Indian Institute of Information Technology, Vadodara

Parallel Programming(cs403)

Prof. Reshmi Mitra

Name - Hemant Kumar
ID - 201352026
Collaborator - Dilip Puri(201351014)

Lab 06

---

1. In the Callgrind tutorial link and the attached file "Lab6_callgrind.pdf" familarize yourself with the callgrind tool.

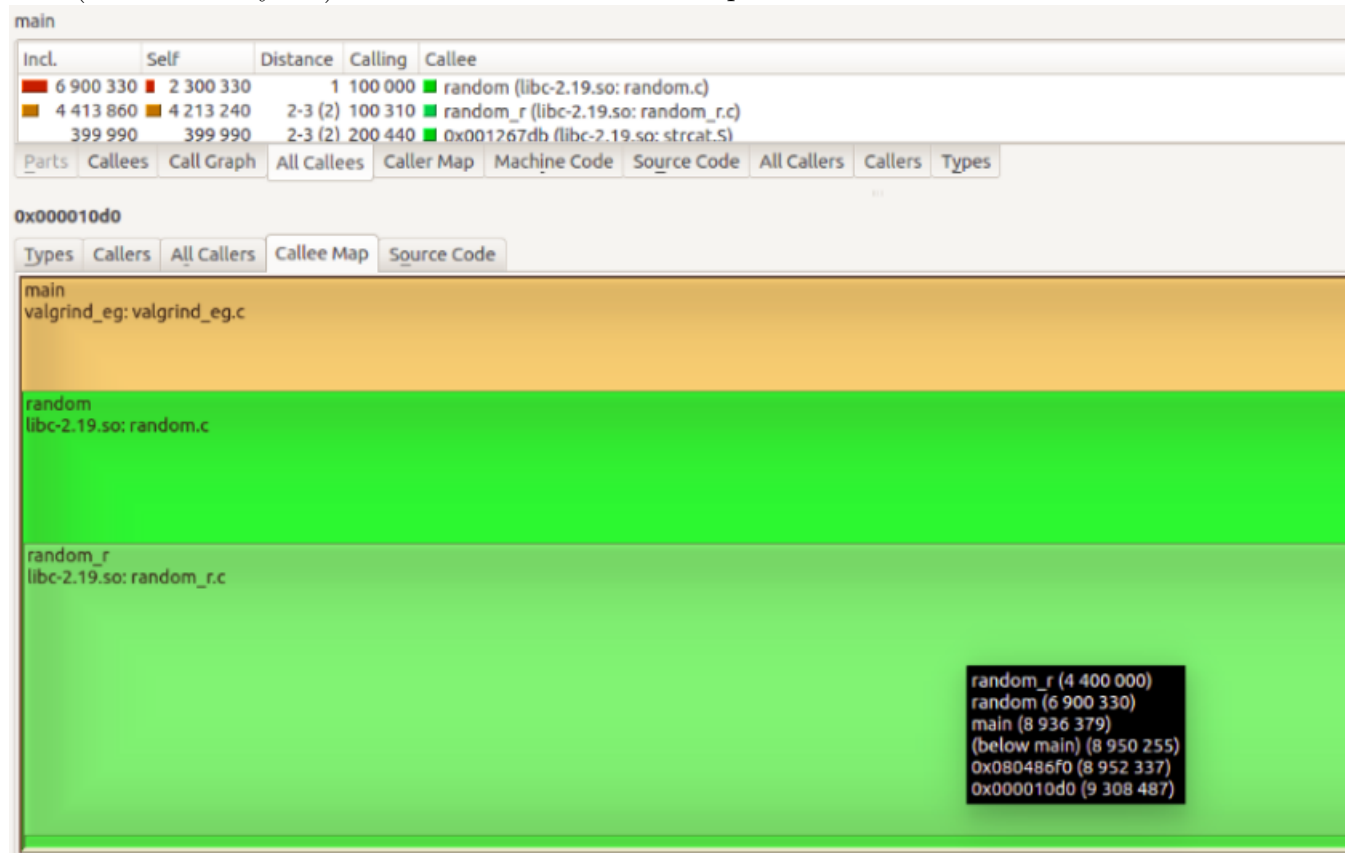2. For the lab exercise, refer to the attached valgrind_eg.c.

