Operating System Project 2 : Racing Threads

Andrew Yihang XU
001180130 (yx513719)

October 19, 2013

University at Albany, SUNY
CSI 500 Operating System
For: Prof. Chaiken

1 Computer's details: At least the speed and type of processor, operating system, and memory size.

Processor: 4 AMD Opteron(tm) Processor 850 running at 2392.147 MHz
Operating system: Linux version 3.7.10-1.16-desktop (openSUSE 12.3)
Memory size: 6152328 kB

2 Q1: Write in your notebook HOW MANY DIFFERENT FILES were unpacked into the new L02 directory by this tar command. Don't count files in subdirectories.

There are 6 files were unpacked into the new L02 directory by this tar command.

3 Q2: Write in your notebook the names of all the files that were just made.

racer.s, racer.o, raceDriver

4 Q3: Write in your notebook the failing condition that I programmed into raceDriver.c to make it print the error message that you saw.

The failing condition is:

```
argc != 2 ||sscanf(argv[1],"%d",&nThreads)!=1
|| nThreads < 0 || nThreads > maxThreads)
```

5 Q4A: Study the racer.s file (together with racer.c) and write into your notebook the assembly language instructions that together work to add 1 to the global variable ring.

```
    movl    ring, %eax
    addl    $1, %eax
    movl    %eax, ring
```

6 Q4B: Study the racer.s file (together with racer.c) and write into your notebook the assembly language instructions that together work to add 1 to the global variable ring.

```
movl    ring, %eax
addl    $1, %eax
```

```
    movl    %eax, ring
```

7 Q5: Study the racer-O3.s file to figure out why the optimized, non-volatile version was (A) so fast and (B) showed no errors at all because of the race condition. (C) How did the optimizing compiler make the computer add 1 to ring 20 million times?

(A)The non-volatile version is so fast because it skip the part .L2: which is process of doing addition and saving new result of summation to the value ring. Additionally, the non-volatile one only do once addition, however, the volatile one do 20000000 times addition and 20000000 times substitution for the counter of for cycle and and saving values. That's why non-volatile one much faster than the volatile one.
(B)No errors because there is no code to read value ring which means skipping this process does not effect the correctness.
(C)The optimizing compiler directly give 20000000 to value ring instead of using those code of doing addition for 20000000 times, in this way, the compiled code can increase the performance and directly put value 20000000 into value ring.

```
ccxuy@mothra:~/proj2/L02> ls
config.h          Cutups    raceDriver.c  racer.h
CS.APP.Ch12Stuff  Makefile  racer.c       RacingLab.html
```

8 Q6: List in your notebook the C-functions that were used in racedriver.c to deal with pthreads. Do some web searching and write in your notebook where you can learn exactly how to use those functions, together with pthread_mutex_init, pthread_mutex_lock and pthread_mutex_unlock

The following functions are used:

- pthread_create
- pthread_join

Learning materials for them:

- POSIX Threads(PThreads)

  `http://comsci.liu.edu/~murali/unix/PThread.htm`
- pthread_create

  `http://linux.die.net/man/3/pthread_create`

  `http://www.thegeekstuff.com/2012/04/create-threads-in-linux/`

  `http://www.amparo.net/ce155/thread-ex.html`
- pthread_join

  `http://linux.die.net/man/3/pthread_join`
- pthread_mutex_init

  `http://linux.die.net/man/3/pthread_mutex_init`
- pthread_mutex_lock

  `http://linux.die.net/man/3/pthread_mutex_lock`
- pthread_mutex_unlock

  `http://linux.die.net/man/3/pthread_mutex_unlock`

9 Q7: Read the comments at the top of the Makefile you got, and then skim the rest of its contents. Do some web searching and write in your notebook where one can learn exactly what the concepts target, dependency and command mean in the context of a rule that would be in a Makefile.

Rule A rule is some kind of method could be used by make consist of target, dependency and command.

