

Exercise 1

Implementation of tasks 1-6 you can find in file task1.c

to execute program you need to type next commands in terminal

mpicc task1.c -o task1 - compile

mpirun -n 1 ./task1 - execute

Result

Task 7

The complexity of the sequential matrix-vector multiplication, in the general case is $m \cdot n$ The complexity of the sequential matrix-vector multiplication, when $m = n$ is n^2

```
set@set: /mnt/C4D630CFD630C388/Studing/Lux/Parallel and Grid Computin/Matrix-Vect
or Multiplication Part 1$ mpirun -n 1 ./task1
Type n:
6
Type m:
7
Type i:
3
Read file base_i_vector_n.dat
0.00 0.00 0.00 1.00 0.00 0.00
Read file random_vector_n.dat
75.00 68.00 44.00 15.00 225.00 170.00
Read file id_matrix_m_n.dat
1.00 0.00 0.00 0.00 0.00 0.00
0.00 1.00 0.00 0.00 0.00 0.00
0.00 0.00 1.00 0.00 0.00 0.00
0.00 0.00 0.00 1.00 0.00 0.00
0.00 0.00 0.00 0.00 1.00 0.00
0.00 0.00 0.00 0.00 0.00 1.00
0.00 0.00 0.00 0.00 0.00 0.00

Read file random_matrix_m_n.dat
107.00 112.00 123.00 170.00 88.00 158.00
30.00 83.00 39.00 77.00 245.00 173.00
165.00 205.00 124.00 59.00 35.00 131.00
184.00 61.00 192.00 76.00 92.00 114.00
45.00 40.00 182.00 89.00 55.00 24.00
132.00 35.00 8.00 0.00 205.00 96.00
30.00 235.00 180.00 197.00 57.00 42.00
Seq matr and vector
70263.00 95300.00 62801.00 67616.00 31893.00 75077.00 49070.00 set@set: /mnt/C4D6
or Multiplication Part 1$
```

Exercise 2

Task1

Task2

To find number of elements of the last processor we need to subtract from k the number of elements of other processors(0..p-2). The number of elements of processor we can find from next formula $\lfloor (i+1) * \frac{k}{p} \rfloor - \lfloor i * \frac{k}{p} \rfloor$

$$k - (\lfloor (0+1) * \frac{k}{p} \rfloor - \lfloor 0 * \frac{k}{p} \rfloor) - (\lfloor (1+1) * \frac{k}{p} \rfloor - \lfloor 1 * \frac{k}{p} \rfloor) - (\lfloor (2+1) * \frac{k}{p} \rfloor - \lfloor 2 * \frac{k}{p} \rfloor) - \dots - (\lfloor (p-m+1) * \frac{k}{p} \rfloor - \lfloor (p-m) * \frac{k}{p} \rfloor) - \dots - (\lfloor (p-2+1) * \frac{k}{p} \rfloor - \lfloor (p-2) * \frac{k}{p} \rfloor)$$

We can see that formula can be simplified to

$$k - \lfloor (p-2+1) * \frac{k}{p} \rfloor = k - \lfloor (p-1) * \frac{k}{p} \rfloor = \text{We need to take upper round value because the are no more processors to process remaining elements} \Rightarrow \lceil k - (p-1) * \frac{k}{p} \rceil = \lceil k - k + \frac{k}{p} \rceil = \lceil \frac{k}{p} \rceil$$

Task3

Implementation of task you can find in file task1.c

```
void create_mixed_count_disp_arrays(int p, size_t k, int **count, int **disp) {
    (*count)[0] = BLOCK_SIZE(0, p, k);
    (*disp)[0] = 0;

    for (int i = 1; i < p; ++i) {
        (*disp)[i] = (*disp)[i-1] + (*count)[i-1];
        (*count)[i] = BLOCK_SIZE(i, p, k);
    }
}
```

Taks4

Implementation of task you can find in file task1.c

```
void create_count_disp_arrays(int id, int p, size_t k, int **count, int **disp) {
    int size = BLOCK_SIZE(id, p, k);
    (*disp)[i] = 0;
    (*count)[i] = size;
    for (int i = 1; i < p; ++i) {
        (*disp)[i] = (*disp)[i-1] + (*count)[i-1];
        (*count)[i] = size;
    }
}
```

Exercise 3

The second approach allows to decrease number of communication between processors. This allows to increase speed of calculation (The bottlenecks in parallel algorithms are often connected with communication delay between processors)

Also two parallel algorithms of matrix multiplication required full vector and row of matrix to find complete element of resulting matrix (Rowwise block-striped decomposition and Columnwise block-striped)

