Khadichabonu Valieva

Dr. Zhang

CSC 410

28 Feburary 2024

Assignment 2

1. An overview of the problem

The goal for this assignment is to estimate pi in parallel using C and MPI. We needed to write three scripts, one of which calculates the sum (Sum\_Serial.c) using the following formula:

A black text with black letters

Description automatically generated

The other two scripts had to validate the serial (Sum\_MPI\_v1.c) and parallel (Sum\_MPI\_v2.c) implementations of the sum formula.

1. Description of parallel algorithm design and implementations.

Table 1: In Sum\_MPI\_v1.c we used point-to-point communication functions like MPI\_Send and MPI\_Recv, while for Sum\_MPI\_v2.c, we implemented collective communication functions. In both cases, we used the same values of n numbers and p processes. We were given values of n =1, 3, 10, 1000, 1000000 as well as p set to 1 and 8.

Table 2: We were given the values of processes, however we had to figure out a constant number of n that will result in T runtime that would be greater than 1 and less than 100 seconds. For the convenience, I chose n=10^8 (100 mln) to get exactly six decimal points for the result.

In both cases, the compilation of the code would give me the value of S (Sum) and T (Elapsed Time) that you can see in screenshots provided for question 5.

As for the script themselves, I used the declaration of variables and the beginning of the script as it was shown during class lectures, used MPI function, basic for loop as well as the calculation of sum.

1. Results in Table 1

A table with numbers and a number on it

Description automatically generated

1. Results in Table 2

A table with numbers and letters

Description automatically generated

1. Six screenshots corresponding to three different cases from Table 1 and Table 2 [show the magnolia account, compilation and execution, inputs and outputs, and the consistent results with the Table 1 and 2 data]

Case 1: Sum\_Serial.c for n=1

A screenshot of a computer

Description automatically generated

Case 2: Sum\_Serial.c for n=1000000

A screenshot of a computer

Description automatically generated

Case 3: Sum\_MPI\_v1.c (Table 1, p=1)

A screenshot of a computer

Description automatically generated

Case 4: Sum\_MPI\_v2.c (Table 1, p=8)

A screenshot of a computer program

Description automatically generated

Case 5: Sum\_MPI\_v1.c (Table 2, n=10^8)

A screenshot of a computer program

Description automatically generated

Case 6: Sum\_MPI\_v2.c (Table 2, n=10^8)

A computer screen shot of a program

Description automatically generated

1. Discussion regarding the parallel performance observed in both implementations.

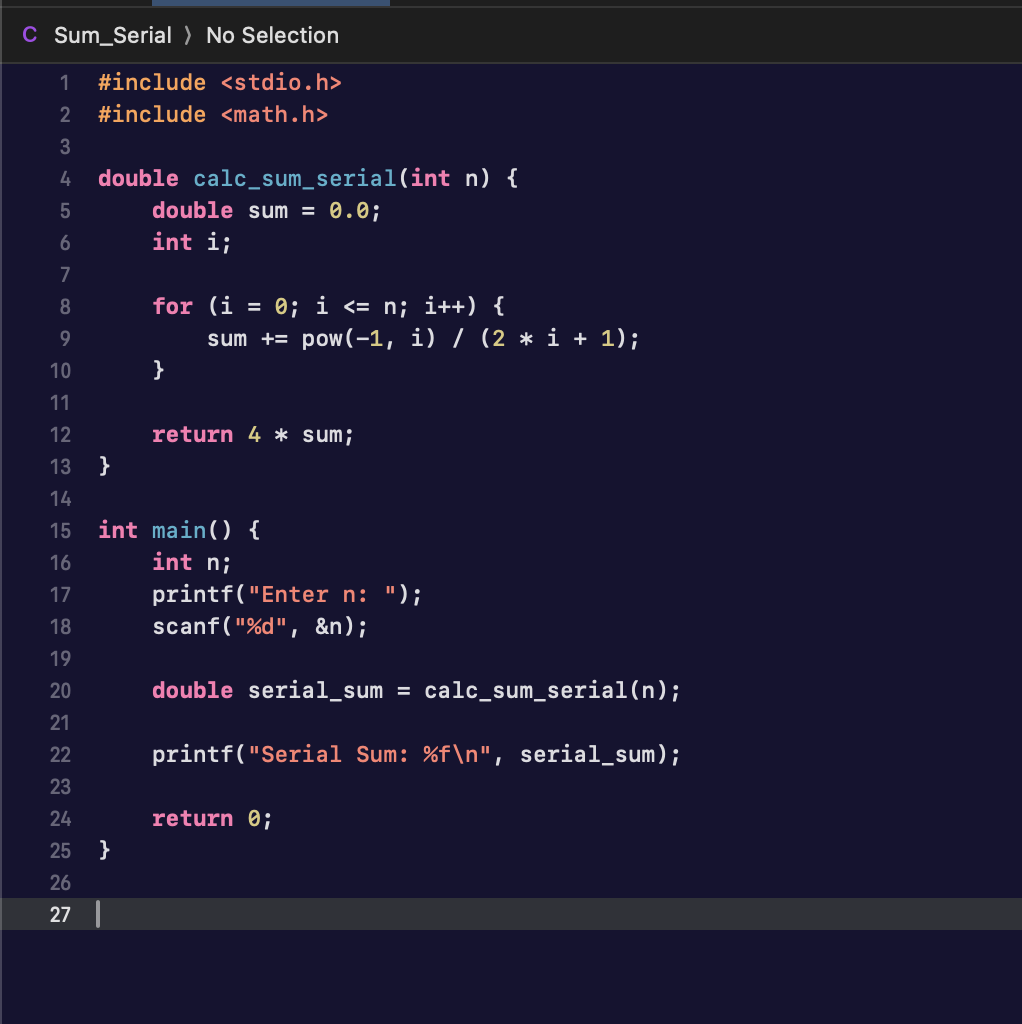
Looking at the Table 1, we can conclude that the answers for scripts with serial and parallel implementation were almost the same. The larger n value we used, the closer to the value of pi we got.

