AGH University of Science and Technology

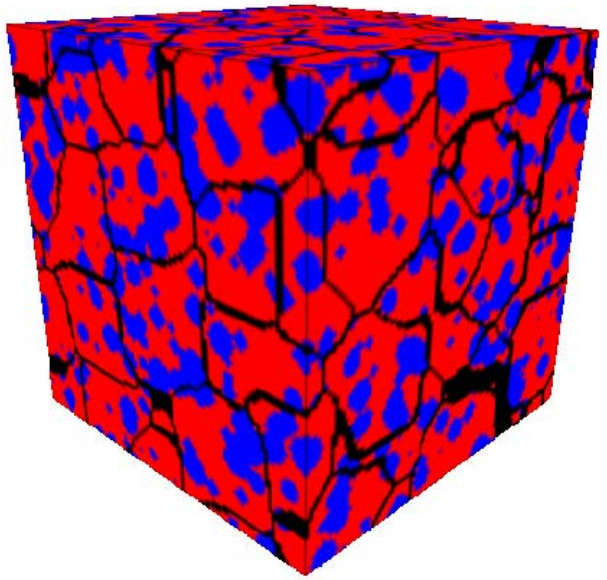
Study of the parallelization possibility of a Monte Carlo grain growth algorithm.

Mateusz Sitko msitko@agh.edu.pl
Łukasz Madej lmadej@agh.edu.pl

Zakopane 01.03.13

Outline

1. Motivation
2. Monte Carlo model for grain growth simulation
3. Parallelization idea
4. Results
5. Conclusions and plans for future work



Thermal operations

Mechanical operations

Thermal operations

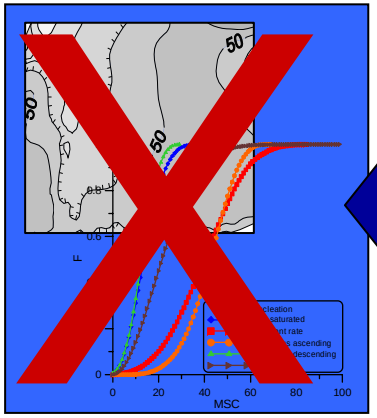
Mechanical operations

1. Texture development
2. Phase transformations
3. Recrystalization processes
   - Dynamic (DRX),
   - Static (SRX),
   - Metadynamic (MDRX).

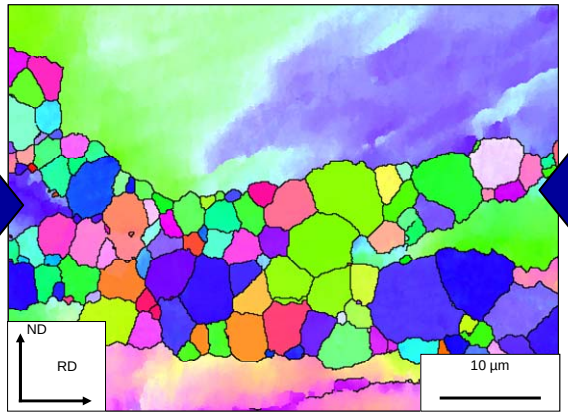
Final exploitation properties

Numerical modelling of static recrystalization (SRX)

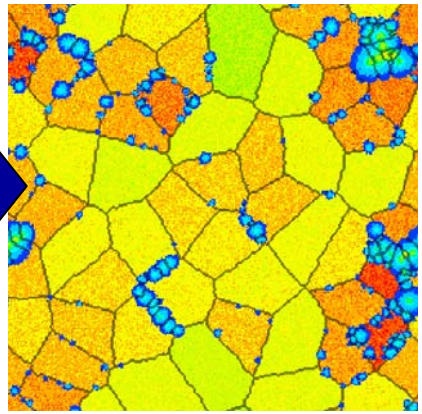
Conventional approaches



Real material



Discrete Models e.g. CA, MC



The main aim of this work is development of MC grain growth algorithm for farther SRX simulation.

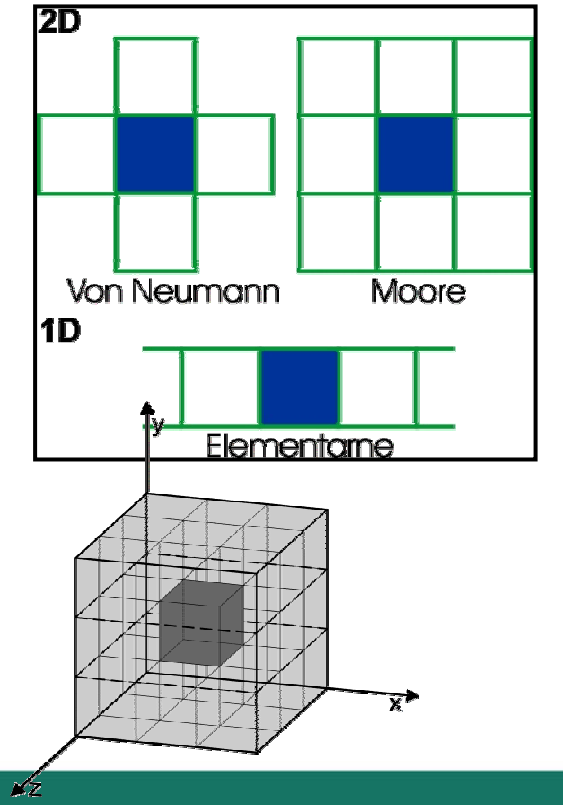
MC method

The main idea of the Monte Carlo technique is to divide a specific part of the material into one-, two-, or three-dimensional lattices of cells, where cells have clearly defined interaction rules between each other. MC simulations have probabilistic character and it is based on minimization of system energy.

