ASSIGNMENT-0 REPORT

BY

KURISETI RAVI SRI TEJA
ENTRY NUMBER-2019CS10369

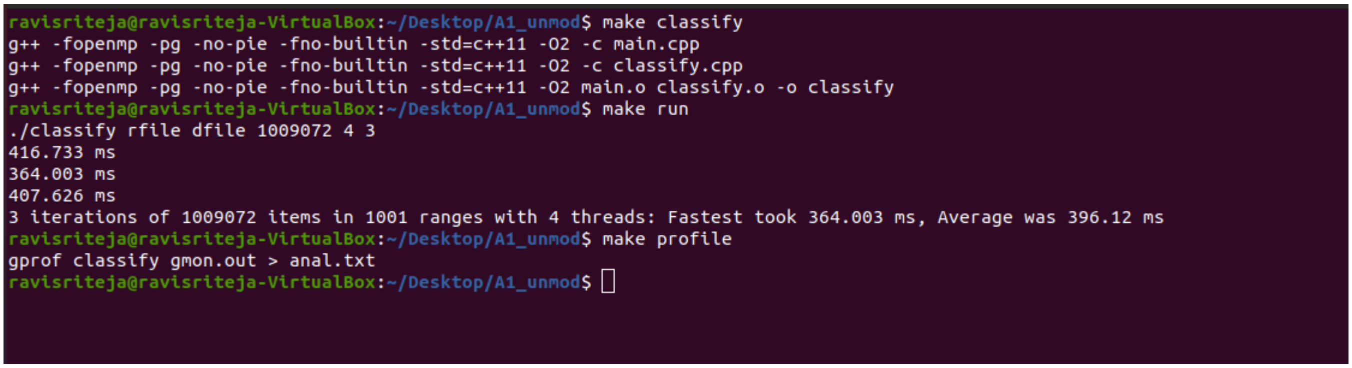
1 Tools used for Analysis

All the analysis done was for the span of **3 iterations** using **4 threads**.

- To compute the time taken by the code,I used the system provided clock in main.cpp
- For the analysis of cache misses,I used the tool **valgrind**
- For analyzing,the time taken by various functions I used the tool **gprof**

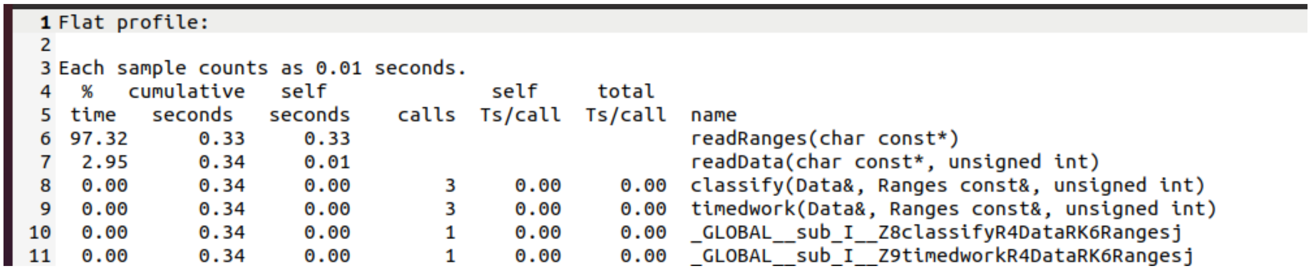
2 Analysis Of Original Code

In the original code,we can see that the D1 miss rate is close to 6% and the LL(L3 cache) miss rate was close to 1%. This can be attributed to the fact that the shared threads in the classify.cpp do not access the contiguous memory blocks but access every r^{th} element in the array(where r is the number of spawned threads). Also since most of the time consumed is at the loops,we can also optimize the **for** loops to make the code faster.