Target The targets are file names, separated by spaces or maybe wildcard characters.

Dependency Dependencies are those files related to the file that need to be compile and linked.

Command Start with a tab character. Used to specify details of compilation commands.

- `http://comsci.liu.edu/~murali/unix/PThread.htm`
- `http://www.chemie.fu-berlin.de/chemnet/use/info/make/make_4.html`

10 FINISH STEP(counts a lot!): Remove the race condition by using a pthreads mutex. Verify that all the counting is consistant. Experiment and report how the performance is different between the program that suffers from races and the error-free program. (Due date and submission to be announced).

After removed the race condition the performance is much faster than the volatile experiment and no error detected from the error accumulation because the ring count is no different from the ideal number. The reason of the non-mutex protection is slower may cause by thread switching latency. I tried to put the mutex protection just around r̈ing = ring + 1;änd the performance dramatically drop to 38.781s while using 10 thread. It is reasonable that most of the cpu time is wasted on thread context switching.
From the following comparison we can tell that the volatile macro would make the compiler no to skip some operation.
From experiment of volatile with OPT=-O3 optimization under racing condition, we can find the real time is almost the user time divide by thread number. Therefore, it is possible that the processor can only deal with one thread at one time. In the 10 thread experiment, our server use four processor and the real time is about 4 times of user time.

thread	volatile	race	OPT	real	user	sys	diff
volatile without optimization under racing condition							
r1-9 1	y	y	n	0m0.064s	0m0.061s	0m0.002s	0
2	y	y	n	0m0.348s	0m0.685s	0m0.001s	-16210516
10	y	y	n	0m2.027s	0m7.746s	0m0.001s	-161759001
volatile with OPT=-O3 optimization under racing condition							
1	y	y	y	0m0.062s	0m0.058s	0m0.002s	0
2	y	y	y	0m0.278s	0m0.549s	0m0.003s	-19587510
10	y	y	y	0m1.765s	0m6.836s	0m0.004s	-159700113
non-volatile without optimization under racing condition							
1	y	y	n	0m0.064s	0m0.061s	0m0.001s	0
2	y	y	n	0m0.351s	0m0.695s	0m0.001s	-16501552
10	y	y	n	0m2.038s	0m7.861s	0m0.003s	-167522996
non-volatile with OPT=-O3 optimization under racing condition							
1	y	y	y	0m0.003s	0m0.000s	0m0.001s	0
2	y	y	y	0m0.002s	0m0.001s	0m0.001s	0
10	y	y	y	0m0.003s	0m0.001s	0m0.001s	0
after remove racing condition without optimization							
1	y	y	n	0m0.064s	0m0.061s	0m0.003s	0
2	y	y	n	0m0.120s	0m0.119s	0m0.000s	0
10	y	y	n	0m0.589s	0m0.585s	0m0.004s	0
after remove racing condition with OPT=-O3 optimization							
1	y	y	y	0m0.003s	0m0.000s	0m0.001s	0
2	y	y	y	0m0.003s	0m0.000s	0m0.002s	0
10	y	y	y	0m0.004s	0m0.000s	0m0.002s	0

11 Conclusion

Mutex can enhance performance and avoid incorrectly using obsolete value or over-writing values due to race condition in multi-threading program.

