1 Heat Equation - 1D

1.1 Round Robin Decomposition:

Question: Write the algorithm that computes heat equation using a Round Robin decomposition.

```
CalculateHeat1D( N, Heat_K, p, P){

if(N%P != 0){
if(p == P-1) {
end = N;
}
}

for(i  = p; i < N; i+=p) {  //for loop that starts working depending on its process id and is incremented
                                    by  total number of processsors to implement round robin process
Heat_K-1[i] = GenerateHeat_K-1();   //Function to generate process specific Heat_K-1 element.

if(i == 0) // condition to check if its the start of the array
{
Isend Heat_K-1[i] to p+1
IRecv Heat_K-1[i+1] from p+1
Heat_K[i] = (2*Heat_K-1[i] +Heat_K-1[i+1])/3;
}
else if ( i== N-1) // condition to check if its the end of the array
{
Isend Heat_K-1[i] to p-1
Irecv Heat_K-1[i-1] from p-1
Heat_K[i] = (2*Heat_K-1[i] + Heay_K-1[i-1])/3;
}
else
{

 Isend Heat_K-1[i] to p-1
 Isend Heat_K-1[i] to p+1
 Irecv Heat_K-1[i+1] from p+1
 Irecv Heat_K-1[i-1] from p-1
 Heat_K[i] = (Heat_K-1[i-1] + Heat_K-1[i] +Heay_K-1[i-1])/3;
}
}
}
```

Question: How much communication happen per iteration of the heat equation for a Round Robin decomposition?

- O(N)

1.2 Block Decomposition

Question: Write the algorithm that computes heat equation using a Block decomposition.
-

```
 CalculateHeat1D( N, Heat_K, p, P){

block_size = N/P;
start  = p * block_size
end = (p+1) * block_size


if(N%P != 0){ // if N/P is not divisible entirely by P processors then the last processor is assigned the
                        extra elements
if(p == P-1) {
end = N;
}
}

arrsize = end - start
Heat_K-1[arrsize]; //creating array K-1 of required size depending on the processid.


for(i  = start ; i< end; i++) {  //for loop runs from its correspoding start to end elements

Heat_K-1[i] = GenerateHeat_K-1();   //Function to generate process specific Heat_K-1 element.

if(i == 0) // condition to check if its the start of the array
{
Isend Heat_K-1[i] to p+1
Recv Heat_K-1[i+1] from p+1
Heat_K[i] = (2*Heat_K-1[i] +Heat_K-1[i+1])/3;
}
else if ( i== N-1) // condition to check if its the end of the array
{
Isend Heat_K-1[i] to p-1
Irecv Heat_K-1[i-1] from p-1
Heat_K[i] = (2*Heat_K-1[i] + Heay_K-1[i-1])/3;
}
else if (  i== p*(block_size))  // condition to check if the element is the start element of any process
                        other than zero inorder to get the value Heat_K-1[i-1] from previous value and
                        give Heat_K-1[i] to p-1
{
 Isend Heat_K-1[i] to p-1
 Irecv Heat_K-1[i-1] from p-1
 Heat_K[i] = (Heat_K-1[i-1] + Heat_K-1[i] +Heay_K-1[i-1])/3;
}
else if ( i== end)
```

```
{
  Isend Heat_K-1[i] to p+1
  Irecv Heat_K-1[i+1] from p+1
 Heat_K[i] = (Heat_K-1[i-1] + Heat_K-1[i] +Heay_K-1[i-1])/3;
}
else
{
  Heat_K[i] = (Heat_K-1[i-1] + Heat_K-1[i] + Heat_K-1[i-1])/3;
}

}
```

Question: How much communication happen per iteration of the heat equation when using a Block decomposition?
- O(N)

1.3 Reflection
Question: What data partitioning would you use?

- I would use Block Partitioning over Roundrobin

2.1 1D partitioning: Horizontal stripes:
Question: Write the algorithm that performs y = Ax;x = y; 10 times in a loop if the data is partitioned horizontally.

```
MatVec(N, X[], p, P)
{
 block_size = N/P;
 start = p* block_size ;
end = (p+1) * block_size ;

if(N%P  != 0)
{
 if(p == P-1)
{
  end = N;
}
}

rows = end – start;
int[][] A = new int[rows][N];
int[] Y= new int[rows];
```

```
for(int i= start ; i< end; i++)
{
 for(int j= 0; j< N; j++)
{
  A[i][j] = Generate();
}
}

for(int iter = 0; iter < 10; iter++)
{


for(int i= start ; i< end; i++)
{
 for(int j= 0; j< N; j++)
{
  Y[i] += A[i][j] *X[j];
}
}
```

using for loop starting from i= 0 to P-1
send the data to all the other processes except for the process itself which can be doe by introducing an If condition of

```
  if(i != p) {
      ISend(Y,i);
}
```

Receive the data similarly from all the nodes and replace the existing X with these values depending upon the datacount and source from where the data was sent.
eg. If the data is from 0 process get the data count sent and using for loop like
for i= source*blocksize to (source* blocksize) + datacount

```
for( t = 0 ; t< P; t++){
if(t != p)
{
 Send Y to t;
}
}

for(r = 0 to p-1)
{
 if (r != p)
{
recv Y from r;
source  = r
for (i= source*blocksize; i< (source* blocksize ) + datacount; i++)
{
memcopy received Y to X
}
```

```
}
}
```

Question: How much memory does each node need if the data is partitioned horizontally?

