Parallel Computing for Science & Engineering Introduction to MPI, Coding Day Spring 2018

Instructors:
Charlie Dey TACC
Lars Koesterke, TACC

Monte Carlo

Pi

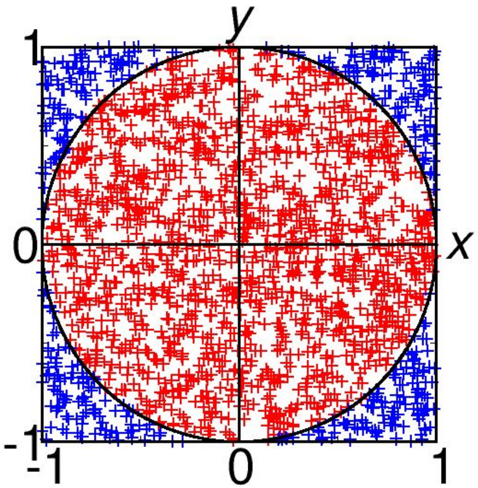
MPI

Sequential Algorithm

A Monte Carlo algorithm for approximating π uniformly generates the points in the square [-1, 1] x [-1, 1]. Then it counts the points which lie in the inside of the unit circle.

Sequential Algorithm

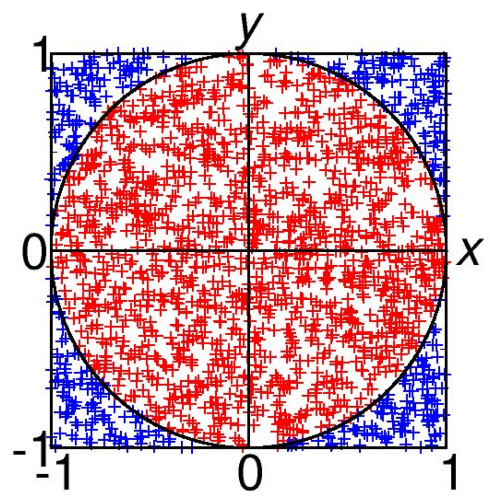
A Monte Carlo algorithm for approximating π uniformly generates the points in the square [-1, 1] x [-1, 1]. Then it counts the points which lie in the inside of the unit circle.



Sequential Algorithm

An approximation of π is then computed by the following formula:

$$4 * \frac{number \quad of \quad points \quad inside}{total \quad number \quad of \quad points}$$



Serial Code

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

void main(int argc, char* argv[])
{
    double n = 10000000;
    double x,y;
    int i;
    int count=0;
    double z;
    double pi;
    srand(time(NULL));
```

```
    //main loop
    for (i=0; i<n; ++i)
    {
        //get random points
        x = (double)random()/RAND_MAX;
        y = (double)random()/RAND_MAX;
        z = sqrt((x*x)+(y*y));
        //check to see if point is in unit circle
        if (z<=1)
        {
            ++count;
        }
    }
    pi = ((double)count/n)*4.0;
    printf("Pi: %f\n", pi);
}
```

How do we parallelize this?

Exercise - Due : 04/03/2018

Parallelize the Monte Carlo method.

Try sample sizes of 10000, 1000000, 1000000000 and see which method is more accurate and which method is faster.

Try node counts of 4, 8, 16, 32

Is there an optimum node/sample size?

Use some strategy:

See if you can implement a MPI_Reduce on *count* and *n*

Try creating a derived data type, so we can send *count* and *n* in one MPI call

Trapezoidal Rule

Pi

MPI

Sequential Algorithm

We can approximate π via $\frac{\pi}{4} = \int_0^1 \sqrt{1-x^2}dx$.

The trapezoidal rule for $\int_a^b f(x)dx \approx \frac{b-a}{2}(f(a)+f(b))$.

Using n subintervals of $[a,b]$:

$$\int_a^b f(x)dx \approx \frac{h}{2}(f(a)+f(b)) + h\sum_{i=1}^{n-1} f(a+ih), \quad h = \frac{b-a}{n}.$$

The Algorithm

this gives us a semi circle

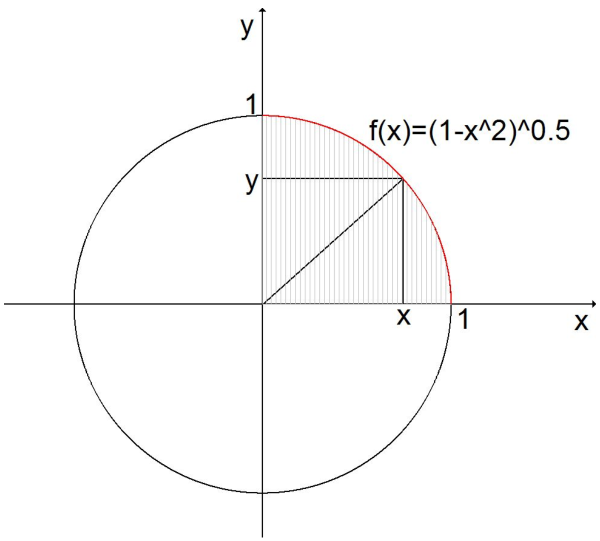
$$f(x) = \sqrt{(1 - x\hat{}2)}$$

this gives us area under the semi circle

$$\int_0^1 f(x)\,dx = \frac{\pi}{4}$$

this would approximate pi

$$PI = 4 * \int_0^1 f(x)\,dx = 4 * \int_0^1 \sqrt{(1 - x\hat{}2)}\,dx$$



Hints for MPI

Basic Housekeeping (initializing)

```
for (i=rank; i<N; i+=size)
{
          x2=d2*i*i;
          result+=sqrt(1-x2);

}
```

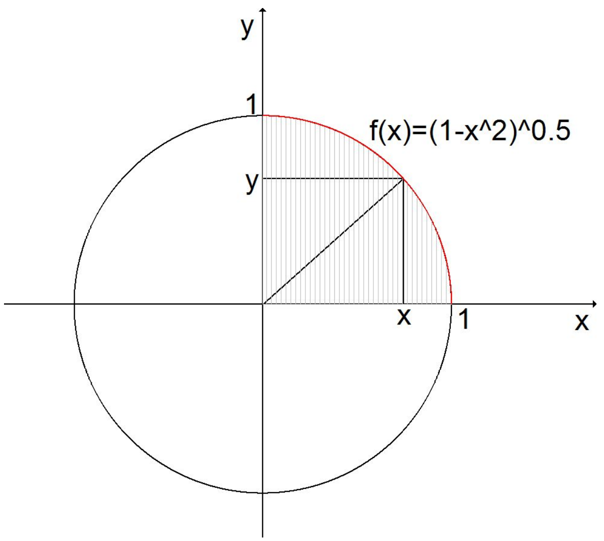
each process will calculate a part of the sum:

```
for (i=rank; i<N; i+=size)
{
          x2=d2*i*i;
          result+=sqrt(1-x2);

}
```

Then, sum up all results using MPI_Reduce:

```
MPI_Reduce(&result, &sum, 1, MPI_DOUBLE, MPI_SUM, 0, MPI_COMM_WORLD);
```



Exercise - Due : 04/03/2018

Find Pi using the Trapezoidal Rule.

Try sample sizes of 10000, 1000000, 1000000000 and see which method is more accurate and which method is faster.

Try node counts of 4, 8, 16, 32

Is there an optimum node/sample size?

How does this compare w/ the Monte Carlo Method?