- MC space - finite set of cells, where each cell is described by a set of internal variables describing the state of a cell.

- Neighborhood – describes the closest neighbors of a particular cell. It can be in 1D, 2D and 3D space.

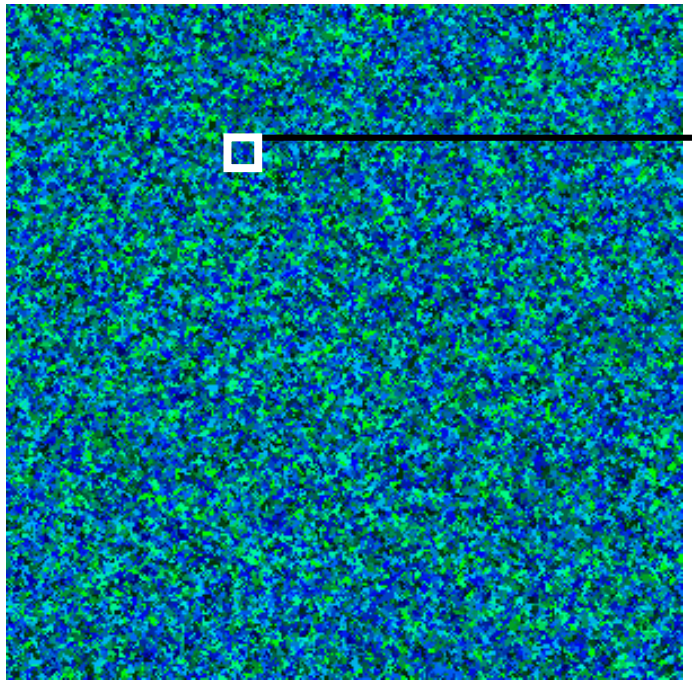
- Energy minimization- E, the energy value of each cell in the lattice is determined by the states of its neighbors and the cell itself.

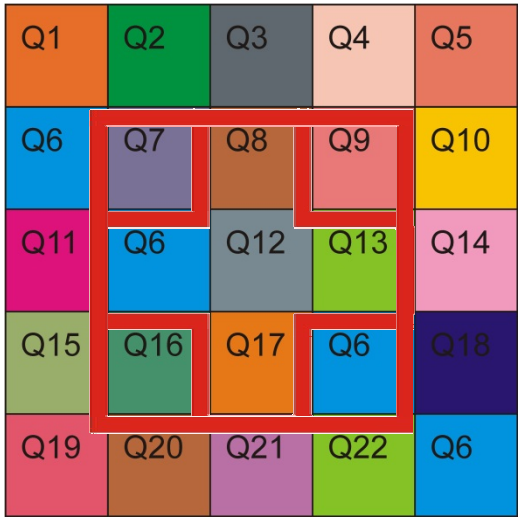


MC method assumptions

n = 50

State number

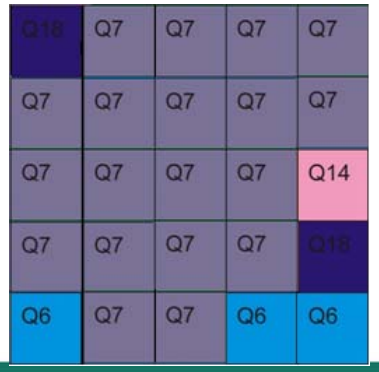




1 MCS

Cells in the same state represent particular grain

$$\Omega = \{Q_0, \ldots, Q_{n-1}\}$$



Algorithm steps:

Q1	Q1	Q2
Q3	Q3	Q2
Q3	Q2	Q2

Step 1: Random selection of element witch specifically orientation.

Step 2: Calculate energy of lattice site surrounding concerned element Q_i.
Energy is calculated using fallowing formula:

$$E = J_{gb} \sum_{<i,j>} \left(1 - \delta_{S_i S_j}\right)$$

Kronecker delta

Grain boundary energy

Surrounding neighbors points

Q1	Q1	Q2
Q3	Q4	Q2
Q3	Q2	Q2

Step 3: The investigated cell changes the state to one of the available states/orientation.
The state/orientation is randomly generated from Ω available states/orientations.

Step 4: Calculate the change in energy Q_i caused by orientation changes

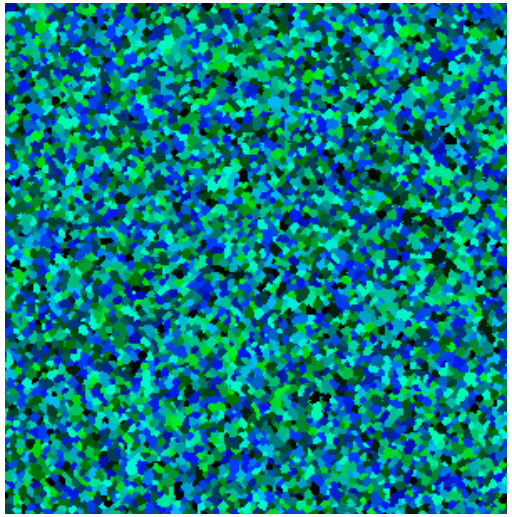
Step 5: The orientation change is accepted with the probability *p*:

$$p(\Delta E) = \begin{cases} 1 & \Delta E \le 0 \\ 0 & \Delta E > 0 \end{cases}$$

