A Parallel Cellular Automaton Model For Adenocarcinomas in Situ with Java: Study of One Case

Antonio J. Tomeu-Hardasmal[1(✉)], Alberto G. Salguero-Hidalgo[1], and Manuel I. Capel[2]

[1] Department of Computer Science, University of Cádiz, 11519 Puerto Real, Spain
{antonio.tomeu,alberto.salguero}@uca.es

[2] Department of Software Engineering, University of Granada, 18071 Granada, Spain
mcapel@ugr.es

Abstract. Adenocarcinomas are tumors that originate in the lining epithelium of the ducts that form the endocrine glands of the human body. Infiltrating breast and one of the most frequent neoplasms among female population, and the early detection of the disease is then fundamental and, for this reason, a profound knowledge of the biology of tumor at this phase is essential. Among the distinct tools that contribute to this knowledge, computational simulation is more frequently used every day. The availability of fast and efficient computations that allow the simulation of tumor dynamics in situ, under a wide range of different parameters, is an important research topic. Based on cellular automata, this paper proposes a generic simulation model for the Adenocarcinomas In Situ (CIS). We applied it to the breast ductal adenocarcinoma in situ (DCIS), modeling our cells with the genomic load that we currently know that the tumor starts, and proposing a numerical coding method for the genome that allows efficient computational management. We propose a parallelization scheme using data parallelism, and we show the acceleration achieved in multiple nodes of our cluster of processors.

Keywords: Adenocarcionomas in situ · Cellular automaton
Data partition · Parallel processing · *Speedup*

1 Introduction

It is estimated that one in eight women [8] will suffer breast cancer, being approximately 80% of them ductal carcinomas. Likewise, one in every nine men will suffer a prostate cancer. Thus, the incidence of these neoplasms in the adult population and the magnitude of the health problem they imply will be very important. Early detection is aimed at identifying the disease when it has not yet acquired infiltrating character, and it is limited to glandular ducts (in situ), i.e. the glandular parenchyma not was infiltrated yet. At this point, the disease can only be root out with surgery that removes the affected segment of the duct, and a safety margin free of disease, while preserving the rest of the patient's breast, with a success rate of more than 90%. In the case of prostate carcinomas, prostate-specific antigen, which has traditionally been used as a tumor marker, it has been found out that is unaccurate in screening the disease for men. In both cases, the characterization of the disease when is still in situ becomes of great interest, and for this, computer simulation can be an excellent tool to investigate it. Carcinogenesis is a phenomenon in which one or multiple mutations on certain genes allow the cells to reproduce and survive abnormally, under a selection process that results in uncontrolled tumor growth characterized by infiltrating nature. There are many mathematical models in the literature that contain the knowledge we currently have about genes involvement in neoplasms [1–3,6,7,9,11], which study the mutations that neoplasms can develop to originate a CIS. In this paper, we propose a three-dimensional cellular automaton based CIS model to simulate a generic glandular duct and to analyze how the mutations in the cells of the simulated duct become CIS. We also apply the model to known breast intraductal adenocarcinoma data, parallelize it and we study its natural development with respect to the parallel model and the acceleration achieved.

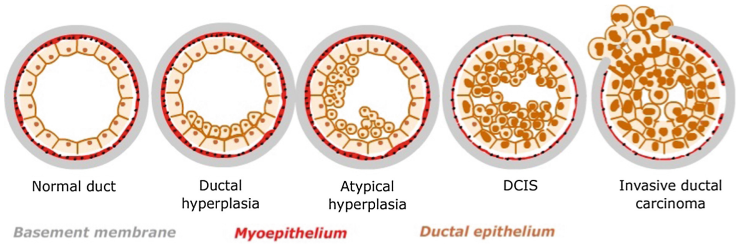


Fig. 1. Natural development of Adenocarcinomas in Situ (CIS), from a normal duct to an infiltrated one.

2 Biology of Breast Adenocarcinomas in Situ

If not detected and treated, the natural development of these tumours is the progression to an infiltrating adenocarcinoma, as shown in Fig. 1. In the case of the human breast, it is now known that within a normal duct the two types of cells that form the ducts originate from a single class of progenitor cell that, by cellular differentiation, leads to two germ lines that conclude in the two cited types of cells. Ductal adenocarcinomas initially have a local character and then grow to infiltrate and reach the duct.

