 │           add      %rdx,%rax
       │           cvtsd2ss %xmm0,%xmm0
  3.96 │           movss    %xmm0,(%rax)
       │
       │         d = fabs(phi[index] - phi_new[index]);
  1.31 │           mov      -0x90(%rbp),%rax
  0.07 │           mov      -0x74(%rbp),%edx
  0.04 │           movslq   %edx,%rdx
  0.01 │           shl      $0x2,%rdx
  1.30 │           add      %rdx,%rax
  0.01 │           movss    (%rax),%xmm0
 15.77 │           mov      -0x40(%rbp),%rax
  0.06 │           mov      -0x74(%rbp),%edx
       │           movslq   %edx,%rdx
       │           shl      $0x2,%rdx
  1.10 │           add      %rdx,%rax
  0.01 │           movss    (%rax),%xmm1
  0.01 │           subss    %xmm1,%xmm0
  4.65 │           movss    _IO_stdin_used+0xb0,%xmm1
  0.11 │           andps    %xmm1,%xmm0
  1.24 │           movss    %xmm0,-0x44(%rbp)
       │         if (d > stepDelta) stepDelta = d;
  1.15 │           movss    -0x44(%rbp),%xmm0
  5.05 │           comiss   -0x70(%rbp),%xmm0
  4.20 │         ↓ jbe      4bf
       │           movss    -0x44(%rbp),%xmm0
       │           movss    %xmm0,-0x70(%rbp)
       │         for (int j = 1; j < N_x - 1; j++) {
  3.61 │4bf:       addl     $0x1,-0x60(%rbp)
       │4c3:       mov      N_x,%eax
       │           sub      $0x1,%eax
  1.18 │           cmp      %eax,-0x60(%rbp)
       │         ↑ jl       23a
       │         for (int i = 1; i < N_y - 1; i++) {
       │           addl     $0x1,-0x64(%rbp)
       │4d9:       mov      N_y,%eax
       │           sub      $0x1,%eax
       │           cmp      %eax,-0x64(%rbp)
```

Как видно из ассемблерного листинга, большая часть тактов уходит на команды связанные:

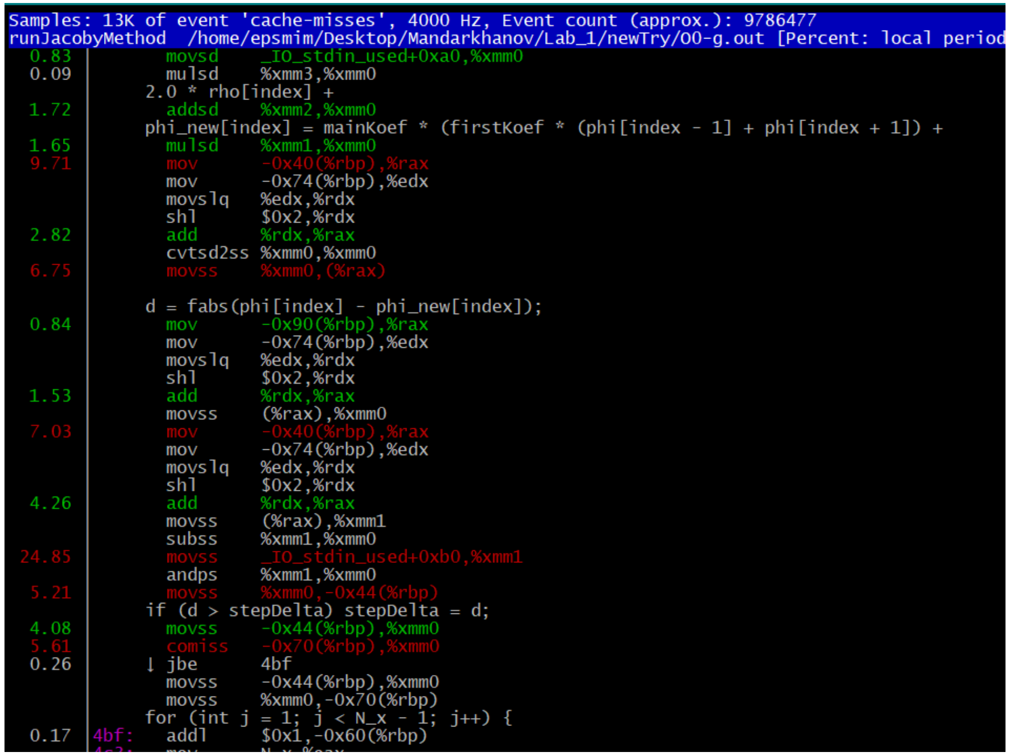
- с работой с памятью (mov, movss)
- арифметикой с плавающей точной (comiss - сравнение, mulsdm addsd)

  Что и логично, ведь основная часть работы программы уходит на арифметические операции и работу с памятью

cache-misses



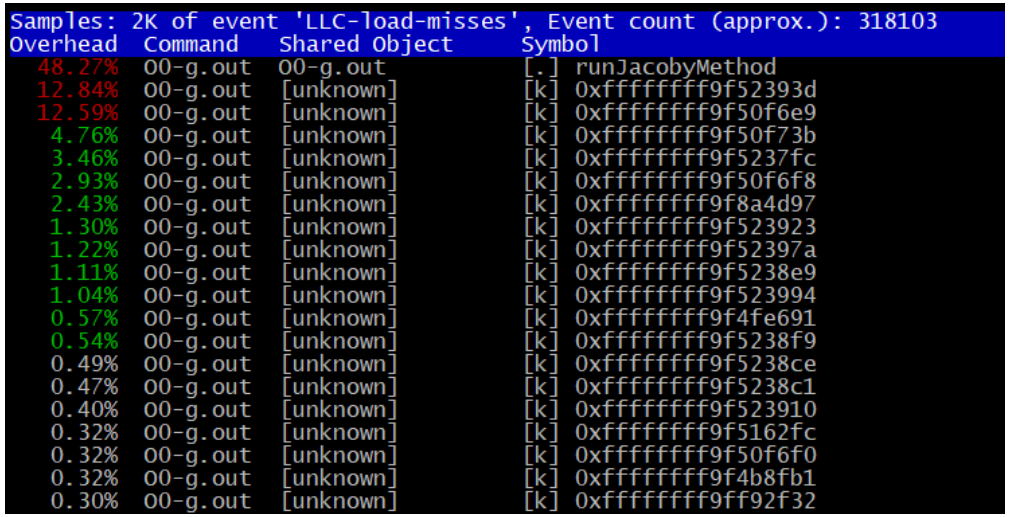
```
Samples: 13K of event 'cache-misses', Event count (approx.): 9786477
  Children      Self  Command   Shared Object       Symbol
+   99.84%     0.00%  00-g.out  libc-2.31.so        [.] __libc_start_main
+   99.84%     0.00%  00-g.out  00-g.out            [.] main
+   91.16%     0.00%  00-g.out  [unknown]           [.] 0xffffffffa0000c07
+   90.28%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9ff928f7
+   90.27%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f2a5ba2
+   90.27%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f504be8
+   81.97%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f504ad4
+   71.43%     0.10%  00-g.out  [unknown]           [.] 0xffffffff9f5495d5
+   70.60%     0.18%  00-g.out  [unknown]           [.] 0xffffffff9f52969b
+   68.14%     0.19%  00-g.out  [unknown]           [.] 0xffffffff9f5266f0
+   66.18%     0.12%  00-g.out  [unknown]           [.] 0xffffffff9f5228d9
+   65.34%    65.34%  00-g.out  [unknown]           [k] 0xffffffff9f8a4d97
+   65.21%     0.05%  00-g.out  [unknown]           [.] 0xffffffff9f4fd4c8
+   58.09%     7.61%  00-g.out  00-g.out            [.] runJacobyMethod
+   41.75%     0.01%  00-g.out  00-g.out            [.] initRho
+   16.35%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f4fd4e7
+   14.60%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f57cecd
+   13.35%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f57ace5
+   12.08%    12.08%  00-g.out  [unknown]           [k] 0xffffffff9f579d07
+    8.30%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f504ae5
+    8.30%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f4ffecf
