Heat Simulation using MPI

Group 6

Gaurav Kukreja
Umbreen Sabir
Viktor Bogischef

Heat Simulation using MPI

Utilized Functions

Creating a 2d cartesion topology:

```
int MPI_Cart_create(MPI_Comm comm_old, int ndims, int *dims, int *periods,
                    int reorder, MPI_Comm *comm_cart)
```

Converting between coordinates and rank:

```
int MPI_Cart_rank(MPI_Comm comm, int *coords, int *rank)
int MPI_Cart_coords(MPI_Comm comm, int rank, int maxdims, int *coords)
```

Nonblocking send and receive commands:

```
int MPI_Isend(void *buf, int count, MPI_Datatype datatype, int dest, int tag,
              MPI_Comm comm, MPI_Request *request)
int MPI_Irecv(void *buf, int count, MPI_Datatype datatype, int source,
              int tag, MPI_Comm comm, MPI_Request *request)
int MPI_Wait(MPI_Request *request, MPI_Status *status)
```

Send and resceive buffers to transfer columns of the matrix

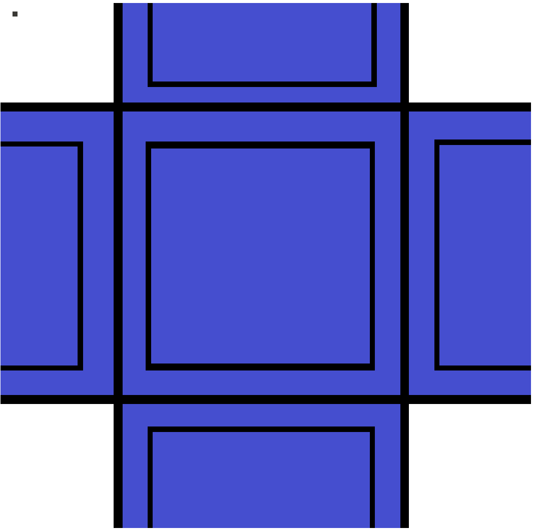
```
// Example: Receiving the left border
for (i = 1; i < size_y -1 ; i++)
        u[i*size_x] = recvbuf_left[i-1];
```

Heat Simulation using MPI

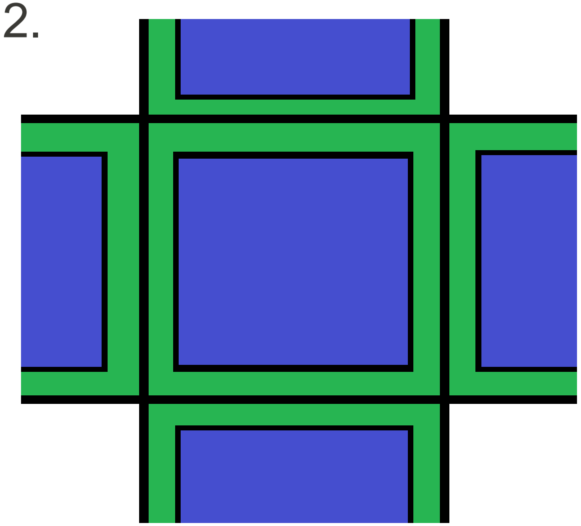
Code Jacobi

```
For each iteration do:
    Calculate the borders
    Send border values to neighbors
    Calculate the inner part
    Receive border values from neighbors
```

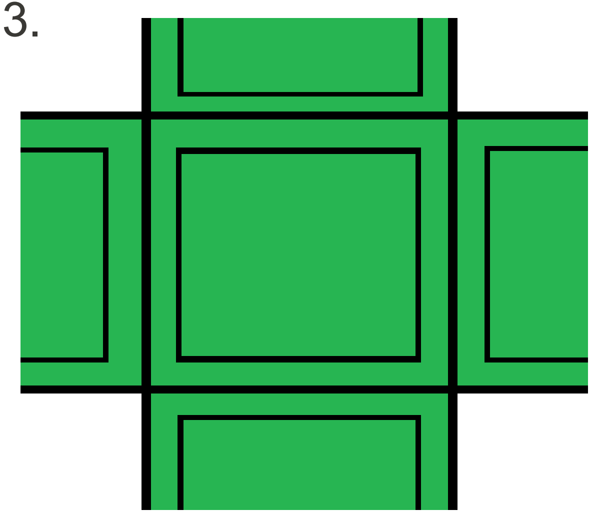
1.



2.