It is known that those women with genetic predisposition to breast cancer accumulate inherited specific mutations [16,17], and thus, an estimate points out that up to 12% the number of cases is due to this circumstance, not mentioning other genes that may be involved. In addition it is now known that mutations in the BRCA1, BRCA2, PTEN and TP53 genes increase the likelihood of suffering from ductal carcinoma. In the model proposed here, this genetic predisposition will be taken into account by means of a logical variable HMG. In our simulation, all the *stem* cells of the duct will be defined with the genetic predisposition incorporated into their genome.

The meaning of the four genes that we will consider in the simulation is illustrated in Table 1. In it, the first and second columns collect the modeled genes and their function in physiological conditions. When one or several of the genes suffer damage, the behavior of the cell that contains it becomes malignant. When a chain of specific mutations occurs it inexorably leads to the proliferation of ductal carcinoma, first local, and then infiltrating, breaking the duct and expanding to the glandular parenchyma.

Table 1. Pathological functioning of genes.

Gen	Operation with damages
BRCA1	The cell dies
BRCA2	Neoplastic reproduction
PTEN	Neoplastic reproduction
PTEN	Does not inhibit neoplastic reproduction
TP53	Cell survival with damage to proto-oncogenes
TP53	Cell survival with damage to the double-layer architecture

3 Cellular Automata

A cellular automaton (CA) [5,15,18] is as a 4-tuple $(\zeta, \varepsilon, N^I, \rho)$ where:

- ζ is a discrete regular network of cells (or nodes) together with some border conditions set for the finite dimension net case, which are of use to define neighboring conditions of cells at the net frontier.In our case, we have the mathematical representation of a 3D-cubic: $\zeta = \{r : r = (r_1, r_2, r_3) \in Z^3\}$.
- ε is a finite set (usually, with an algebraic Abelian ring structure) of states that the network of cells can take on.
- N^I is a finite set of cells that define the neighbor cells with which a given cell of the network can interaction.
- The transition function ρ that defines how any cell's state can change depending on time and on its neighbor cells own state N^I.

Given the previous definitions, any area of cells can be defined as the network ζ included in the real 3D space R^d that uniformly covers a portion of the d-dimensional Euclidean space. Each cell is labeled by its position $r \in \zeta$. The layout of cells is spatially specified by the connections that any cell holds with its closest neighbors, which are obtained by connecting pairs of cells following a regular pattern. For any spatial coordinate r, the neighborhood grid $N_b(r)$ consists of a list of neighbor cells that is defined by

$$N_b(r) = \{r + c_i : c_i \in N_b, i = 1, \cdots, b\} \tag{1}$$

Where b is the *coordination number* or, in other words, the number of the grid neighbors that directly interact with the cell at coordinate r. N_b denotes the elements in that pattern as $c_i \in R^d$, $i = 1, \cdots, b$.

$$\zeta = \{r : r \in \mathbb{Z}^3\} \tag{2}$$

The total number of cells available is usually denoted by $|\zeta|$. The entire set of neighboring cells whose states affect any cell r is defined by the *interaction vicinity* $N_b^I(r)$ function, $N_b^I(r) = \{r + c_i : c_i \in N_b^I\}$.

Any cell's neighborhood can be chosen in different ways, though we choose for our simulation the vicinity schema of Moore [5], where any cell has as neighbors only its surrounding cells. Furthermore, each cell $r \in \zeta$ has a state $s(r) \in \varepsilon$. A global configuration of the automaton $s \in \varepsilon^{|\zeta|}$ is determined by the state of all the cells on the grid. Finally, model's temporal evolution dynamics is determined by the function of transition ρ that specifies the changes in any cell state according to its previous state, and the interaction with closest-cells neighborhood given by $\rho : \varepsilon^{\mu} \rightarrow \varepsilon$ where $\mu = |N_b^I|$. The rule is proved to be spatially homogeneous and does not therefore explicitly depends on the position of a given cell [14]. Extensions of the previous definition to include temporary or spatial homogeneity are feasible. If the CA is deterministic, the function of transition yields only one feasible change of state, whereas if it is stochastic, the new cell's state is given by a specific distribution of probability.