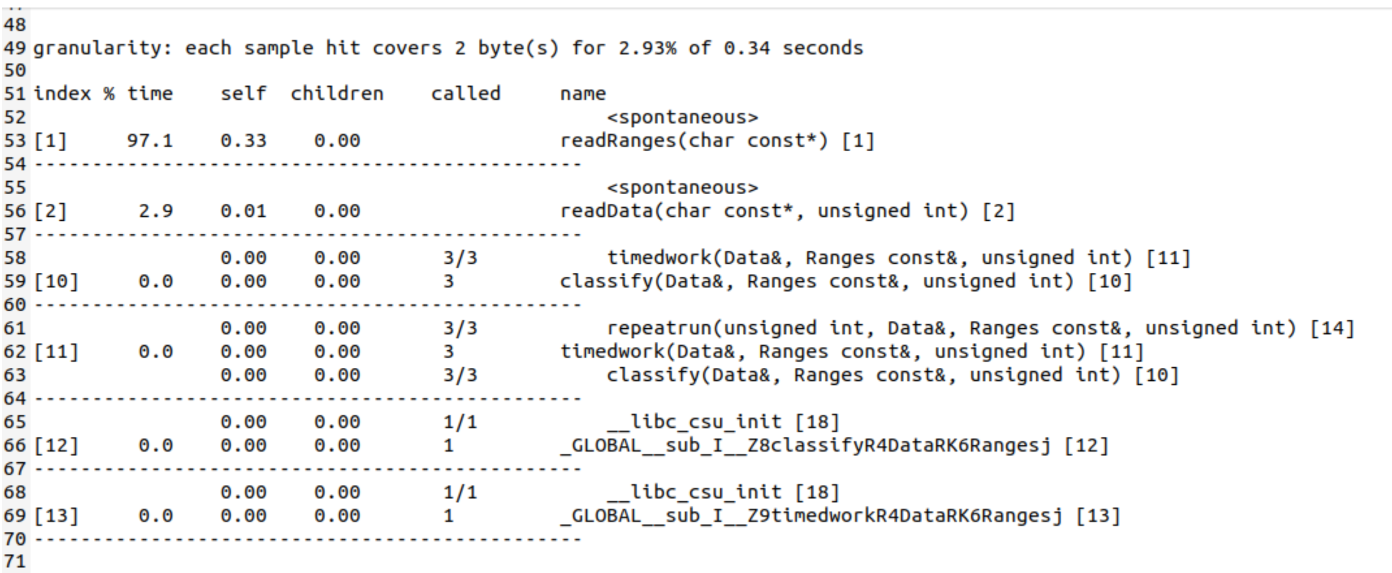


```
ravisriteja@ravisriteja-VirtualBox:~/Desktop/A1_unmod$ make classify
g++ -fopenmp -pg -no-pie -fno-builtin -std=c++11 -O2 -c main.cpp
g++ -fopenmp -pg -no-pie -fno-builtin -std=c++11 -O2 -c classify.cpp
g++ -fopenmp -pg -no-pie -fno-builtin -std=c++11 -O2 main.o classify.o -o classify
ravisriteja@ravisriteja-VirtualBox:~/Desktop/A1_unmod$ make run
./classify rfile dfile 1009072 4 3
416.733 ms
364.003 ms
407.626 ms
3 iterations of 1009072 items in 1001 ranges with 4 threads: Fastest took 364.003 ms, Average was 396.12 ms
ravisriteja@ravisriteja-VirtualBox:~/Desktop/A1_unmod$ make profile
gprof classify gmon.out > anal.txt
ravisriteja@ravisriteja-VirtualBox:~/Desktop/A1_unmod$ 
```

