Homework 2 Report

Logan Murray

1. Descriptions
   a. Jacobi: I began coding the Jacobi part of the assignment by copying the code that was provided to us. I looked over the iterative implementation and tried to decide where the parallel parts of the code needed to go. I decided that I would create the parallel region containing the while loop and the two inner for loops. I also made both inner for loops in parallel. However, on the second for loop, I decided that a reduction needed to be made on the addition of the error. I then made another variable called final error that I could store my final error in, so I would not have to worry about messing up the while loop when resetting my error. I also realized that I needed to have a omp single around the last calculation and output of the function because otherwise the loop would end incorrectly.
   b. Histogram: This one was straightforward. I started using my original code, and I put the parallel section around the main for loops. I dealt with the race condition by making a local value for each of the RBG histograms, added them up separately in the loop, and then had a critical section where I summed them all together.
   c. Smoothing: I decided to go for the extra credit here. I first made the function parallel by adding the parallel section and making the first for loop of the calculation parallel. I then added the ability to choose the path of the data folder so that I could loop over multiple images. After that, I made it where only a single thread would read and write the images. However, I made the threads barrier before the calculation began so that the image could be fully loaded, otherwise the image would be distorted.
2. Performance

Execution Time (Fill in this table ONLY)

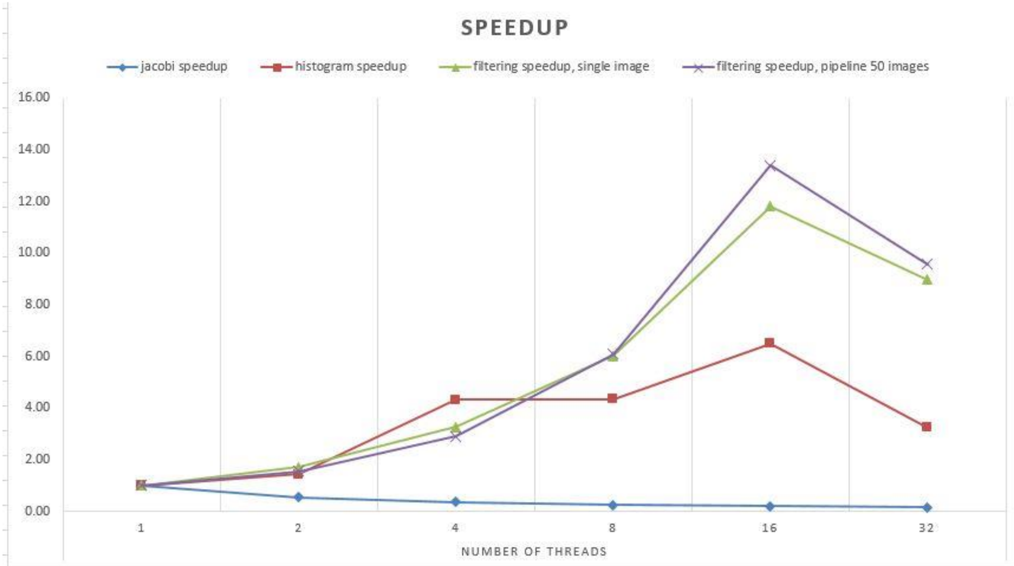
	number of threads					
	1	**2**	**4**	**8**	**16**	**32**
jacobi execution time (s)	5.00	9	14	19	23	30
histogram execution time (ms)	13.00	9	3	2.99	2	4
filtering execution time (deci-s), single image	15.00	8.7	4.58	2.49	1.27	1.67
filtering execution time (s), pipeline 50 images	67.00	43	23	11	5	7

NOTE: If the execution times of the three programs vary dramatically, putting them on one figure may not look go
Please try to use different time unit (ms, s, or even something like 0.5*s) of the three programs so those bars fit v
But for the same program, the execution times MUST use the same time unit.

Speedup (automatically calculated from the Executime Time table)

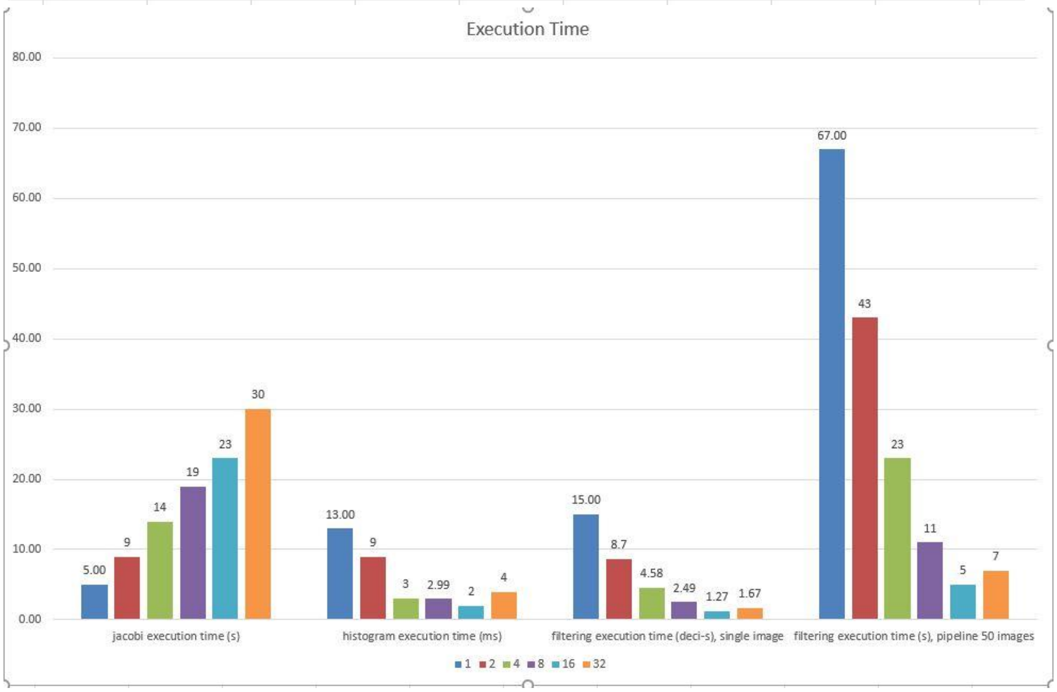
	number of threads					
	1	**2**	**4**	**8**	**16**	**32**
jacobi speedup	1.00	0.56	0.36	0.26	0.22	0.17
histogram speedup	1.00	1.44	4.33	4.35	6.50	3.25
filtering speedup, single image	1.00	1.72	3.28	6.02	11.81	8.98
filtering speedup, pipeline 50 images	1.00	1.56	2.91	6.09	13.40	9.57