4 Modelling Breast Adeconocarcinoma Ductal in Situ with Cellular Automata

To model the duct, a cellular automaton [12,13] assuming a three-dimensional ζ grid with $20 \times 20 \times 200$ nodes is used, which is built from the two-dimensional model proposed in [14], by just adding an additional dimension. Each node may contains a cell. Although a human ductal cell has a genome composed of multiple genes with millions of DNA bases, we will limit ourselves to consider only the four genes in the model involved in the pathogenesis of the DICS, which are encoded by 32 bits integers. The genetic load of a cell is then modeled by an ordered tuple of the form $GC = (brca1, brca2, pten, tp53)$. The tuple GC is encoded in its turn by a single integer using the pairing function given by the Eq. 3. The three dimensional version of *Moore*'s neighborhood and a null boundary condition is used to give the ends of the duct a biological coherence.

$$\langle x, y \rangle = 2^x(2y + 1) - 1 \tag{3}$$

This function, which is a bijection, may be nested by means of the expression.

$$\langle \langle brca1, brca2, \rangle, \langle pten, tp53 \rangle \rangle \tag{4}$$

It allows the encoding of the entire genome of a cell in a single positive integer by using a compact and reasonably efficient way. In this way, each node of the ζ grid of the cellular automaton contains a pair of positive numbers that respectively code the cell type and its genetic load, which in turn are re-encoded by applying the pairing function to both data, so that the node contains a unique number in $\mathbb{Z}^+$. Decoding the integer to update that genetic load when a cell mutates or for any other reason is trivial, given the encoding technique exposed, by simply using the decoding functions $r(z)$ and (z), described in the Eqs. 5 and 6. The set of possible cell types[1] of the grid is $S = \{free, basal, luminal, myoephitelial\}$ and, as we have said, they are numerically coded.

$$l(z) = min_{x \leq z}[(\exists y)_{\leq z}](z = \langle x, y \rangle) \tag{5}$$

$$r(z) = min_{y \leq z}[(\exists x)_{\leq z}](z = \langle x, y \rangle) \tag{6}$$

The three given functions are primitive recursive [4] and, therefore, computable. The nodes of the grid are synchronously updated node by node of the duct. The cells are updated according to the probability of mutation and its neighborhood environment. Both variables define the transition function ρ. The selection and updating of the 8×10^4 nodes of the grid defines a generation. The number of generations varies depending on the length of the natural history of the tumor being simulated, increasing or decreasing the number of generations of the simulation. The grid is initialized by a completely deterministic algorithm that creates a base membrane and places a small number of progenitor cells in its interior, which reproduce to form a double layer duct, though we will apply it to the breast ducts here. Each section of the duct contains approximately 45–50 luminal cells. Originally, all the cells are located on the duct have a healthy genome, which is represented by 32 bits equal to zero. Mutations are modeled by nullifying the value of one or several bits of a gene. The `HMG` *flag* allows us to execute the model considering an inherited genetic predisposition to contract the disease, using a Monte-Carlo method. The algorithm to obtain a simulation of the duct compatible with the histological structure of a human breast is shown below[2].

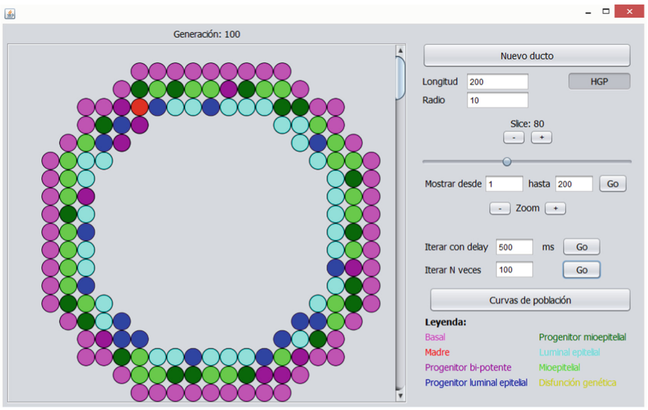
```
Algorithm SetUp
Input: empty grid
Output: grid with initial states for nodes
Method:

1. With radial symmetry put basal_cells to define basal
   membrane;
```

[1] Basal cells: they form the outer layer of tissue that surrounds the duct; luminal and myoephitelial cells: form the internal structure of a normal duct (see Fig. 1); free represents the internal space of the duct that is empty.

