Agent-Oriented Stem Cell Computational Modeling

Maíra A. de C. Gatti, Geisa M. Faustino, Diego Bispo, José Eurico Vasconcellos, and Carlos J.P. de Lucena

Laboratório de Engenharia de Software - LES
Departamento de Informática – PUC-Rio – RJ – Brazil

{mgatti,gfaustino,jfilho,lucena}@inf.puc-rio.br,
diego@les.inf.puc-rio.br

Abstract. *The stem cell ability of differentiation in different kinds of cells is crucial for therapeutic application. Stem cells can be used to find a cure for several diseases, such as cardiovascular, neurodegenerative, and others. The promise of the use of the capacity of computational power and the applicability of the recent techniques in computational and mathematic modeling in order to understand and forecast important aspects of biological systems is a great opportunity for the research in medicine and biology. In this context, stem cells are biological systems that are open, distributed, and complex natural systems. They produce an emergent behavior and the design and development of the computational modeling of these systems is a non-trivial task that requires by definition a specific software engineering approach, including specific modeling techniques. This paper presents the first experiments for developing a computational modeling for embryonic stem cell behavior using a specific (agent-based) approach, its challenges and targeted expected contributions in a long term. This work is being developed in the context of the Second Grand Challenge of building computational modeling of complex natural systems.*

Resumo. *Devido à capacidade de diferenciação em diferentes tipos de células, as células-tronco têm tido papel cada vez mais relevante como ferramenta terapêutica. São potencialmente úteis em terapias de combate a diversas doenças cardiovasculares, neurodegenerativas dentre outras. A possibilidade de utilização do poder computacional e da aplicabilidade das recentes técnicas de modelagem computacionais e matemáticas para entender e predizer aspectos importantes do comportamento de sistemas biológicos surgem como um novo ferramental para a pesquisa medica e biológica. Neste contexto, células-tronco podem ser caracterizadas como sistemas naturais, abertos, distribuídos, e complexos compostos de múltiplos elementos autonômicos interagindo entre si, e que exibem comportamento emergente. O projeto da modelagem computacional destes sistemas é uma tarefa não-trivial que, por definição, requer abordagens de engenharia de software específicas, incluindo técnicas de modelagem específicas. Este artigo apresenta os primeiros experimentos obtidos na modelagem computacional do comportamento de células-tronco embrionárias utilizando tal abordagem (orientada agentes), seus desafios e contribuições alvo esperadas a longo prazo. Este trabalho se situa no Segundo Grande Desafio da construção de modelos computacionais de sistemas naturais complexos.*

1. Introduction

The aim of the Second Grand Research Challenge in Computer Science [26] stated by the Brazilian Computer Society (SBC) is to create, evaluate, modify, compose, manage and exploit computational models for a variety of domains and applications, ranging from experiments on artificial systems through to those concerning natural or social interactions.

The idea of formulating Grand Research Challenges has been adopted in several countries, in distinct fields. It has been observed that the formulation of these challenges has led to establishing a long-term research agenda, with positive consequences not only in terms of advancing knowledge, but also to educate new generations of researchers. For instance, the In Vivo in Silico Grand Challenge [27] stated by the UK Computing Research Committee and the British Computer Society is to realize fully detailed, accurate and predictive models of some of the most studied life in biology.

In this context, the systems biology community is building increasingly complex models and simulations of cells and other biological entities [15]. This community is beginning to look at alternatives to traditional representations, such as those provided by ordinary differential equations (ODE).

On the other hand, stem cells play a prominent role in biology and life sciences. Their importance is growing more and more, not only in basic research fields such as cell or developmental biology, but also in medicine and clinical research. The main reason underlying this broad interest in stem cells is their capacity to reconstitute functional tissues after disturbance or injury. They are able to produce a huge number of differentiated, functional cells and, at the same time, they maintain or even re-establish their own population.

A stem cell is a primitive cell that can either self-renew (reproduce itself) or give rise to more specialized cell types. The new perspective on stem cell systems as networks of different cell types and their interactions implies that stem ness should not be treated as an explicit cellular property, but rather as the result of a dynamic self-organization process. The microenvironment interactions and their specific effects on proliferation and differentiation have to be embedded in the concept.