+    6.63%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f4fc1c8
+    3.11%     0.04%  00-g.out  [unknown]           [.] 0xffffffff9f57cebc
+    1.89%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9ff92efd
+    1.89%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9ff92ec9
+    1.83%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f374976
+    1.83%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f2edf30
+    1.83%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f314f03
+    1.83%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f5482e9
+    1.83%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f51006f
+    1.83%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f50fbe9
+    1.65%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f4fbf5e
+    1.32%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f5264b5
+    1.32%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9f52443e
+    1.21%     1.21%  00-g.out  [unknown]           [k] 0xffffffff9f574971
+    1.08%     0.00%  00-g.out  [unknown]           [.] 0xffffffffa0000f0b
+    0.95%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9ff91fce
+    0.88%     0.00%  00-g.out  [unknown]           [.] 0xffffffff9ff92909
+    0.76%     0.76%  00-g.out  [unknown]           [k] 0xffffffff9f5749a8
+    0.75%     0.75%  00-g.out  [unknown]           [k] 0xffffffff9f52393d
+    0.68%     0.68%  00-g.out  [unknown]           [k] 0xffffffff9f50f6f8
+    0.68%     0.68%  00-g.out  [unknown]           [k] 0xffffffff9f50f6e9
+    0.67%     0.67%  00-g.out  [unknown]           [k] 0xffffffff9f579cf0
     0.40%     0.40%  00-g.out  [unknown]           [k] 0xffffffff9f579cfe
```



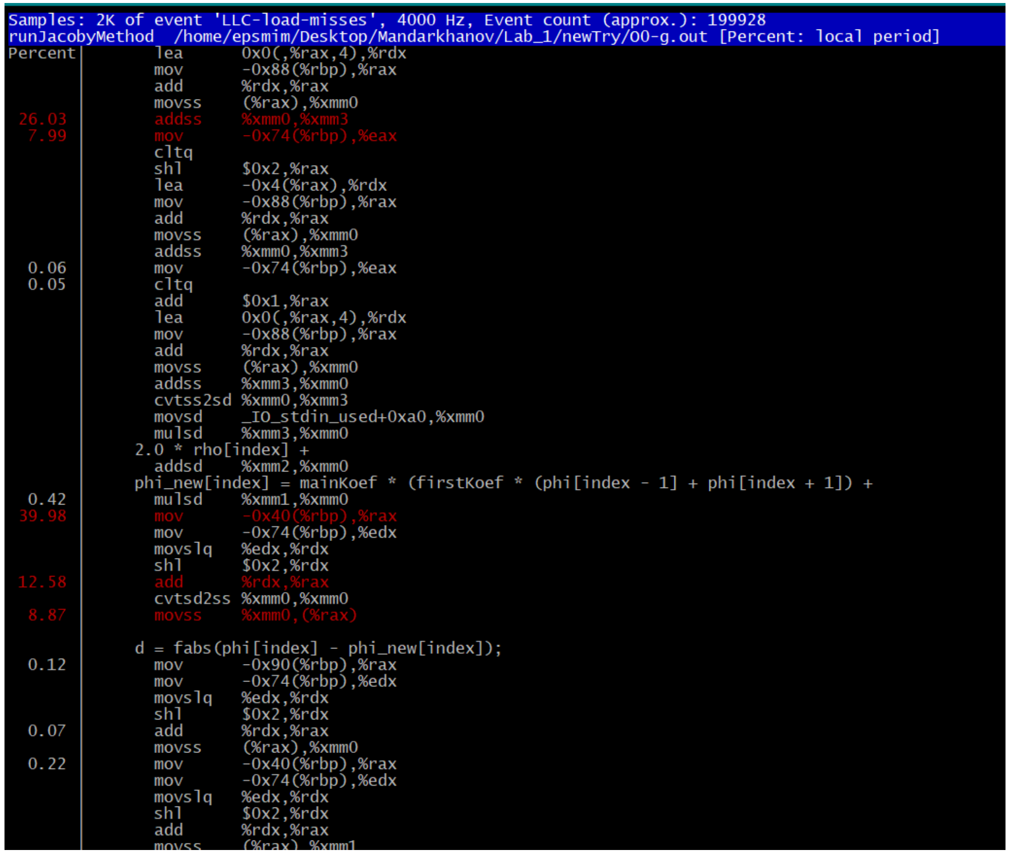
```
Samples: 13K of event 'cache-misses', 4000 Hz, Event count (approx.): 9786477
runJacobyMethod  /home/epsmim/Desktop/Mandarkhanov/Lab_1/newTry/O0-g.out [Percent: local period
  0.83 │       movsd    _IO_stdin_used+0xa0,%xmm0
  0.09 │       mulsd    %xmm3,%xmm0
       │     2.0 * rho[index] +
  1.72 │       addsd    %xmm2,%xmm0
       │     phi_new[index] = mainKoef * (firstKoef * (phi[index - 1] + phi[index + 1]) +
  1.65 │       mulsd    %xmm1,%xmm0
  9.71 │       mov      -0x40(%rbp),%rax
       │       mov      -0x74(%rbp),%edx
       │       movslq   %edx,%rdx
       │       shl      $0x2,%rdx
  2.82 │       add      %rdx,%rax
       │       cvtsd2ss %xmm0,%xmm0
  6.75 │       movss    %xmm0,(%rax)
       │
       │     d = fabs(phi[index] - phi_new[index]);
  0.84 │       mov      -0x90(%rbp),%rax
       │       mov      -0x74(%rbp),%edx
       │       movslq   %edx,%rdx
       │       shl      $0x2,%rdx
  1.53 │       add      %rdx,%rax
       │       movss    (%rax),%xmm0
  7.03 │       mov      -0x40(%rbp),%rax
       │       mov      -0x74(%rbp),%edx
       │       movslq   %edx,%rdx
       │       shl      $0x2,%rdx
  4.26 │       add      %rdx,%rax
       │       movss    (%rax),%xmm1
       │       subss    %xmm1,%xmm0
 24.85 │       movss    _IO_stdin_used+0xb0,%xmm1
       │       andps    %xmm1,%xmm0
  5.21 │       movss    %xmm0,-0x44(%rbp)
       │     if (d > stepDelta) stepDelta = d;
  4.08 │       movss    -0x44(%rbp),%xmm0
  5.61 │       comiss   -0x70(%rbp),%xmm0
  0.26 │     ↓ jbe      4bf
       │       movss    -0x44(%rbp),%xmm0
       │       movss    %xmm0,-0x70(%rbp)
       │     for (int j = 1; j < N_x - 1; j++) {
  0.17 │4bf:   addl     $0x1,-0x60(%rbp)
```

Из асемблерного листинга видно, что основная часть кэш-промахов идет на операции с памятью

LLC-load-misses

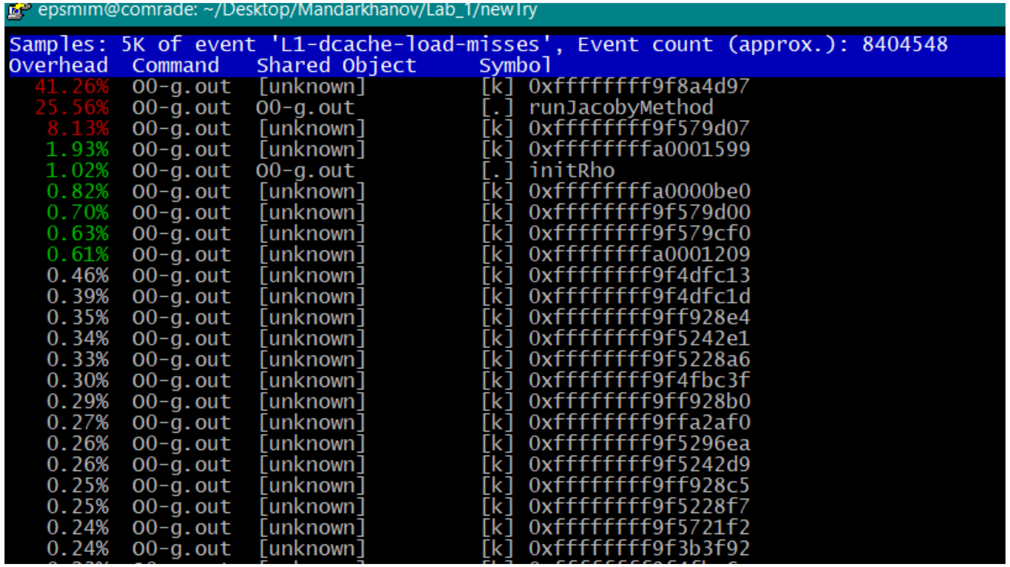