   a.



SPEEDUP
jacobi speedup
histogram speedup
filtering speedup, single image
filtering speedup, pipeline 50 images
16.00
14.00
12.00
10.00
8.00
6.00
4.00
2.00
0.00
1
2
4
8
16
32
NUMBER OF THREADS

b.



Execution Time
80.00
70.00
60.00
50.00
40.00
30.00
20.00
10.00
0.00
5.00
9
14
19
23
30
13.00
9
3
2.99
2
4
15.00
8.7
4.58
2.49
1.27
1.67
67.00
43
23
11
5
7
jacobi execution time (s)
histogram execution time (ms)
filtering execution time (deci-s), single image
filtering execution time (s), pipeline 50 images
1
2
4
8
16
32

c.

3.

```
login as: lamurray
Using keyboard-interactive authentication.
XSEDE Authentication
password:
******************************** W A R N I N G ********************************
You have connected to br005.pvt.bridges.psc.edu

This computing resource is the property of the Pittsburgh Supercomputing Center.

It is for authorized use only.  By using this system, all users acknowledge
notice of, and agree to comply with, PSC polices including the Resource Use
Policy, available at http://www.psc.edu/index.php/policies. Unauthorized or
improper use of this system may result in administrative disciplinary action,
civil charges/criminal penalties, and/or other sanctions as set forth in PSC
policies. By continuing to use this system you indicate your awareness of and
consent to these terms and conditions of use.

LOG OFF IMMEDIATELY if you do not agree to the conditions stated in this warning



Please contact remarks@psc.edu with any comments/concerns.

******************************** W A R N I N G ********************************
[lamurray@br005 ~]$ cat/proc/cpuinfo
-bash: cat/proc/cpuinfo: No such file or directory
[lamurray@br005 ~]$ cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 63
model name      : Intel(R) Xeon(R) CPU E5-2695 v3 @ 2.30GHz
stepping        : 2
microcode       : 0x3a
cpu MHz         : 2300.000
cache size      : 35840 KB
physical id     : 0
siblings        : 14
core id         : 0
cpu cores       : 14
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 15
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpelgb rdt
scp lm constant_tsc arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc ap
erfmperf eagerfpu pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 fma
cx16 xtpr pdcm pcid dca sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes
 xsave avx f16c rdrand lahf_lm abm epb invpcid_single tpr_shadow vnmi flexpriori
ty ept vpid fsgsbase tsc_adjust bmi1 avx2 smep bmi2 erms invpcid cqm xsaveopt cq
m_llc cqm_occup_llc dtherm ida arat pln pts
bogomips        : 4594.71
clflush size    : 64
cache_alignment : 64
address sizes   : 46 bits physical, 48 bits virtual
power management:

processor       : 1
vendor_id       : GenuineIntel
cpu family      : 6
model           : 63
model name      : Intel(R) Xeon(R) CPU E5-2695 v3 @ 2.30GHz
stepping        : 2
microcode       : 0x3a
cpu MHz         : 2300.000
cache size      : 35840 KB
physical id     : 0
siblings        : 14
core id         : 2
cpu cores       : 14
apicid          : 4
initial apicid  : 4
fpu             : yes
fpu_exception   : yes
cpuid level     : 15
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpelgb rdt
scp lm constant_tsc arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc ap
erfmperf eagerfpu pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 fma
cx16 xtpr pdcm pcid dca sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes
 xsave avx f16c rdrand lahf_lm abm epb invpcid_single tpr_shadow vnmi flexpriori
ty ept vpid fsgsbase tsc_adjust bmi1 avx2 smep bmi2 erms invpcid cqm xsaveopt cq
```

4. My results were somewhat surprising. Other than Jacobi, my results made sense. However, the runtimes for all the 32 thread sessions were slower than for 16. I can only image that this is due to some sort of memory cap or allocation cap for the threads. In general, as the threads increased so did the performance. For Jacobi however, this is the exact opposite. As the number of threads increased, the performance decreased. I believe this to be due to the large overhead for the reduction operation. However, I am not sure. I also ran all my tests on the supercomputer from XSEDE. I also allow for the pipeline algorithm in my smoothing. Simply input the path to the folder that you want as the first argument when running the smoothing algorithm, and the output images will be in the output folder.