Multiscale modeling report

Jakub Witas, Informatyka Stosowana NS

II Rok, 3 Semestr, 279048

1. Selected technologies

Program for multiscale modeling is implemented in Java language. It has been chosen, because java has GUI framework called Swing. It was easier to use framework that has all ready to use components, like inputs or text field.

It was written in IDE called NetBeans. That code editor allowed us to manipulate GUI, with our mouse and just “paste” selected element to user interface.

In order to run the application on Your computer You need to install Java SDK and NetBeans 8.2 or newer. After installing of all those required components You need to simply open the project with NetBeans and press Run. Window with main view of the application will pop up and simulation will be ready to started.

2. GUI Description

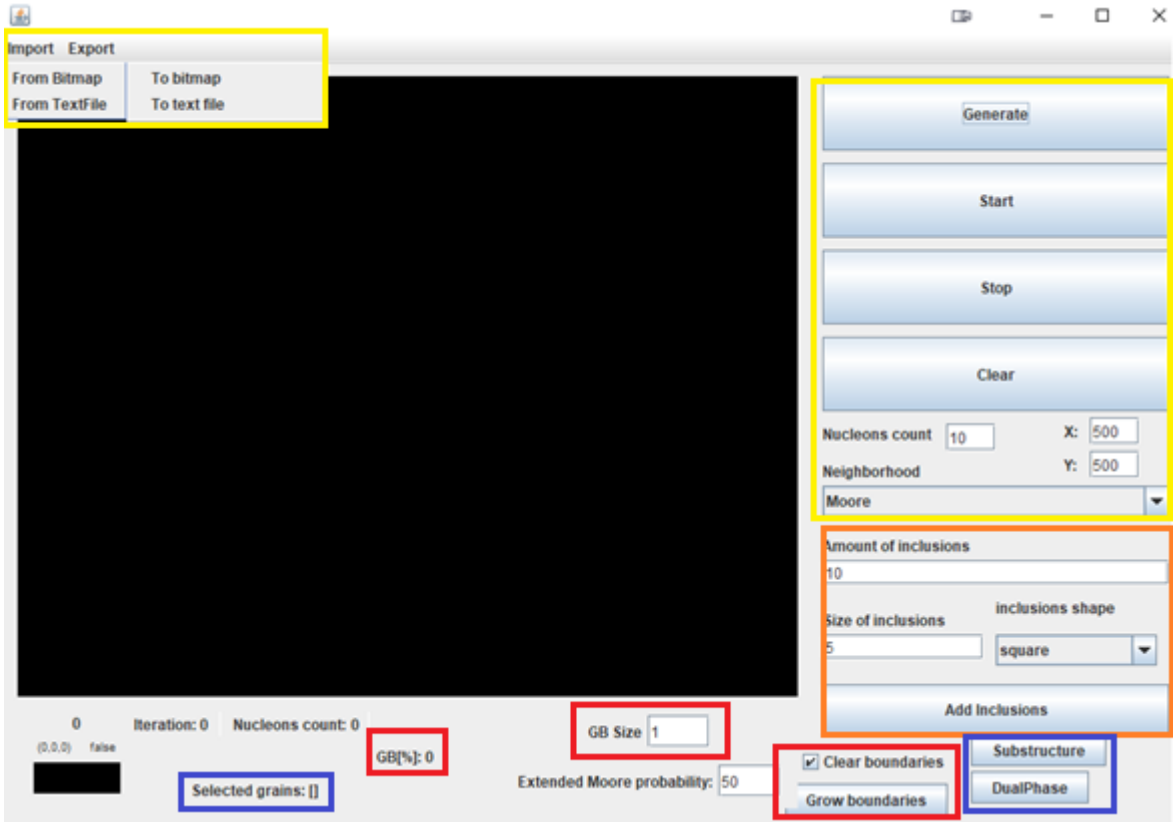
Our graphic interface allows user to:

1) **First classes (Yellow color on image 1)**

- Change size of generated area to growth grain. There is a X/Y variables, localized below Clear button. User can enter his own width, height of area.
- Four buttons:

- Generate – that button allows user to generate new nucleons on map. Also when user change X/Y map, it’ll refresh it.
- Start – when user will click that button, simulation of CA will began.
- Stop – when user will click that button, simulation will be stopped.
- Clear – when user will click that button, map will be cleared from all nucleons. That button have two phase of working. First when the checkbox “Clear boundaries” is

checked, it'll clear whole map. Second when the checkbox is disabled, it'll clear all grains, expect grain boundaries.