```
Samples: 2K of event 'LLC-load-misses', Event count (approx.): 318103
Overhead  Command   Shared Object      Symbol
  48.27%  O0-g.out  O0-g.out           [.] runJacobyMethod
  12.84%  O0-g.out  [unknown]          [k] 0xffffffff9f52393d
  12.59%  O0-g.out  [unknown]          [k] 0xffffffff9f50f6e9
   4.76%  O0-g.out  [unknown]          [k] 0xffffffff9f50f73b
   3.46%  O0-g.out  [unknown]          [k] 0xffffffff9f5237fc
   2.93%  O0-g.out  [unknown]          [k] 0xffffffff9f50f6f8
   2.43%  O0-g.out  [unknown]          [k] 0xffffffff9f8a4d97
   1.30%  O0-g.out  [unknown]          [k] 0xffffffff9f523923
   1.22%  O0-g.out  [unknown]          [k] 0xffffffff9f52397a
   1.11%  O0-g.out  [unknown]          [k] 0xffffffff9f5238e9
   1.04%  O0-g.out  [unknown]          [k] 0xffffffff9f523994
   0.57%  O0-g.out  [unknown]          [k] 0xffffffff9f4fe691
   0.54%  O0-g.out  [unknown]          [k] 0xffffffff9f5238f9
   0.49%  O0-g.out  [unknown]          [k] 0xffffffff9f5238ce
   0.47%  O0-g.out  [unknown]          [k] 0xffffffff9f5238c1
   0.40%  O0-g.out  [unknown]          [k] 0xffffffff9f523910
   0.32%  O0-g.out  [unknown]          [k] 0xffffffff9f5162fc
   0.32%  O0-g.out  [unknown]          [k] 0xffffffff9f50f6f0
   0.32%  O0-g.out  [unknown]          [k] 0xffffffff9f4b8fb1
   0.30%  O0-g.out  [unknown]          [k] 0xffffffff9ff92f32
```



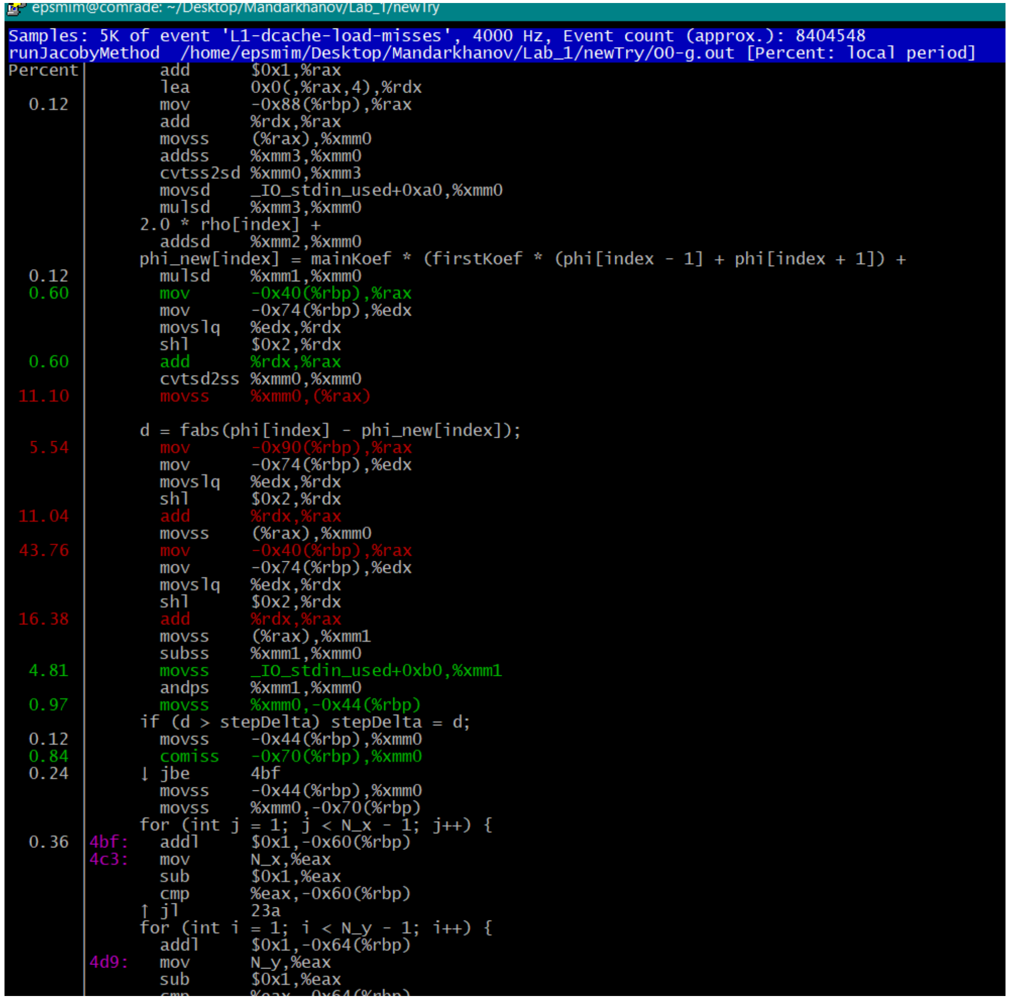
```
Samples: 2K of event 'LLC-load-misses', 4000 Hz, Event count (approx.): 199928
runJacobyMethod  /home/epsmim/Desktop/Mandarkhanov/Lab_1/newTry/O0-g.out [Percent: local period]
Percent│         lea      0x0(,%rax,4),%rdx
       │         mov      -0x88(%rbp),%rax
       │         add      %rdx,%rax
       │         movss    (%rax),%xmm0
  26.03│         addss    %xmm0,%xmm3
   7.99│         mov      -0x74(%rbp),%eax
       │         cltq
       │         shl      $0x2,%rax
       │         lea      -0x4(%rax),%rdx
       │         mov      -0x88(%rbp),%rax
       │         add      %rdx,%rax
       │         movss    (%rax),%xmm0
       │         addss    %xmm0,%xmm3
   0.06│         mov      -0x74(%rbp),%eax
   0.05│         cltq
       │         add      $0x1,%rax
       │         lea      0x0(,%rax,4),%rdx
       │         mov      -0x88(%rbp),%rax
       │         add      %rdx,%rax
       │         movss    (%rax),%xmm0
       │         addss    %xmm3,%xmm0
       │         cvtss2sd %xmm0,%xmm3
       │         movsd    _IO_stdin_used+0xa0,%xmm0
       │         mulsd    %xmm3,%xmm0
       │       2.0 * rho[index] +
       │         addsd    %xmm2,%xmm0
       │       phi_new[index] = mainKoef * (firstKoef * (phi[index - 1] + phi[index + 1]) +
   0.42│         mulsd    %xmm1,%xmm0
  39.98│         mov      -0x40(%rbp),%rax
       │         mov      -0x74(%rbp),%edx
       │         movslq   %edx,%rdx
       │         shl      $0x2,%rdx
  12.58│         add      %rdx,%rax
       │         cvtsd2ss %xmm0,%xmm0
   8.87│         movss    %xmm0,(%rax)
       │
       │       d = fabs(phi[index] - phi_new[index]);
   0.12│         mov      -0x90(%rbp),%rax
       │         mov      -0x74(%rbp),%edx
       │         movslq   %edx,%rdx
       │         shl      $0x2,%rdx
   0.07│         add      %rdx,%rax
       │         movss    (%rax),%xmm0
   0.22│         mov      -0x40(%rbp),%rax
       │         mov      -0x74(%rbp),%edx
       │         movslq   %edx,%rdx
       │         shl      $0x2,%rdx
       │         add      %rdx,%rax
       │         movss    (%rax),%xmm1
```

L1-dcache-load-misses



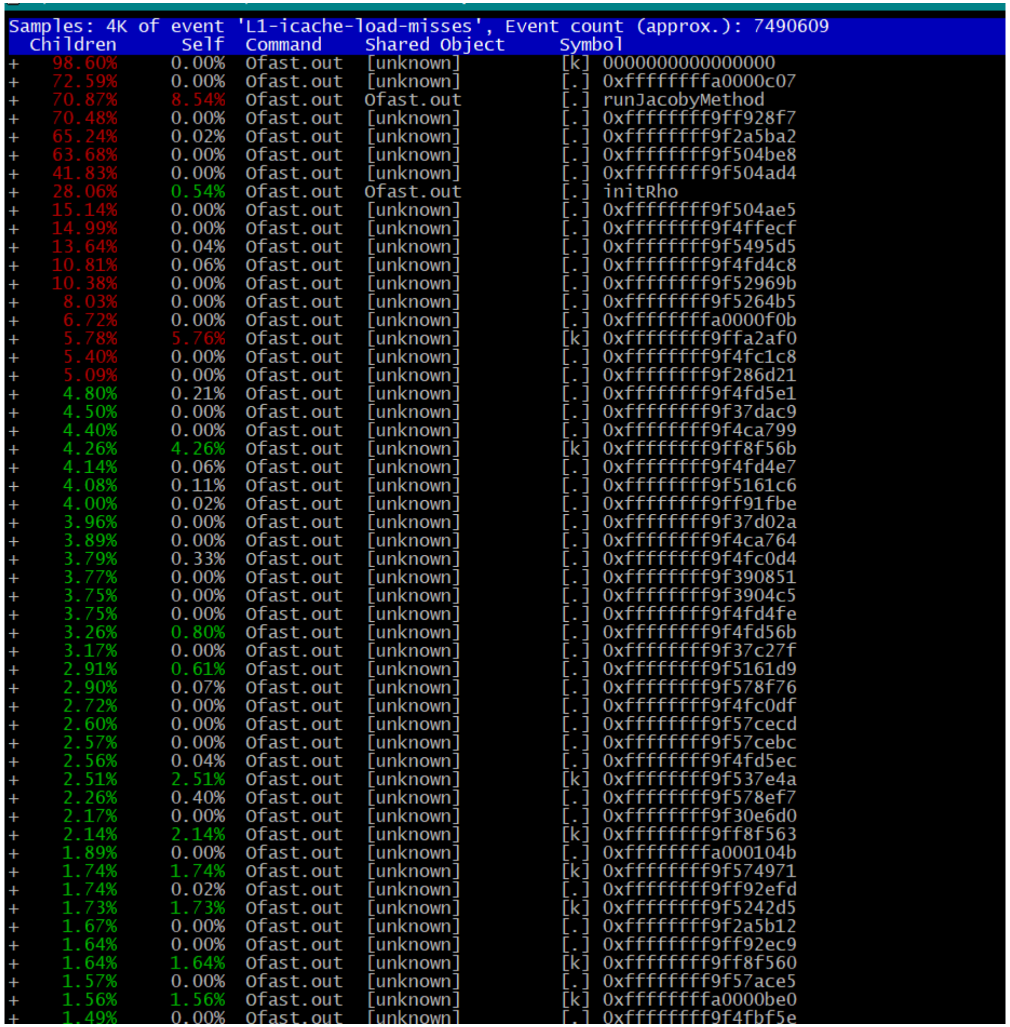
epsmim@comrade: ~/Desktop/Mandarkhanov/Lab_1/newTry

