Result Analysis

Purpose: Discuss Results: 1) Analysis of the effectiveness of your parallelization. 2) Possible performance improvement of your program 3)Limitations of your program.

The results ran from different computer configuration as shown below:

Result from 4-CPUs PC	Result from 2-CPUs PC
Architecture: x86_64 CPU op-mode(s): 32-bit, 64-bit Byte Order: Little Endian CPU(s): 4	Architecture: x86_64 CPU op-mode(s): 32-bit, 64-bit Byte Order: Little Endian CPU(s): 2

yinkaihua@ubuntu:~/uwb$ ./convex 100
do you want to display initial data? n
elapsed time = 957
do you want to display result data? y
point[47].x = 7739.000000 .y = 12.000000
point[72].x = 9503.000000 .y = 19.000000
point[27].x = 9956.000000 .y = 1873.000000
point[45].x = 9932.000000 .y = 5060.000000
point[74].x = 9708.000000 .y = 6715.000000
point[28].x = 6862.000000 .y = 9170.000000
point[99].x = 5928.000000 .y = 9529.000000
point[15].x = 3929.000000 .y = 9802.000000
point[64].x = 709.000000 .y = 8927.000000
point[32].x = 336.000000 .y = 6505.000000
point[85].x = 124.000000 .y = 4914.000000
point[53].x = 97.000000 .y = 2902.000000
point[33].x = 846.000000 .y = 1729.000000
point[50].x = 795.000000 .y = 570.000000
point[5].x = 2362.000000 .y = 27.000000
yinkaihua@ubuntu:~/uwb$ ./convex_mp 100 4
do you want to display initial data? n
elapsed time = 1496
do you want to display result data? y
point[47].x = 7739.000000 .y = 12.000000
point[72].x = 9503.000000 .y = 19.000000
point[27].x = 9956.000000 .y = 1873.000000
point[45].x = 9932.000000 .y = 5060.000000
point[74].x = 9708.000000 .y = 6715.000000
point[28].x = 6862.000000 .y = 9170.000000
point[99].x = 5928.000000 .y = 9529.000000
point[15].x = 3929.000000 .y = 9802.000000
point[64].x = 709.000000 .y = 8927.000000
point[32].x = 336.000000 .y = 6505.000000
point[85].x = 124.000000 .y = 4914.000000
point[53].x = 97.000000 .y = 2902.000000
point[33].x = 846.000000 .y = 1729.000000
	[zhengsq@csslab9 os1]$ ./convex 100
	do you want to display initial data? n
	elapsed time = 377
	do you want to display result data? y
	point[47].x = 7739.000000 .y = 12.000000
	point[72].x = 9503.000000 .y = 19.000000
	point[27].x = 9956.000000 .y = 1873.000000
	point[45].x = 9932.000000 .y = 5060.000000
	point[74].x = 9708.000000 .y = 6715.000000
	point[28].x = 6862.000000 .y = 9170.000000
	point[99].x = 5928.000000 .y = 9529.000000
	point[15].x = 3929.000000 .y = 9802.000000
	point[64].x = 709.000000 .y = 8927.000000
	point[32].x = 336.000000 .y = 6505.000000
	point[85].x = 124.000000 .y = 4914.000000
	point[53].x = 97.000000 .y = 2902.000000
	point[33].x = 846.000000 .y = 1729.000000
	point[50].x = 795.000000 .y = 570.000000
	point[5].x = 2362.000000 .y = 27.000000
	[zhengsq@csslab9 os1]$ ./convex_mp 100 4
	do you want to display initial data? n
	elapsed time = 1115
	do you want to display result data? y
	point[47].x = 7739.000000 .y = 12.000000
	point[72].x = 9503.000000 .y = 19.000000
	point[27].x = 9956.000000 .y = 1873.000000
	point[45].x = 9932.000000 .y = 5060.000000
	point[74].x = 9708.000000 .y = 6715.000000
	point[28].x = 6862.000000 .y = 9170.000000
	point[99].x = 5928.000000 .y = 9529.000000
	point[15].x = 3929.000000 .y = 9802.000000
	point[64].x = 709.000000 .y = 8927.000000
	point[32].x = 336.000000 .y = 6505.000000
	point[85].x = 124.000000 .y = 4914.000000
	point[53].x = 97.000000 .y = 2902.000000
	point[33].x = 846.000000 .y = 1729.000000

```
point[50].x = 795.000000 .y = 570.000000
point[5].x = 2362.000000 .y = 27.000000
yinkaihua@ubuntu:~/uwb$ ./convex 20000
do you want to display initial data? n
elapsed time = 202360
do you want to display result data? n
yinkaihua@ubuntu:~/uwb$ ./convex  20000 1
do you want to display initial data? n
elapsed time = 202451
do you want to display result data? n
yinkaihua@ubuntu:~/uwb$ ./convex_mp 20000 1
do you want to display initial data? n
elapsed time = 204014
do you want to display result data? n
yinkaihua@ubuntu:~/uwb$ ./convex_mp 20000 2
do you want to display initial data? n
elapsed time = 115311
do you want to display result data? n
yinkaihua@ubuntu:~/uwb$ ./convex_mp 20000 4
do you want to display initial data? n
elapsed time = 65547
do you want to display result data? n
```

```
point[50].x = 795.000000 .y = 570.000000
point[5].x = 2362.000000 .y = 27.000000
[zhengsq@csslab9 os1]$ ./convex 20000
do you want to display initial data? n
elapsed time = 71947
do you want to display result data? n
[zhengsq@csslab9 os1]$ ./convex 20000 1
do you want to display initial data? n
elapsed time = 71522
do you want to display result data? n
[zhengsq@csslab9 os1]$ ./convex_mp 20000 1
do you want to display initial data? n
elapsed time = 71772
do you want to display result data? n
[zhengsq@csslab9 os1]$ ./convex_mp 20000 2
do you want to display initial data? n
elapsed time = 37379
do you want to display result data? n
[zhengsq@csslab9 os1]$ ./convex_mp 20000 4
do you want to display initial data? n
elapsed time = 38082
do you want to display result data? n
```

Compare two results set, we can see, the number of CPUs matters a lot when executing multiple processes, when going up to 4 processes, it seems that the time spent by 2-CPUs computer is much more than executing 2 processes itself, it’s because one core only can execute one process at one time, for 2 core PC, the best way making full use of it is to execute 2 processes at the same time. Whereas, as shown above, the 4-CPUs computer performances very well when taking on 4 tasks at the same time.

From the statistics the 4-CPUs computer provides, for 20000 points,

Leaves	1	2	4
time	204014	115311	65547
Performance improvement		1.77	3.11

Another thought is, multiple processes may improve performance when computing big data set, for small data set, single process may work better, because change between kernel mode and user mode would add to overhead, which is another concern. As statistics shown for 100 points, 4 process approach reduce the performance, contrary to single process approach.

To sum up, the modified program may performance better on big data set, with enough CPUs supported.