Q1	Q1	Q2
Q3	Q3	Q2
Q3	Q2	Q2

Q1	Q1	Q2
Q3	Q4	Q2
Q3	Q2	Q2

$n_Q = 50$

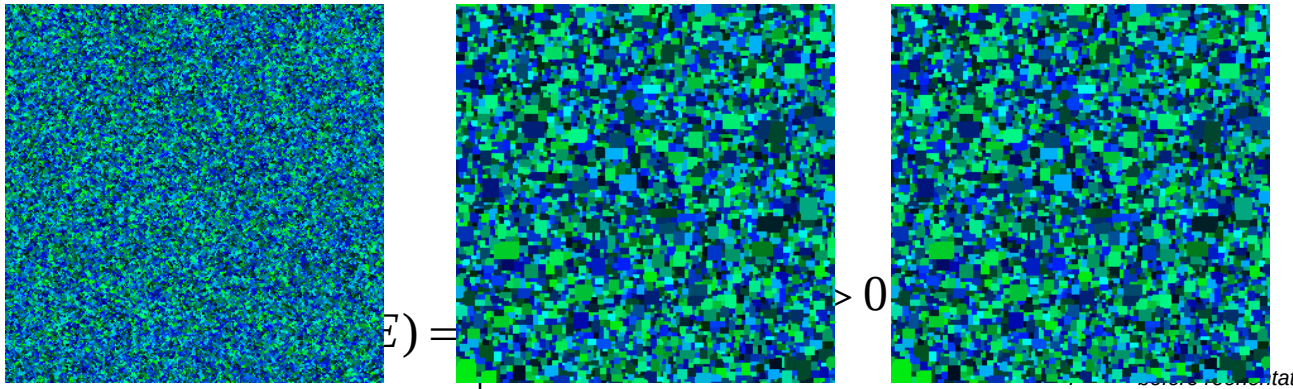


Moore's neighborhood

MC method - algorithm

1 MCS 200 MCS 400 MCS

Q = 50



$E) =$ > 0

ΔE - $E_{before\ reorientation}$ - $E_{after\ reorientation}$,

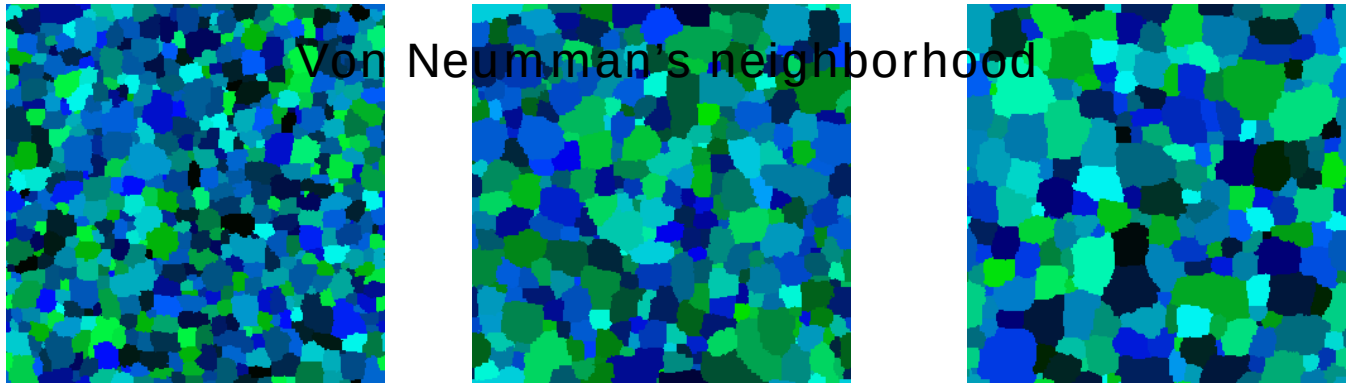
$\begin{cases} 1 & \Delta E \le 0 \end{cases}$ k_b - Boltzmann constant, T - temperature.

pinning effect

200 MCS 400 MCS 600 MCS

Q =50

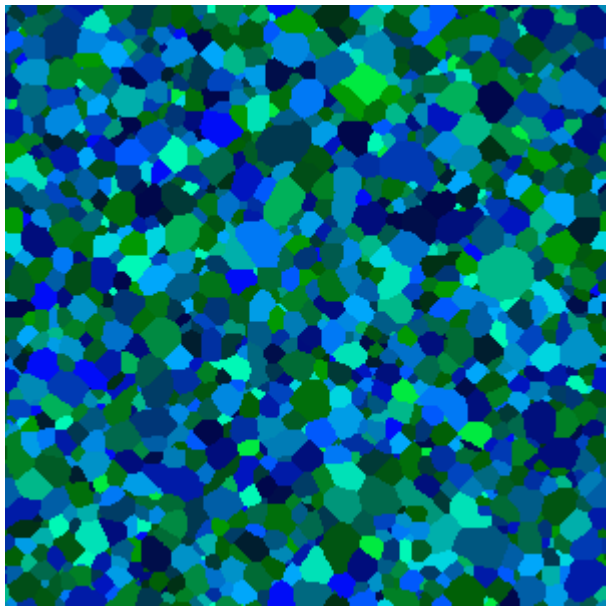
kbT=0.6



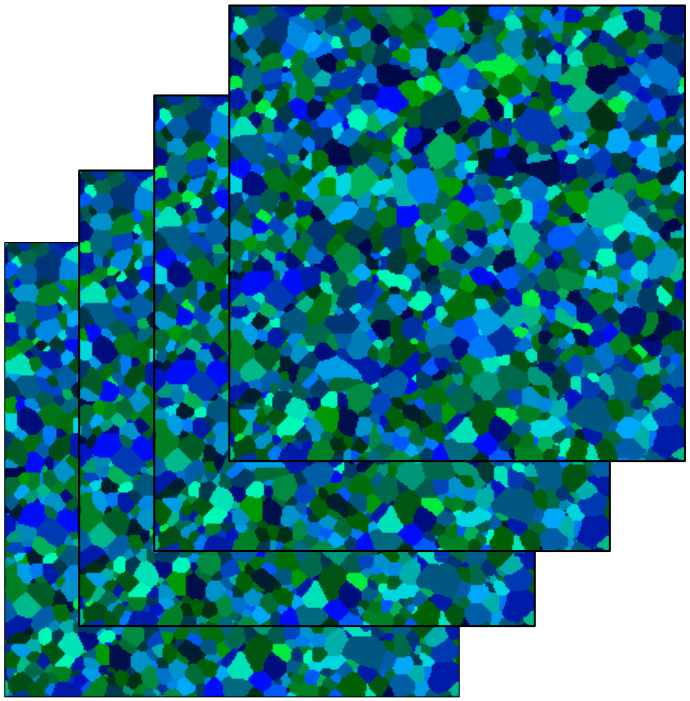
Von Neumman's neighborhood

How can we improve computing time?