```
Samples: 5K of event 'L1-dcache-load-misses', Event count (approx.): 8404548
Overhead  Command   Shared Object      Symbol
  41.26%  00-g.out  [unknown]          [k] 0xffffffff9f8a4d97
  25.56%  00-g.out  00-g.out           [.] runJacobyMethod
   8.13%  00-g.out  [unknown]          [k] 0xffffffff9f579d07
   1.93%  00-g.out  [unknown]          [k] 0xffffffffa0001599
   1.02%  00-g.out  00-g.out           [.] initRho
   0.82%  00-g.out  [unknown]          [k] 0xffffffffa0000be0
   0.70%  00-g.out  [unknown]          [k] 0xffffffff9f579d00
   0.63%  00-g.out  [unknown]          [k] 0xffffffff9f579cf0
   0.61%  00-g.out  [unknown]          [k] 0xffffffffa0001209
   0.46%  00-g.out  [unknown]          [k] 0xffffffff9f4dfc13
   0.39%  00-g.out  [unknown]          [k] 0xffffffff9f4dfc1d
   0.35%  00-g.out  [unknown]          [k] 0xffffffff9ff928e4
   0.34%  00-g.out  [unknown]          [k] 0xffffffff9f5242e1
   0.33%  00-g.out  [unknown]          [k] 0xffffffff9f5228a6
   0.30%  00-g.out  [unknown]          [k] 0xffffffff9f4fbc3f
   0.29%  00-g.out  [unknown]          [k] 0xffffffff9ff928b0
   0.27%  00-g.out  [unknown]          [k] 0xffffffff9ffa2af0
   0.26%  00-g.out  [unknown]          [k] 0xffffffff9f5296ea
   0.26%  00-g.out  [unknown]          [k] 0xffffffff9f5242d9
   0.25%  00-g.out  [unknown]          [k] 0xffffffff9ff928c5
   0.25%  00-g.out  [unknown]          [k] 0xffffffff9f5228f7
   0.24%  00-g.out  [unknown]          [k] 0xffffffff9f5721f2
   0.24%  00-g.out  [unknown]          [k] 0xffffffff9f3b3f92
```

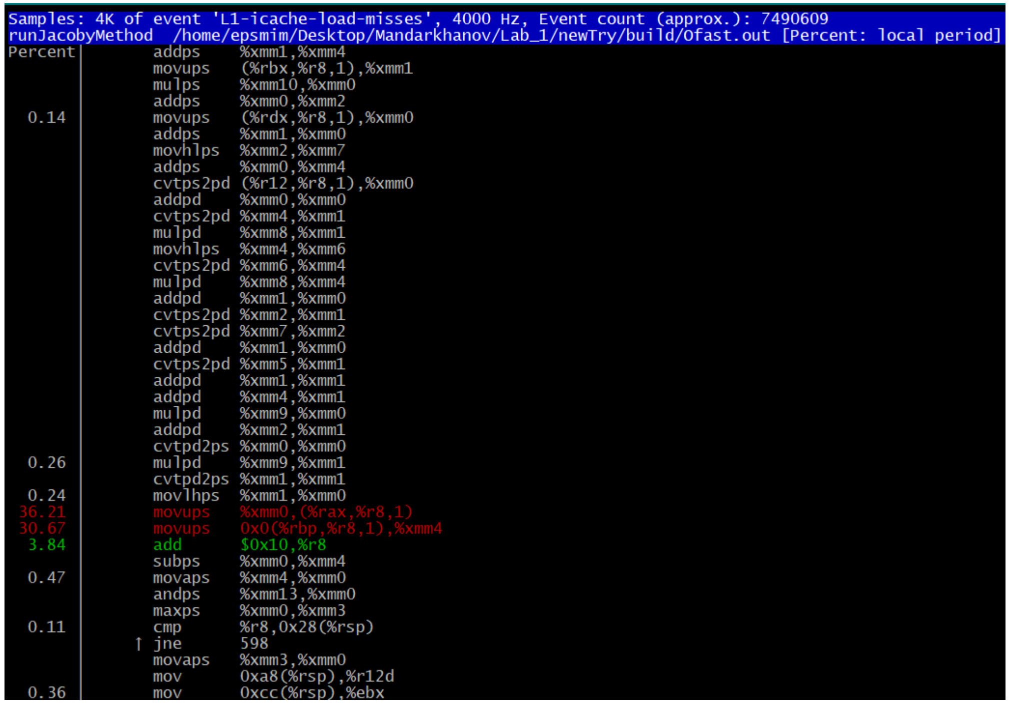