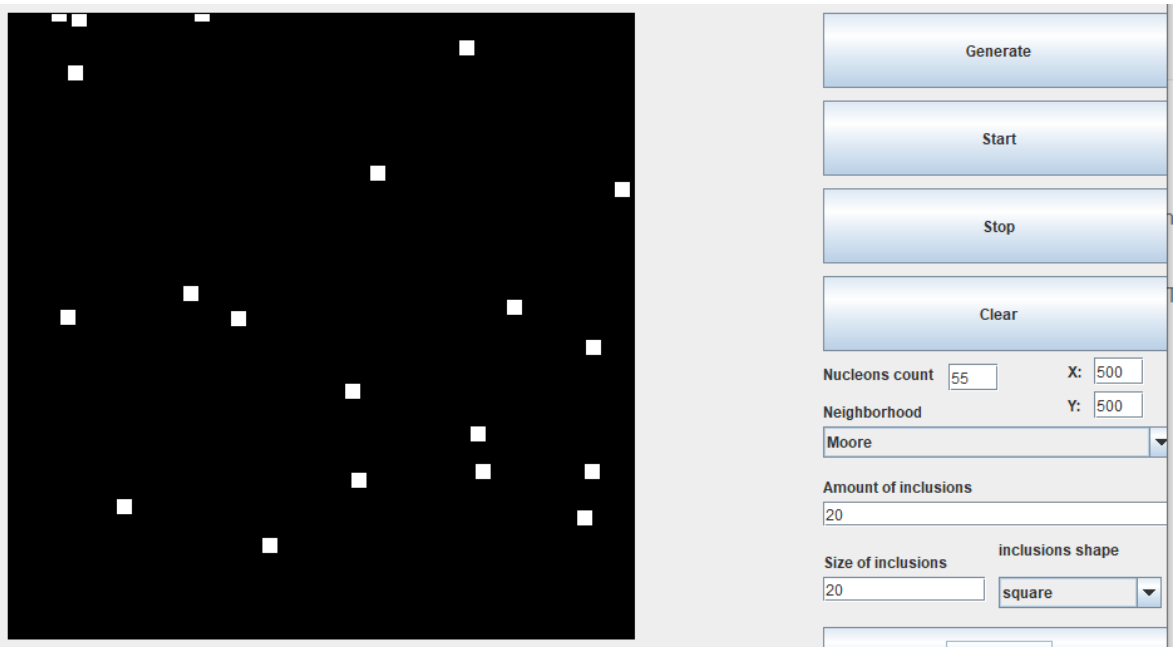


1. Image of Graphic User Interface from CA Software

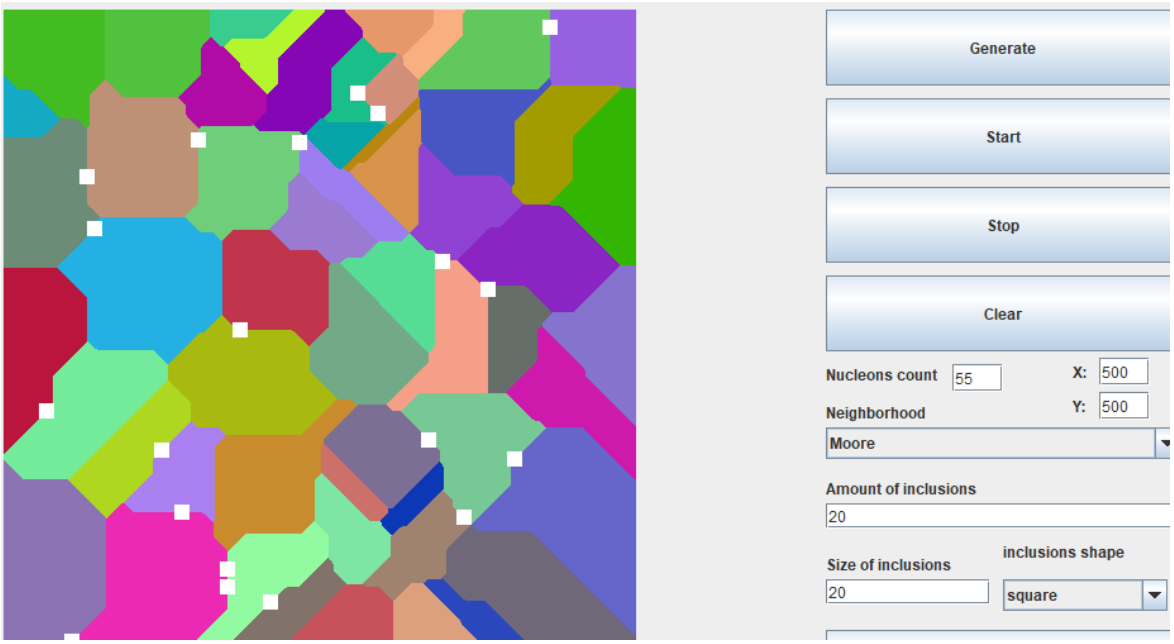
- Neighborhood multiselect – user can select three neighborhoods from there. Available options are: Moore, von Neumann and extended Moore.
- Nucleon count – that input tells the program, how many nucleons will be generated when generate button will be clicked.
- Import and export map to file. There is two options to import and export generated map. First option is to export generated map to bitmap, second option is to export map to text file.