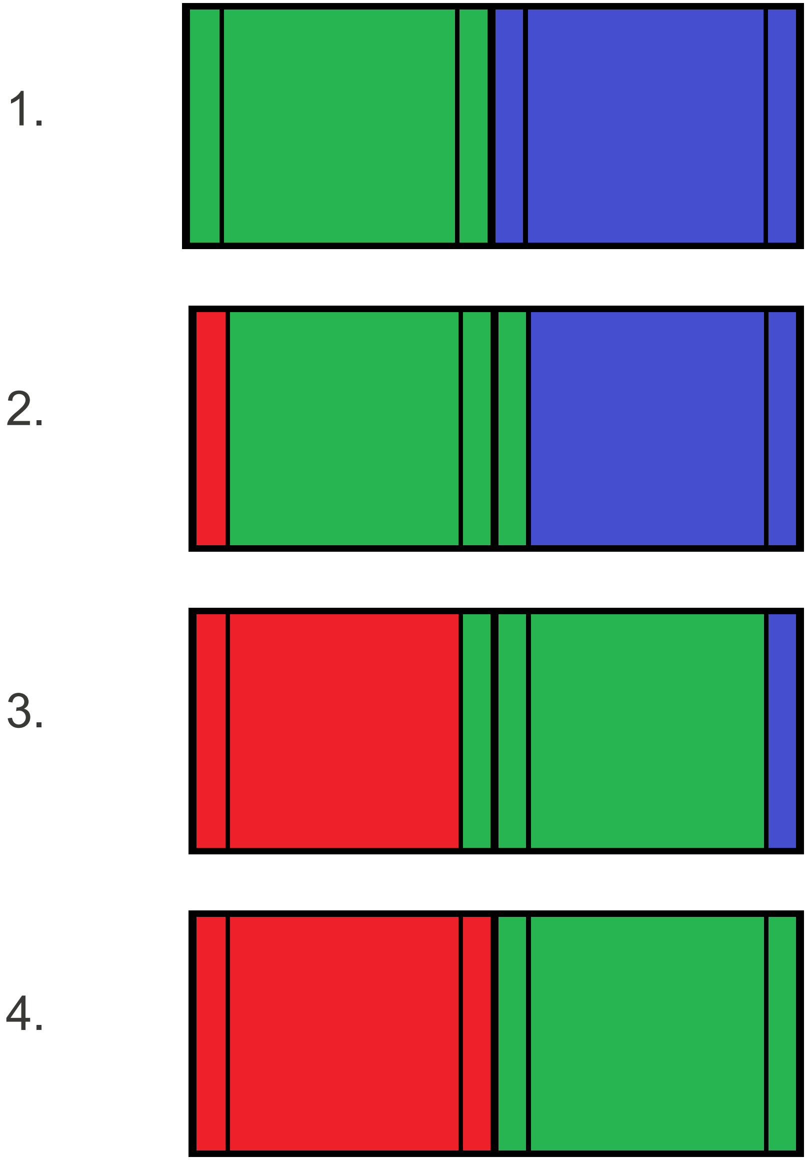
3.



Heat Simulation using MPI

Code 1D-Gauss

```
For each iteration do:
    Receive border values from left neighbor
    Calculate left border
    Send border values to left neighbor
    Calculate inner part
    Receive border values from right neighbor
    Calculate right border
    Send border values to right neighbor
```

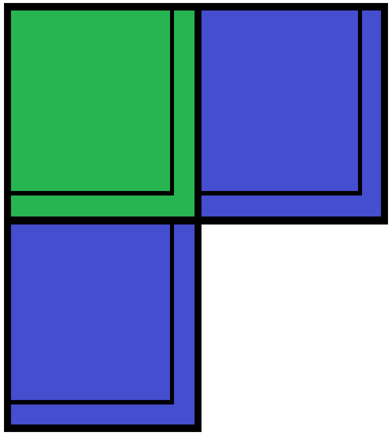


Heat Simulation using MPI

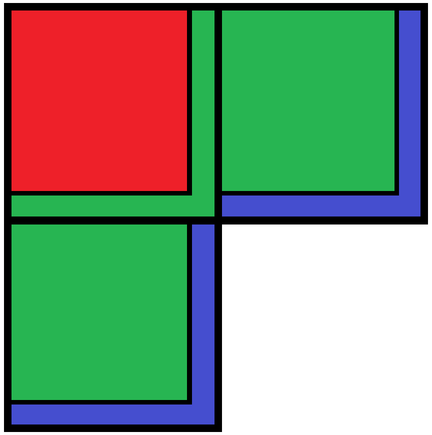
Code 2D-Gauss

```
For each iteration do:
    Receive border values from top and left neighbor
    Calculate top and left border and inner part
    Receive border values from right and bottom neighbor
    Calculate bottom and right border
    Send border values to all neighbors
```