```
epsmim@comrade: ~/Desktop/Mandarkhanov/Lab_1/newTry
Samples: 5K of event 'L1-dcache-load-misses', 4000 Hz, Event count (approx.): 8404548
runJacobyMethod  /home/epsmim/Desktop/Mandarkhanov/Lab_1/newTry/O0-g.out [Percent: local period]
Percent│        add      $0x1,%rax
       │        lea      0x0(,%rax,4),%rdx
  0.12 │        mov      -0x88(%rbp),%rax
       │        add      %rdx,%rax
       │        movss    (%rax),%xmm0
       │        addss    %xmm3,%xmm0
       │        cvtss2sd %xmm0,%xmm3
       │        movsd    _IO_stdin_used+0xa0,%xmm0
       │        mulsd    %xmm3,%xmm0
       │      2.0 * rho[index] +
       │        addsd    %xmm2,%xmm0
       │      phi_new[index] = mainKoef * (firstKoef * (phi[index - 1] + phi[index + 1]) +
  0.12 │        mulsd    %xmm1,%xmm0
  0.60 │        mov      -0x40(%rbp),%rax
       │        mov      -0x74(%rbp),%edx
       │        movslq   %edx,%rdx
       │        shl      $0x2,%rdx
  0.60 │        add      %rdx,%rax
       │        cvtsd2ss %xmm0,%xmm0
 11.10 │        movss    %xmm0,(%rax)
       │
       │      d = fabs(phi[index] - phi_new[index]);
  5.54 │        mov      -0x90(%rbp),%rax
       │        mov      -0x74(%rbp),%edx
       │        movslq   %edx,%rdx
       │        shl      $0x2,%rdx
 11.04 │        add      %rdx,%rax
       │        movss    (%rax),%xmm0
 43.76 │        mov      -0x40(%rbp),%rax
       │        mov      -0x74(%rbp),%edx
       │        movslq   %edx,%rdx
       │        shl      $0x2,%rdx
 16.38 │        add      %rdx,%rax
       │        movss    (%rax),%xmm1
       │        subss    %xmm1,%xmm0
  4.81 │        movss    _IO_stdin_used+0xb0,%xmm1
       │        andps    %xmm1,%xmm0
  0.97 │        movss    %xmm0,-0x44(%rbp)
       │      if (d > stepDelta) stepDelta = d;
  0.12 │        movss    -0x44(%rbp),%xmm0
  0.84 │        comiss   -0x70(%rbp),%xmm0
  0.24 │      ↓ jbe      4bf
       │        movss    -0x44(%rbp),%xmm0
       │        movss    %xmm0,-0x70(%rbp)
       │      for (int j = 1; j < N_x - 1; j++) {
  0.36 │4bf:    addl     $0x1,-0x60(%rbp)
       │4c3:    mov      N_x,%eax
       │        sub      $0x1,%eax
       │        cmp      %eax,-0x60(%rbp)
       │      ↑ jl       23a
       │      for (int i = 1; i < N_y - 1; i++) {
       │        addl     $0x1,-0x64(%rbp)
       │4d9:    mov      N_y,%eax
       │        sub      $0x1,%eax
       │        cmp      %eax,-0x64(%rbp)
```

L1-icache-load-misses

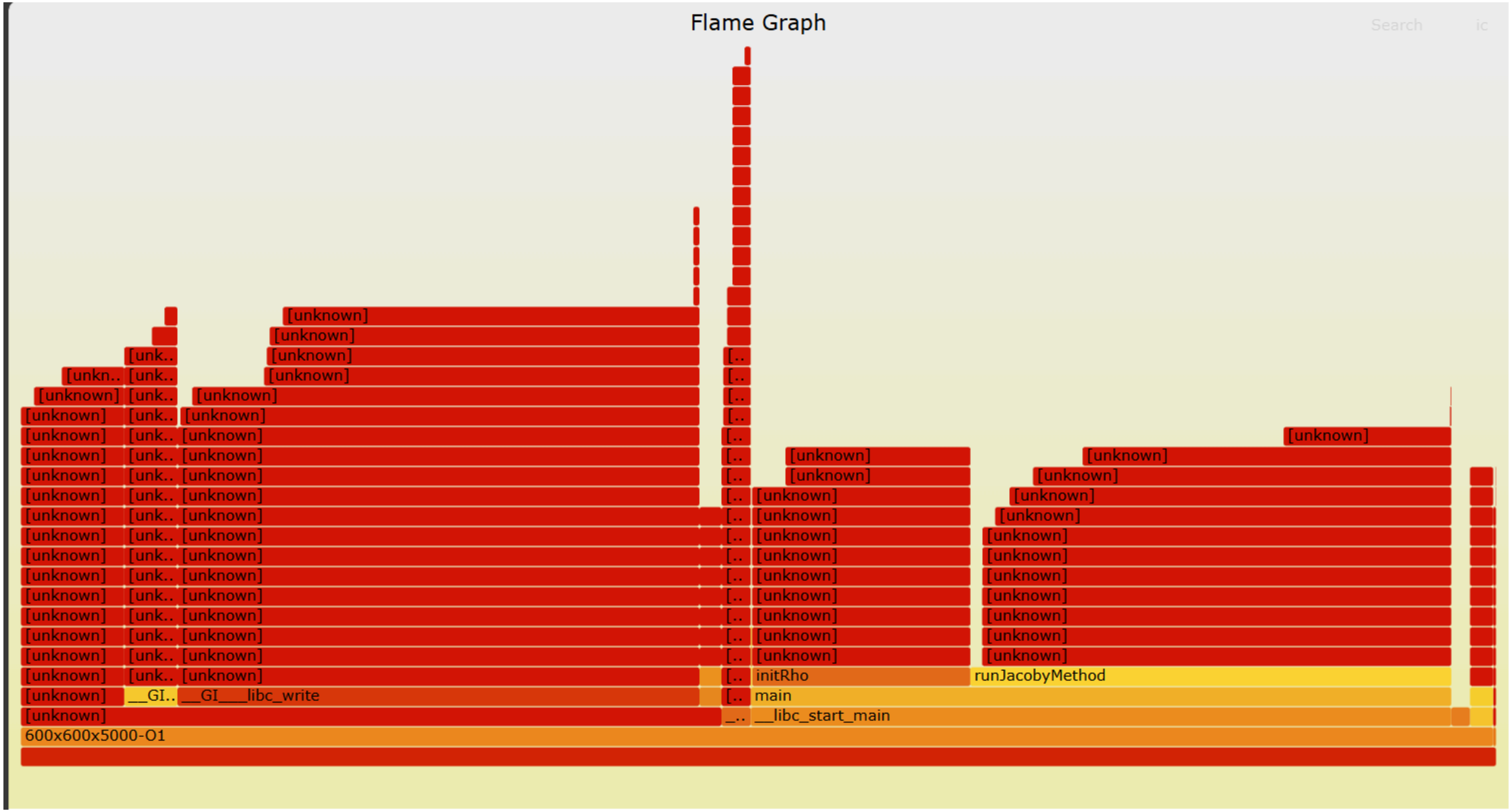