2) Second classes (Orange color on image 1)

- There are two available implemented types of inclusion shapes. We can select square or circular inclusion shape (**image 2 and 3**).
- Size of inclusions – user can define how big inclusions will be. Depends on that number inclusions size will increase.
- Amount of inclusions – user can specific the amount of inclusions, that will be generated.
- Add inclusion button – button allow user to add inclusions randomly to map when simulation is not generated or when simulation is ended, inclusions will be added to grain boundaries.



2. Inclusions added before simulation



3. Inclusions added after simulation

3) **Third classes**

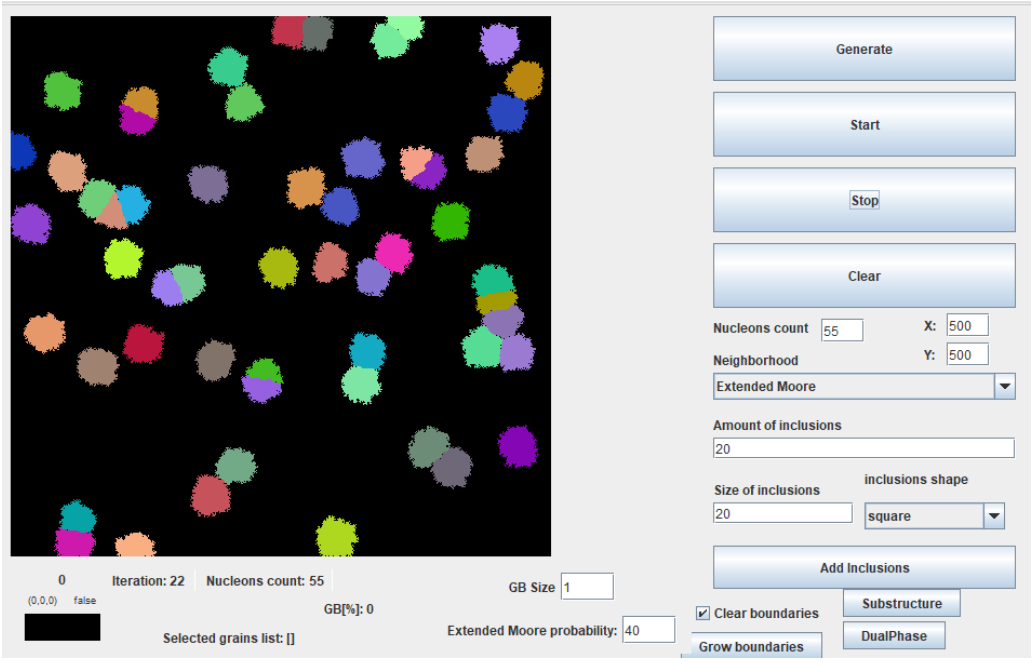
- When user will select extended Moore from Neighborhood select, it will be checking three rules.

  First – Moore: The id of particular cell depends on its all neighbors. If five to eight of the cells neighbors id's is equal to S, then cell transforms to the state S

  Second – Nearest Moore: The id of particular cell depends on its nearest neighbors. If at least three of the cells neighbors id's is equal to S, then cell transforms to the state S

  Third – Further Moore: The id of particular cell depends on its further neighbors. If at least three of the cells neighbors id's is equal to S, then cell transforms to the state S

  Fourth – Probability rule: The id of particular cell depends on its all neighbors, and has X % probability chance to change.