Part I

Appendix

A Experiment result and commands

A.1 Computer configuration test

```
ccxuy@mothra:~/proj2/L02> lscpu
Architecture:          i686
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                4
On-line CPU(s) list:   0-3
Thread(s) per core:    1
Core(s) per socket:    1
Socket(s):             4
NUMA node(s):          1
Vendor ID:             AuthenticAMD
CPU family:            15
Model:                 5
Stepping:              10
CPU MHz:               2392.147
BogoMIPS:              4783.98
L1d cache:             64K
L1i cache:             64K
L2 cache:              1024K
NUMA node0 CPU(s):     0-3

ccxuy@mothra:~/proj2/L02> cat /proc/version
Linux version 3.7.10-1.16-desktop (geeko@buildhost) (gcc version 4.7.2 20130108 [gcc-

ccxuy@mothra:~/proj2/L02> lsb_release -a
LSB Version:    n/a
Distributor ID: openSUSE project
Description:    openSUSE 12.3 (i586)
Release:        12.3
```

```
Codename:       Dartmouth

ccxuy@mothra:~/proj2/L02> more /proc/cpuinfo
processor       : 0
vendor_id       : AuthenticAMD
cpu family      : 15
model           : 5
model name      : AMD Opteron(tm) Processor 850
stepping        : 10
microcode       : 0x3a
cpu MHz         : 2392.147
cache size      : 1024 KB
fdiv_bug        : no
hlt_bug         : no
f00f_bug        : no
coma_bug        : no
fpu             : yes
fpu_exception   : yes
cpuid level     : 1
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush mmx fxsr sse sse2 syscall nx mmxext lm 3dnowext 3dnow extd_api
cid
bogomips        : 4784.29
clflush size    : 64
cache_alignment : 64
address sizes   : 40 bits physical, 48 bits virtual
power management: ts fid vid ttp

processor       : 1
vendor_id       : AuthenticAMD
cpu family      : 15
model           : 5
model name      : AMD Opteron(tm) Processor 850
stepping        : 10
microcode       : 0x3a
cpu MHz         : 2392.147
cache size      : 1024 KB
fdiv_bug        : no
hlt_bug         : no
f00f_bug        : no
coma_bug        : no
```

```
fpu             : yes
fpu_exception   : yes
cpuid level     : 1
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush mmx fxsr sse sse2 syscall nx mmxext lm 3dnowext 3dnow extd_api
cid
bogomips        : 4783.98
clflush size    : 64
cache_alignment : 64
address sizes   : 40 bits physical, 48 bits virtual
power management: ts fid vid ttp

processor       : 2
vendor_id       : AuthenticAMD
cpu family      : 15
model           : 5
model name      : AMD Opteron(tm) Processor 850
stepping        : 10
microcode       : 0x3a
cpu MHz         : 2392.147
cache size      : 1024 KB
fdiv_bug        : no
hlt_bug         : no
f00f_bug        : no
coma_bug        : no
fpu             : yes
fpu_exception   : yes
cpuid level     : 1
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush mmx fxsr sse sse2 syscall nx mmxext lm 3dnowext 3dnow extd_api
cid
bogomips        : 4783.98
clflush size    : 64
cache_alignment : 64
address sizes   : 40 bits physical, 48 bits virtual
power management: ts fid vid ttp

processor       : 3
vendor_id       : AuthenticAMD
cpu family      : 15
```

```
model           : 5
model name      : AMD Opteron(tm) Processor 850
stepping        : 10
microcode       : 0x3a
cpu MHz         : 2392.147
cache size      : 1024 KB
fdiv_bug        : no
hlt_bug         : no
f00f_bug        : no
coma_bug        : no
fpu             : yes
fpu_exception   : yes
cpuid level     : 1
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush mmx fxsr sse sse2 syscall nx mmxext lm 3dnowext 3dnow extd_api
cid
bogomips        : 4783.98
clflush size    : 64
cache_alignment : 64
address sizes   : 40 bits physical, 48 bits virtual
power management: ts fid vid ttp

ccxuy@mothra:~/proj2/L02> cat /proc/meminfo
MemTotal:        6152328 kB
MemFree:         5003524 kB
Buffers:          339776 kB
Cached:           608660 kB
SwapCached:            0 kB
Active:           689168 kB
Inactive:         351220 kB
Active(anon):     103984 kB
Inactive(anon):     3372 kB
Active(file):     585184 kB
Inactive(file):   347848 kB
Unevictable:           0 kB
Mlocked:               0 kB
HighTotal:       5322184 kB
HighFree:        4609792 kB
LowTotal:         830144 kB
LowFree:          393732 kB
SwapTotal:       4192252 kB
```

```
SwapFree:        4192252 kB
Dirty:                 4 kB
Writeback:             0 kB
AnonPages:         91980 kB
Mapped:            37880 kB
Shmem:             15412 kB
Slab:              81744 kB
SReclaimable:      69176 kB
SUnreclaim:        12568 kB
KernelStack:        2216 kB
PageTables:         3740 kB
NFS_Unstable:          0 kB
Bounce:                0 kB
WritebackTmp:          0 kB
CommitLimit:     7268416 kB
Committed_AS:     429228 kB
VmallocTotal:     122880 kB
VmallocUsed:       10616 kB
VmallocChunk:     110328 kB
HardwareCorrupted:     0 kB
AnonHugePages:     20480 kB
HugePages_Total:       0
HugePages_Free:        0
HugePages_Rsvd:        0
HugePages_Surp:        0
Hugepagesize:       2048 kB
DirectMap4k:        8184 kB
DirectMap2M:      894976 kB

ccxuy@mothra:~/proj2/L02> free -k
             total       used       free     shared    buffers     cached
Mem:       6152328    1148804    5003524          0     339796     677844
-/+ buffers/cache:     131164    6021164
Swap:      4192252          0    4192252
```