```
Samples: 4K of event 'L1-icache-load-misses', Event count (approx.): 7490609
  Children      Self  Command     Shared Object        Symbol
+   98.60%     0.00%  Ofast.out   [unknown]            [k] 0000000000000000
+   72.59%     0.00%  Ofast.out   [unknown]            [.] 0xffffffffa0000c07
+   70.87%     8.54%  Ofast.out   Ofast.out            [.] runJacobyMethod
+   70.48%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9ff928f7
+   65.24%     0.02%  Ofast.out   [unknown]            [.] 0xffffffff9f2a5ba2
+   63.68%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f504be8
+   41.83%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f504ad4
+   28.06%     0.54%  Ofast.out   Ofast.out            [.] initRho
+   15.14%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f504ae5
+   14.99%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f4ffecf
+   13.64%     0.04%  Ofast.out   [unknown]            [.] 0xffffffff9f5495d5
+   10.81%     0.06%  Ofast.out   [unknown]            [.] 0xffffffff9f4fd4c8
+   10.38%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f52969b
+    8.03%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f5264b5
+    6.72%     0.00%  Ofast.out   [unknown]            [.] 0xffffffffa0000f0b
+    5.78%     5.76%  Ofast.out   [unknown]            [k] 0xffffffff9ffa2af0
+    5.40%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f4fc1c8
+    5.09%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f286d21
+    4.80%     0.21%  Ofast.out   [unknown]            [.] 0xffffffff9f4fd5e1
+    4.50%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f37dac9
+    4.40%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f4ca799
+    4.26%     4.26%  Ofast.out   [unknown]            [k] 0xffffffff9ff8f56b
+    4.14%     0.06%  Ofast.out   [unknown]            [.] 0xffffffff9f4fd4e7
+    4.08%     0.11%  Ofast.out   [unknown]            [.] 0xffffffff9f5161c6
+    4.00%     0.02%  Ofast.out   [unknown]            [.] 0xffffffff9ff91fbe
+    3.96%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f37d02a
+    3.89%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f4ca764
+    3.79%     0.33%  Ofast.out   [unknown]            [.] 0xffffffff9f4fc0d4
+    3.77%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f390851
+    3.75%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f3904c5
+    3.75%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f4fd4fe
+    3.26%     0.80%  Ofast.out   [unknown]            [.] 0xffffffff9f4fd56b
+    3.17%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f37c27f
+    2.91%     0.61%  Ofast.out   [unknown]            [.] 0xffffffff9f5161d9
+    2.90%     0.07%  Ofast.out   [unknown]            [.] 0xffffffff9f578f76
+    2.72%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f4fc0df
+    2.60%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f57cecd
+    2.57%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f57cebc
+    2.56%     0.04%  Ofast.out   [unknown]            [.] 0xffffffff9f4fd5ec
+    2.51%     2.51%  Ofast.out   [unknown]            [k] 0xffffffff9f537e4a
+    2.26%     0.40%  Ofast.out   [unknown]            [.] 0xffffffff9f578ef7
+    2.17%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f30e6d0
+    2.14%     2.14%  Ofast.out   [unknown]            [k] 0xffffffff9ff8f563
+    1.89%     0.00%  Ofast.out   [unknown]            [.] 0xffffffffa000104b
+    1.74%     1.74%  Ofast.out   [unknown]            [k] 0xffffffff9f574971
+    1.74%     0.02%  Ofast.out   [unknown]            [.] 0xffffffff9ff92efd
+    1.73%     1.73%  Ofast.out   [unknown]            [k] 0xffffffff9f5242d5
+    1.67%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f2a5b12
+    1.64%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9ff92ec9
+    1.64%     1.64%  Ofast.out   [unknown]            [k] 0xffffffff9ff8f560
+    1.57%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f57ace5
+    1.56%     1.56%  Ofast.out   [unknown]            [k] 0xffffffffa0000be0
+    1.49%     0.00%  Ofast.out   [unknown]            [.] 0xffffffff9f4fbf5e
```

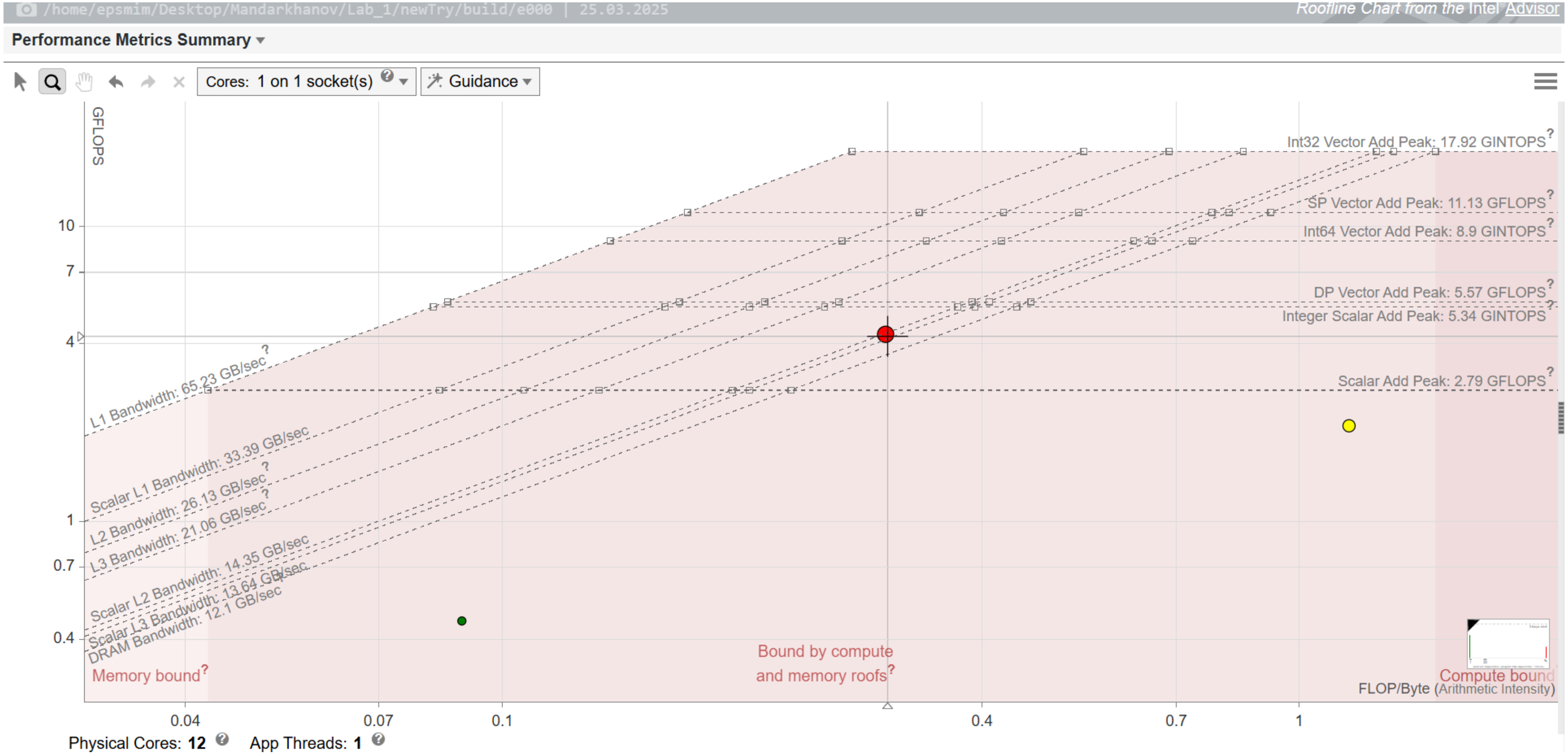