   ```
   $ gcc -g -o valgrind_eg valgrind_eg.c -pthread
   $ ./valgrind_eg 1
   completion time for put phase = 2.758351
   0: 0 keys missing
   completion time for get phase = 2.270119
   This is the same program as used in the previous lab.
   valgrind --tool=callgrind --dump-instr=yes --simulate-cache=yes ./val_eg <>
   ```

   Use the above command on this sample C-code to find the call-graph. Toggle the "Cycle Detection" and "% Relative" menu in the top of kcachegrind to observe to menu output. Using this option, find

   - callee map

- cost (in terms of cycles) associated with the most expensive function calls.

- cost (in terms of cycles) associated with pthread_mutex_lock.c and pthread_mutex_unlock.c

```
document.pdf  function_call_kcachegrind  function_call_kcachegrind.txt~  Helpful_commands  Lab06.pdf  valgrind_eg
raghuvar@raghuvar-HCL-ME-Laptop:~/Desktop/PP/PP_lab/lab06$ valgrind --tool=callgrind --dump-instr=yes --simulate-
==4219== Callgrind, a call-graph generating cache profiler
==4219== Copyright (C) 2002-2015, and GNU GPL'd, by Josef Weidendorfer et al.
==4219== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info
==4219== Command: ./valgrind_eg 2
==4219==
--4219-- warning: L3 cache found, using its data for the LL simulation.
==4219== For interactive control, run 'callgrind_control -h'.
completion time for put phase = 407.358839
1: 0 keys missing
0: 0 keys missing
completion time for get phase = 423.969385
==4219==
==4219== Events    : Ir Dr Dw I1mr D1mr D1mw ILmr DLmr DLmw
==4219== Collected : 30032474217 9009883757 2005841782 1273 373521843 358345 1209 20317 6703
==4219==
==4219== I   refs:      30,032,474,217
==4219== I1  misses:             1,273
==4219== LLi misses:             1,209
==4219== I1  miss rate:          0.00%
==4219== LLi miss rate:          0.00%
==4219==
==4219== D   refs:      11,015,725,539  (9,009,883,757 rd + 2,005,841,782 wr)
==4219== D1  misses:       373,880,188  (  373,521,843 rd +       358,345 wr)
==4219== LLd misses:            27,020  (       20,317 rd +         6,703 wr)
==4219== D1  miss rate:            3.4% (          4.1%  +           0.0%  )
==4219== LLd miss rate:           0.0% (          0.0%  +           0.0%  )
==4219==
==4219== LL refs:          373,881,461  (  373,523,116 rd +       358,345 wr)
==4219== LL misses:             28,229  (       21,526 rd +         6,703 wr)
==4219== LL miss rate:            0.0% (          0.0%  +           0.0%  )
```

Incl.	Self	Called	Function	Location
33 764 803 104	2 387	3 start_thread		libpthread-2.19.so: pth
16 887 953 964	3 254 640	2 get_thread		valgrind_eg: valgrind_e
16 884 691 092	16 873 728 058	100 000 get		valgrind_eg: valgrind_e
16 876 830 759	3 153 314	2 put_thread		valgrind_eg: valgrind_e
16 873 677 445	16 873 677 445	100 000 put		valgrind_eg: valgrind_e
8 207 846 900	10	1 clone'2		libc-2.19.so: clone.S
8 207 846 890	225	1 start_thread'2		libpthread-2.19.so: pth
9 308 487	274	(0) 0x000010d0		ld-2.19.so
8 952 337	458	1 0x080486f0		valgrind_eg
8 950 255	633	1 (below main)		libc-2.19.so: libc-start.c
8 936 379	1 789 655	1 main		valgrind_eg: valgrind_e
6 900 330	2 300 330	100 000 random		libc-2.19.so: random.c
6 669 987	6 451 399	100 002 pthread_mutex_lock		libpthread-2.19.so: pth
4 413 860	4 213 240	100 310 random_r		libc-2.19.so: random_r.c
4 290 906	300 126	100 002 pthread_mutex_unlock		libpthread-2.19.so: pth
3 990 780	3 766 826	100 002 __pthread_mutex_unloc...		libpthread-2.19.so: pth

Function Name	Inclusive Cost(In terms of Cycles)	Self Cost(In terms of Cycles)	Ca
get	16 884 691 092	16 873 728 058	
get_thread	16 887 953 964	3 254 640	
put	16 873 677 445	16 873 677 445	
put_thread	16 876 830 759	3 153 314	
main	8 936 379	1 789 655	