Studies in Logic, Vol. x, No. x (xxxx): 1–10
PII: 1674-3202(xxxx)-0x-0001-10

Parallel Programming Project 2

Eren Aytüre
CS, Bilkent Univercity
21200559
eren.ayture@ug.bilkent.edu.tr

Abstract.

1 Introduction

In following sections, i will be discussing about monte carlo simulation which are parallel and serial execution. First I will discuss what kind of parallelism, does my strategy apply for approximating pi in terms of data and task parallelism; then, what is my communication scheme, along with MPI communication functions did i use and why; How MPI address space is used between processes; What would be in a distributed environment where i had many different computers connected together? Would function pointers, to part of the code segment of virtual memory space, still work if you had a truly distributed environment. and my timing results and my graphs according to number of processes.

2 Parallelism Strategy Applied

In this scenario, i divide unit circle by row vise such as from 0.0,(any floating number between 0 to 1) to 1/n,(any floating number between 0 to 1) according to number of processes. Each row partitioning task goes to a responsible processes. In serial part, trial size which is (number of random float numbers/world size)*2 (numbers are between 0 and 1 and *2 is for x y coordinates), and world size which is the number of processes is shared as data memory. Each task is responsible for generating random numbers using MPI MAP FUNCTION, then calculates the hits in circle and creates an array puts them in to array represented by 1.0 using MPI FILTER FUNCTION, THEN MPI FOLD FUNCTION sums 1.0's and adds to result in both serial an parallel execution. On the other hand, in parallel execution, only trial size*2/world size is send to the processes. Number of processes is determined by MPI Comm size.

Received xxxx-xx-xx

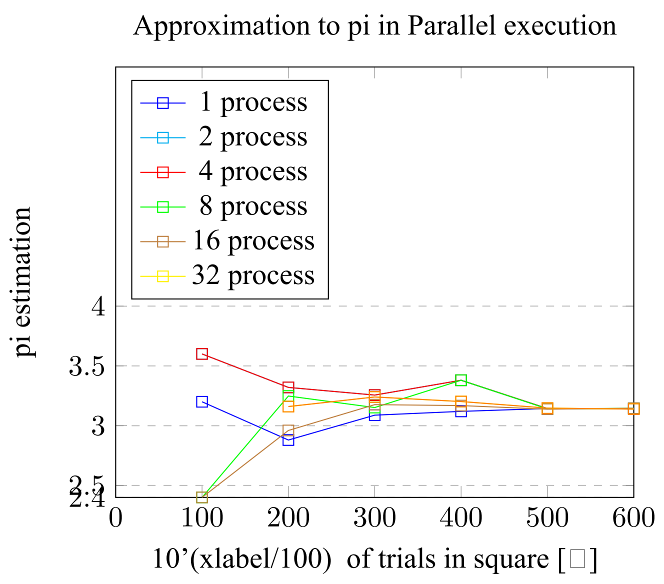
3 MPI Communication Scheme Used

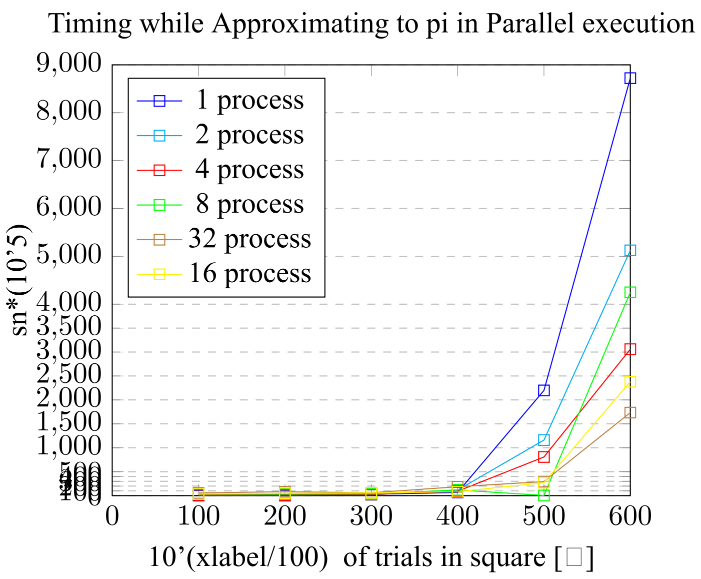
In this scenario, I used point to point communication. Rank zero distributes tasks in row vise. MPI Send and MPI Recv are used. I used MPI received due to MPI Reduce and MPI Allreduce is not allowed. each task send the result to rank 0 and result is calculated. I could have used MPI Scatter but message size increases that effects timing.