2.1 Gprof Analysis

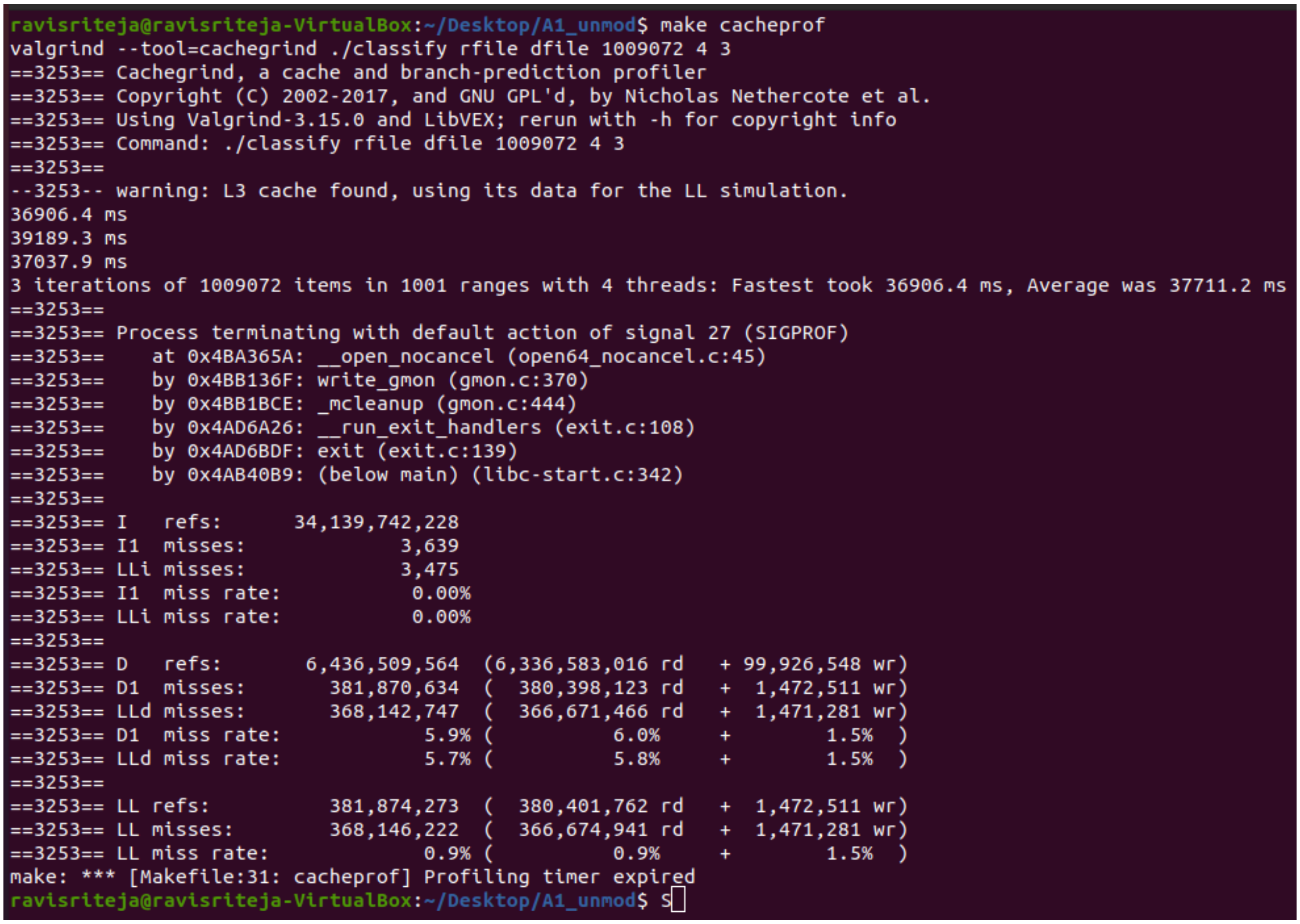


```
Flat profile:

Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  Ts/call  Ts/call  name
 97.32      0.33     0.33                             readRanges(char const*)
  2.95      0.34     0.01                             readData(char const*, unsigned int)
  0.00      0.34     0.00        3     0.00     0.00  classify(Data&, Ranges const&, unsigned int)
  0.00      0.34     0.00        3     0.00     0.00  timedwork(Data&, Ranges const&, unsigned int)
  0.00      0.34     0.00        1     0.00     0.00  _GLOBAL__sub_I__Z8classifyR4DataRK6Rangesj
  0.00      0.34     0.00        1     0.00     0.00  _GLOBAL__sub_I__Z9timedworkR4DataRK6Rangesj
```



```

granularity: each sample hit covers 2 byte(s) for 2.93% of 0.34 seconds

index % time    self  children    called     name
                                                 <spontaneous>
[1]     97.1    0.33    0.00                 readRanges(char const*) [1]
-----------------------------------------------
                                                 <spontaneous>
[2]      2.9    0.01    0.00                 readData(char const*, unsigned int) [2]
-----------------------------------------------
                0.00    0.00       3/3           timedwork(Data&, Ranges const&, unsigned int) [11]
[10]     0.0    0.00    0.00       3         classify(Data&, Ranges const&, unsigned int) [10]
-----------------------------------------------
                0.00    0.00       3/3           repeatrun(unsigned int, Data&, Ranges const&, unsigned int) [14]
[11]     0.0    0.00    0.00       3         timedwork(Data&, Ranges const&, unsigned int) [11]
                0.00    0.00       3/3           classify(Data&, Ranges const&, unsigned int) [10]
-----------------------------------------------
                0.00    0.00       1/1           __libc_csu_init [18]
[12]     0.0    0.00    0.00       1         _GLOBAL__sub_I__Z8classifyR4DataRK6Rangesj [12]
-----------------------------------------------
                0.00    0.00       1/1           __libc_csu_init [18]
[13]     0.0    0.00    0.00       1         _GLOBAL__sub_I__Z9timedworkR4DataRK6Rangesj [13]
-----------------------------------------------