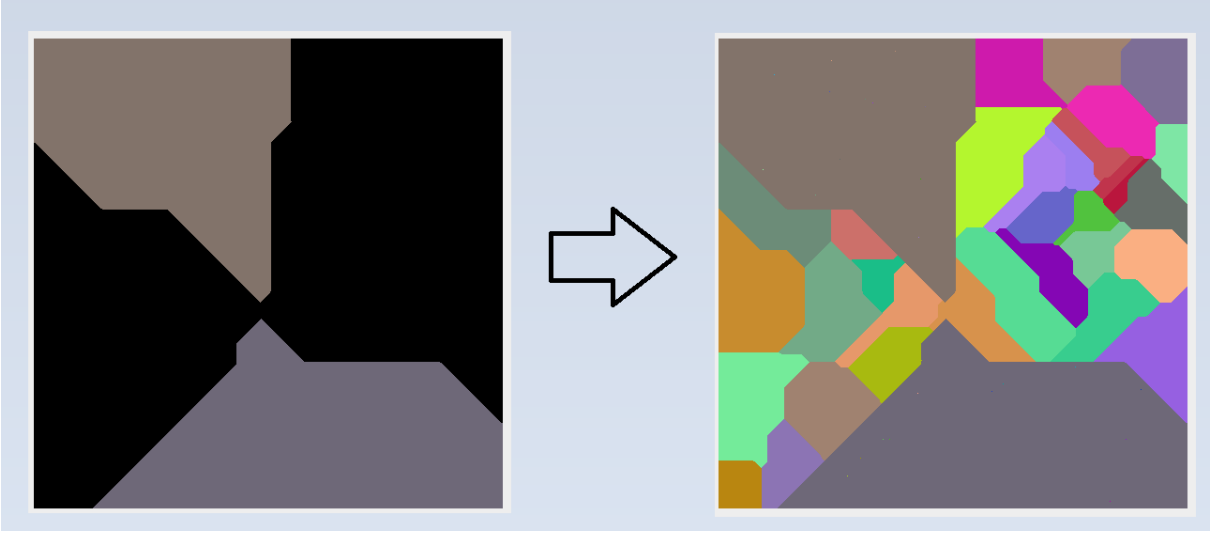
4. Extended Moore simulation with 40% probability rate

4) **Fourth classes (Color blue on image 1)**

- After simulation is completed, user can select grains to get Substructure or DualPhase of that grain.

Substructure:

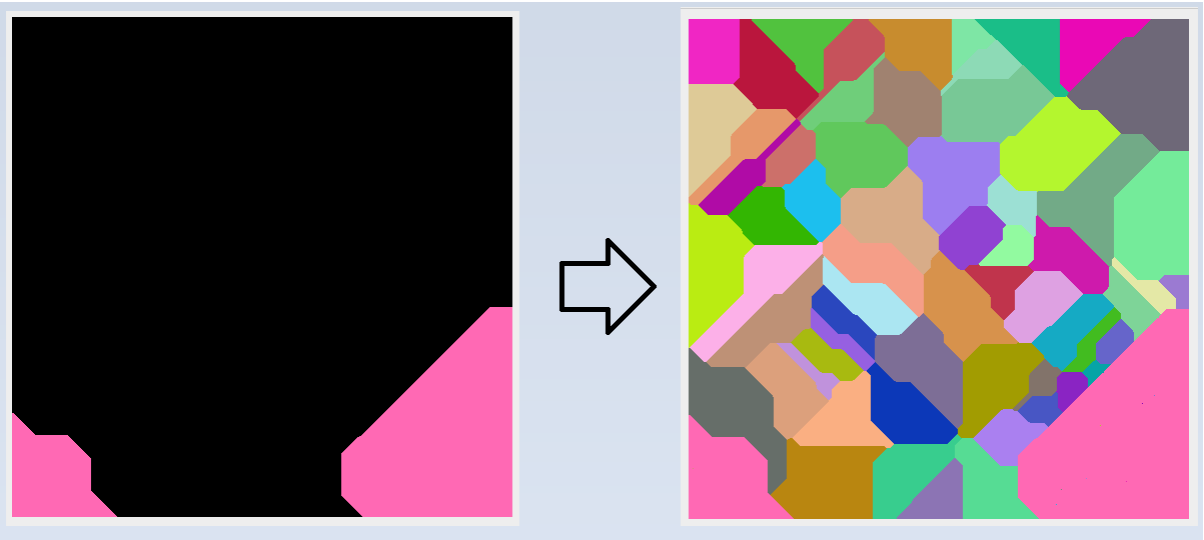
Selected grains from simulation will stay on map after clearing map. When the second simulation is over grains will remains the same, but other nucleons will rise (image 5).