Nowadays, stem cells are cultivated in the lab in order to differentiate into a specific mature cell. Today it is hard to predict stem cell behavior under some substances. Moreover, the entire infrastructure necessary to maintain a stem cell culture is very expensive and many stem cells are wasted if the injected substance does not lead the culture to the desired mature cell. Thus, stem cell simulation is a powerful tool for reducing such costs and accelerating the stem cell therapy process.

There have been several attempts to build formal models (for instance, ordinary differential equations [5] and cellular automata [6]) so that predictions can be made about how and why stem cells behave either individually or collectively. Moreover, the model has to consistently explain the broad variety of experimental observations. It is intended to link these macroscopic phenomena to underlying (latent) microscopic mechanisms. To include experimental observations, which describe individual cell behavior, into the analysis, the model must be able to describe single cells as well as cell population behavior.

If the model is sufficiently detailed and accurate, it serves as a reference, a guide for interpreting experimental results and a powerful means of suggesting new hypotheses supporting the physicians and researchers. Moreover, the simulation lets physicians test experimentally unfeasible scenarios and can potentially reduce experimental costs and time (experiments in vitro can last weeks; the same experiments can take just a few minutes if they are done *in silico*). Besides, such a technique avoids ethical problems about the use of embryonic stem cells.

An Agent-Based (AB) simulation is a simulation with many intelligent agents interacting among themselves and with the environment. In a typical AB simulation of social behavior, the agents are the individuals that take rational decisions based on their neighbors' decisions. Very interesting social phenomena have been recently investigated, such as, for example, cooperation [7], social instability [8] and crowd modeling [9].

The great advantage of this modeling technique is that the emergent phenomena can be modeled through very simple rules governing the behavior of each agent. The global effect resulting from the interaction of the individuals is often unpredictable. More specifically, an agent is a high-level software abstraction that provides a convenient and powerful way to describe a complex software entity in terms of its behavior within a contextual computational environment [10]. The dynamic structures present in biological systems can be intuitively represented and efficiently implemented in agent-oriented simulators [11] [12].

However, in order to achieve reuse we need a general framework for the modeling and simulation of the intra-cellular processes and the stem cell proliferation and differentiation, i.e., a set of classes, their relationships and common behavior that can be reused and which speed up the development of different cell and different cell differentiation processes.

This paper presents the first results achieved towards the construction of an agent-based stem cell computational modeling, its challenges and targeted expected contributions.

What are the main challenges?

There are a number of research strands in the Computing Sciences needed to support the aspirations of the Second Grand Challenge. Some strands will work bottom-up, paying great attention to biological data. Other strands will work top-down, studying minimal abstractions capable of generating the phenomena exhibited in vivo. This work is 'middle-out', balancing the desire to be simple, elegant and general with the desire to be faithful to the data. As a largely top down attack to this challenge, we have:

First step: reverse engineer identifying a simple architecture which is capable of exhibiting a wide range of the phenomena observed by experimental stem cell researchers.

Second step: identify a core architecture for modeling cell and stem cell behavior, a second candidate is presented later in this paper. The first one can be found in [1].

Third step: refine the core architecture by attempting to incorporate experimental test.

Fourth step: build a predictive demonstrator to display the core architecture. Three milestone levels of achievement might be:

▪ Exhibiting the phenomena used as data to build the model (good performance on the training set).

▪ Exhibiting other stem cell behavior phenomena known to stem cell researchers (good performance on the holdback set).

▪ Exhibiting new phenomena not presently known to stem cell researchers, as a disease (good performance on the unknown set).

Fifth step: review progress and evaluate the approach.

The main software engineering specific challenges in order to build such a computational model consist of:

1. Modeling the emergent behavior. Emergence can not be reduced into local parts. It must be a result of the local interactions.

2. Verifying the model. It is hard both to construct a verification model and to observe the simulation results considering distribution and emergence.

