

Board

1 Board

This exercise is about parallelizing a code that executes a sequence of operations on the cells of a board of size $(n + 2) \times (n + 2)$. Given a cell $c_{i,j}$, $1 \leq i, j \leq n$, three different operators can be used:

update \odot : performs one of the following operations that use, respectively, the cell on the left, right, up or down $c_{i,j}$

$$\text{left } \odot : c_{i,j} = c_{i,j} \odot c_{i,j-1}$$

$$\text{right } \odot : c_{i,j} = c_{i,j} \odot c_{i,j+1}$$

$$\text{up } \odot : c_{i,j} = c_{i,j} \odot c_{i+1,j}$$

$$\text{down } \odot : c_{i,j} = c_{i,j} \odot c_{i-1,j}$$

gather \oplus : performs the operation $c_{i,j} = c_{i,j} \oplus c_{i-1,j} \oplus c_{i+1,j} \oplus c_{i,j-1} \oplus c_{i,j+1}$

scatter \otimes : performs these four operations at once $c_{i-1,j} = c_{i-1,j} \otimes c_{i,j}$; $c_{i+1,j} = c_{i+1,j} \otimes c_{i,j}$; $c_{i,j-1} = c_{i,j-1} \otimes c_{i,j}$; $c_{i,j+1} = c_{i,j+1} \otimes c_{i,j}$

None of the above operators is commutative nor associative.

The list of operations to perform is contained in the **operations** array; each element of this array is of type **operation_t** whose members are:

- **i, j**: the coordinates of the cell
- **optype**: the type of operation (i.e., left, right, up, down, gather, scatter)

The provided sequential code trivially consists of a main loop where, at each iteration, an operation is taken from the list and then the corresponding operator is applied on the corresponding cell (and the concerned neighbors). The content of the board is also printed at the beginning, after the sequential code and after the parallel code you are required to implement as instructed below.

2 Package content

In the **board** directory you will find the following files:


- `main.c`: This file contains the code that implements the process described above. This reads from command line the size of the board and the number of operations in the sequence it generates a random sequence of operations and then call the sequential and parallel routine to execute the operations in the list. At the beginning, the sequential and parallel routines are identical and implement the algorithm described above. **Only this file must be modified for this exercise.**
- `aux.c`, `aux.h`: these two files contain auxiliary routines and declarations and **must not be modified.**

The code can be compiled with the `make` command: just type `make` inside the `board` directory; this will generate a `main` program that can be run like this:

```
$ ./main n nops
```

where `n` is the size of the board (which thus contains $(n+2) \times (n+2)$ cells) and `nops` the number of operations to perform. `n` must be smaller than or equal to 10.

3 Assignment

-  The objective of this exercise is to parallelize the code of the `main` program in order to reduce its execution time. Make sure that the content of the board is the same after the sequential and the parallel execution. To make sure everything works as expected, make sure you increase the number of operations to at least a few tens.