```

2.2 Valgrind Analysis



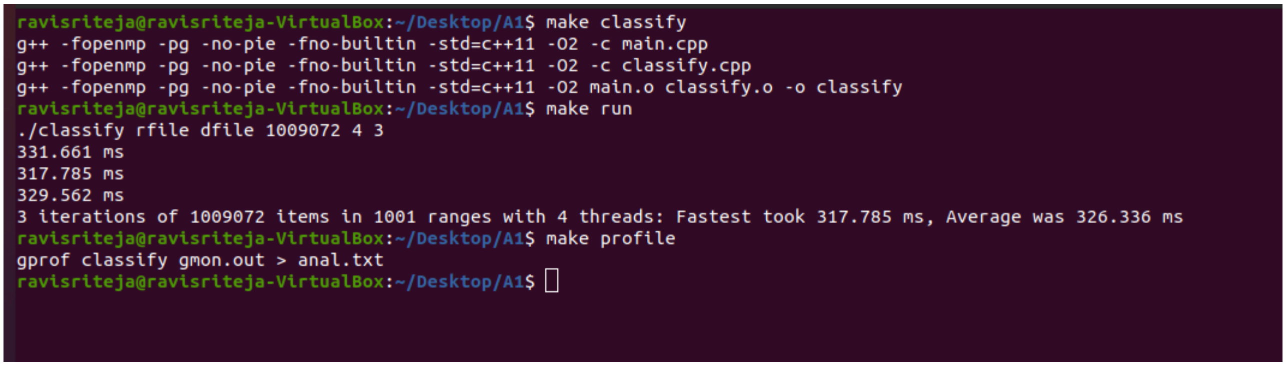
```
ravisriteja@ravisriteja-VirtualBox:~/Desktop/A1_unmod$ make cacheprof
valgrind --tool=cachegrind ./classify rfile dfile 1009072 4 3
==3253== Cachegrind, a cache and branch-prediction profiler
==3253== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas Nethercote et al.
==3253== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==3253== Command: ./classify rfile dfile 1009072 4 3
==3253==
--3253-- warning: L3 cache found, using its data for the LL simulation.
36906.4 ms
39189.3 ms
37037.9 ms
3 iterations of 1009072 items in 1001 ranges with 4 threads: Fastest took 36906.4 ms, Average was 37711.2 ms
==3253==
==3253== Process terminating with default action of signal 27 (SIGPROF)
==3253==    at 0x4BA365A: __open_nocancel (open64_nocancel.c:45)
==3253==    by 0x4BB136F: write_gmon (gmon.c:370)
==3253==    by 0x4BB1BCE: _mcleanup (gmon.c:444)
==3253==    by 0x4AD6A26: __run_exit_handlers (exit.c:108)
==3253==    by 0x4AD6BDF: exit (exit.c:139)
==3253==    by 0x4AB40B9: (below main) (libc-start.c:342)
==3253==
==3253== I   refs:      34,139,742,228
==3253== I1  misses:             3,639
==3253== LLi misses:             3,475
==3253== I1  miss rate:           0.00%
==3253== LLi miss rate:           0.00%
==3253==
==3253== D   refs:       6,436,509,564  (6,336,583,016 rd   + 99,926,548 wr)
==3253== D1  misses:       381,870,634  (  380,398,123 rd   +  1,472,511 wr)
==3253== LLd misses:       368,142,747  (  366,671,466 rd   +  1,471,281 wr)
==3253== D1  miss rate:            5.9% (          6.0%     +        1.5%  )
==3253== LLd miss rate:            5.7% (          5.8%     +        1.5%  )
==3253==
==3253== LL refs:          381,874,273  (  380,401,762 rd   +  1,472,511 wr)
==3253== LL misses:        368,146,222  (  366,674,941 rd   +  1,471,281 wr)
==3253== LL miss rate:             0.9% (          0.9%     +        1.5%  )
make: *** [Makefile:31: cacheprof] Profiling timer expired
ravisriteja@ravisriteja-VirtualBox:~/Desktop/A1_unmod$ $
```