2 D – very time consuming



3 D – even worse



2 options to solve this issue

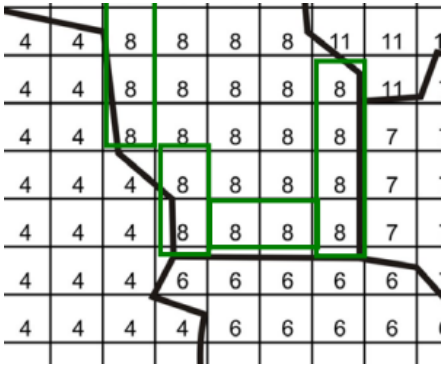
modification of the algorithm

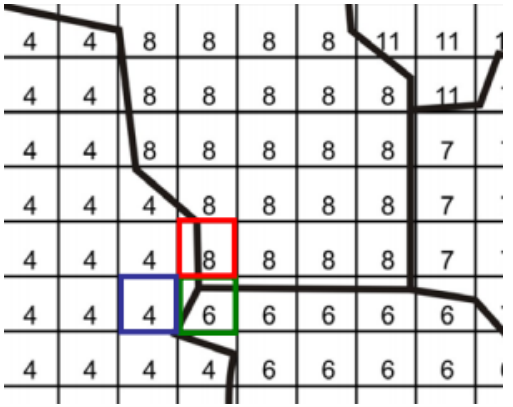
parallelization

Modification the Algorithm

Reduction of computing time :

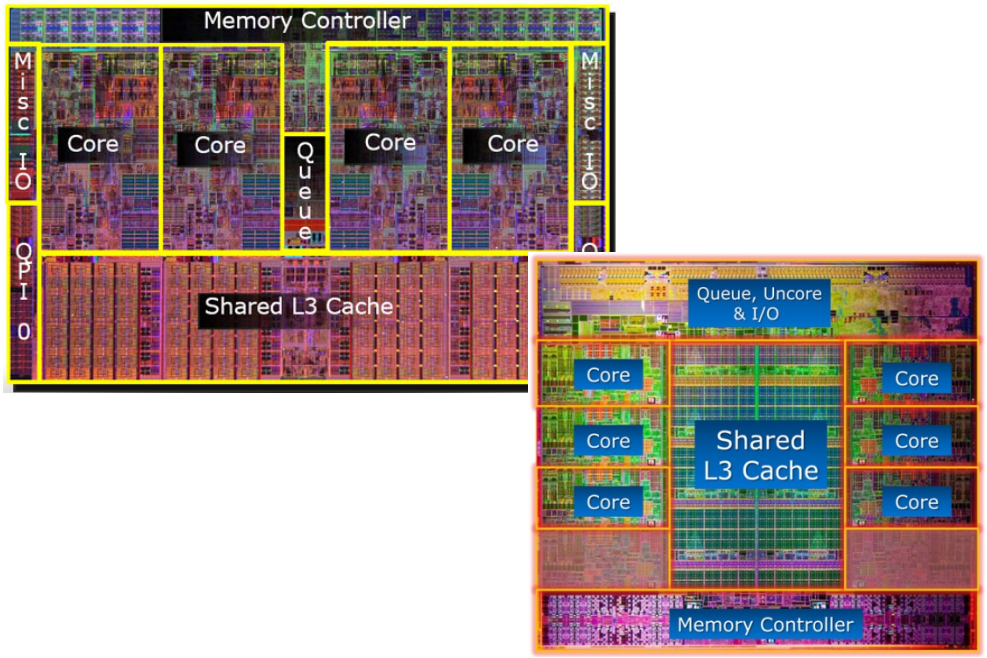
- consider only cells on grains boundary
- selection of a new orientation only from neighbors
- cell is randomly generated from (*W-k*) available cells, where k is a number of cells already considerated in currently MC step.





Parallelization possibility

Multi threads application
single machine - openMP

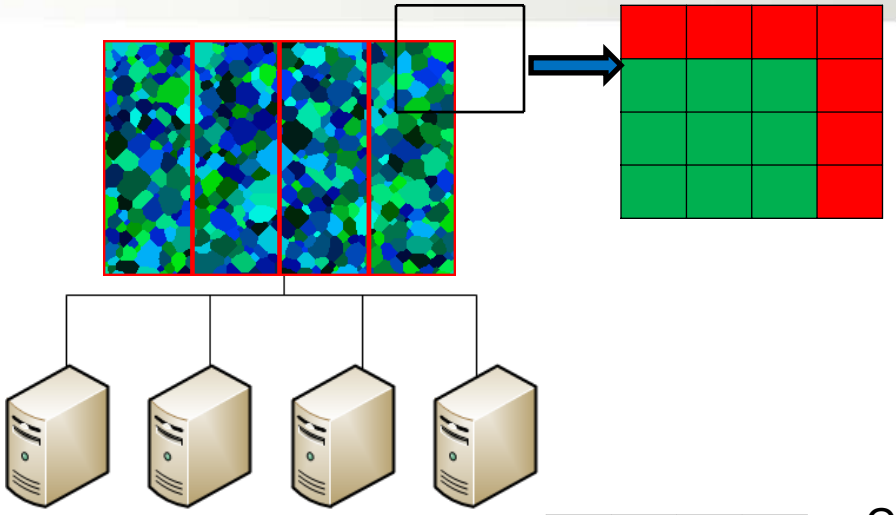


Multi node system - MPI

Parallelization idea – division work between threads (periodic boundary condition)

