

CSCI 455: Lab #3 — Pi Calculation

Darwin Jacob Groskleg

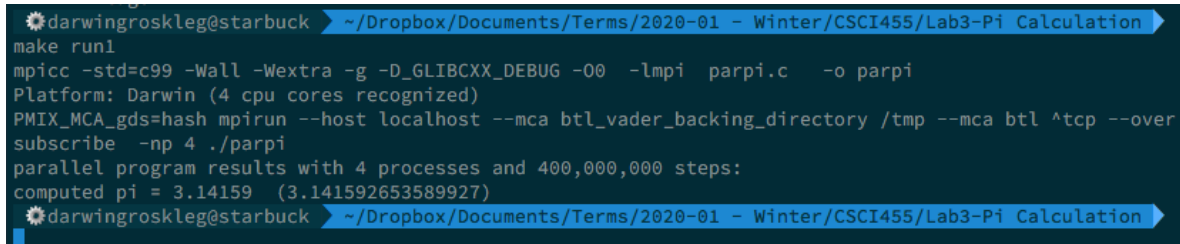
Winter 2020

Contents

Pi Calculation	1
Q1. Any speedup or unusual changes?	1
Q2. Observations after doubling NUM_STEPS	1
parpi.c	2

Pi Calculation

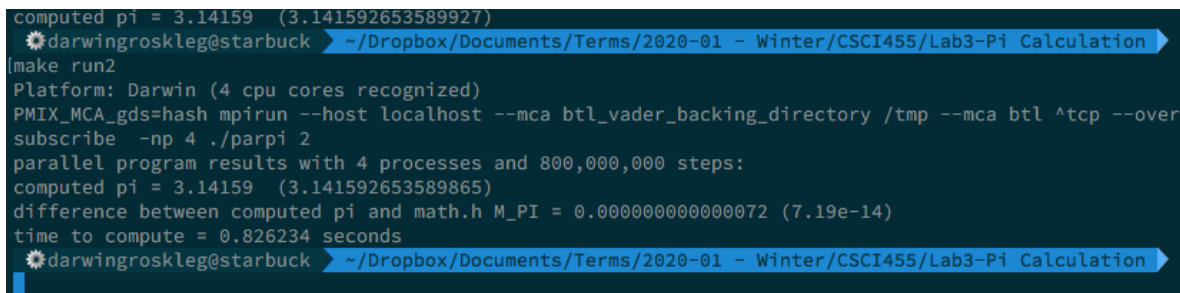
Q1. Any speedup or unusual changes?



```
darwin@groskleg@starbuck > ~/Dropbox/Documents/Terms/2020-01 - Winter/CSCI455/Lab3-Pi Calculation
make run1
mpicc -std=c99 -Wall -Wextra -g -D_GLIBCXX_DEBUG -O0 -lmpi parpi.c -o parpi
Platform: Darwin (4 cpu cores recognized)
PMIX_MCA_gds=hash mpirun --host localhost --mca btl_vader_backing_directory /tmp --mca btl ^tcp --over
subscribe -np 4 ./parpi
parallel program results with 4 processes and 400,000,000 steps:
computed pi = 3.14159 (3.141592653589927)
darwin@groskleg@starbuck > ~/Dropbox/Documents/Terms/2020-01 - Winter/CSCI455/Lab3-Pi Calculation
```

Figure 1: Code with 5, 10, 20, 40, 60 processors

Q2. Observations after doubling **NUM_STEPS**



```
computed pi = 3.14159 (3.141592653589927)
darwin@groskleg@starbuck > ~/Dropbox/Documents/Terms/2020-01 - Winter/CSCI455/Lab3-Pi Calculation
make run2
Platform: Darwin (4 cpu cores recognized)
PMIX_MCA_gds=hash mpirun --host localhost --mca btl_vader_backing_directory /tmp --mca btl ^tcp --over
subscribe -np 4 ./parpi 2
parallel program results with 4 processes and 800,000,000 steps:
computed pi = 3.14159 (3.141592653589865)
difference between computed pi and math.h M_PI = 0.0000000000000072 (7.19e-14)
time to compute = 0.826234 seconds
darwin@groskleg@starbuck > ~/Dropbox/Documents/Terms/2020-01 - Winter/CSCI455/Lab3-Pi Calculation
```