3 Improvements Done

To reduce the cache misses,I modified the code toallocate the contiguous blocks of array for every thread(For example if there are N elements in array and q threads,then the first r threads will access (N/q)+1 elements and the remaining q-r threads will access (N/q) elements(where a/b is the integer division and N=q*(N/q)+r $0 \leq r \leq q-1$).For decreasing the time taken by the loops,I used the OpenMP built-in optimizer # **pragma omp for** that decreases the time taken by the **for** loops using optimizations of GCC.

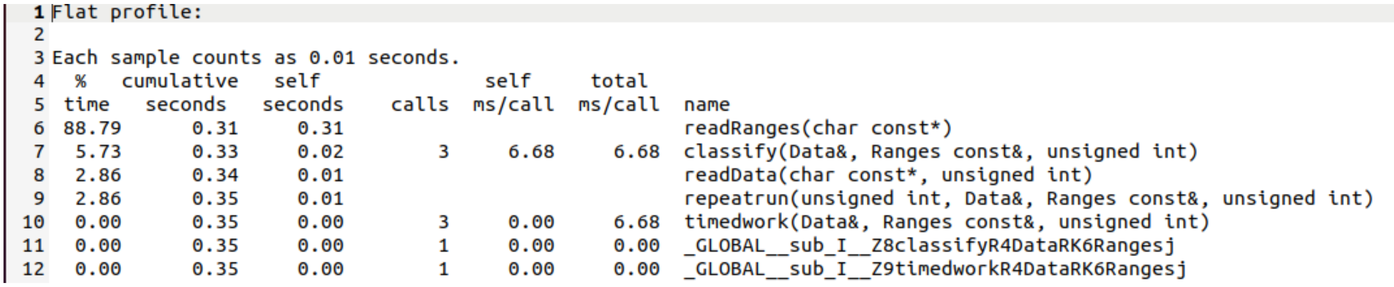
4 Analysis Of Final Code

After the above optimizations were done,I could see a improvement in time taken for execution of code and cache miss rate.The L3 cache miss rate dropped to 0.4% from 0.9% and average execution time dropped from 396 ms to 326 ms.The time taken for the function readRanges() also decreased(from 0.33 to 0.31 in my documented runs) and overall there was an improvement by a factor of 1.21 times for 4 threads and this increases with the increase in the number of cores utilized.