3. Visualizing the phenomena. There are million of cells and substances to be visualized in a 3D visualization. The visualization must be decoupled from the model and also distributed. Moreover, the movements of local parts are very hard to visualize.

The next section offers a more detailed picture of what and how is to be modeled in the proposed approach, and expands some of the arguments for proceeding in the direction proposed.

We describe three main contributions already achieved towards the first and third challenge: (1) an agent-based framework for the stem cell modeling and simulation; (2) a stem cell behavior 3D visualization tool; and (3) a software engineering approach inserted in the stem cell wet-lab research processes.

Outline

First, Section 2 describes briefly the stem-cell self-organization problem overview. Section 3 highlights the related work. Section 4 presents the agent-based computational modeling developed. Section 5 describes the results achieved. And finally, Section 6 concludes the paper, presents ongoing work and future work in long term.

2. The Stem-Cell Self-Organization Problem Overview

In this section we show the stem cell differentiation and division process. An important entity with an active influence in the stem cell differentiation process is the Niche [14]. The Niche is a microenvironment in which stem cells are found, which interacts with them to regulate their fate and contains proteins that constitute the extra cellular environment. It saves stem cells from depletion, while protecting the host from over-exuberant stem-cell proliferation. The word “niche” can be in reference to the in vivo or in vitro stem cell microenvironment.

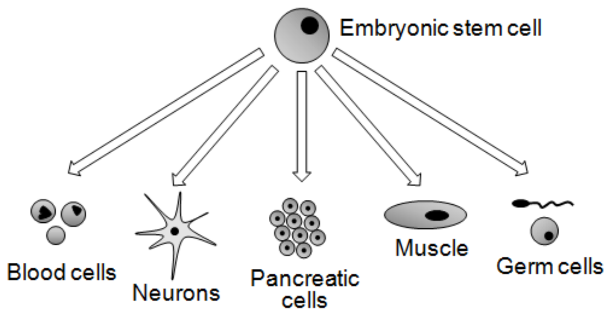


Figure 1. Stem cell differentiation: note the several kinds of cells that the stem cell can be differentiated into.

In developmental biology, cellular differentiation is the process by which a less specialized cell becomes a more specialized cell type [16],[17],[18]. The stem cells can be specialized in several kinds of cells, such as heart cells, skin cells, or nerve cells. Figure 1 shows some of the kinds of cells that a stem cell can differentiate itself into. We are interested in the case of a stem cell that differentiates into a mature neuron.

We based our model layer in on the cell life cycle and stem cell process conceptual model designed with stem cell researchers. An agent-based stem cell conceptual model and a multi-agent-based framework for support applications that simulate the stem cells behavior were developed.

At the agent-based stem cell computational model, there are four kinds of cells: multi-potent cells are cells with a high power of differentiation that can give rise to several other cell types; precursor cells are cells be that are able to self- differentiate into a specific kind of cell, for instance, a blood cell; progenitor cells are stem cells that have developed to the stage where they are committed to forming a particular kind of new specific cell; differentiated cells are specialized cells with no power of differentiation.