Figure 2: Same code as Q1 but **NUM_STEPS** is doubled

parpi.c

```

1  /* parpi.c
2  * -----
3  * Authors: Darwin Jacob Groskleg
4  * CSCI 455
5  * Lab 3 - Calculation of Pi
6  *
7  * Purpose: to determine the value of pi. The lab uses the method to evaluate
8  *           the integral of  $4/(1+x^2)$  between 0 and 1.
9  *
10 * See 'lecture09-Ch3-Embarassingly Parallel.ppt' for algorithm details.
11 *
12 * TODO:
13 * - [x] Complete the skeleton code.
14 * - [x] Can you run the code with 5, 10, 20, 40, 60 or more processors,
15 *       respectively, to see the speedups? Do you see any unusual changes on
16 *       speedup?
17 * - [x] Can you double the NUM_STEPS with the same number of processors as
18 *       above, then compare and observe the speed ups?
19 */
20 #include <stdio.h>
21 #include <stdlib.h>
22 #include <math.h>
23 #include <time.h>
24 #include <locale.h>
25
26 #include <mpi.h>
27
28 #ifndef M_PI
29 /* copied from not-strictly-standard part of math.h */
30 #define M_PI 3.14159265358979323846
31 #endif /* M_PI */
32
33 #define BASE_NUM_STEPS 400000000
34 int NUM_STEPS = BASE_NUM_STEPS;
35
36 int main(int argc, char *argv[]) {
37     /* initialize for MPI */
38     MPI_Init(&argc, &argv);
39
40     if (argc>1)
41         NUM_STEPS *= atoi(argv[1]);
42
43     /* get number of processes */
44     int cluster_size;
45     MPI_Comm_size(MPI_COMM_WORLD, &cluster_size);
46
47     /* get this process's number (ranges from 0 to cluster_size - 1) */
48     int myid;
49     MPI_Comm_rank(MPI_COMM_WORLD, &myid);
50
51     /* record start time */
52     double start_time = MPI_Wtime();
53
54     /* do computation */
55     double step = 1.0/(double) NUM_STEPS;

```

```

56     double x;
57     double sum = 0.0;
58     for (int i=myid+1; i<=NUM_STEPS; i+=cluster_size) {
59         x = step * ((double) i - 0.5);
60         sum += 4.0 / (1.0 + x*x);
61     }
62     double piece_of_pi = step * sum;
63     double pi = 0.0;
64     MPI_Reduce(
65         &piece_of_pi,
66         &pi,      // receiver
67         1,      // data count
68         MPI_DOUBLE,
69         MPI_SUM,
70         0,      // root node rank (receiver)
71         MPI_COMM_WORLD
72     );
73
74     /* record end time */
75     double end_time = MPI_Wtime();
76
77     /* print results */
78     if (myid == 0) {
79         setlocale(LC_NUMERIC, ""); /* thousand separator for easy reading */
80         printf("parallel program results with '%d' processes and '%d' steps:\n",
81             cluster_size, NUM_STEPS);
82         printf("computed pi = %g (%'17.15f)\n", pi, pi);
83         long double delta_err = fabs(pi - M_PI);
84         printf("difference between computed pi and math.h M_PI = ");
85         printf("%'17.15Lf (%1.2Le)\n", delta_err, delta_err);
86         printf("time to compute = %g seconds\n", end_time - start_time);
87     }
88
89     /* clean up for MPI */
90     MPI_Finalize();
91
92     return 0;
93 }

```