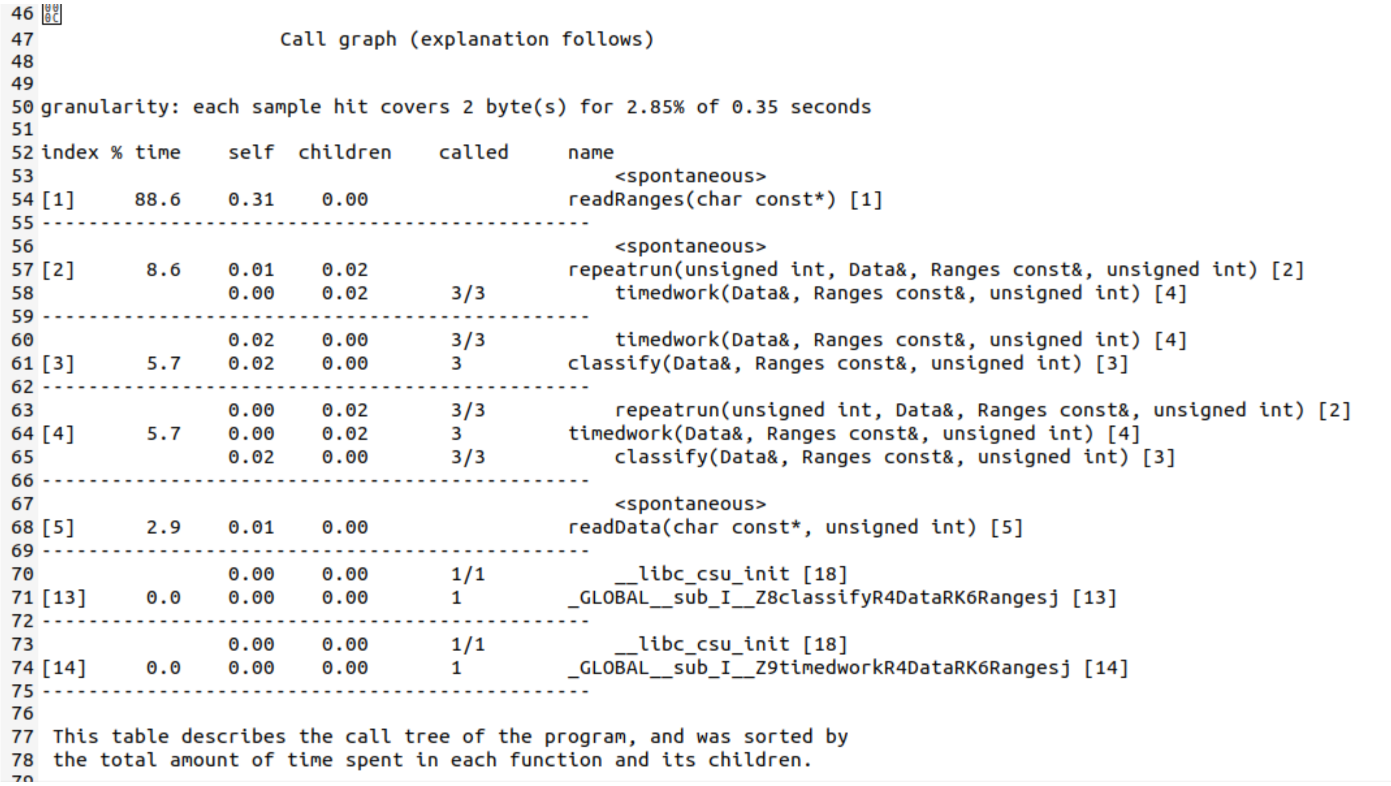


```
ravisriteja@ravisriteja-VirtualBox:~/Desktop/A1$ make classify
g++ -fopenmp -pg -no-pie -fno-builtin -std=c++11 -O2 -c main.cpp
g++ -fopenmp -pg -no-pie -fno-builtin -std=c++11 -O2 -c classify.cpp
g++ -fopenmp -pg -no-pie -fno-builtin -std=c++11 -O2 main.o classify.o -o classify
ravisriteja@ravisriteja-VirtualBox:~/Desktop/A1$ make run
./classify rfile dfile 1009072 4 3
331.661 ms
317.785 ms
329.562 ms
3 iterations of 1009072 items in 1001 ranges with 4 threads: Fastest took 317.785 ms, Average was 326.336 ms
ravisriteja@ravisriteja-VirtualBox:~/Desktop/A1$ make profile
gprof classify gmon.out > anal.txt
ravisriteja@ravisriteja-VirtualBox:~/Desktop/A1$ 
```