A.2 Unzipping files

```
ccxuy@mothra:~/proj2> tar xvf L02.tar
L02/
L02/racer.c
L02/CS.APP.Ch12Stuff/
L02/CS.APP.Ch12Stuff/csapp.h
L02/CS.APP.Ch12Stuff/busy
L02/CS.APP.Ch12Stuff/csapp.c
L02/CS.APP.Ch12Stuff/badcnt.c
L02/CS.APP.Ch12Stuff/badcnt
L02/CS.APP.Ch12Stuff/23-sync-basic.pdf
L02/CS.APP.Ch12Stuff/busy.c
L02/CS.APP.Ch12Stuff/22-concurrent-programming.pdf
L02/config.h
L02/raceDriver.c
L02/RacingLab.html
L02/Cutups/
L02/Cutups/ring.fig
L02/Cutups/blueCutMeUp.fig
L02/Cutups/blueCutMeUp.pdf
L02/Cutups/redCutMeUp.pdf
L02/Cutups/redCutMeUp.fig
L02/Cutups/ring.pdf
L02/Makefile
L02/racer.h
```

A.3 After unzip files

```
ccxuy@mothra:~/proj2/L02> ls
config.h          Cutups    raceDriver.c  racer.h
CS.APP.Ch12Stuff  Makefile  racer.c       RacingLab.html

ccxuy@mothra:~/proj2/L02> make
gcc -c -S  racer.c -o racer.s
gcc  -c racer.s
gcc -o raceDriver -lpthread raceDriver.c racer.o

ccxuy@mothra:~/proj2/L02> ls
```

```
config.h          Cutups  Makefile    raceDriver.c  racer.h  racer.s
CS.APP.Ch12Stuff  ls      raceDriver  racer.c       racer.o  RacingLab.html
```

A.4 Looking for error message

```
ccxuy@mothra:~/proj2/L02> ./raceDriver
./raceDriver nThreads
where nThreads in [0,20] is the
number of pthreads to race each other.
The printed difference is the accumulated error
```