[2] For the sake of clarity, we have abstracted the necessary coding and decoding steps that allow us to modify the state of a node of the reticle or the genome of a cell located in that node. However, the reader should always bear in mind that any reading or writing to node in the grid requires that state modification.

```
    stem_cells=[];
    //seeding stem cells...
 2. for(i=0; i<200; i++){
     cx=random(0, 19);
     cy=random(0, 19);
     cz=random(0,199);
     grid[x][y][z]=stem;
     stem_cells.add((x, y, z));
    }
    //putting mutations in stem cells...
 3. if(HMG==true)
      for iterator in stem_cells{
        x=iterator(x);
        y=iterator(y);
        z=iterator(z);
        mutate(grid[x][y][z], all_gens, 15%);
      }
    //making the rest of duct...
 4. While(free_places){
    5. for all cells in grid
         reproduce(grid[x][y][z], adjacent, hierarchy);
    6. for all !(stem_cell) in grid
         migrate(grid[x][y][z], vacant_neighboring,
             radial_symemtry);
    }
```


Generación: 100

Longitud 200
Radio 10
Slice: 80
Mostrar desde 1 hasta 200
Iterar con delay 500 ms
Iterar N veces 100

Fig. 2. Initial state for a layer of the duct.

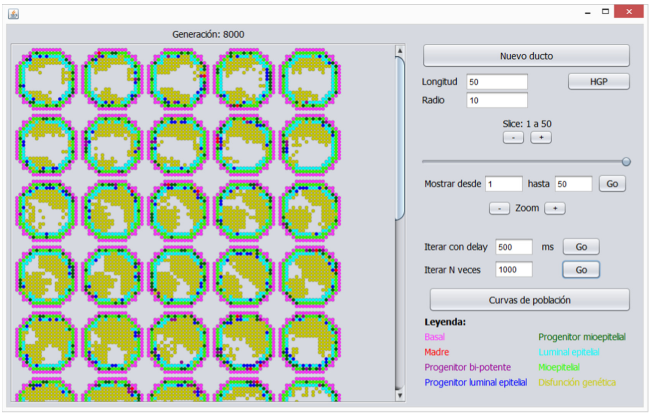
When the previous simulation is executed in the Java language, a grid is obtained that coherently models the normal histological structure of a human duct (Fig. 2), and all the cells generated in the grid remain inside, adopting the double layer structure illustrated in Fig. 1 for the normal state of the duct.

During the reproduction phase, modeled on line number 5, all the cells of the grid get divided and take up the adjacent places, wherever there is enough room for them. The inherited genetic load can be mutated, according to the mutation rate set as a parameter of the `mutate` method, to which we have given a value of 15%, which reasonably encompasses the various real causes that can lead to this type of mutations, and that include the environment, the genetics and even the type of the cell [7]. Since all genes are mutated through the method, including those that control both mitosis and programmed cell death, the cells of the duct resulting from the routine `SetUp` will eventually lead to neoplastic pathology. Once the grid is in its initial state, it is necessary to make it evolve over time, which is the responsibility of the `Evolve` algorithm.

```
Algorithm Evolve
Input: grid in t−time
Output: grid in (t+1)−time
Method:

//now, the transition function...
1. for all cells in grid{
      //normal apoptosis...
     2. if((mutations(BRCA1(grid[x][y][z]))
           +mutations(TP53(grid[x][y][z]))))>32)
          grid[x][y][z]=free; //cell dies
    //normal apoptosis...
     3. if(mutations(TP53(grid[x][y][z]))<16 && (
           !adjacent_basal(grid[x][y][z]) ||
           !adjacent_myopithelial(grid[x][y][z])))
           grid[x][y][z]=free; //cell dies
      //anormal apoptosis...
     4. if(stem(grid[x][y][z])){
          5. if(adjacent_free(grid[x][y][z]))
               normal_reproduction();
          6. if((mutations(BRCA1(grid[x][y][z]))+
                 mutations(BRCA1(grid[x][y][z])+
               mutations(PTEN(grid[x][y][z])+
               mutations(TP53(grid[x][y][z]))))>64)
               cancerous_reproduction()
       }
      7. for all !(stem_cell) in grid
           migrate();
   }
```