5. Subtraction of structure

Dual phase:

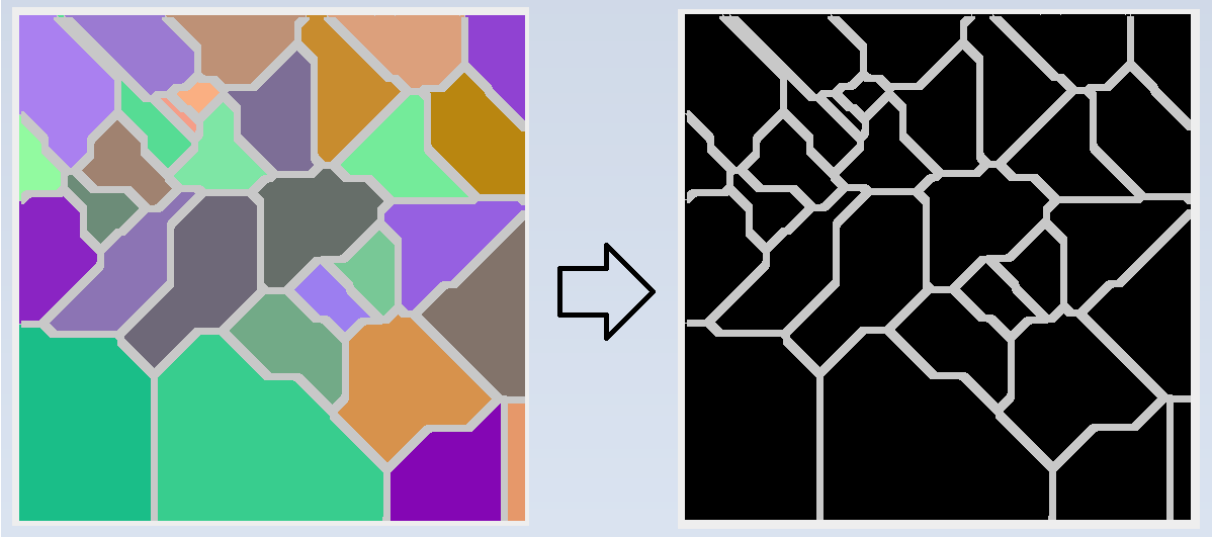
Selected grain will change the phase and stay after clearing map. After second simulation they, will cover the rest of grains (Image 6).