A.5 Volatile experiment

```
ccxuy@mothra:~/proj2/L02> script
Script started, file is typescript
```

```
ccxuy@mothra:~/proj2/L02> time ./raceDriver 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.064s
user    0m0.061s
sys     0m0.002s
ccxuy@mothra:~/proj2/L02> time ./raceDriver 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.061s
user    0m0.059s
sys     0m0.001s
ccxuy@mothra:~/proj2/L02> time ./raceDriver 1
```

```
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.061s
user    0m0.059s
sys     0m0.002s

ccxuy@mothra:~/proj2/L02> time ./raceDriver 2
ring=23789484 ideal=40000000 diff=-16210516 reldiff=-4.052629e-01

real    0m0.348s
user    0m0.685s
sys     0m0.001s
ccxuy@mothra:~/proj2/L02> time ./raceDriver 2
ring=26592850 ideal=40000000 diff=-13407150 reldiff=-3.351788e-01

real    0m0.348s
user    0m0.665s
sys     0m0.001s
ccxuy@mothra:~/proj2/L02> time ./raceDriver 2
ring=23565355 ideal=40000000 diff=-16434645 reldiff=-4.108661e-01

real    0m0.350s
user    0m0.693s
sys     0m0.001s

ccxuy@mothra:~/proj2/L02> time ./raceDriver 10
ring=38240999 ideal=200000000 diff=-161759001 reldiff=-8.087950e-01

real    0m2.027s
user    0m7.746s
sys     0m0.001s
ccxuy@mothra:~/proj2/L02> time ./raceDriver 10
ring=40188982 ideal=200000000 diff=-159811018 reldiff=-7.990551e-01

real    0m1.936s
user    0m7.525s
sys     0m0.001s
ccxuy@mothra:~/proj2/L02> time ./raceDriver 10
ring=40498207 ideal=200000000 diff=-159501793 reldiff=-7.975090e-01

real    0m1.939s
user    0m7.526s
```

```
sys     0m0.004s
```

```
ccxuy@mothra:~/proj2/L02> make OPT=-O3
gcc -c -S -O3 racer.c -o racer-O3.s
gcc -O3 -c racer-O3.s
gcc -o raceDriver-O3 -lpthread raceDriver.c racer-O3.o
```

```
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.062s
user    0m0.058s
sys     0m0.002s
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.062s
user    0m0.060s
sys     0m0.001s
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.061s
user    0m0.058s
sys     0m0.003s
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 2
ring=20412490 ideal=40000000 diff=-19587510 reldiff=-4.896877e-01

real    0m0.278s
user    0m0.549s
sys     0m0.003s
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 2
ring=20389790 ideal=40000000 diff=-19610210 reldiff=-4.902552e-01

real    0m0.278s
```

```
user    0m0.550s
sys     0m0.001s
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 2
ring=20423252 ideal=40000000 diff=-19576748 reldiff=-4.894187e-01

real    0m0.278s
user    0m0.549s
sys     0m0.002s
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 10
ring=40299887 ideal=200000000 diff=-159700113 reldiff=-7.985006e-01

real    0m1.765s
user    0m6.836s
sys     0m0.004s
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 10
ring=47336623 ideal=200000000 diff=-152663377 reldiff=-7.633169e-01

real    0m1.749s
user    0m6.689s
sys     0m0.003s
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 10
ring=45316480 ideal=200000000 diff=-154683520 reldiff=-7.734176e-01

real    0m1.745s
user    0m6.694s
sys     0m0.003s
```