```
Samples: 4K of event 'L1-icache-load-misses', 4000 Hz, Event count (approx.): 7490609
runJacobyMethod  /home/epsmim/Desktop/Mandarkhanov/Lab_1/newTry/build/Ofast.out [Percent: local period]
Percent│        addps    %xmm1,%xmm4
       │        movups   (%rbx,%r8,1),%xmm1
       │        mulps    %xmm10,%xmm0
       │        addps    %xmm0,%xmm2
  0.14 │        movups   (%rdx,%r8,1),%xmm0
       │        addps    %xmm1,%xmm0
       │        movhlps  %xmm2,%xmm7
       │        addps    %xmm0,%xmm4
       │        cvtps2pd (%r12,%r8,1),%xmm0
       │        addpd    %xmm0,%xmm0
       │        cvtps2pd %xmm4,%xmm1
       │        mulpd    %xmm8,%xmm1
       │        movhlps  %xmm4,%xmm6
       │        cvtps2pd %xmm6,%xmm4
       │        mulpd    %xmm8,%xmm4
       │        addpd    %xmm1,%xmm0
       │        cvtps2pd %xmm2,%xmm1
       │        cvtps2pd %xmm7,%xmm2
       │        addpd    %xmm1,%xmm0
       │        cvtps2pd %xmm5,%xmm1
       │        addpd    %xmm1,%xmm1
       │        addpd    %xmm4,%xmm1
       │        mulpd    %xmm9,%xmm0
       │        addpd    %xmm2,%xmm1
       │        cvtpd2ps %xmm0,%xmm0
  0.26 │        mulpd    %xmm9,%xmm1
       │        cvtpd2ps %xmm1,%xmm1
  0.24 │        movlhps  %xmm1,%xmm0
 36.21 │        movups   %xmm0,(%rax,%r8,1)
 30.67 │        movups   0x0(%rbp,%r8,1),%xmm4
  3.84 │        add      $0x10,%r8
       │        subps    %xmm0,%xmm4
  0.47 │        movaps   %xmm4,%xmm0
       │        andps    %xmm13,%xmm0
       │        maxps    %xmm0,%xmm3
  0.11 │        cmp      %r8,0x28(%rsp)
       │      ↑ jne      598
       │        movaps   %xmm3,%xmm0
       │        mov      0xa8(%rsp),%r12d
  0.36 │        mov      0xcc(%rsp),%ebx
```

FlameGraph



Roofline модель



/home/epsmim/Desktop/Mandarkhanov/Lab_1/newTry/build/e000 | 25.03.2025

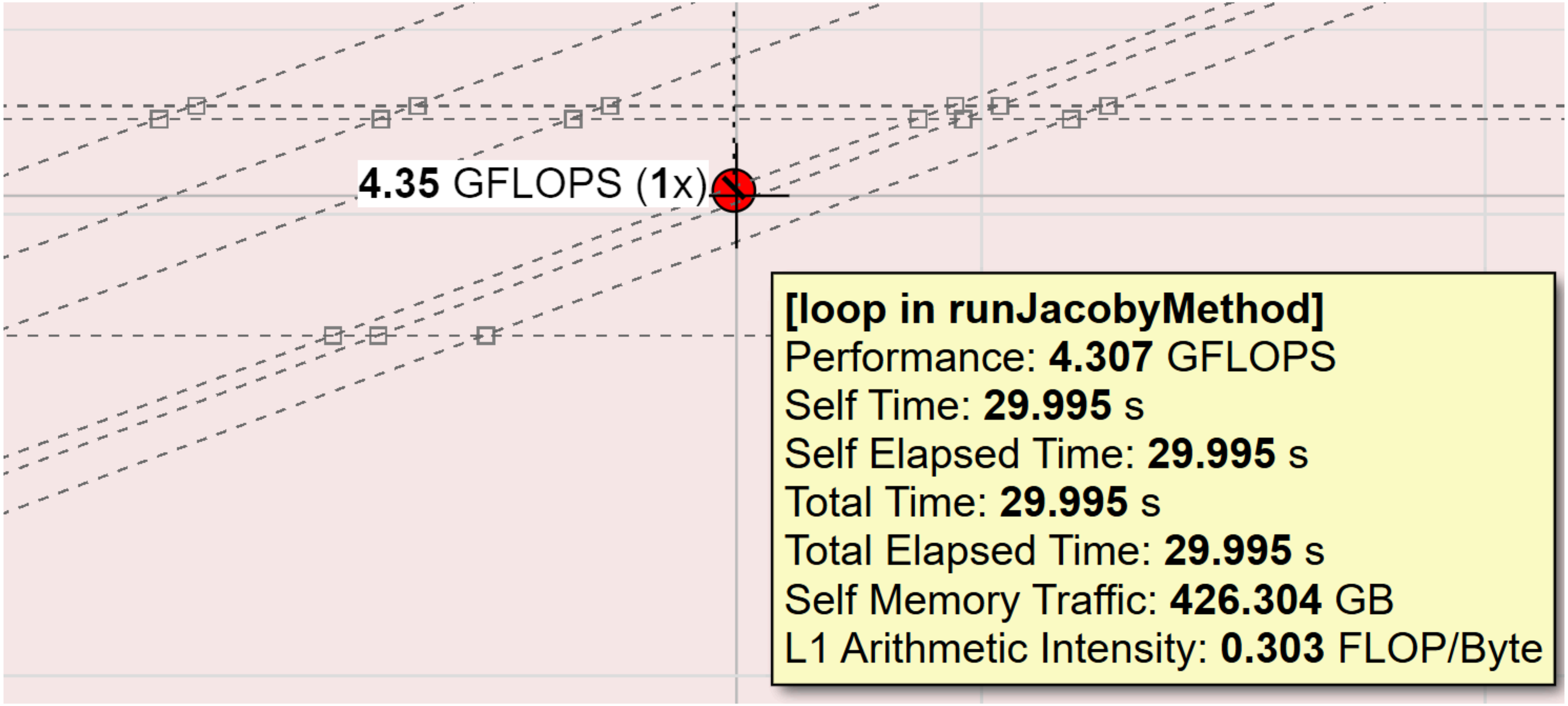
Roofline Chart from the Intel Advisor

Performance Metrics Summary

Cores: 1 on 1 socket(s)

Physical Cores: **12** App Threads: **1**

Guidance
GFLOPS
Int32 Vector Add Peak: 17.92 GINTOPS
SP Vector Add Peak: 11.13 GFLOPS
Int64 Vector Add Peak: 8.9 GINTOPS
DP Vector Add Peak: 5.57 GFLOPS
Integer Scalar Add Peak: 5.34 GINTOPS
Scalar Add Peak: 2.79 GFLOPS
L1 Bandwidth: 65.23 GB/sec
Scalar L1 Bandwidth: 33.39 GB/sec
L2 Bandwidth: 26.13 GB/sec
L3 Bandwidth: 21.06 GB/sec
Scalar L2 Bandwidth: 14.35 GB/sec
Scalar L3 Bandwidth: 13.64 GB/sec
DRAM Bandwidth: 12.1 GB/sec
10
7
4
1
0.7
0.4
Memory bound
Bound by compute and memory roofs
Compute bound
FLOP/Byte (Arithmetic Intensity)
0.04
0.07
0.1
0.4
0.7
1



[loop in runJacobyMethod]
Performance: **4.307** GFLOPS
Self Time: **29.995** s
Self Elapsed Time: **29.995** s
Total Time: **29.995** s
Total Elapsed Time: **29.995** s
Self Memory Traffic: **426.304** GB
L1 Arithmetic Intensity: **0.303** FLOP/Byte

Приложение 1 - Листинг программы

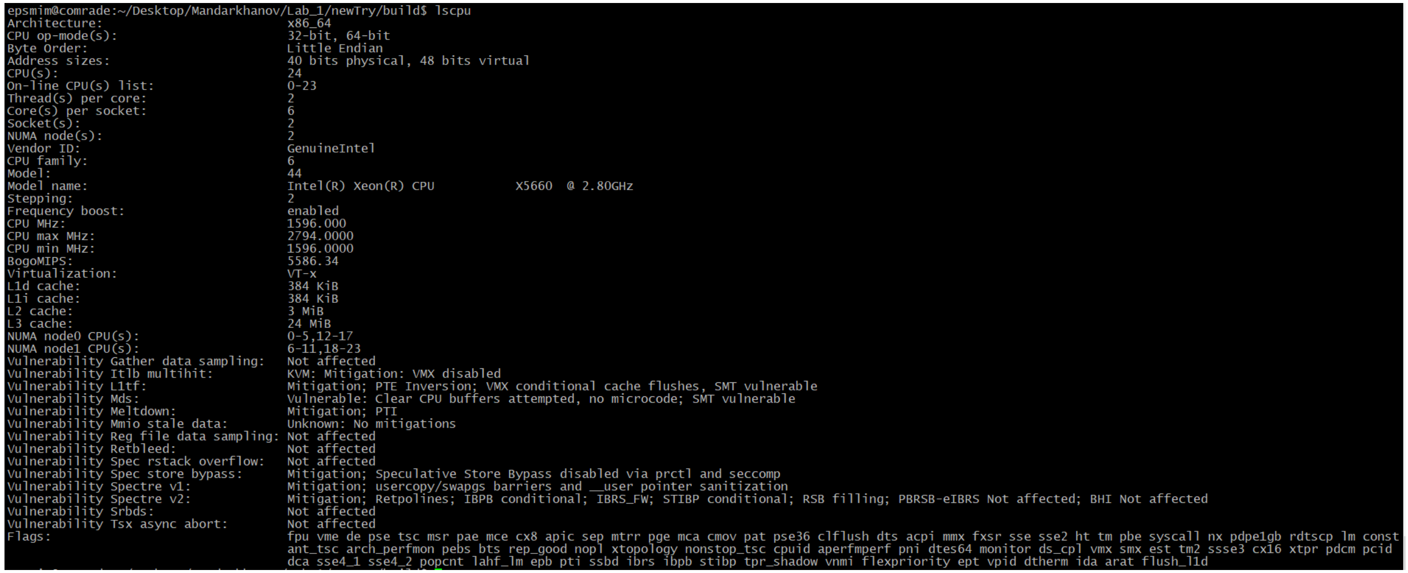
```
#include<stdio.h>
#include<math.h>
#include<malloc.h>
#include<stdbool.h>
#include <time.h>