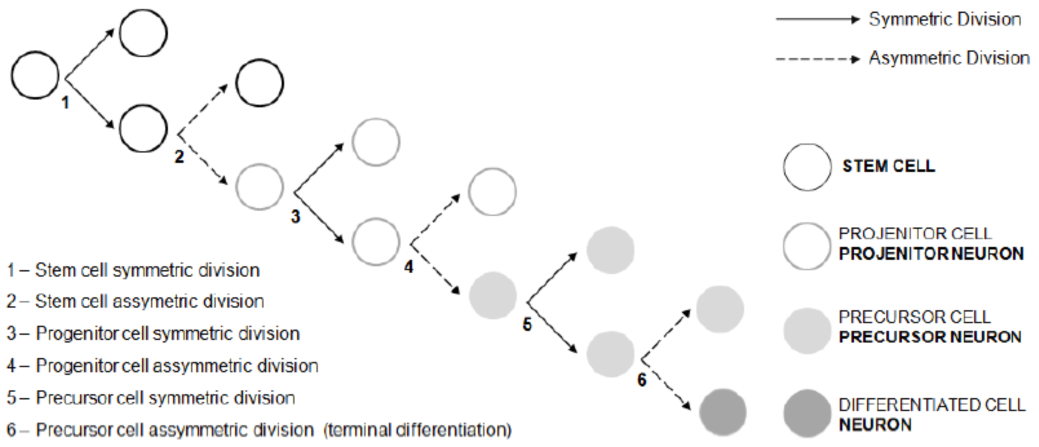

Symmetric Division
Asymmetric Division
STEM CELL
PROJENITOR CELL
PROJENITOR NEURON
PRECURSOR CELL
PRECURSOR NEURON
DIFFERENTIATED CELL
NEURON
1 – Stem cell symmetric division
2 – Stem cell assymetric division
3 – Progenitor cell symmetric division
4 – Progenitor cell assymmetric division
5 – Precursor cell symmetric division
6 – Precursor cell assymmetric division (terminal differentiation)

Figure 2. Stem cell process: on the right, the cells present in the process until the mature cell and (in bold) the cells present until the neuron generation; and on the left, all division steps until the mature cell generation.

In the case that the differentiation process generates a neuron we have: a stem cell like a multi-potent cell; a precursor neuron like a precursor cell; a progenitor neuron like a progenitor cell and a neuron like a differentiated cell - for instance, that would be a neuroblast and the neuron itself, respectively. Figure 2 shows the stem cell division process until the mature cell generation with no differentiation potentiality and for the case when a neuron is generated.

3. The Stem Cell Computational Model

In cooperation with stem cell researchers, we developed an agent-based solution to model and simulate the stem cell processes and the internal cell life-cycle and we considered it as first-order abstraction that simplifies the intracellular processes and stem cell behavior.

We used some AUML Diagrams [2] to help us in the designing of some structures and dynamics, such as the agent use case diagram, agent class diagram and agent interaction diagram. We developed our framework on top of MASON, which is a fast discrete-event multi-agent simulation library core in Java, designed to be the foundation for large custom-purpose Java simulations, and also to provide more than enough functionality for many lightweight simulation needs. MASON contains both a model library and an optional suite of visualization tools in 2D and 3D [3]. Both layers (model and visualization) are independent; therefore, it is possible to run the simulation without visualizing it.

Our agent-based cell computational model is a distributed autonomous entity composed of reactive and pro-active agents. It can perceive and signal the environment. To determine how cells behave and interact, we need to understand how information is transferred among and within cells and how it changes the environment and other cells states (cellular data and hypotheses, and qualitative modeling).

In order to simplify the model we considered only the more active components during the cell life-cycle and the mitosis division (which is the stem cell division during the self-renew process). By actives components we mean components that more directly contribute and influence the cellular differentiation during the cycle. Cells, proteins, DNA and complexes such CDKs-Cyclins are agents. The cells can be a proliferative cell or non-proliferative cell. The stem, progenitor and precursor cells are proliferative, while the neuron cells are non-proliferative. CDK, Retinoic Acid, Cyclin and LIF are proteins.

All the cell life-cycle concepts were implemented in a multi-agent-based framework for the cell simulation. On top of this framework we developed the stem cell behavior framework. We needed a framework because there are different kinds of stem cells providing for a broad spectrum of proliferated progenitor cells that ultimately lead to the creation of every adult cell type necessary for sustaining life (see Figure 1, pg. 3).

Those frameworks allow the reuse of structural relationships and dynamic interactions modeling and design of different cells types within those entities and different cellular process and also the reuse of the stem cell processes modeling, which can be instantiated to different differentiation processes rather than only to the neuron generation that is our instance. It also allows us to integrate different signal transduction pathways, and speed the reverse engineering of bio-molecular regulatory networks through those integrations.