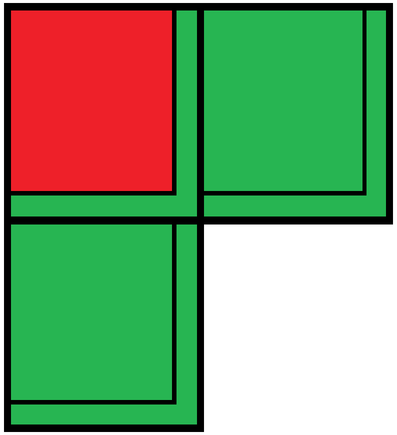
1.



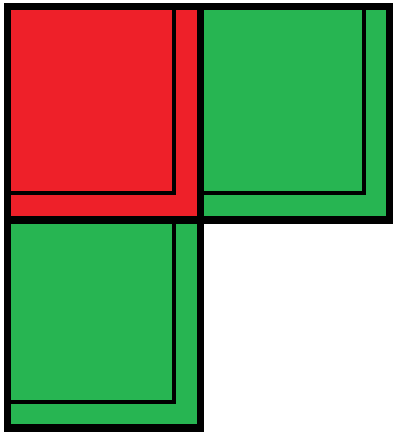
2.



3.

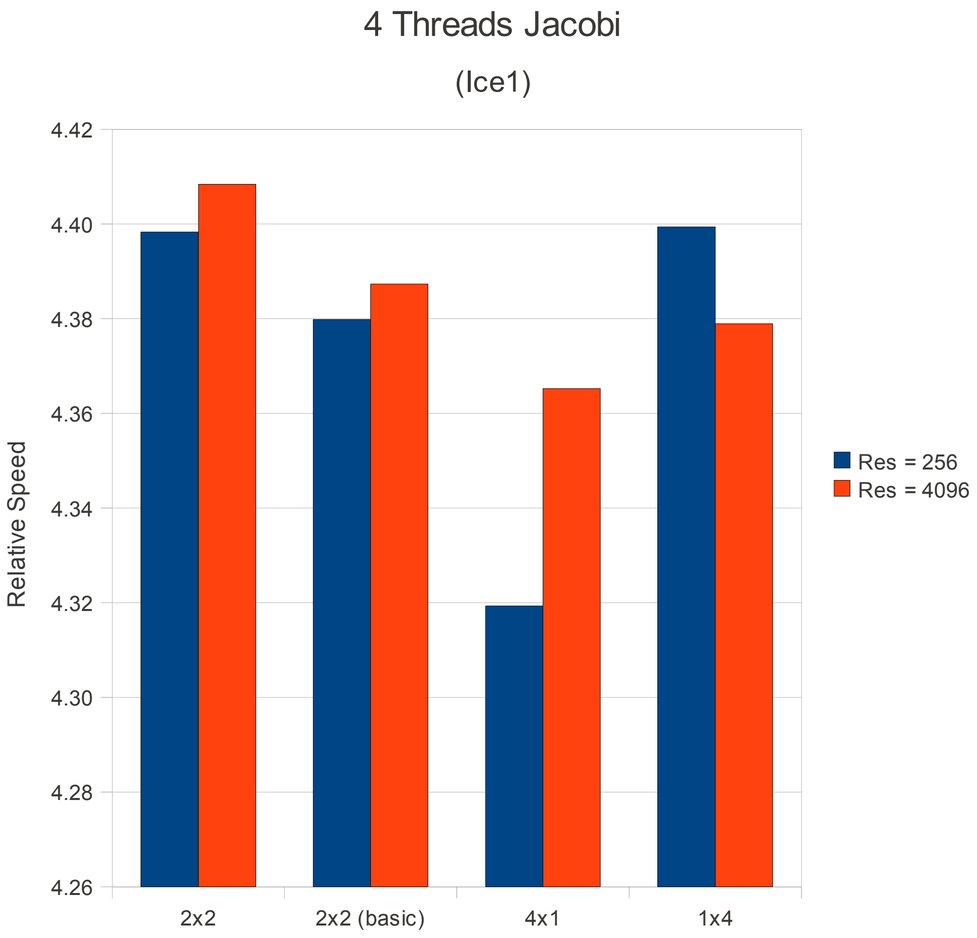


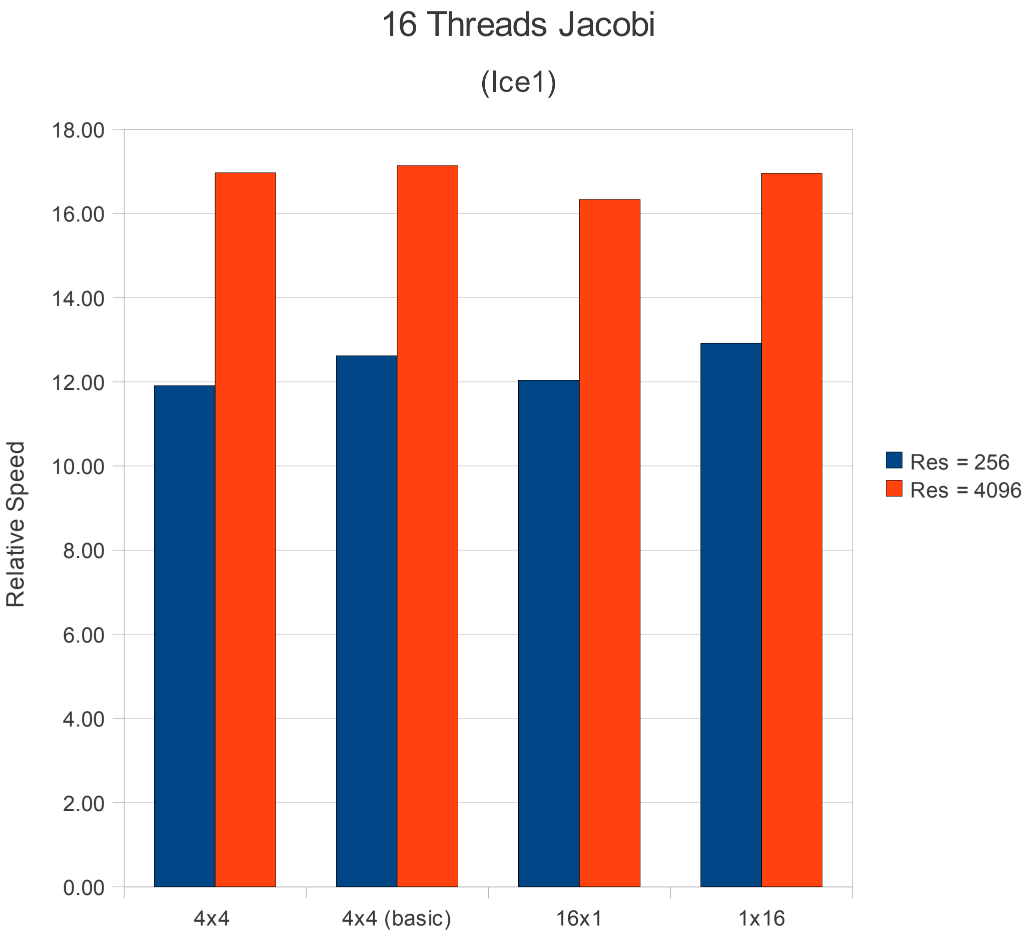
4.