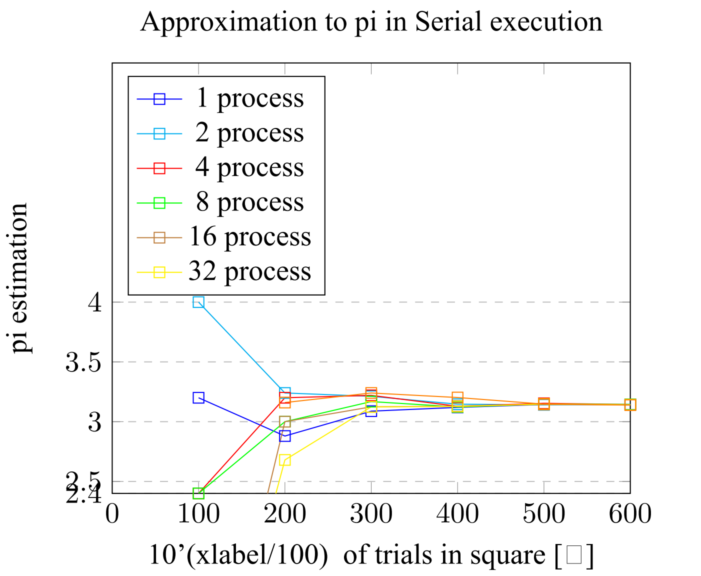
4 How MPI address space is used between processes?

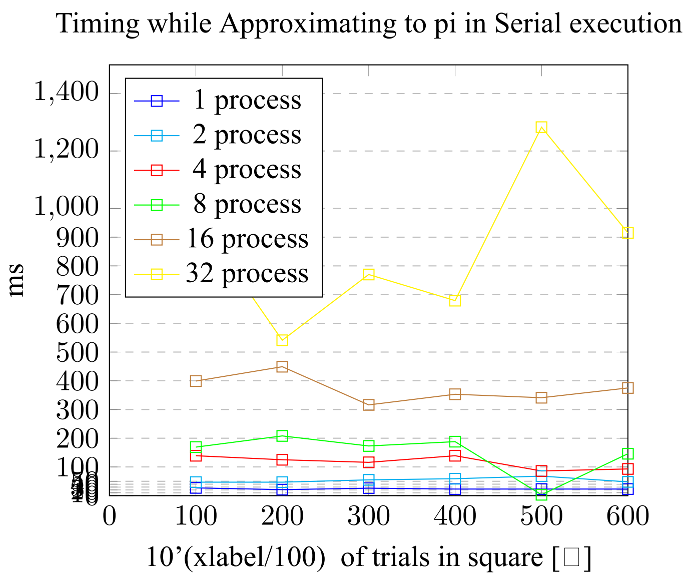
They are different os level processes because they have non uniform memory access. Function pointers and functions will work in mpi environment that have helper.h file. However, trial size must be bigger than number of processes because trial size / number of process may produce zero already.Since, Each task have the same cost of work wit same size input data and functions, it can work on truly distributed environment.

5 Graphs




Timing while Approximating to pi in Parallel execution
sn*(10'5)
10'(xlabel/100) of trials in square [□]
1 process
2 process
4 process
8 process
32 process
16 process
9,000
8,000
7,000
6,000
5,000
4,000
3,500
3,000
2,500
2,000
1,500
1,000
0
100
200
300
400
500
600


Approximation to pi in Serial execution
pi estimation
10'(xlabel/100) of trials in square [□]
1 process
2 process
4 process
8 process
16 process
32 process
4
3.5
3
2.5
2.4
0
100
200
300
400
500
600



In these graphs i see that approximating pi is depends on trial size the more hit on unit circle the more chance by approximating pi . However, in mpi processes, the number of process affects the timing in approximation of pi. The more process divides the work in to distributed levels and efficiency increases.However, on serial execution, since we lose scalability, the more process due to number of threads decreases efficiency. In addition, serial execution is still faster than parallel execution using mpi due to message passing of distributed processes.

eren.ayture@ug.bilkent.edu.tr

摘　　要