In the previous algorithm, the methods `BRCA1`, `BRCA2`, `PTEN` and `TP53` takes the integer that encodes the genome of a cell in the grid and extracts the 32-bit integer that encodes the gene according to the name of the method; the `mutations` method takes a numerically coded gene as its argument and returns the number of mutations it presents, as an integer between 0 and 32. The methods `adjacent_basal` and `adjacent_myopithelial` allow us to know the type of the cells that form the *Moore*'s neighborhood cube of a given cell while the method `adjacent_free` gets the free nodes on the neighborhood of the node given as argument. On the other hand, there is also available a set of four methods that allow us to know the type of the cell that is in a node of the grid. The subroutine `normal_reproduction` allows parents to be reproduced correctly around their local cubic neighborhood, preserving the double layer structure of the duct. The subroutine `cancerous_reproduction`, allows parents to be reproduced at points in their local cubic neighborhood, but does not respect the double-layer structure of the duct, and ended up forming an intraductal carcinoma in situ. Note that for a parent to reproduce in this way, it is necessary that the total sum of mutations present in their four genes is greater than half the positions that the four genes encode. Finally, a simulation is carried out using a given number of discrete time steps, in which each of them the described algorithm `Evolve` is executed. When a critical number of mutations is reached, cells begin to proliferate uncontrollably, filling the duct lumen and forming the carcinoma in situ. Figure 3 illustrates this for a segment of the duct consisting of fifty layers, where the neoplastic transformation has taken place and the malignant cells have begun to fill the duct lumen, without infiltrating the base membrane.


Generación: 8000

Longitud 50
Radio 10
Slice: 1 a 50
Mostrar desde 1 hasta 50
Iterar con delay 500 ms
Iterar N veces 1000

Leyenda:

Basal	Progenitor mioepitelial
Madre	Luminal epitelial
Progenitor bi-potente	Mioepitelial
Progenitor luminal epitelial	Disfunción genética

Fig. 3. Simulation for a full duct by layers. Neoplastic cells (yellow) are filling the inside of the duct. (Color figure online)

5 Implementation

The previous model was implemented using the Java programming language, and parallelizing the initial, sequential version. The parallelization employed the principles of symmetric multiprocessing with data parallelism, dividing the grid in its longitudinal dimension z into cubic sections that were processed by different threads on a dedicated *core* to each one of them [18].

A security condition is implemented to ensure the consistency of the simulation, which consists of forcing a thread trying to write in the bilayer section of the neighboring sub-table to consult the state of the node in which it intends to write after the acquisition of the lock, since the thread responsible for that sub-grid reticle could have occupied that node in its own writing time [15]. The execution tests were developed on four different nodes of the cluster of our university. Each node has two Intel®Xeon^{TM} E5 processors at 2.6 GHz, which yield 20.8 Gflops together, with 128 GB memory and without hyperthreading activated. The entry node operates the HP Cluster Management Utility on Red Hat Enterprise Linux for HPC and the processing nodes the version *Compute Node* of the same operating system. The version of the Java development kit used was Oracle 1.8.0.151-1.b12.el7_4.

6 Measurements and Results Discussion

Figures 4 and 5 show the average times and *speedups* obtained, for 5×10^3 generations. Once computed, the simulation stopped. Average times ans *speedups* were obtained by computing the same simulation in different nodes of our cluster It can be seen that both time and speedup curves reach their bests values for eight parallel threads and that these values get worse if the number of parallel tasks is increased. In other words, the optimal average time is 4.18 s for a maximum speedup of 5.85, over a theoretical maximum of 16, which is the number of cores available in each node, and starting from an optimal sequential time of 24.45 s. One might think that the parallelization of the model, which barely gets up to half of the theoretical maximum *speedup*, could be easily improved. However, the following items clarify the obtained results rationale:

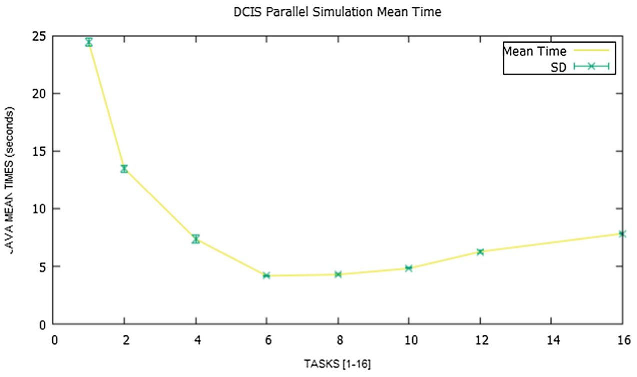


Fig. 4. Execution times (Mean ± Standard Deviation).