Figure 3 shows the Stem Cell and Cyclin agent classes. The Stem Cell agent plays the Stem Cell role. For instance, it has the *volume* attribute, which is updated when the cell synthesizes substances, and the *potentiality* attribute, which contains how many times the cell can still differentiate into until it reaches a mature cell. The *Execute Cell Cycle* Protocol is responsible for describing all the messages exchanged during the cell cycle execution and, for instance, the *Execute G2* Protocol is responsible for describing all the messages exchanged when the cell is in the Gap 2 phase during the cell life cycle. Notice that the Cyclin agent class also contains the *Execute G2* Protocol, which is the same in the Stem Cell agent class and shows the Cyclins' roles existent in this protocol.

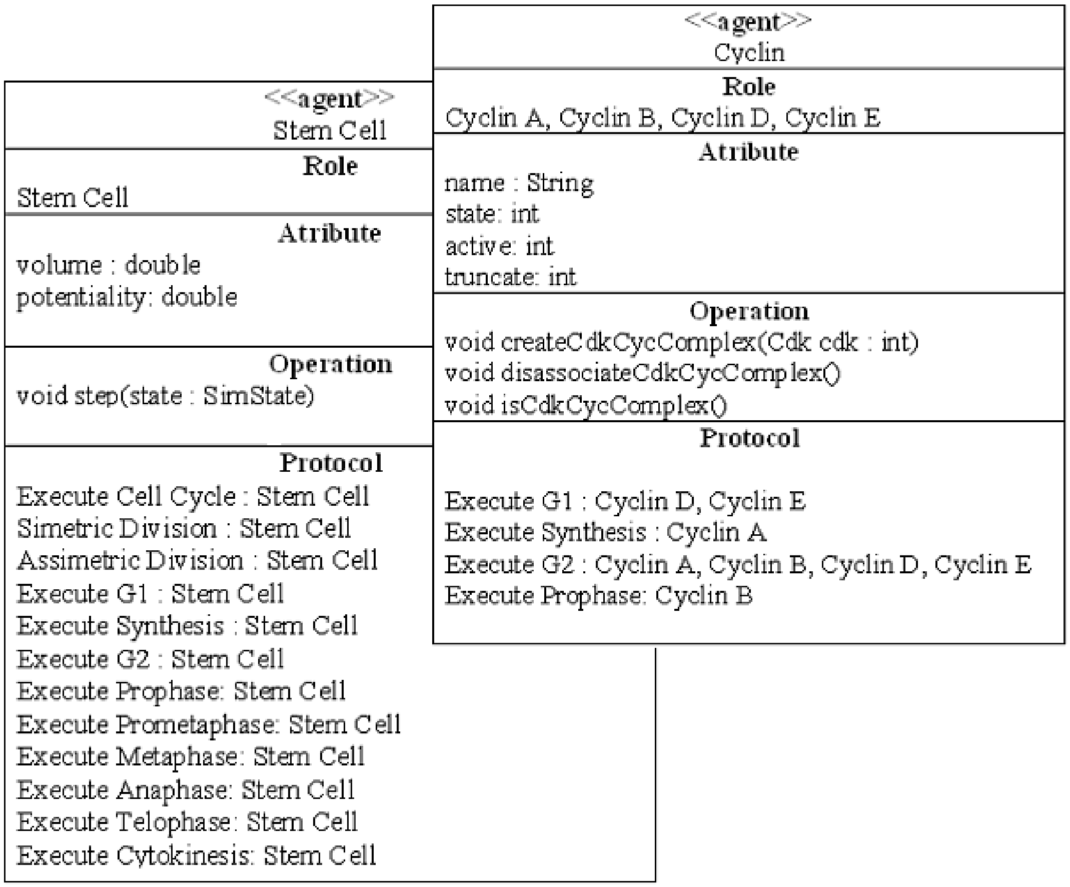

<<agent>>
Stem Cell
Role
Stem Cell
Atribute
volume : double
potentiality: double
Operation
void step(state : SimState)
Protocol
Execute Cell Cycle : Stem Cell
Simetric Division : Stem Cell
Assimetric Division : Stem Cell
Execute G1 : Stem Cell
Execute Synthesis : Stem Cell
Execute G2 : Stem Cell
Execute Prophase: Stem Cell
Execute Prometaphase: Stem Cell
Execute Metaphase: Stem Cell
Execute Anaphase: Stem Cell
Execute Telophase: Stem Cell
Execute Cytokinesis: Stem Cell
<<agent>>
Cyclin
Role
Cyclin A, Cyclin B, Cyclin D, Cyclin E
Atribute
name : String
state: int
active: int
truncate: int
Operation
void createCdkCycComplex(Cdk cdk : int)
void disassociateCdkCycComplex()
void isCdkCycComplex()
Protocol
Execute G1 : Cyclin D, Cyclin E
Execute Synthesis : Cyclin A
Execute G2 : Cyclin A, Cyclin B, Cyclin D, Cyclin E
Execute Prophase: Cyclin B

Figure 3. The Stem Cell and Cyclin Agent Classes (partial view)