4.1 Gprof Analysis

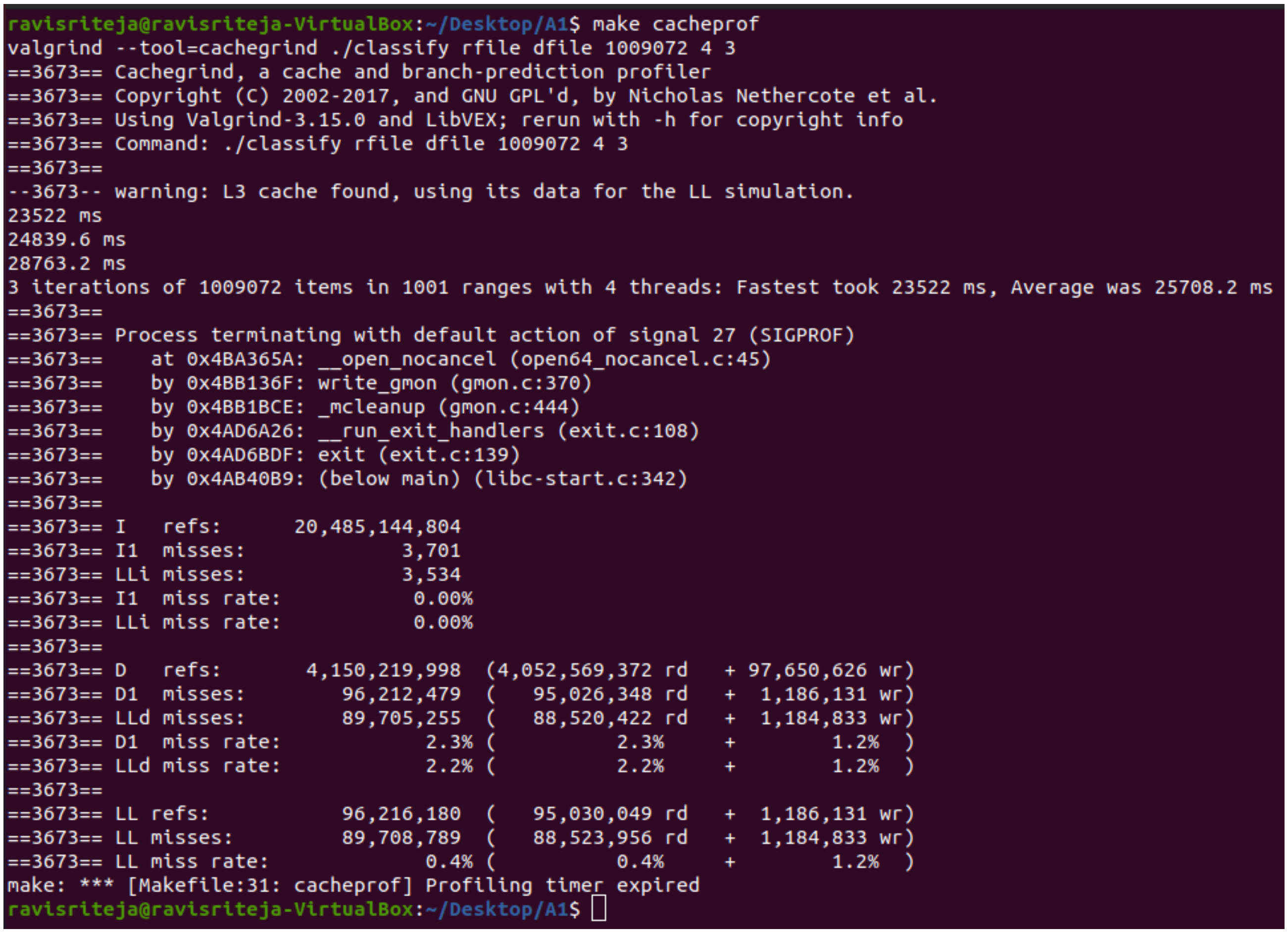


```
Flat profile:

Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  ms/call  ms/call  name
 88.79      0.31     0.31                             readRanges(char const*)
  5.73      0.33     0.02        3     6.68     6.68  classify(Data&, Ranges const&, unsigned int)
  2.86      0.34     0.01                             readData(char const*, unsigned int)
  2.86      0.35     0.01                             repeatrun(unsigned int, Data&, Ranges const&, unsigned int)
  0.00      0.35     0.00        3     0.00     6.68  timedwork(Data&, Ranges const&, unsigned int)
  0.00      0.35     0.00        1     0.00     0.00  _GLOBAL__sub_I__Z8classifyR4DataRK6Rangesj
  0.00      0.35     0.00        1     0.00     0.00  _GLOBAL__sub_I__Z9timedworkR4DataRK6Rangesj
```



```

                     Call graph (explanation follows)


granularity: each sample hit covers 2 byte(s) for 2.85% of 0.35 seconds

index % time    self  children    called     name
                                                 <spontaneous>
[1]     88.6    0.31    0.00                 readRanges(char const*) [1]
-----------------------------------------------
                                                 <spontaneous>
[2]      8.6    0.01    0.02                 repeatrun(unsigned int, Data&, Ranges const&, unsigned int) [2]
                0.00    0.02       3/3           timedwork(Data&, Ranges const&, unsigned int) [4]
-----------------------------------------------
                0.02    0.00       3/3           timedwork(Data&, Ranges const&, unsigned int) [4]
[3]      5.7    0.02    0.00       3         classify(Data&, Ranges const&, unsigned int) [3]
-----------------------------------------------
                0.00    0.02       3/3           repeatrun(unsigned int, Data&, Ranges const&, unsigned int) [2]
[4]      5.7    0.00    0.02       3         timedwork(Data&, Ranges const&, unsigned int) [4]
                0.02    0.00       3/3           classify(Data&, Ranges const&, unsigned int) [3]
-----------------------------------------------
                                                 <spontaneous>
[5]      2.9    0.01    0.00                 readData(char const*, unsigned int) [5]
-----------------------------------------------
                0.00    0.00       1/1           __libc_csu_init [18]
[13]     0.0    0.00    0.00       1         _GLOBAL__sub_I__Z8classifyR4DataRK6Rangesj [13]
-----------------------------------------------
                0.00    0.00       1/1           __libc_csu_init [18]
[14]     0.0    0.00    0.00       1         _GLOBAL__sub_I__Z9timedworkR4DataRK6Rangesj [14]
-----------------------------------------------

 This table describes the call tree of the program, and was sorted by
 the total amount of time spent in each function and its children.
```