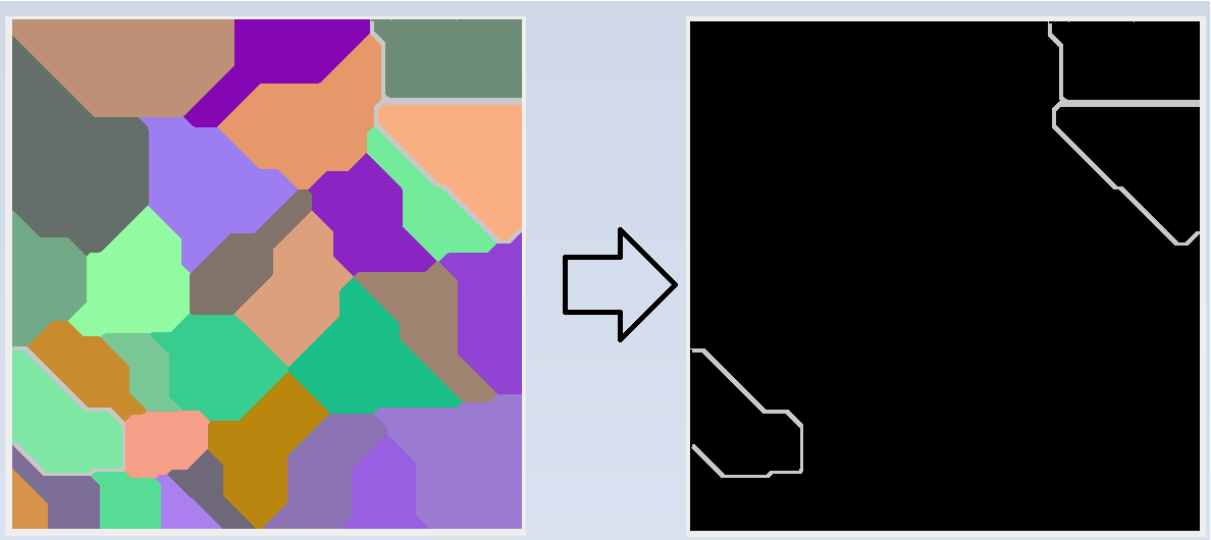
6. Dual phase of structure

5) Fifth classes (Color red on image 1)

When simulation is over there is a possibility to add grain boundaries to map. There is input called GB Size which add possibility to change grain boundaries size. If user don't choose any of grain, then grain boundaries will, be added to every grain (Image 7). When user will select some of grain, then boundaries will be added only to them (Image 8). There is a also a possibility to clear map, but leave grain boundaries on it. User need to uncheck the Clear boundaries checkbox to make it happen.



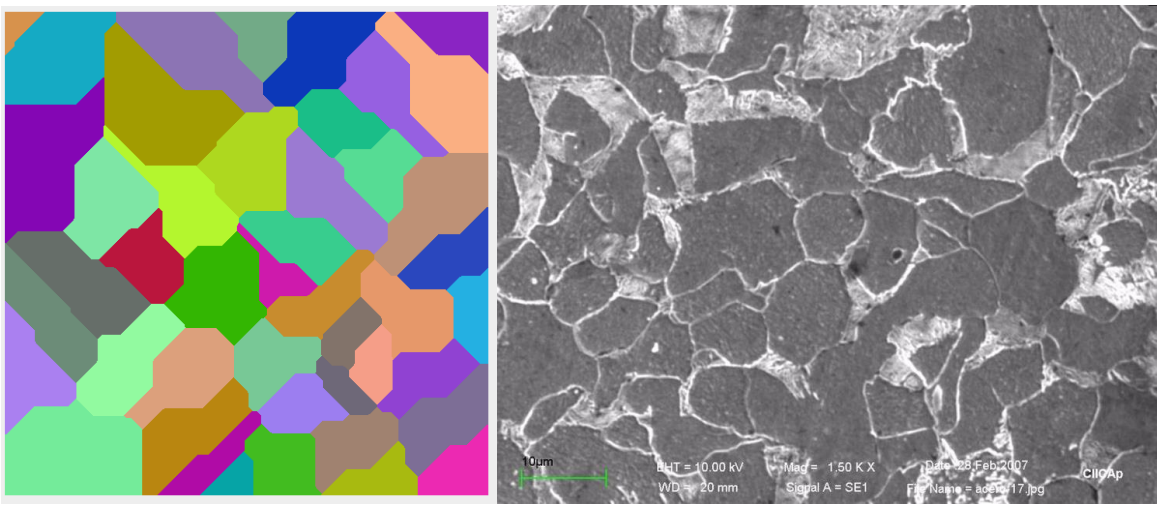
7. Grain boundaries without selected grains



8. Grain boundaries with selected three grains

3. Comparing with real microstructures

Down below are two microstructures. First one is from program and the second one is the real steel microstructure.



9.Two microstructures. First is generated by program, second is real one

- Shapes of both microstructures are pretty similar to each other, but the size of grains differ very much between those two pictures.
- Grain borders in real microstructure are much thinner and more recognizable then in generated one. Borders on the structure generated by the program are poorly visible.
- In real microstructure it can be noticed that there are several grayish colors that represent different grains. In my application the grains are much more distinguishable with the usage of RGB colors

4. Conclusions

Implementation of program help us generate multiple variants of microstructures. It is very helpful to recognize, check and show growing of the microstructure.

The main advantage of the approach presented is the assumption of finite sizes of cell grids, whose states change synchronously in a discretely defined time step, and by assuming interaction only with the closest neighbors, they also ignore the effects of long-range interactions.

Since the evolution of CA depends on the rules of the automaton as well as on the initial state, there is an attempt to describe specific features of the automaton only on the basis of its rules.