Basically, the framework uses four design patterns [4]. The pattern Prototype is used to implement the stem cell's substances because this facilitates the objects' clone process. The pattern State is used to implement the states the cell can be in. The pattern Strategy is used to implement the cell's asymmetric division strategy. Each type of cell has one distinct asymmetric division strategy. The singleton is used to implement the niche, because the niche is unique.

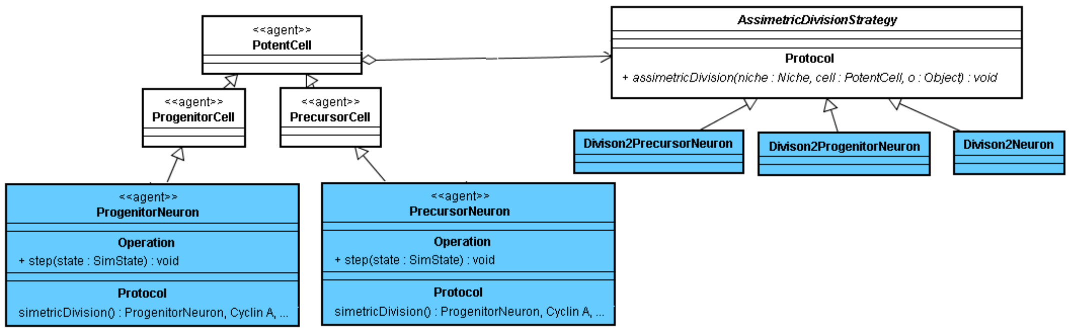

<<agent>>
PotentCell
AssimetricDivisionStrategy
Protocol
+ assimetricDivision(niche : Niche, cell : PotentCell, o : Object) : void
<<agent>>
ProgenitorCell
<<agent>>
PrecursorCell
Divison2PrecursorNeuron
Divison2ProgenitorNeuron
Divison2Neuron
<<agent>>
ProgenitorNeuron
Operation
+ step(state : SimState) : void
Protocol
simetricDivision() : ProgenitorNeuron, Cyclin A, ...
<<agent>>
PrecursorNeuron
Operation
+ step(state : SimState) : void
Protocol
simetricDivision() : ProgenitorNeuron, Cyclin A, ...

Figure 4. Hot spots' framework (partial view)

Briefly, the first hot spot that must be implemented is the environment where the cells represented by the agents live: the niche. Other important hot spots are the progenitor and precursor cells, which must be instantiated depending on the process for a specific type of cell to be simulated. It is necessary to add features and behaviors depending on the type of cells to be instantiated. The strategy for asymmetric divisions from stem cells, progenitor (neurons) and precursor (neurons) cells also must be instantiated (Figure 4).

We are interested in a 3D visualization of the differentiation and division processes for the neuron generation in a higher level of detail. The visualization in a macro level is fundamental for analyzing the differentiation process. Sphere primitives represent the cell entities. Each kind of cell in the conceptual model is represented by different colors. Each state of the cell life cycle is represented by different tones of the same color and some of them by different sizes.