4.2 Valgrind Analysis



```
ravisriteja@ravisriteja-VirtualBox:~/Desktop/A1$ make cacheprof
valgrind --tool=cachegrind ./classify rfile dfile 1009072 4 3
==3673== Cachegrind, a cache and branch-prediction profiler
==3673== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas Nethercote et al.
==3673== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==3673== Command: ./classify rfile dfile 1009072 4 3
==3673==
--3673-- warning: L3 cache found, using its data for the LL simulation.
23522 ms
24839.6 ms
28763.2 ms
3 iterations of 1009072 items in 1001 ranges with 4 threads: Fastest took 23522 ms, Average was 25708.2 ms
==3673==
==3673== Process terminating with default action of signal 27 (SIGPROF)
==3673==    at 0x4BA365A: __open_nocancel (open64_nocancel.c:45)
==3673==    by 0x4BB136F: write_gmon (gmon.c:370)
==3673==    by 0x4BB1BCE: _mcleanup (gmon.c:444)
==3673==    by 0x4AD6A26: __run_exit_handlers (exit.c:108)
==3673==    by 0x4AD6BDF: exit (exit.c:139)
==3673==    by 0x4AB40B9: (below main) (libc-start.c:342)
==3673==
==3673== I   refs:      20,485,144,804
==3673== I1  misses:             3,701
==3673== LLi misses:             3,534
==3673== I1  miss rate:           0.00%
==3673== LLi miss rate:           0.00%
==3673==
==3673== D   refs:       4,150,219,998  (4,052,569,372 rd   + 97,650,626 wr)
==3673== D1  misses:        96,212,479  (   95,026,348 rd   +  1,186,131 wr)
==3673== LLd misses:        89,705,255  (   88,520,422 rd   +  1,184,833 wr)
==3673== D1  miss rate:            2.3% (          2.3%     +        1.2%  )
==3673== LLd miss rate:            2.2% (          2.2%     +        1.2%  )
==3673==
==3673== LL refs:           96,216,180  (   95,030,049 rd   +  1,186,131 wr)
==3673== LL misses:         89,708,789  (   88,523,956 rd   +  1,184,833 wr)
==3673== LL miss rate:             0.4% (          0.4%     +        1.2%  )
make: *** [Makefile:31: cacheprof] Profiling timer expired
ravisriteja@ravisriteja-VirtualBox:~/Desktop/A1$ 
```