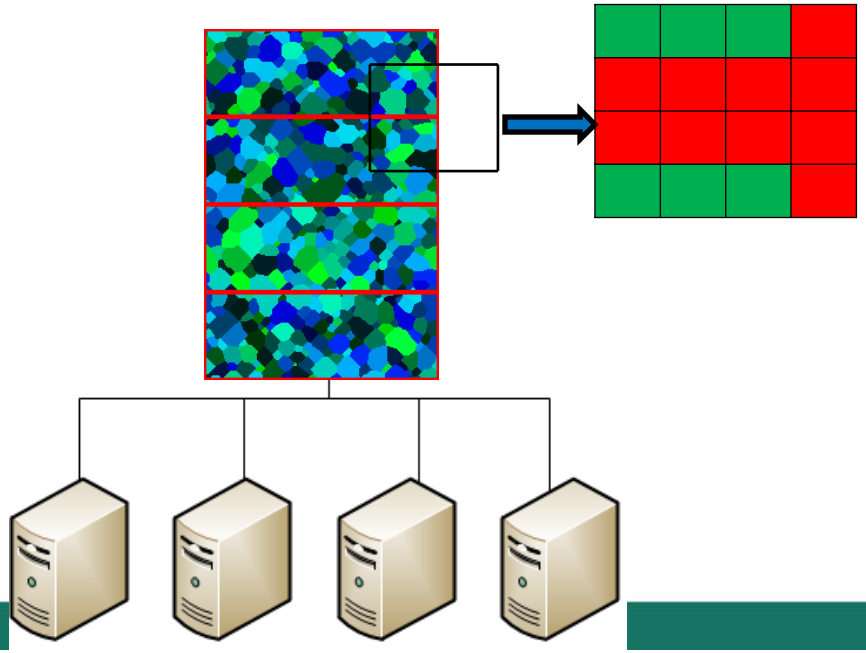
Case 1:

If(x>y)



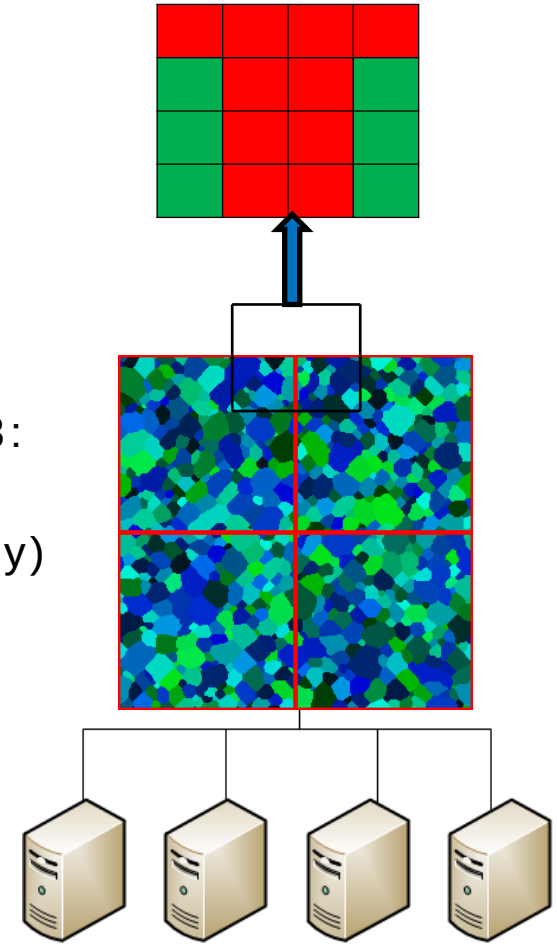
Case 2:

If(y>x)



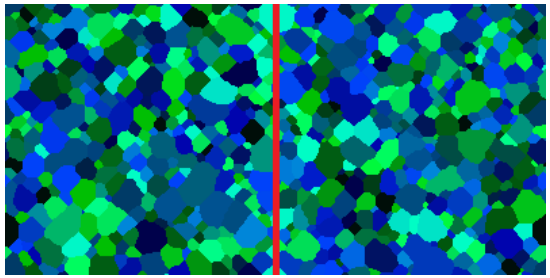
Case 3:

If(x==y)

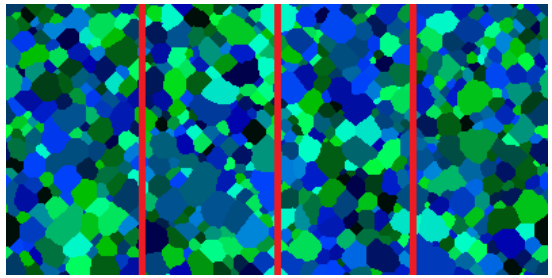


Parallelization idea – non periodic boundary condition

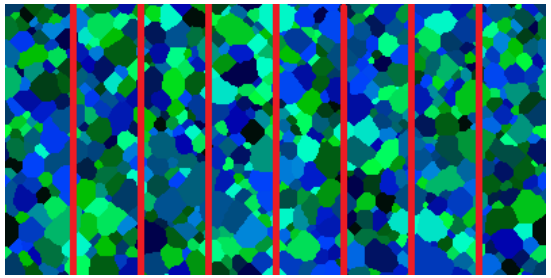
Case 1:



2 threads



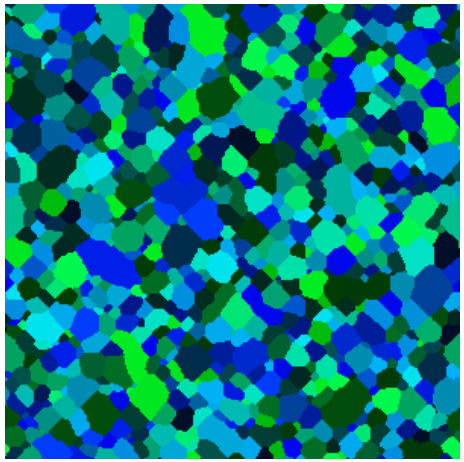
4 threads



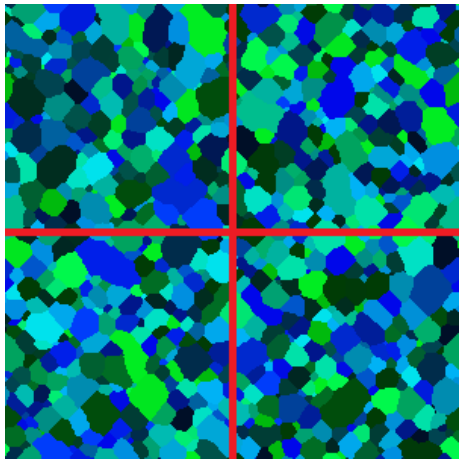
8 threads

n threads

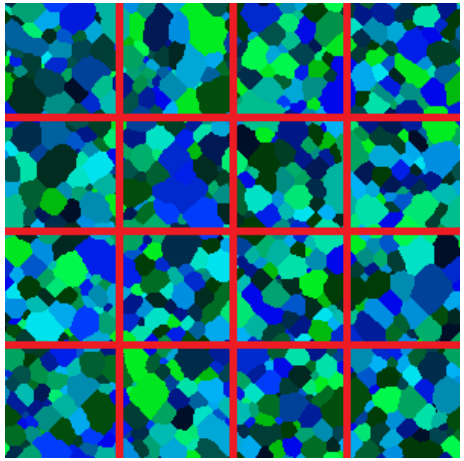
Case 3:



1 thread

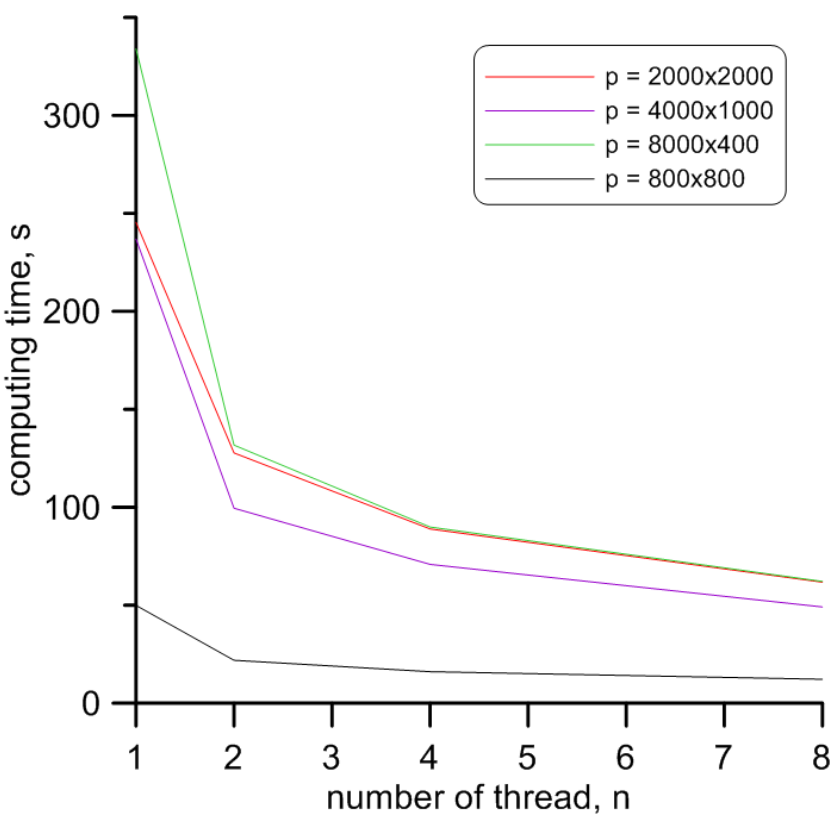


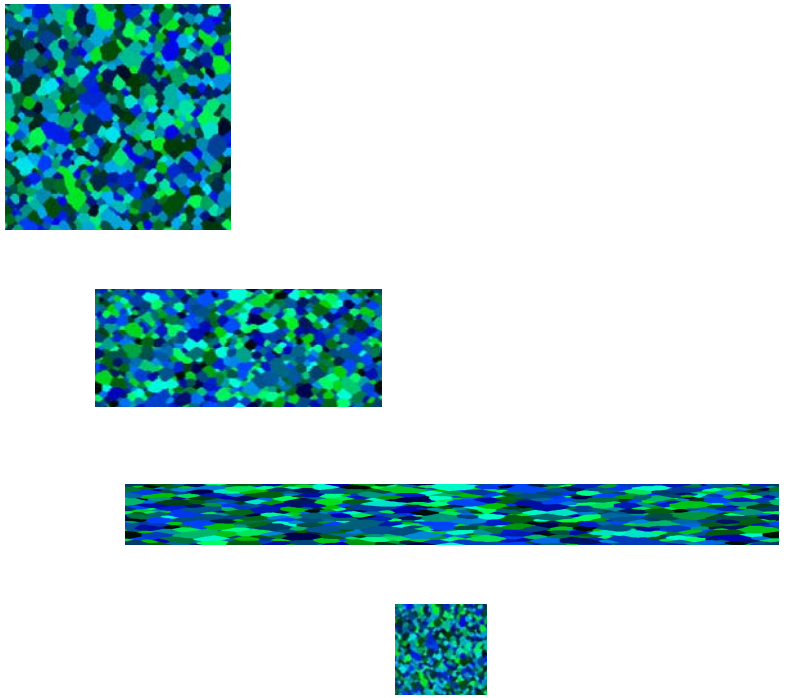
4 threads



16 threads

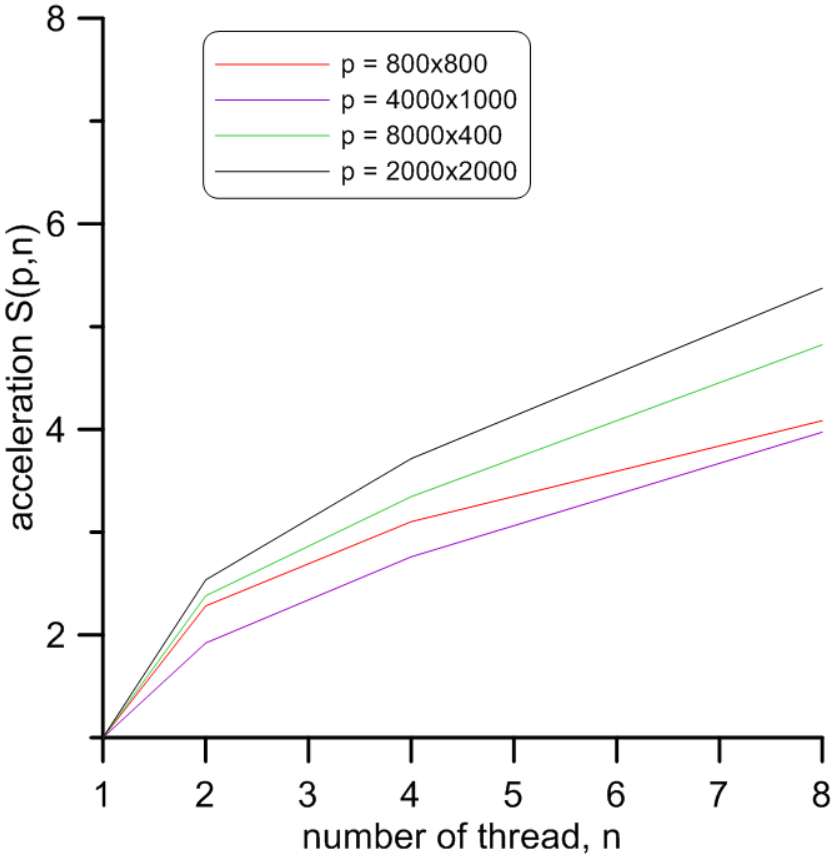
Result – Computing time





Computing time for 100MCS with different space size and number of thread

Result – acceleration



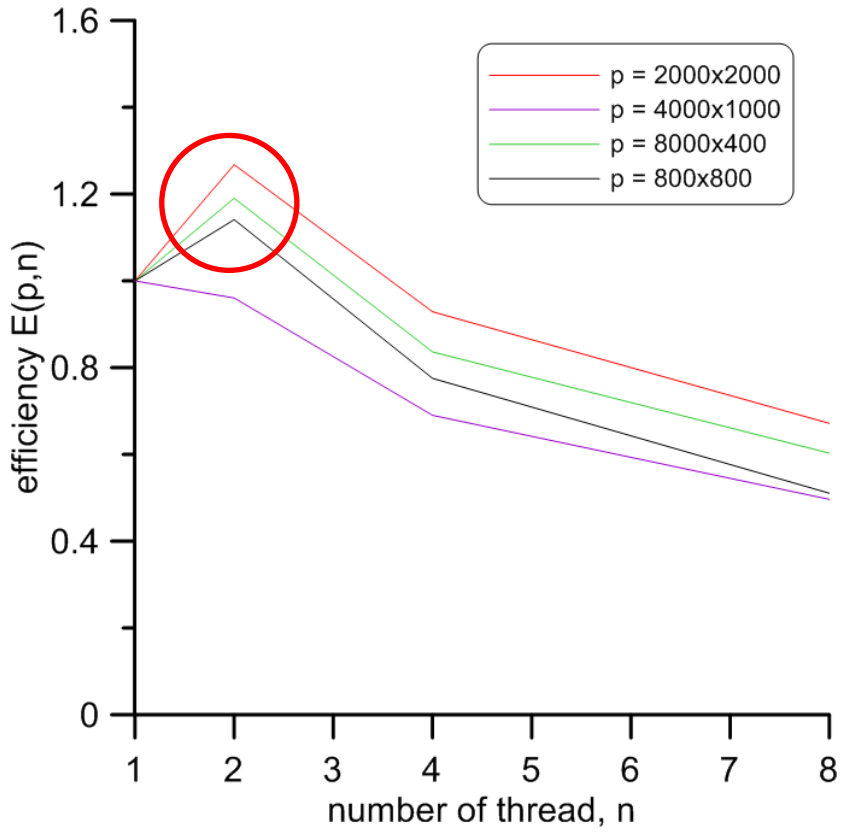
Acceleration :

$$S(p) = T_i / T_{||}(p)$$

Where: T_s - execution time for one processor,
$T_{||}(p)$ - execution time for p processors