Heat Simulation using MPI

Results Jacobi

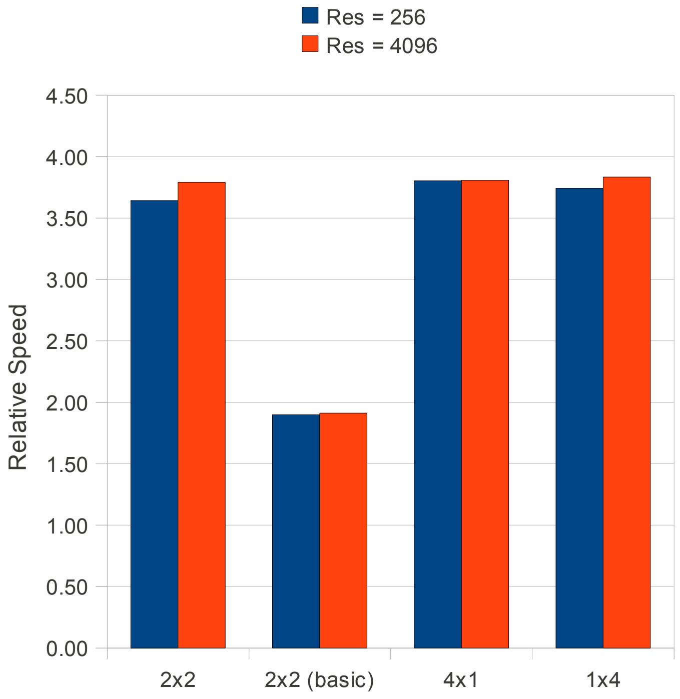




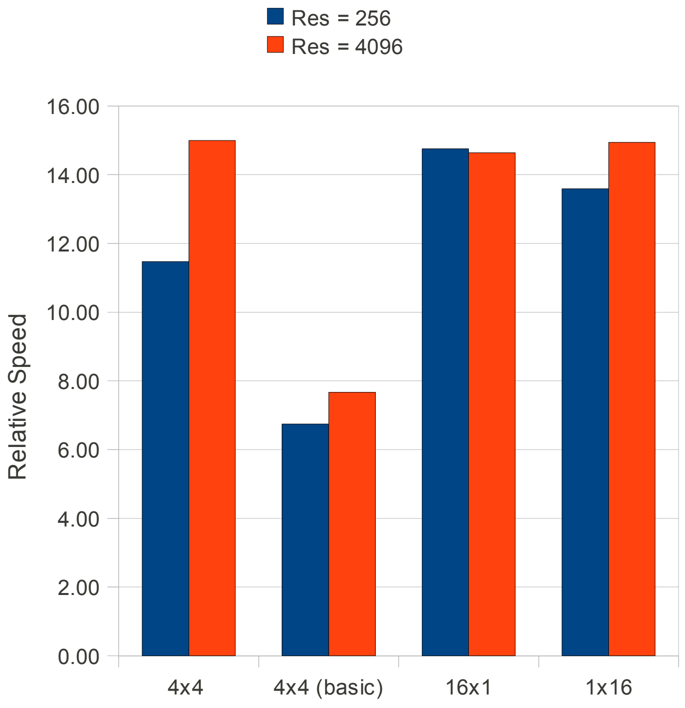
„Basic“ means that the blocks are not devided into inner part and border, but calculated at once.

Heat Simulation using MPI

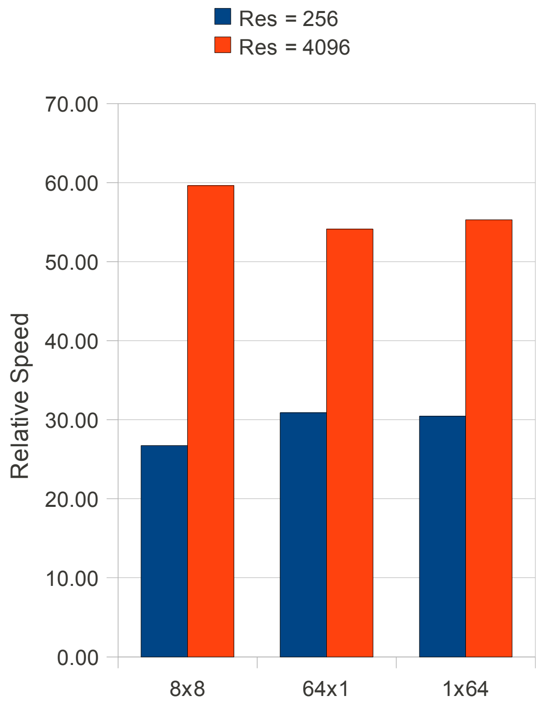
Results Gauss


4 Threads Gauss-Seidel

(Ice1)


16 Threads Gauss-Seidel

(Ice1)


64 Threads Gauss-Seidel

(Ice1)

„Basic“ means that the blocks are not devided into upper left part and lower right border, but calculated at once.

Raw Results in MFlops for Gauss (Ice1)

4 Threads

Resolution	sequential	2x2	2x2 (basic)	4x1	1x4
256	842	3068	1600	3202	3152
4096	819	3104	1567	3118	3139

16 Threads

Resolution	sequential	4x4	4x4 (basic)	16x1	1x16
256	842	9657	5680	12425	11443
4096	819	12281	6281	11987	12234

64 Threads

Resolution	sequential	8x8	64x1	1x64
256	842	22521	26017	25639
4096	819	48842	44315	45296

Raw Results in MFlops for Jacobi (Ice1)

4 Threads

Resolution	sequential	2x2	2x2 (basic)	4x1	1x4
256	974	4284	4266	4207	4285
4096	950	4188	4168	4147	4160

16 Threads

Resolution	sequential	4x4	4x4 (basic)	16x1	1x16
256	974	11603	12293	11726	12584
4096	950	16122	16283	15517	16111