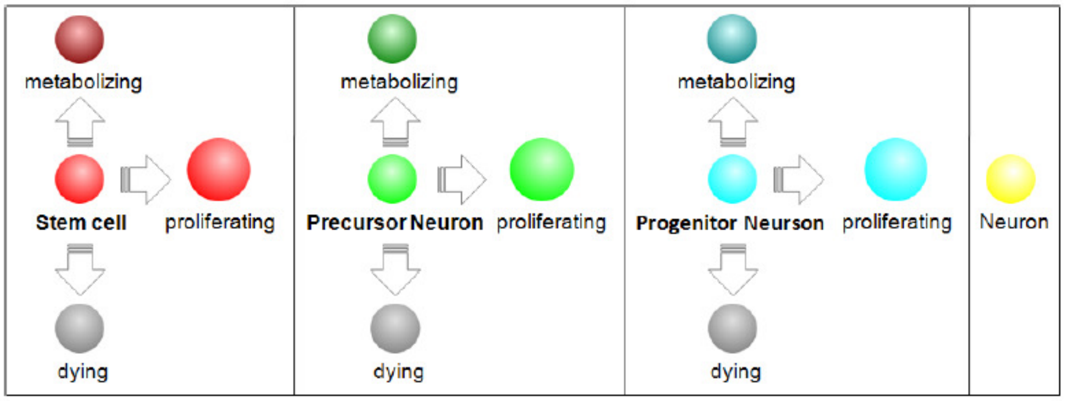


Figure 5. Cell visualization representation

Figure 5 shows how we represented the cells. Note that not all states of the cell life cycle are visualized. Nevertheless they are represented in the conceptual model. Also note that the proliferation state is represented by a larger sphere. This is a more faithful representation of the entities.

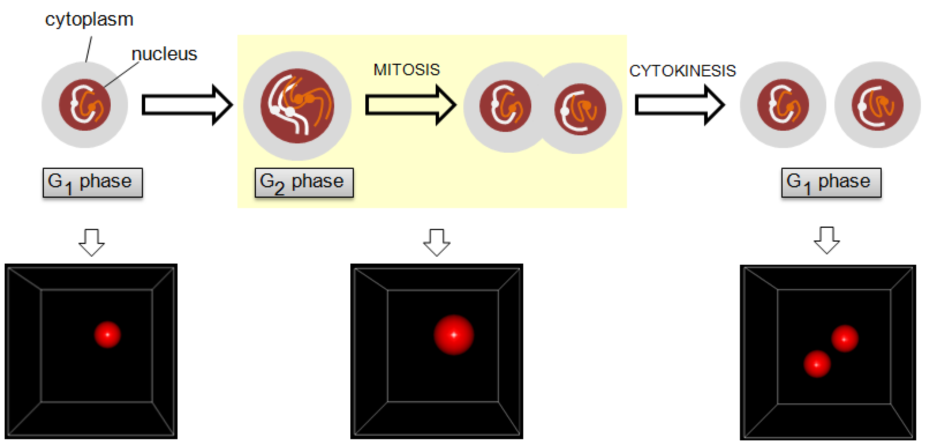


Figure 6. Division process: on the left, the figures represent the division steps in real life; on the right the figures are their representation in the system.

The strategy to simulate the spatial self-organizing developed in the previous system was reformulated and extended to a 3D space. The real life, spatial self-organization of stem cells is roughly explained as follows. The stem cell divides itself into two news cells, due to the mitoses division process (Figure 6). Both cells assume new positions: one assumes the parent position and the other an adjacency position. Before the division process, the cell grows up, pushing its neighbors away and occupying the required space for the new cell.

In order to model this spatial self-organization we must find the best direction in which the new cell is going to push into its neighborhoods. The best direction criterion is to determine out of all possible directions the one that minimizes the cell effort. Choosing the closest empty cell grid in its neighborhoods minimizes the cell effort. If there is more than one possible direction, we choose randomly among the possible directions.