- It must be remembered that the contact zones between sub-reticles, controlled by different threads, are protected by a mutually exclusive lock, which introduces undesirable latencies that are however necessary to guarantee the coherence of the state of the nodes in the grid. This directly induces an

overload of the execution time that is proportional to the number of parallel threads (and contact zones). This shortcoming was identified in previous work [15] and by other authors in [10].
- Although the introduction of different reading and writing grids allow the threads to process the nodes that are not in the contact areas of the grid in a fully parallel manner, this is not the case of the nodes in those areas. Here, the thread that wants to write to a node located in the contact area, once it has got the permission to do it, cannot just use the neighborhood data in the reading grid, but has also to check the writing grid to verify that there is enough free space for the modification, because another thread may have occupied that space as the result of a mitosis. This also induces execution overloads.
- It is also worth mentioning that each node of the grid encodes a lot of information by using a single positive integer (type of cell that occupies the node, and BRCA1, BRCA, PTEN and TP53 genes). This introduces a heavy load process for decoding (and encoding, when appropriate) the information in the neighborhood of a cell.

One could think that the representation of the state and the genome of a cell by means of a class `gridCell.java` could improve the results, although the measures obtained by using an alternative implementation, discarded that. The space occupied in the *heap* of the Java Virtual Machine by the nodes modeled as classes, and the need of navigate through their respective references to reach them, increases the global process times and decreases the *speedups*. In short, the three previous items justify why those *seepdups* have been obtained, being the second one of particular relevance, and also being coherent with models that develop similar simulation dynamics in two dimensions, such as the results published in [10] and in [15]. The parallelization is worthwhile by itself, besides the proposed method is very general, and applicable to other types of tumors where, depending on the transition function with which they are modeled, the speedup could be slightly improved.

Regarding the biological fidelity of the model, we have compared the simulation with a real specimen, using the number of neoplastic cells in the duct as a as variable that changes over time (number of generations). In the real case, the genetic predisposition was verified by means of immunohistochemical method, while in our case the corresponding *flag* of the simulation was activated. The classic gompertzian behavior [7] that describes the tumor dynamics for both *in vivo* and *in vitro* were observed, which tends to occupy all available tissue domain with a quasi-exponential acceleration from a specific time instant. We see that the simulation *in silico* is compatible with biological and histological observations, with an acceptable degree of fidelity in terms of the global dynamics of neoplastic growth *in situ* refers.

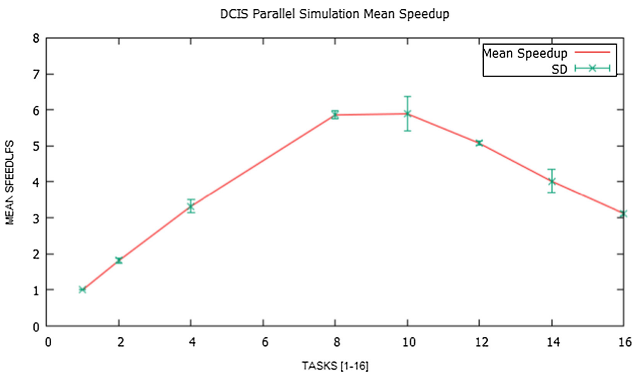


Fig. 5. Speedups (Mean ± Standard Deviation).

7 Conclusions and Future Works

In this work we have proposed a general procedure for the parallel simulation of adenocarcinomas *in situ* by using cellular automata-based model. A change of the transition function of the cellular automaton and the genetic load model allows us to adapt it to different types of glandular neoplasms, before they adopt infiltrating character. From the proposed algorithms, a parallel implementation has been developed using the Java language with symmetric multiprocessing by means of data parallelism for the study of a case: breast ductal adenocarcinoma *in situ*. The parallel simulation in the cluster of our University achieved a significant reduction of the processing times (Fig. 4), getting a maximum *speedup* factor of 5.85 (Fig. 5); it has also allowed us to identify an important limitation to the scalability of the proposed method, derived from the need to have under mutual exclusion control the nodes of the simulation grid located in contact zones that separate the data spaces reserved for different threads, which we already had identified in a previous work [15] for a two dimensional simulation. This limitation is typical of the nature of the problem and, therefore, cannot be ignored. The fidelity of the proposed model to the biological reality has also been checked, showing that the simulation achieves a more than acceptable fidelity with respect to the usual behavior in this type of neoplasms. Our future work is focused in:

– the application of the developed model to other glandular neoplasms *in situ*.
– the development of a data partitioning scheme that allows parallel simulations on GPU architectures.