A.6 Before removed the race condition:

```
ccxuy@mothra:~/proj2/L02> script nonvolatile.results
Script started, file is nonvolatile.results
ccxuy@mothra:~/proj2/L02> make
make: ‘raceDriver’ is up to date.
ccxuy@mothra:~/proj2/L02> time ./raceDriver 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.064s
```

```
user    0m0.061s
sys     0m0.001s
ccxuy@mothra:~/proj2/L02> time ./raceDriver 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.061s
user    0m0.058s
sys     0m0.003s
ccxuy@mothra:~/proj2/L02> time ./raceDriver 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.061s
user    0m0.058s
sys     0m0.002s
ccxuy@mothra:~/proj2/L02> time ./raceDriver 2
ring=23498448 ideal=40000000 diff=-16501552 reldiff=-4.125388e-01

real    0m0.351s
user    0m0.695s
sys     0m0.001s
ccxuy@mothra:~/proj2/L02> time ./raceDriver 2
ring=23480317 ideal=40000000 diff=-16519683 reldiff=-4.129921e-01

real    0m0.351s
user    0m0.696s
sys     0m0.001s
ccxuy@mothra:~/proj2/L02> time ./raceDriver 2
ring=23479253 ideal=40000000 diff=-16520747 reldiff=-4.130187e-01

real    0m0.351s
user    0m0.695s
sys     0m0.002s
ccxuy@mothra:~/proj2/L02> time ./raceDriver 10
ring=32477004 ideal=200000000 diff=-167522996 reldiff=-8.376150e-01

real    0m2.038s
user    0m7.861s
sys     0m0.003s
ccxuy@mothra:~/proj2/L02> time ./raceDriver 10
ring=35666283 ideal=200000000 diff=-164333717 reldiff=-8.216686e-01

real    0m1.984s
```

```
user    0m7.835s
sys     0m0.001s
ccxuy@mothra:~/proj2/L02> time ./raceDriver 10
ring=33122845 ideal=200000000 diff=-166877155 reldiff=-8.343858e-01

real    0m2.068s
user    0m7.886s
sys     0m0.002s
ccxuy@mothra:~/proj2/L02> make OPT=-O3
gcc -c -S -O3 racer.c -o racer-O3.s
gcc -O3 -c racer-O3.s
gcc -o raceDriver-O3 -lpthread raceDriver.c racer-O3.o
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.003s
user    0m0.000s
sys     0m0.001s
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.003s
user    0m0.001s
sys     0m0.001s
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.002s
user    0m0.000s
sys     0m0.001s
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 2
ring=40000000 ideal=40000000 diff=0 reldiff=0.000000e+00

real    0m0.002s
user    0m0.001s
sys     0m0.001s
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 2
ring=40000000 ideal=40000000 diff=0 reldiff=0.000000e+00

real    0m0.002s
user    0m0.000s
sys     0m0.001s
```

```
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 2
ring=40000000 ideal=40000000 diff=0 reldiff=0.000000e+00

real    0m0.002s
user    0m0.000s
sys     0m0.002s
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 10
ring=200000000 ideal=200000000 diff=0 reldiff=0.000000e+00

real    0m0.003s
user    0m0.001s
sys     0m0.001s
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 10
ring=200000000 ideal=200000000 diff=0 reldiff=0.000000e+00

real    0m0.003s
user    0m0.001s
sys     0m0.002s
ccxuy@mothra:~/proj2/L02> time ./raceDriver-O3 10
ring=200000000 ideal=200000000 diff=0 reldiff=0.000000e+00

real    0m0.003s
user    0m0.000s
sys     0m0.003s
ccxuy@mothra:~/proj2/L02> exit
Script done, file is nonvolatile.results
```