float X_a = 0.0;
```

```c
float X_b = 4.0;
float Y_a = 0.0;
float Y_b = 4.0;

int N_x = 8000;
int N_y = 8000;
int N_t = 100;

void swapFloatPointers(float** a, float** b) {
    float* tmp = *a;
    *a = *b;
    *b = tmp;
}

void fillFile(float* matrix, char* filename) {
    FILE *fp;
    fp = fopen(filename, "wb");

    if (fp == NULL) {
        printf("Error opening file!\n");
        return;
    }

    int index;
    for (int i = 0; i < N_y; i++) {
        index = i * N_x;
        for (int j = 0; j < N_x; j++) {
            fprintf(fp, "%f ", matrix[index]);
            index++;
        }
        fprintf(fp, "\n");
    }
    fclose(fp);
}
```

```

void initRho(float* rho) {
    float h_x = (X_b - X_a) / (N_x - 1);
    float h_y = (Y_b - Y_a) / (N_y - 1);

    float X_s1 = X_a + (X_b - X_a) / 3.0;
    float Y_s1 = Y_a + (Y_b - Y_a) * 2.0 / 3.0;
    float X_s2 = X_a + (X_b - X_a) * 2.0 / 3.0;
    float Y_s2 = Y_a + (Y_b - Y_a) / 3.0;

    float R = 0.1 * fmin(X_b - X_a, Y_b - Y_a);

    int index;

    for (float i = 0; i < N_y; i++) {
        index = i * N_x;
        for (float j = 0; j < N_x; j++) {

            if ((X_a + j * h_x - X_s1) * (X_a + j * h_x - X_s1) + (Y_a + i * h_y - Y_s1) * (Y_a + i * h_y - Y_s1) < R * R) {
                rho[index] = 1.0;
            }
            else if ((X_a + j * h_x - X_s2) * (X_a + j * h_x - X_s2)  + (Y_a + i * h_y - Y_s2) * (Y_a + i * h_y - Y_s2) < R * R) {
                rho[index] = -1.0;
            }
            else {
                rho[index] = 0.0;
            }

            index ++;
        }
    }
}
```

```c

void runJacobyMethod (float *rho, float* phi) {
    float *phi_new = (float*)calloc(N_x * N_y, sizeof(float));

    float h_x = (X_b - X_a) / (N_x - 1);
    float h_y = (Y_b - Y_a) / (N_y - 1);

    int index;
    float d, stepDelta;
    float globalDelta = 1.0;

    float mainKoef = 0.2 / (1.0 / (h_x * h_x) + 1.0 / (h_y * h_y));
    float firstKoef = 2.5 / (h_x * h_x) - 0.5 / (h_y * h_y);
    float secondKoef = 2.5 / (h_y * h_y) - 0.5 / (h_x * h_x);
    float thirdKoef = 0.25 / (h_x * h_x) + 0.25 / (h_y * h_y);

    int iterNumber = 0;
    bool isSuccess = true;

    long long t1, t2;
    double tDiff;
    struct timespec curTime;
    clock_gettime(CLOCK_BOOTTIME, &curTime);
    t1 = curTime.tv_sec * 1000000000 + curTime.tv_nsec;

    while (iterNumber <= N_t) {

        stepDelta = -1.0;
        for (int i = 1; i < N_y - 1; i++) {
            index = i * N_x;
            for (int j = 1; j < N_x - 1; j++) {
                index++;

                phi_new[index] = mainKoef * (firstKoef * (phi[index - 1] + phi[index + 1]) +
                                            secondKoef * (phi[index - N_x] + phi[index + N_x]) +
                                            thirdKoef * (phi[index - N_x - 1] + phi[index - N_x + 1] + phi[index + N_x - 1] + phi[index + N_x + 1]) +
                                            2.0 * rho[index] +
                                            (rho[index - N_x] + rho[index + N_x] + rho[index - 1] + rho[index + 1]) * 0.25);

                d = fabs(phi[index] - phi_new[index]);
                if (d > stepDelta) stepDelta = d;
            }
        }

        if ((stepDelta - globalDelta) < 0.0000001) {
            globalDelta = stepDelta;
            // printf("[%d]globalDelta = %f\n", iterNumber, globalDelta);
            swapFloatPointers(&phi, &phi_new);
            iterNumber++;
        }
        else {
            isSuccess = false;
            break;
        }
    }

    clock_gettime(CLOCK_BOOTTIME, &curTime);
    t2 = curTime.tv_sec * 1000000000 + curTime.tv_nsec;
    tDiff = (double) (t2 - t1) / 1000000000.0;=
    printf("Time = %g s\n", tDiff);

    if (isSuccess) printf("Jacoby method finished\n");
    else printf("Jacoby method failed\n");

    if ((iterNumber % 2) != 0) {
        swapFloatPointers(&phi, &phi_new);
    }
    free(phi_new);
    return;
}

int main() {
    float *rho = (float*)malloc(N_x * N_y * sizeof(float));
    initRho(rho);
    fillFile(rho, "rho.dat");

    float *phi = (float*)malloc(N_x * N_y * sizeof(float));
    runJacobyMethod(rho, phi);
    fillFile(phi, "phi.dat");

    free(phi);
    free(rho);
    return 0;
}
}
```

Приложение 2 lscpu



```
epsmim@comrade:~/Desktop/Mandarkhanov/Lab_1/newTry/build$ lscpu
Architecture:                         x86_64
CPU op-mode(s):                       32-bit, 64-bit
Byte Order:                           Little Endian
Address sizes:                        40 bits physical, 48 bits virtual
CPU(s):                               24
On-line CPU(s) list:                  0-23
Thread(s) per core:                   2
Core(s) per socket:                   6
Socket(s):                            2
NUMA node(s):                         2
Vendor ID:                            GenuineIntel
CPU family:                           6
Model:                                44
Model name:                           Intel(R) Xeon(R) CPU           X5660  @ 2.80GHz
Stepping:                             2
Frequency boost:                      enabled
CPU MHz:                              1596.000
CPU max MHz:                          2794.0000
CPU min MHz:                          1596.0000
BogoMIPS:                             5586.34
Virtualization:                       VT-x
L1d cache:                            384 KiB
L1i cache:                            384 KiB
L2 cache:                             3 MiB
L3 cache:                             24 MiB
NUMA node0 CPU(s):                    0-5,12-17
NUMA node1 CPU(s):                    6-11,18-23
Vulnerability Gather data sampling:   Not affected
Vulnerability Itlb multihit:          KVM: Mitigation: VMX disabled
Vulnerability L1tf:                   Mitigation; PTE Inversion; VMX conditional cache flushes, SMT vulnerable
Vulnerability Mds:                    Vulnerable: Clear CPU buffers attempted, no microcode; SMT vulnerable
Vulnerability Meltdown:               Mitigation; PTI
Vulnerability Mmio stale data:        Unknown: No mitigations
Vulnerability Reg file data sampling: Not affected
Vulnerability Retbleed:               Not affected
Vulnerability Spec rstack overflow:   Not affected
Vulnerability Spec store bypass:      Mitigation; Speculative Store Bypass disabled via prctl and seccomp
Vulnerability Spectre v1:             Mitigation; usercopy/swapgs barriers and __user pointer sanitization
Vulnerability Spectre v2:             Mitigation; Retpolines; IBPB conditional; IBRS_FW; STIBP conditional; RSB filling; PBRSB-eIBRS Not affected; BHI Not affected
Vulnerability Srbds:                  Not affected
Vulnerability Tsx async abort:        Not affected
Flags:                                fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ht tm pbe syscall nx pdpe1gb rdtscp lm const
                                      ant_tsc arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc cpuid aperfmperf pni dtes64 monitor ds_cpl vmx smx est tm2 ssse3 cx16 xtpr pdcm pcid
                                      dca sse4_1 sse4_2 popcnt lahf_lm epb pti ssbd ibrs ibpb stibp tpr_shadow vnmi flexpriority ept vpid dtherm ida arat flush_l1d
```

Вывод

Исходя из результатов roofline модели и результатов профилирования видно, что проблема программа упирается в scalar L2 bandwidth
Это также подтверждается большим количеством кэш-промахов

Скорее всего это связано с тем, что у нас топология вычисления - "крест", что может плохо влиять на работу кэша, ведь верхний и нижний сосед клетки не находятся рядом, а смещены на строку массива