As for Table 2, the values of S were the same compared both parallel computing and to the actual value of pi. One slight difference was in runtime T for each value p (p=1,2,4,8,16), where the difference was from 0.01 being the least to 0.2 being the most.

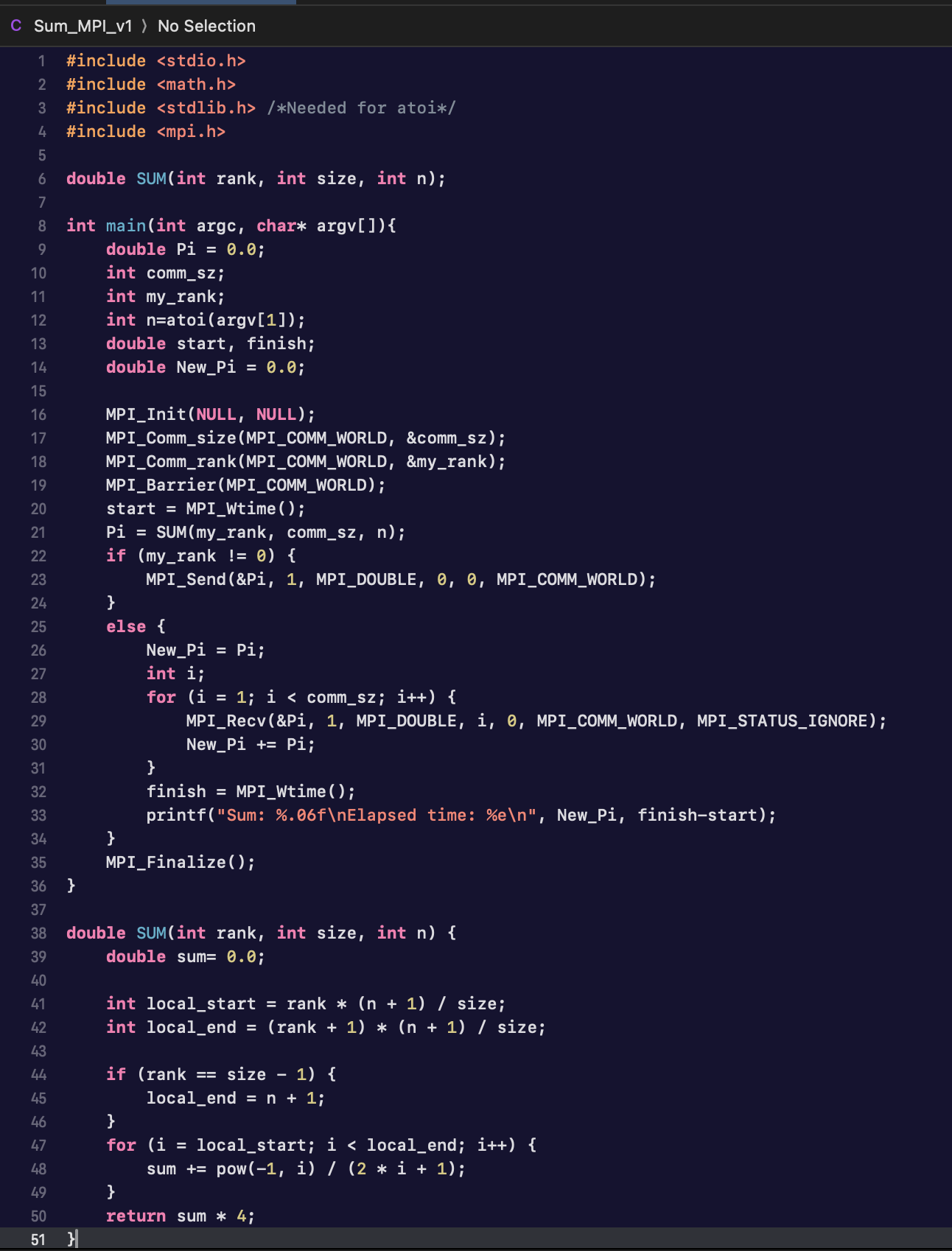
We used the following to find out the values of S and E: S=Ts/Tp, where Ts = T when p=1, and E=S/p.

1. Appendix that includes the serial and parallel programs

* Screenshot of Sum\_Serial.c script:



* Screenshot of Sum\_MPI\_v1.c script:



* Screenshot of Sum\_MPI\_v2.c script:

