Parallelization Approach: Some examples

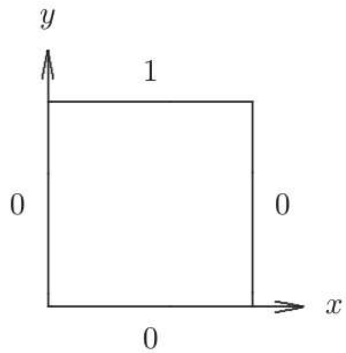
V. Venkatesh Shenoi

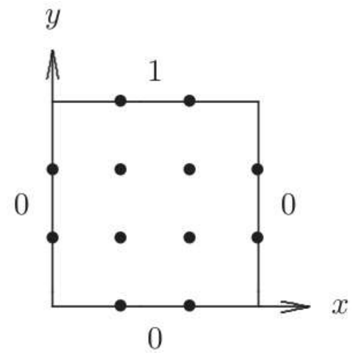
HPC - Infrastructure & Ecosystem, C-DAC, Pune

venkateshs@cdac.in

High Performance Computing for scientists and engineers

PDE Solver: Soln. Laplace equation





Laplace Equation in two dimensions

$\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = 0,$ with boundary conditions,

$U(x,\ 0) = 0, U(x,\ 1) = 1,$

$U(0,\ y) = 0, U(1,\ y) = 0$

Finite Differences method, the solution $\boxed{0 < x < 1, 0 < y < 1}$ can be obtained.

Discretize along x and y directions, $M \times M$ grid with $\Delta x = \Delta y$, $U(x,\ y) \equiv U(i,\ j)$

$x = i\Delta x$, $i \in [0,\ M]$, with $\Delta x = 1/M$,

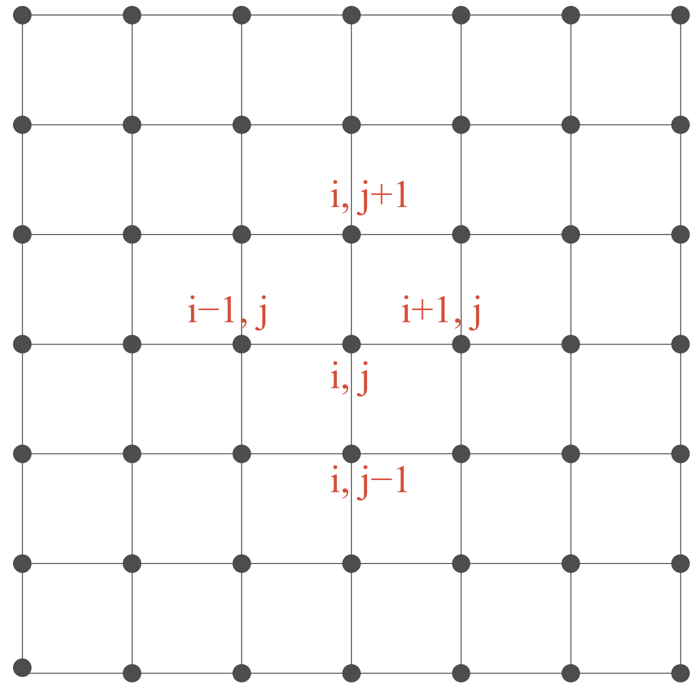
$y = j\Delta y$, $j \in [0,\ M]$, with $\Delta y = 1/M$,

$$\frac{(U_{i+1,\ j} - 2U_{i,\ j} + U_{i-1,\ j})}{(\Delta x)^2} + \frac{(U_{i,\ j+1} - 2U_{i,\ j} + U_{i,\ j-1})}{(\Delta y)^2} = 0$$

$$U_{i,\ j} = \frac{1}{4}(U_{i+1,\ j} + U_{i-1,\ j} + U_{i,\ j+1} + U_{i,\ j-1})$$

$\boxed{5-\textbf{ point stencil: a point and its neighbours}}$

Soln. Laplace equation (stencil computation)

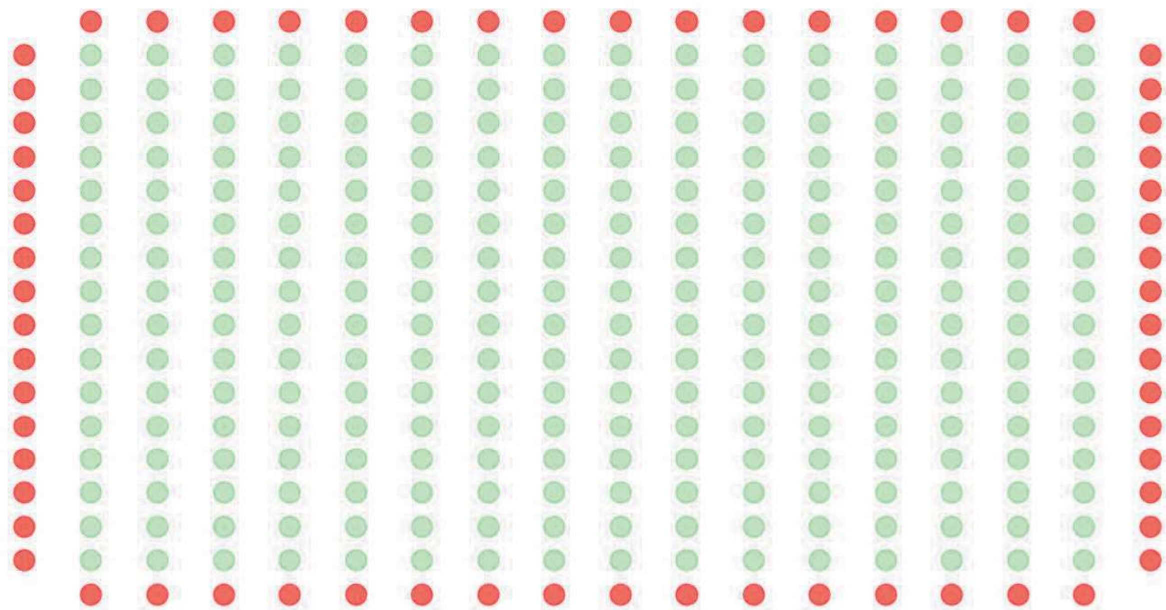


$$U_{i,\,j}^{(n+1)} \;=\; \frac{1}{4}\left(U_{i+1,\,j}^{(n)} + U_{i-1,\,j}^{(n)} + U_{i,\,j+1}^{(n)} + U_{i,\,j-1}^{(n)}\right)$$

Solution can be obtained **iteratively** ($U_{i,\,j}^{(n)} \rightarrow U_{i,\,j}^{(n+1)}$) for the points on the grid using the values from previous iteration for the neighbouring points.

Error: $$dU_{i,\,j}^{(n+1)} = (U_{i-1,\,j}^{(n)} + U_{i+1,\,j}^{(n)} + U_{i,\,j-1}^{(n)} + U_{i,\,j+1}^{(n)})/4 - U_{i,\,j}^{(n)} = U_{i,\,j}^{(n+1)} - U_{i,\,j}^{(n)}$$

Jacobi Solver



```
for (it=1;it<itmax;it++){
  for (j=1;j<M;j++){
    for(i=1;i<M;i++){
      u[i,j]=0.25*(up[i+1,j]+up[i-1,j]+up[i,j+1]+up[i,j-1]);
   }
  }
 up=u;
}
```

Jacobi solver (serial)

To find $U(x,\ y)$ which satisfies Laplace Equation (two dimensions) in unit square, $0 < x < 1, 0 < y < 1$,

$$\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = 0, \text{ with boundary conditions,}$$

$$\begin{aligned} U(x,\ 0) &= 0,\ U(x,\ 1) = \sin(2\pi x), \\ U(0,\ y) &= 0,\ U(1,\ y) = -\sin(\pi y). \end{aligned}$$

Analytical solution $U(x,\ y)$:

$$U(x,\ y) = \frac{\sinh(2\pi y)\sin(2\pi x)}{\sinh(2\pi)} - \frac{\sinh(\pi x)\sin(\pi y)}{\sinh(\pi)}$$

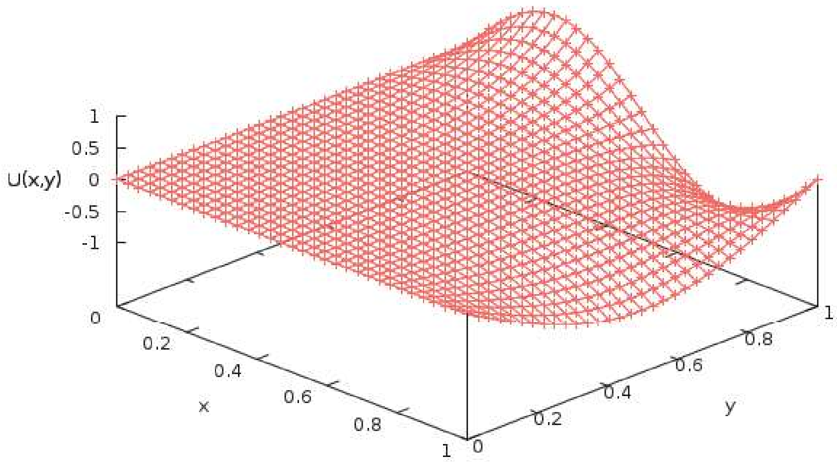
Sample codes:

1. level 0: initialize, data layout, identifying the upper and right boundaries.
2. level 1: intialize boundary condition, data for $U(i,\ j)$ with boundaries.
3. level 2: evolution of $U(i,\ j)$ with iterations, interchange read and write grid after every iteration.

.

Source: from some lecture notes in the web (unknown)

Jacobi solver (serial): code segments




```
// parameters
#define NR 31
#define NC 31
#define itmax 120

// READ, WRITE variables
double *u1;
double *u2;

// temporary variable
double *tmp;

//TWO grids: READ(right) and WRITE(left)
u1=(double *)malloc((NR+2)*(NC+2)*sizeof(double));
u2=(double *)malloc((NR+2)*(NC+2)*sizeof(double));

//Difference between succesive iterations (error)
du=(double *)malloc((NR+2)*(NC+2)*sizeof(double));

//Discretization
dx=1.0/(double)(NC+1);
dy=1.0/(double)(NR+1);
```

```
//Jacobi iteration
for(iter=1;iter<=itmax;iter++){

   //Jacobi kernel
    for(i=1;i<=NR;i++){
        b=i*(NC+2)+1;
        for(j=1;j<=NC;j++){
            u2[b]=0.25*(u1[b-1]+u1[b+1]+u1[b-(NC+2)]+u1[b+(NC+2)]);
            du[b]=u2[b]-u1[b];
            b++;
  }
     }

    // Interchange READ & WRITE => u1<=>u2
         tmp=u1;
         u1=u2;
         u2=tmp;
}
```

Jacobi solver (serial): output

```
OUTPUT 0:

layout
 0        1        2        3        4        5
 6        7        8        9        10       11
 12       13       14       15       16       17
 18       19       20       21       22       23
 24       25       26       27       28       29
 30       31       32       33       34       35


30       0.000000
31       0.200000
32       0.400000
33       0.600000
34       0.800000
35       1.000000


5        0.000000
11       0.200000
17       0.400000
23       0.600000
29       0.800000
35       1.000000
```

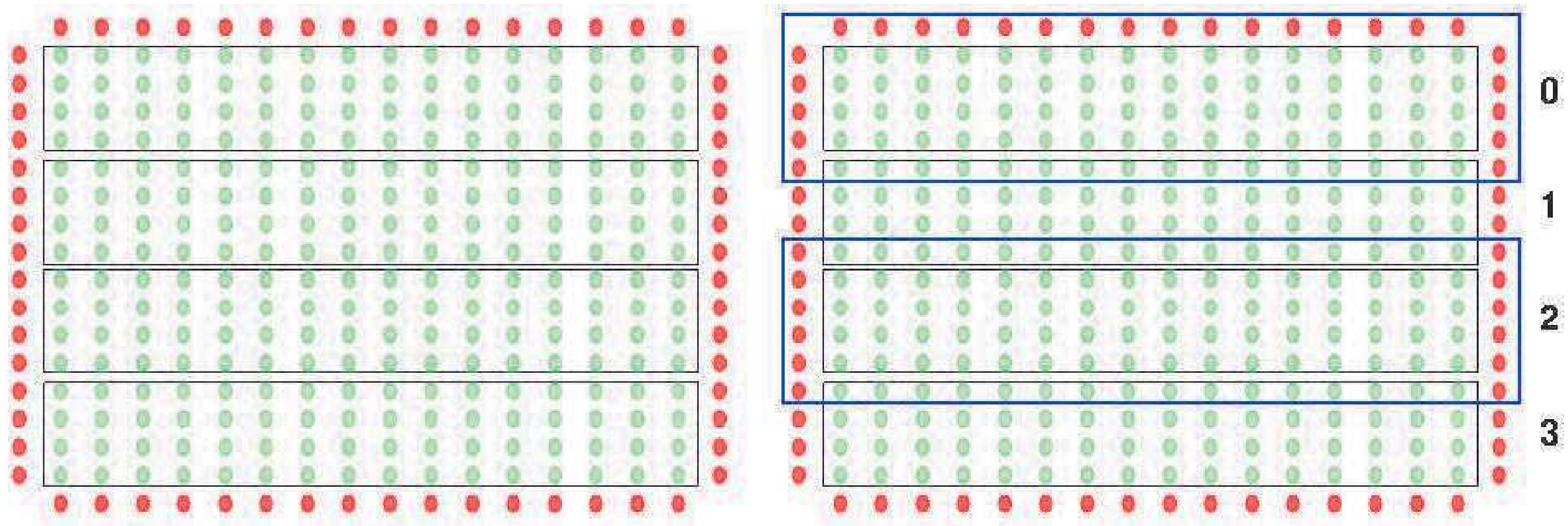
```
OUTPUT 1:

layout
 0        1        2        3        4        5
 6        7        8        9        10       11
 12       13       14       15       16       17
 18       19       20       21       22       23
 24       25       26       27       28       29
 30       31       32       33       34       35


30       0.000000        0.000000
31       0.200000        0.950859
32       0.400000        0.588816
33       0.600000        -0.586238
34       0.800000        -0.951841
35       1.000000        -0.003185


5        0.000000        -0.000000
11       0.200000        -0.587528
17       0.400000        -0.950859
23       0.600000        -0.951351
29       0.800000        -0.588816
35       1.000000        -0.001593

u1 boundary initialized
0.000000        0.000000        0.000000        0.000000        0.000000        -0.000000
0.000000        0.000000        0.000000        0.000000        0.000000        -0.587528
0.000000        0.000000        0.000000        0.000000        0.000000        -0.950859
0.000000        0.000000        0.000000        0.000000        0.000000        -0.951351
0.000000        0.000000        0.000000        0.000000        0.000000        -0.588816
0.000000        0.950859        0.588816        -0.586238       -0.951841       -0.001593
```

Jacobi solver (serial): output

```
OUTPUT 2:

layout
 0       1       2       3       4       5
 6       7       8       9       10      11
 12      13      14      15      16      17
 18      19      20      21      22      23
 24      25      26      27      28      29
 30      31      32      33      34      35

u1
0.000000        0.000000        0.000000        0.000000        0.000000        0.000000
0.000000        0.000000        0.000000        0.000000        0.000000        0.000000
0.000000        0.000000        0.000000        0.000000        0.000000        0.000000
0.000000        0.000000        0.000000        0.000000        0.000000        0.000000
0.000000        0.000000        0.000000        0.000000        0.000000        0.000000
0.000000        0.000000        0.000000        0.000000        0.000000        0.000000

u2
0.000000        0.000000        0.000000        0.000000        0.000000        0.000000
0.000000        0.000000        0.000000        0.000000        0.000000        0.000000
0.000000        0.000000        0.000000        0.000000        0.000000        0.000000
0.000000        0.000000        0.000000        0.000000        0.000000        0.000000
0.000000        0.000000        0.000000        0.000000        0.000000        0.000000
0.000000        0.000000        0.000000        0.000000        0.000000        0.000000

30      0.000000        0.000000
31      0.200000        0.950859
32      0.400000        0.588816
33      0.600000        -0.586238
34      0.800000        -0.951841
35      1.000000        -0.003185


5       0.000000        -0.000000
11      0.200000        -0.587528
17      0.400000        -0.950859
23      0.600000        -0.951351
29      0.800000        -0.588816
35      1.000000        -0.001593

u1 boundary initialized
0.000000        0.000000        0.000000        0.000000        0.000000        -0.000000
0.000000        0.000000        0.000000        0.000000        0.000000        -0.587528
0.000000        0.000000        0.000000        0.000000        0.000000        -0.950859
0.000000        0.000000        0.000000        0.000000        0.000000        -0.951351
0.000000        0.000000        0.000000        0.000000        0.000000        -0.588816
0.000000        0.950859        0.588816        -0.586238       -0.951841       -0.001593

#u2 modified

#u2 modified:iteration 1
0.000000        0.000000        0.000000        0.000000        0.000000        -0.000000
0.000000        0.000000        0.000000        0.000000        -0.146882       -0.587528
0.000000        0.000000        0.000000        0.000000        -0.237715       -0.950859
0.000000        0.000000        0.000000        0.000000        -0.237838       -0.951351
0.000000        0.237715        0.147204        -0.146559       -0.385164       -0.588816
0.000000        0.950859        0.588816        -0.586238       -0.951841       -0.001593

#u2 modified:iteration 2
0.000000        0.000000        0.000000        0.000000        0.000000        -0.000000
0.000000        0.000000        0.000000        -0.036720       -0.206311       -0.587528
0.000000        0.000000        0.000000        -0.059429       -0.333895       -0.950859
0.000000        0.059429        0.036801        -0.096099       -0.393558       -0.951351
0.000000        0.274516        0.169993        -0.206050       -0.481263       -0.588816
0.000000        0.950859        0.588816        -0.586238       -0.951841       -0.001593

#u2 modified:iteration 3
0.000000        0.000000        0.000000        0.000000        0.000000        -0.000000
0.000000        0.000000        -0.009180       -0.066435       -0.239536       -0.587528
0.000000        0.014857        -0.005657       -0.116679       -0.402539       -0.950859
0.000000        0.077829        0.033331        -0.155559       -0.465652       -0.951351
0.000000        0.295070        0.173521        -0.248402       -0.535066       -0.588816
0.000000        0.950859        0.588816        -0.586238       -0.951841       -0.001593

#u2 modified:iteration 4
0.000000        0.000000        0.000000        0.000000        0.000000        -0.000000
0.000000        0.001419        -0.018023       -0.091349       -0.264125       -0.587528
0.000000        0.018043        -0.019418       -0.157547       -0.443182       -0.950859
0.000000        0.085814        0.022534        -0.199351       -0.511129       -0.951351
0.000000        0.300552        0.167204        -0.275835       -0.563678       -0.588816
0.000000        0.950859        0.588816        -0.586238       -0.951841       -0.001593

#u2 modified:iteration 5
0.000000        0.000000        0.000000        0.000000        0.000000        -0.000000
0.000000        0.000005        -0.027337       -0.109924       -0.280514       -0.587528
0.000000        0.016954        -0.033748       -0.188325       -0.470915       -0.950859
0.000000        0.085282        0.008562        -0.230495       -0.539390       -0.951351
0.000000        0.300969        0.159016        -0.295516       -0.581905       -0.588816
0.000000        0.950859        0.588816        -0.586238       -0.951841       -0.001593

#u2 modified:iteration 6
0.000000        0.000000        0.000000        0.000000        0.000000        -0.000000
0.000000        -0.002596       -0.035917       -0.124044       -0.292092       -0.587528
0.000000        0.012885        -0.047536       -0.211271       -0.489772       -0.950859
0.000000        0.081621        -0.004986       -0.253667       -0.558667       -0.951351
0.000000        0.298790        0.150708        -0.309905       -0.593891       -0.588816
0.000000        0.950859        0.588816        -0.586238       -0.951841       -0.001593

#u2 modified:iteration 7
0.000000        0.000000        0.000000        0.000000        0.000000        -0.000000
0.000000        -0.005758       -0.043544       -0.134820       -0.300336       -0.587528
0.000000        0.007872        -0.059822       -0.228755       -0.503222       -0.950859
0.000000        0.076672        -0.017218       -0.271207       -0.572170       -0.951351
0.000000        0.295797        0.143178        -0.320772       -0.602307       -0.588816
0.000000        0.950859        0.588816        -0.586238       -0.951841       -0.001593

#u2 modified:iteration 8
0.000000        0.000000        0.000000        0.000000        0.000000        -0.000000
0.000000        -0.008918       -0.050100       -0.143159       -0.306392       -0.587528
0.000000        0.002773        -0.070411       -0.242268       -0.513030       -0.950859
0.000000        0.071613        -0.027795       -0.284729       -0.582022       -0.951351
0.000000        0.292677        0.136656        -0.329143       -0.608400       -0.588816
0.000000        0.950859        0.588816        -0.586238       -0.951841       -0.001593
```

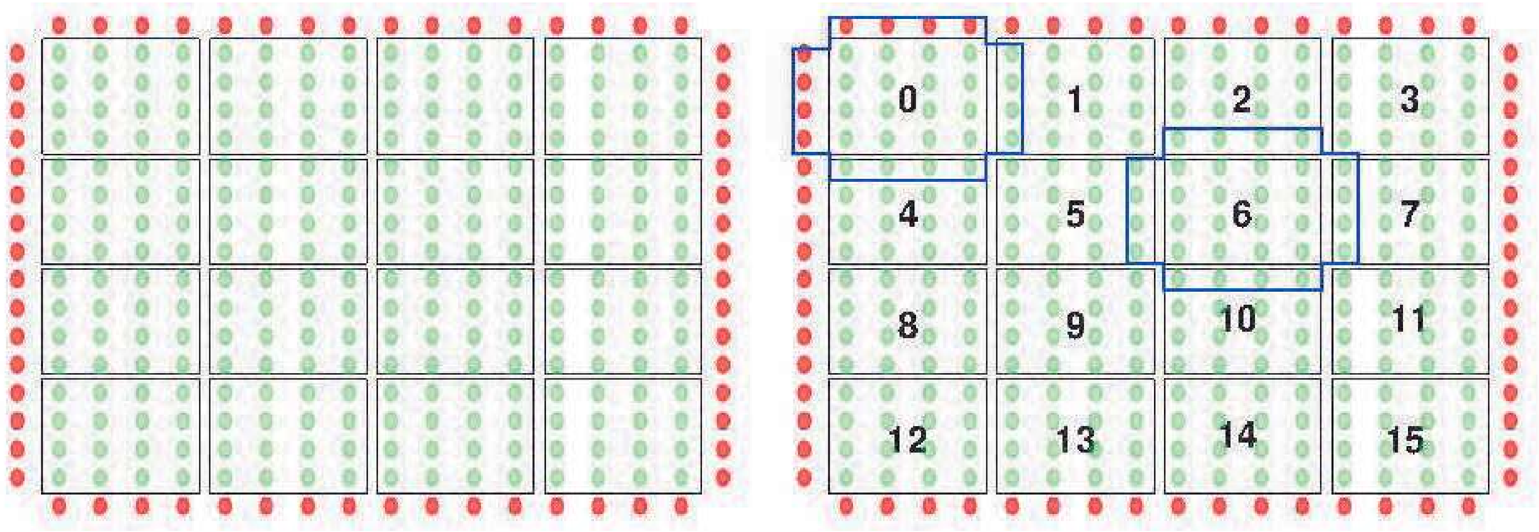
Parallel Jacobi solver

Jacobi Solver (MPI: 1D decomposition)



- Geometric decomposition of the domain (across p processes) 'p' strips.
- Working data for each process includes: **its domain** + **neighbour rows/columns** around the domain (ghost regions).
- For computing $(M-1)^2/p$ points, require $2(M-1)$ points (along i direction) +$2(M-1)/p$ points (along j direction) known as **halo** region.
- Could be boundary points (fixed)/on the neighbour process (changes with every iteration).

Jacobi solver (MPI: 2D decomposition)



- Geometric decomposition of the domain (across p processes) 'p' tiles.
- Working data for each process: **its domain** + **neighbour rows/columns** (halo regions).
- For computing $(M-1)^2/p$ points requires $4(M-1)/\sqrt{p}$ points around the subdomain (known as **halo**), with $(M-1)/\sqrt{p}$ pt.s (different neighbour processes).
- Could be boundary points (fixed)/on the neighbour process (changes every iteration).
- Mapped the processes to different subdomains on cartesian grid, given a process (rank), find its coordinates, neighbours as well as enable transfer of data (**EIGHT** exchanges).

$1D$ decomposition: data exchange

	*	*	*	*	*	*	*	*	*	*	*	*	
*	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	*
*	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	*
*	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	*
	1	1	1	1	1	1	1	1	1	1	1	1	

	0	0	0	0	0	0	0	0	0	0	0	0	
*	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	*
*	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	*
*	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	*
	2	2	2	2	2	2	2	2	2	2	2	2	

	1	1	1	1	1	1	1	1	1	1	1	1	
*	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	*
*	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	*
*	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	*
	3	3	3	3	3	3	3	3	3	3	3	3	

	2	2	2	2	2	2	2	2	2	2	2	2	
*	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	*
*	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	*
*	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	*
	*	*	*	*	*	*	*	*	*	*	*	*	

12×12 domain $\Rightarrow$ **4** [3×12 sub-domains, 5×14 domain with boundaries (ghost regions)]

2D decomposition: data exchange

	*	*	*	
*	**0**	**0**	**0**	1
*	**0**	**0**	**0**	1
*	**0**	**0**	**0**	1
	4	4	4	

	*	*	*	
0	**1**	**1**	**1**	2
0	**1**	**1**	**1**	2
0	**1**	**1**	**1**	2
	5	5	5	

	*	*	*	
1	**2**	**2**	**2**	3
1	**2**	**2**	**2**	3
1	**2**	**2**	**2**	3
	6	6	6	

	*	*	*	
2	**3**	**3**	**3**	*
2	**3**	**3**	**3**	*
2	**3**	**3**	**3**	*
	7	7	7	

	0	0	0	
*	**4**	**4**	**4**	5
*	**4**	**4**	**4**	5
*	**4**	**4**	**4**	5
	8	8	8	

	1	1	1	
4	**5**	**5**	**5**	6
4	**5**	**5**	**5**	6
4	**5**	**5**	**5**	6
	9	9	9	

	2	2	2	
5	**6**	**6**	**6**	7
5	**6**	**6**	**6**	7
5	**6**	**6**	**6**	7
	10	10	10	

	3	3	3	
6	**7**	**7**	**7**	*
6	**7**	**7**	**7**	*
6	**7**	**7**	**7**	*
	11	11	11	

	4	4	4	
*	**8**	**8**	**8**	9
*	**8**	**8**	**8**	9
*	**8**	**8**	**8**	9
	12	12	12	

	5	5	5	
8	**9**	**9**	**9**	10
8	**9**	**9**	**9**	10
8	**9**	**9**	**9**	10
	13	13	13	

	6	6	6	
9	**10**	**10**	**10**	11
9	**10**	**10**	**10**	11
9	**10**	**10**	**10**	11
	14	14	14	

	7	7	7	
10	**11**	**11**	**11**	*
10	**11**	**11**	**11**	*
10	**11**	**11**	**11**	*
	15	15	15	

	8	8	8	
*	**12**	**12**	**12**	13
*	**12**	**12**	**12**	13
*	**12**	**12**	**12**	13
	*	*	*	

	9	9	9	
12	**13**	**13**	**13**	14
12	**13**	**13**	**13**	14
12	**13**	**13**	**13**	14
	*	*	*	

	10	10	10	
13	**14**	**14**	**14**	15
13	**14**	**14**	**14**	15
13	**14**	**14**	**14**	15
	*	*	*	

	11	11	11	
14	**15**	**15**	**15**	*
14	**15**	**15**	**15**	*
14	**15**	**15**	**15**	*
	*	*	*	

12×12 domain $\Rightarrow$ **16** [3×3 sub-domains, 5×5 domain with boundaries (ghost regions)]