EPL372
Lab Exercise 4:
gprof and gdb for pThreads

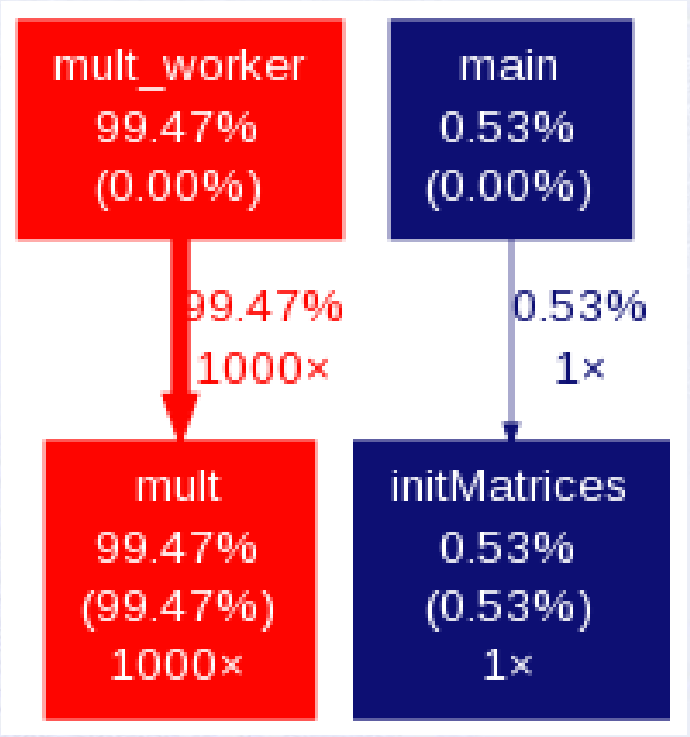
Πέτρος Παναγή

References:

https://sourceware.org/binutils/docs/gprof/
https://code.google.com/p/jrfonseca/wiki/Gprof2Dot
https://sourceware.org/gdb/onlinedocs/gdb/Threads.html
http://users.ece.utexas.edu/~adnan/gdb-refcard.pdf

Profiling with gprof (-pg)

```
gcc -lpthread -g -pg -Wall -Werror
matrix_threads_gprof.c -o
matrix_threads_gprof.out

./matrix_threads_gprof.out

gprof matrix_threads_gprof.out

=====================================

GRAPHICALY:

gprof ./matrix_threads_gprof.out |
python gprof2dot.py | dot -Tpng -o
output.png

From:
http://gprof2dot.jrfonseca.googlecod
e.com/git/gprof2dot.py
```



Profiling with gprof

```
Flat profile:

Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  ms/call  ms/call  name
100.46      7.63     7.63      998     7.65     7.65  mult
  0.00      7.63     0.00        1     0.00     0.00  elapsedTime
  0.00      7.63     0.00        1     0.00     0.00  initMatrices
  0.00      7.63     0.00        1     0.00     0.00  startTime
  0.00      7.63     0.00        1     0.00     0.00  stopTime
```

Call graph

```
index % time    self  children    called     name
                7.63    0.00     998/998         mult_worker [2]
[1]    100.0    7.63    0.00     998         mult [1]
-----------------------------------------------
                                                <spontaneous>
[2]    100.0    0.00    7.63                 mult_worker [2]
                7.63    0.00     998/998         mult [1]
-----------------------------------------------
                0.00    0.00       1/1           main [11]
[3]      0.0    0.00    0.00       1         elapsedTime [3]
-----------------------------------------------
                0.00    0.00       1/1           main [11]
[4]      0.0    0.00    0.00       1         initMatrices [4]
-----------------------------------------------
                0.00    0.00       1/1           main [11]
[5]      0.0    0.00    0.00       1         startTime [5]
-----------------------------------------------
                0.00    0.00       1/1           main [11]
[6]      0.0    0.00    0.00       1         stopTime [6]
-----------------------------------------------
```

GDB (-g)

`gdb matrix_threads_gprof.out`

.....