A.7 After removed the race condition:

```
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.064s
user    0m0.061s
sys     0m0.003s
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00
```

```
real    0m0.063s
user    0m0.062s
sys     0m0.001s
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.063s
user    0m0.061s
sys     0m0.001s
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver 2
ring=40000000 ideal=40000000 diff=0 reldiff=0.000000e+00

real    0m0.120s
user    0m0.119s
sys     0m0.000s
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver 2
ring=40000000 ideal=40000000 diff=0 reldiff=0.000000e+00

real    0m0.119s
user    0m0.117s
sys     0m0.002s
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver 2
ring=40000000 ideal=40000000 diff=0 reldiff=0.000000e+00

real    0m0.122s
user    0m0.120s
sys     0m0.002s
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver 10
ring=200000000 ideal=200000000 diff=0 reldiff=0.000000e+00

real    0m0.589s
user    0m0.585s
sys     0m0.004s
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver 10
ring=200000000 ideal=200000000 diff=0 reldiff=0.000000e+00

real    0m0.589s
user    0m0.586s
sys     0m0.003s
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver 10
ring=200000000 ideal=200000000 diff=0 reldiff=0.000000e+00
```

```
real    0m0.589s
user    0m0.585s
sys     0m0.004s




ccxuy@mothra:~/things/proj2/L02> time ./raceDriver-O3 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.003s
user    0m0.000s
sys     0m0.001s
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver-O3 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.003s
user    0m0.000s
sys     0m0.001s
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver-O3 1
ring=20000000 ideal=20000000 diff=0 reldiff=0.000000e+00

real    0m0.002s
user    0m0.000s
sys     0m0.002s
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver-O3 2
ring=40000000 ideal=40000000 diff=0 reldiff=0.000000e+00

real    0m0.003s
user    0m0.000s
sys     0m0.002s
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver-O3 2
ring=40000000 ideal=40000000 diff=0 reldiff=0.000000e+00

real    0m0.003s
user    0m0.000s
sys     0m0.002s
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver-O3 2
ring=40000000 ideal=40000000 diff=0 reldiff=0.000000e+00

real    0m0.002s
```

```
user    0m0.001s
sys     0m0.001s
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver-O3 10
ring=200000000 ideal=200000000 diff=0 reldiff=0.000000e+00

real    0m0.004s
user    0m0.002s
sys     0m0.000s
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver-O3 10
ring=200000000 ideal=200000000 diff=0 reldiff=0.000000e+00

real    0m0.003s
user    0m0.000s
sys     0m0.002s
ccxuy@mothra:~/things/proj2/L02> time ./raceDriver-O3 10
ring=200000000 ideal=200000000 diff=0 reldiff=0.000000e+00

real    0m0.003s
user    0m0.002s
sys     0m0.000s
```

B Experiment Source Code

The whole project source code is available from github: `https://github.com/ccxuy/csi500hw/tree/master/proj2/L02`

Several important files are show below:

B.1 Source code of raceDriver.c

```
#include <stdio.h>
#include "racer.h" /*defines nLoops, ring, and racer*/

#define maxThreads 20

int main(int argc, char * argv[])
```

```
{
  pthread_t threads[maxThreads];
  int nThreads;
  void * status;
  int i;

  if(argc != 2 ||
     sscanf(argv[1],"%d",&nThreads)!=1 ||
     nThreads < 0 || nThreads > maxThreads)
    {
      printf("%s nThreads\nwhere nThreads in [0,%d] is the \n"
             "number of pthreads to race each other.\n"
             "The printed difference is the accumulated error\n"
             "due to races",
             argv[0], maxThreads);
      return 1;
    }
  pthread_mutex_init(&mutex,NULL);
  for(i = 0; i < nThreads; i++)
    {
      pthread_create(&threads[i], NULL, racer, (void *) i);
    }
  for(i = 0; i < nThreads; i++)
    {
      pthread_join(threads[i],&status);
    }
  /* Now, all threads have exited. */

  printf("ring=%d ideal=%d diff=%d reldiff=%e\n",
         ring, nLoops*nThreads,
         ring - nLoops*nThreads,
         ((float)(ring - nLoops*nThreads))/(nLoops*nThreads)
         );
  return 0;
}
```

B.2 Source code of racer.c

```
#include "racer.h"
```

```
#include <pthread.h>

#ifdef CONFIG_VOLATILE
volatile
#endif
int ring = 0;

void  * racer(void *tid)
{
  int count;
  pthread_mutex_lock(&mutex);
  for(count = nLoops; count > 0; count--)
    {
      ring = ring + 1;
    }
  pthread_mutex_unlock(&mutex);
}
```

B.3 Source code of racer.h

```
#include "config.h"
#include <pthread.h>
#define nLoops 20000000

#ifdef CONFIG_VOLATILE
volatile
#endif
int ring;
/* ring has static lifetime.
   ring is initialized in racer.c
   All the new threads run racer so
   they all share ring.
*/
pthread_mutex_t mutex;
void * racer( void * arg );
```