Acceleration as a function of thread number for 100MCS with different space size

Result – efficiency



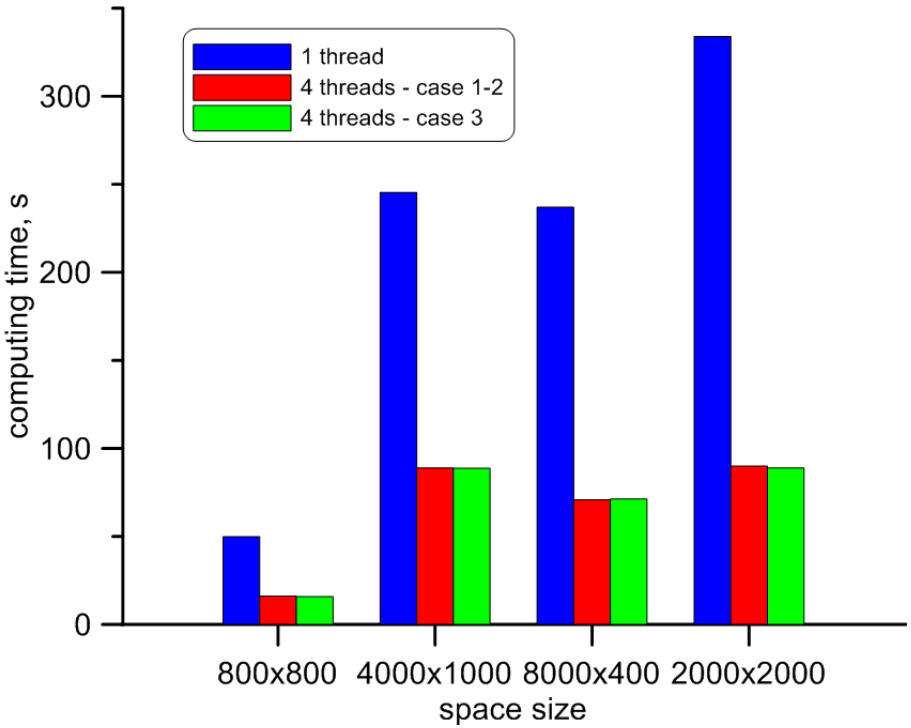
Parallel efficiency:

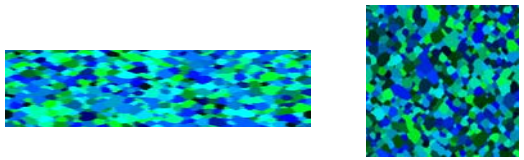
$$E(p) = S(p) / p$$

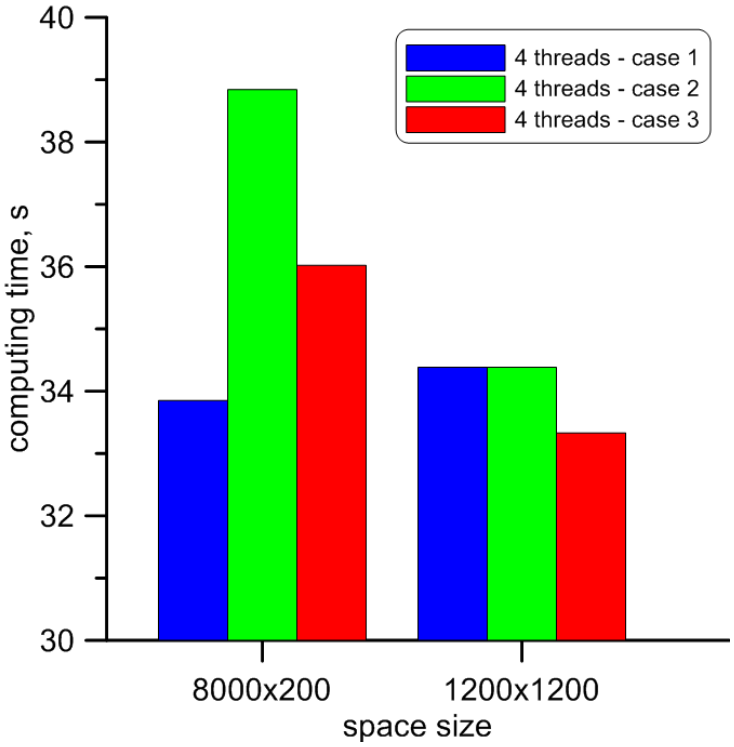
Where: *S(p)* - acceleration, *p* processors number

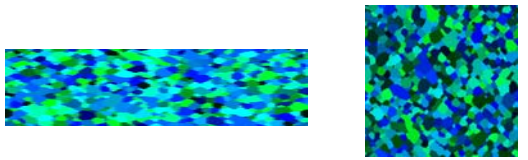
Computing efficiency for 100MCS with different space size and number of thread

Comparison – three approaches to space division









Conclusions

•MC algorithm can be successfully applied to create polycrystalline material representation.

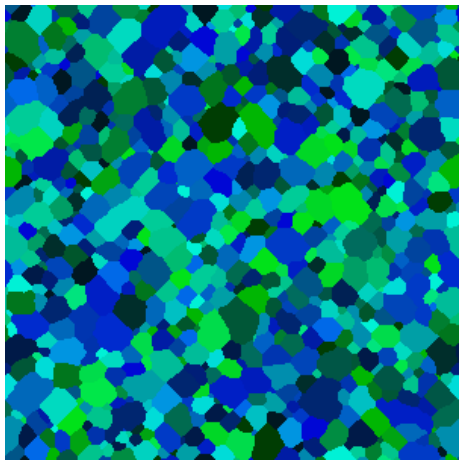
•The major problem of this method is long computation time.

•Computation time can be reduced by modification or parallelization of the algorithm

•Each MC space division scheme caused decrease in calculation time.

•The different division of the MC space between the threads significantly affect speedup calculation.

•Proper selction of space division can decrease calculation time almost 6 times.



Future work plans

•Scattering calculation on multi node system.

•Extension of different space division and parallel algorithm developed within this work to model static recrystalization phenomenon.