`b 98` (set a break point on line 98 or `b mult`)

`r` (run the program up to the first break point. Ig you do not have a breakpoint then it will show the creation and exit of the pThreads)

`s` (Step Over)

`c` (Continue)

`info threads` (information about threads)

`thread 3` (switch to thread 3)

Switch to bash shell if you get an error from gdb with $**bash**

GDB (-g)

Use `print row` to read the variable row. (you will need to run c several times for the creation of more threads)

```
(gdb) info threads
[New Thread 0x7fffee225700 (LWP 15719)]
  17 Thread 0x7fffee225700 (LWP 15719)  0x00000037e1ce88c1 in clone () from /lib64/libc.so.6
* 16 Thread 0x7fffeec26700 (LWP 15718)  mult (size=1000, row=14, MA=0x6013a0, MB=0x9d1ca0, MC=0xda25a0) at matrix_threads_gprof.c:98
  15 Thread 0x7fffef627700 (LWP 15717)  mult (size=1000, row=13, MA=0x6013a0, MB=0x9d1ca0, MC=0xda25a0) at matrix_threads_gprof.c:98
  14 Thread 0x7ffff0028700 (LWP 15716)  mult (size=1000, row=12, MA=0x6013a0, MB=0x9d1ca0, MC=0xda25a0) at matrix_threads_gprof.c:98
  13 Thread 0x7ffff0a29700 (LWP 15713) (Exiting)  0x00000037e2405dd0 in __nptl_death_event () from /lib64/libpthread.so.0
  12 Thread 0x7ffff142a700 (LWP 15712)  mult (size=1000, row=10, MA=0x6013a0, MB=0x9d1ca0, MC=0xda25a0) at matrix_threads_gprof.c:98
  11 Thread 0x7ffff1e2b700 (LWP 15711)  mult (size=1000, row=9, MA=0x6013a0, MB=0x9d1ca0, MC=0xda25a0) at matrix_threads_gprof.c:98
  10 Thread 0x7ffff282c700 (LWP 15710)  mult (size=1000, row=8, MA=0x6013a0, MB=0x9d1ca0, MC=0xda25a0) at matrix_threads_gprof.c:98
  9 Thread 0x7ffff322d700 (LWP 15709)  mult (size=1000, row=7, MA=0x6013a0, MB=0x9d1ca0, MC=0xda25a0) at matrix_threads_gprof.c:98
  8 Thread 0x7ffff3c2e700 (LWP 15708)  mult (size=1000, row=6, MA=0x6013a0, MB=0x9d1ca0, MC=0xda25a0) at matrix_threads_gprof.c:98
  7 Thread 0x7ffff462f700 (LWP 15705) (Exiting)  0x00000037e2405dd0 in __nptl_death_event () from /lib64/libpthread.so.0
  6 Thread 0x7ffff5030700 (LWP 15704)  0x00000037e1ceb960 in profil_counter () from /lib64/libc.so.6
  5 Thread 0x7ffff5a31700 (LWP 15703)  0x0000000000400b58 in mult (size=1000, row=3, MA=0x6013a0, MB=0x9d1ca0, MC=0xda25a0) at matrix_threads_gprof.c:101
  4 Thread 0x7ffff6432700 (LWP 15702) (Exiting)  0x00000037e2405dd0 in __nptl_death_event () from /lib64/libpthread.so.0
  1 Thread 0x7ffff7fd8700 (LWP 15695)  0x00000037e1ce88c1 in clone () from /lib64/libc.so.6
```

GDB Exercise

Set one breakpoint on `pthread_create()` and one inside `mult()` i.e. line 98. Run the debuger and monitor the creaton of threads using commands `c` and `info thread`

Switch to any thread with `thread num` and step over the thread using `s`

List the code using `l`

Step multiple lines using `s num`

Use `bt` to see the backtrace of the thread and the values on the arguments when called

Switch between different threads and repeat the steps above.

```
gdb matrix_threads_gprof.out
b 98
r
c
c
c
layout asm
ctrl-x 2
s
info local
print row
bt
Ctrl-x2 (to switch to assembly)
si (one assembly line)
```