References

1. Altrock, P.M., Liu, L.L., Michor, F.: The mathematics of cancer: integrating quantitative models. Nat. Rev. Cancer **15**, 730–745 (2015)
2. Byrne, H.M., Alarcon, T., Owen, M.R., Webb, S.D., Maini, P.K.: Modelling aspects of cancer dynamics: a review. Philos. Trans. Roy. Soc. **364**(1843), 1563–1578 (2006)
3. Byrne, H.M.: Dissecting cancer through mathematics: from the cell to the animal model. Nat. Rev. Cancer **10**, 221–230 (2010)
4. Davis, M., Sigal, R., Weyuker, E.: Computability, Complexity and Languages. Academic Press, Cambridge (1994)
5. Deutsch, A., Dormann, S.: Cellular Automaton Modeling of Biological Pattern Formation. Birkhäuser, Boston (2005)
6. Edelman, L.B., Eddy, J.A., Price, N.D.: *In silico* models of cancer. Syst. Biol. Med. Adv. Rev. **2**, 438–439 (2010)
7. Enderling, H., Chaplain, M., Anderson, A., Vaidya, J.: A mathematical model of breast cancer development, local treatment and recurrence. J. Theor. Biol. **246**(2), 245–259 (2007)
8. Erbas, B., Provenzano, E., Armes, J., Gertig, D.: The natural history of ductal carcinoma in situ of the breast: a review. Breast Cancer Res. Treat. **97**(2), 135–144 (2006)
9. Gatenby, R.A., Gawlinski, E.T.: A reaction-diffusion model of cancer invasion. Cancer Res. **56**(24), 5745–5753 (1996)
10. Giordano, A., et al.: Parallel execution of cellular automata through space partitioning: the landslide simulation Sciddicas3-Hex case study. In: Proceeding of 25th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), pp. 505–510 (2017)
11. Gevertz, J.L., Gillies, G.T., Torquato, S.: Simulating tumor growth in confined heterogeneous environments. Phys. Biol. **5**(3), 036010 (2008)
12. Kam, Y., Rejniak, K.A., Anderson, A.R.A.: Cellular modeling of cancer invasion: integration of *in silico* and *in vitro* approaches. J. Cell. Physiol. **227**(2), 431–438 (2012)
13. Monteagudo, A., Santos, J.: Studying the capability of different cancer hallmarks to initiate tumor growth using a cellular automaton simulation. Application in a cancer stem cell context. Biosystems **115**, 46–58 (2014)
14. Norton, K.-A., Wininger, M., Bhanot, G., Ganesan, S., Barnardh, N., Shinbrotb, T.: A 2D mechanistic model of breast ductal carcinoma in situ (DCIS) morphology and progression. J. Theor. Biol. **263**(4), 393–406 (2009)
15. Salguero-Hidalgo, A.G., Capel-Tunon, M.I., Tomeu-Hardasmal, A.J.: Parallel cellular automaton tumor growth model. In: Proceedings of 12th International Conference on Practical Applications of Computational Biology and Bioinformatics (to appear)
16. Silva, A.S., Gatenby, R.A., Gillies, R.J., Yunesa, J.A.: A quantitative theoretical model for the development of malignancy in ductal carcinoma in situ. J. Theor. Biol. **262**(4), 601–613 (2009)
17. Sottoriva, A., et al.: Cancer stem cell tumor model reveals invasive morphology and increased phenotypical heterogeneity. Cancer Res. **70**(1), 46–56 (2010)
18. Tomeu-Hardasmal, A.J., Salguero-Hidalgo, A.G., Capel-Tunon, M.I.: Speeding up tumor growth simulations using parallel programming and cellular automata. IEEE Lat. Am. Trans. **14**(11), 4603–4619 (2016)