-

N^2

Question: How much communication does the algorithm do per iteration if the data is partitioned horizontally

-

O(N)

2.2 1D partitioning: vertical stripes

Question: Write the algorithm that performs y = Ax;x = y; 10 times in a loop if the data is partitioned vertically.

-

```
MatVec(N, Y[], p, P)
{

 block_size = N/P;

start = p* block_size ;
end = (p+1) * block_size ;

if(N%P  != 0)
{
 if(p == P-1)
{
 end = N;
}
}

size = end-start;

int[][] A = new [N][size];
int[] X= new int[size];


for(int i= 0; i< N; i++)
{
 for(int j= start ; j< end; j++)
{
  A[i][j] = Generate();
}
}

 for(int j= start ; j< end; j++)
```

```
{
  X[j] = Generate();
}

for(int iter = 0; iter < 10; iter++)
{
 for(int i= 0; i< N; i++)
{
 for(int j= start ; j< end; j++)
{
  Y[i] += A[i][j] *X[j];
}
}
```

using for loop starting from i= 0 to P-1
send the data to all the other processes except for the process itself which can be doe by introducing an If condition of

```
  if(i != p) {
       Send(Y, I);
}
```

Receive the data similarly from all the nodes, reduce the values received from all the nodes and replace the existing X with the reduced value.

```
for( t = 0 ; t< P; t++)
{
if(t != p)
{
 Send Y to t;
}
}

for( t = 0 ; t< P; t++)
{
if(t != p)
{
 Recv  Y from t;
for(i=0 ; i< N; i++)
{
ReduceAll(Y[i] into X[i]);
}

}
}

}
}
```

Question: How much memory does each node need if the data is partitioned vertically?

-
N^2

Question: How much communication does the algorithm do per iteration if the data is partitioned vertically?

-
O(N)

2.3 2D partitioning: blocks

Question: Write the algorithm that performs y = Ax;x = y; 10 times in a loop if the data is partitioned in blocks.

```
MatVec(N, Y[], p, P)
{

int square = sqrt(P);
int block_size = N/square;
int div = p / square;
int mod = fmod(p, square);
int rstart = (p - (p - div)) * block_size;
int rend = rstart + (block_size-1);
int cstart = (p - (p - mod)) * block_size;
int cend = cstart + (block_size-1);


int[][] A = new [block_size][block_size];
int[] X= new int[block_size];
int[] Y= new int[block_size];


for(int i= rstart; i< rend; i++)
{
 for(int j= cstart ; j< cend; j++)
{
  A[i][j] = Generate();
}
}

 for(int j= colstart ; j< colend; j++)
{
  X[j] = Generate();
}

for(int iter = 0; iter < 10; iter++)
{
 for(int i= rstart; i< rend; i++)
{
```

```
 for(int j= cstart ; j< cend; j++)
{
  Y[i] += A[i][j] *X[j];
}
}

fake_p = p / square; //to get the exact row in which te data has to be sent
//finding the processor id s to which the data needs to be sent
start_elem = fake_p * square;
end_elem = start_elem + square;

for(e=start_elem ; e<end_elem; e++)
{
  if(e != p)
  {
    send Yto e; // send to the processes here given by e
  }
}

for(e=start_elem ; e<end_elem; e++)
{
  if(e != p)
  {
    Recv Yfrom e;
  for(f = 0; f< datacount; f++) //datacount is the number of elements received
{
     Reduceall(Y[f] to X[f]); Reduce the value in X received in Y
}
  }
}

}
}
```

Question: How much memory does each node need if the data is partitioned in blocks?

- N^2

Question: How much communication does the algorithm do per iteration if the data is partitioned in blocks?

- O(N)

3 Reduction

Question: Fill the following table. For each algorithm and each network structure, answer the following questions. Run a small example if you have difficulty seeing how communication happens; but express all answers for the case with P processors.

Case	How much data on most loaded link	How much data on most loaded node	How long is the longest chain of communication.
Reduce-star on chain	O(P-1)	O(P-1)	P
Reduce-star on clique	O(1)	O(P-1)	1
Reduce-chain on chain	O(1)	O(1)	P
Reduce-chain on clique	O(1)	O(1)	P
Reduce-tree on chain	O(P/2)	O(Log (P))	P/2
Reduce-tree on clique	O(1)	O(Log (P))	1

Question: What do you think is the best algorithm for each network structure? (One of the given algorithm or a different one.)

- Among the above algorithms the best fitting algorithm is Reduce-Tree Algorithm.