CSE 422- Studio 4

1. Selamawit Tegegn(sntegegn@wustl.edu) , Leiquan Pan(lpan22@wustl.edu), yiying Lu([lu.yiying@wustl.edu](mailto:lu.yiying@wustl.edu))

2.

dense\_mm:

input the sizeof matrix

Generating matrices

Multiplying matrices

Multiplication done!

parallel\_dense\_mm:

input the sizeof matrix

Generating matrices

Multiplying matrices

Verification process

Multiplication done!

sort:

input the size of array to sort

Generating array

Sorting array

Verifying array is sorted

Sort done!

sing:

input the number of verse you want to display

output the verse for several times that you want

3.

time ./dense\_mm 100

Generating matrices...

Multiplying matrices...

Multiplication done!

real 0m0.006s

user 0m0.004s

sys 0m0.000s

time ./parallel\_dense\_mm 100

Generating matrices...

Multiplying matrices...

Multiplication done!

real 0m0.044s

user 0m0.008s

sys 0m0.000s

time ./parallel\_dense\_mm 100

Generating matrices...

Multiplying matrices...

Multiplication done!

real 0m0.044s

user 0m0.008s

sys 0m0.000s

time ./sing 100^C

...

-Bilbo, The Lord of the Rings

real 0m0.001s

user 0m0.000s

sys 0m0.004s

4.

real: The time interval between the start time and the end time of your program

user: The amount of time your program running on CPU at a user mode

sys:The amount of time your program running on CPU at a kernel mode

5.

time ./parallel\_dense\_mm 800

Generating matrices...

Multiplying matrices...

Multiplication done!

real 0m4.044s

user 0m4.028s

sys 0m0.000s

time ./dense\_mm 800

Generating matrices...

Multiplying matrices...

Multiplication done!

real 0m3.916s

user 0m3.896s

sys 0m0.000s

Description:

Since the program is running on just one core, which means normally real is bigger than user, the usage of CPU can be calculated as (sys+user)/real.

But when it comes to multiple cores, user sometimes may bigger than real.

6.

time ./sing 1000

real 0m0.446s

user 0m0.000s

sys 0m0.004s

Description: In this case, there are huge amount of operations using I/O, so the sys time increase compare to previous programs. Compare to its calculation complexity, the user time is comparatively small.

7.

CLOCK\_MONOTONIC

This clock won’t be affected user program.

8.

clockid\_t clock\_types[]={CLOCK\_REALTIME,CLOCK\_MONOTONIC,CLOCK\_MONOTONIC\_COARSE,CLOCK\_PROCESS\_CPUTIME\_ID,CLOCK\_THREAD\_CPUTIME\_ID};

Clock 0, Seconds: 0, Nanoseconds: 1

Clock 1, Seconds: 0, Nanoseconds: 1

Clock 2, Seconds: 0, Nanoseconds: 4000000

Clock 3, Seconds: 0, Nanoseconds: 1

Clock 4, Seconds: 0, Nanoseconds: 1

Meanning: The resolution values that outputs show the time precision clock can provide.

9.

CLOCK\_MONOTONIC has a really fine-grained timestamp, which CLOCK\_MONOTONIC\_COARSE doesn't. It's because for some reason, engineers just need a low-res tick grained time resolution, rather than a precise but slow time type. Therefore they introduced \_coarse.

10.

Spend Seconds: 0,Nanoseconds: 300

Method:

I use clock\_gettime function twice, the first time I record its value as the start time, the second time I record its value as the end time. At last the amount of time between them is the time to call a clock\_gettime task.

11.

./time\_parallel\_dense\_mm 100 100

Generating matrices...

Multiplying matrices...

The maximum is Seconds: 0, Nanoseconds: 10800490

The minimum is Seconds: 0, Nanoseconds: 4049540

The mean is Seconds: 0, Nanoseconds: 4314828

Multiplication done!

Description: I tend to think that the mean time is the correct time for a common running. Because it represents the average time to process a task. Compare to the mean time, max time or mini time sometimes can be influences by some software problems or hardware problems, thus those can't be a common running time.