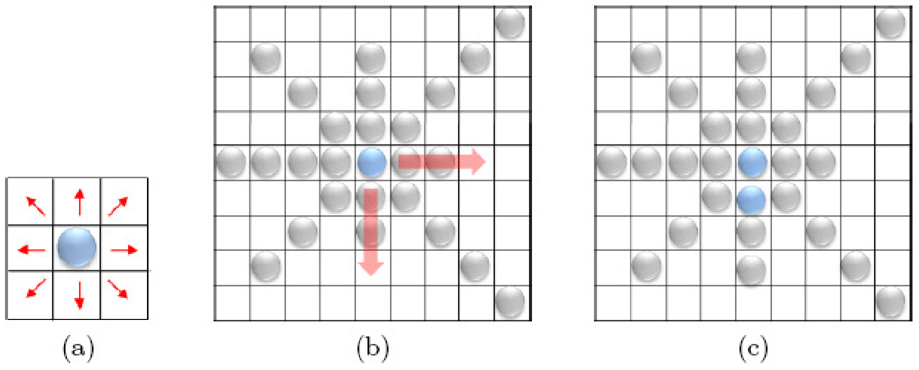


Figure 7. 2D spatial self-organizing simulation strategy: (a) it shows the 8 possible directions in a 2D space; (b) it shows an example with two possible directions in which the cell can move; (c) it shows the space reorganized after the south direction was selected.

The niche is now extended to a 3D grid. Figure 7 shows the 2D case in which there are 8 possible adjacency directions. In the 3D space, there are 26 possible directions. It is more than three times the number of directions in the 2D space, which makes the problem computationally intensive. Note that the problem was already an exponential complexity time due to the cell division process that duplicates the number of agents at each iteration.

The spatial self-organizing strategy is described as follows: through the cell's current position, represented by a triple (x, y, z), which is being divided, we find which among 26 possible directions minimizes the cell effort. The 26 possible directions are represented by canonical base (e.g. the north direction is represented by (1, 0, 0); the southwest direction is represented by (1,–1, 0) and so on). Once the best direction is defined, the cells are shifted away by 1 unit in the defined direction. So, the new cell is inserted in the free adjacent cell grid.

Let $\mathbf{d_k}$ represent the number of possible directions in dimension k (e.g. when k = 3, the $\mathbf{d_k}$ is 26). The best case of the algorithm occurs when the cell that is dividing itself is located in the center of the empty grid and it visits all empty adjacent cell grids, that is, it runs, in at least, ($\mathbf{d_k}$). The worst case occurs when the cell that is dividing itself is located in the center of the full grid and it visits the cells grids up to the limit of the grid in all $\mathbf{d_k}$ directions, that is, it runs, at most O(max{width, height, length}).

The objectives of the development of a stem cell behavior simulator also include: the offer to the users of a visual environment through which it is possible to follow the Macro and Micro levels of the simulation of stem cell's cellular life cycle in the niche (Figure 8); the perception of the difficulties of implementation of the proposed model; the validation of the model, observing if the simulated behavior has similarities with the behavior of the real entities (stem cell's); and the testing of several scenarios.

We understand as Macro scale the emergent behavior preceding the interactions between the simulated entities. The simulator presents this scale to the users by means of a visualization area (3D) that represents the niche where the cells evolve in their life-cycles.

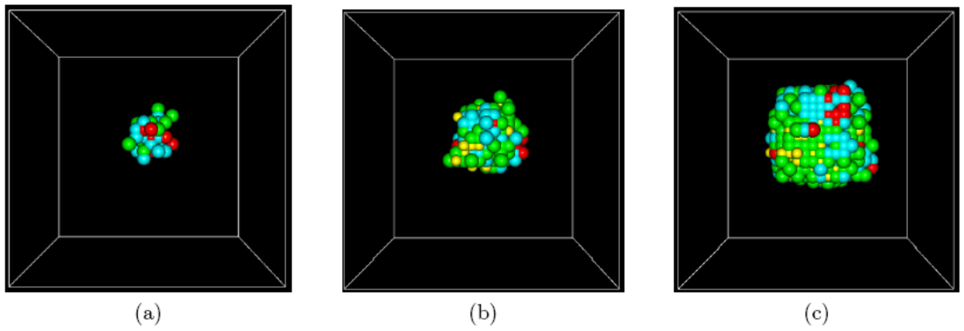
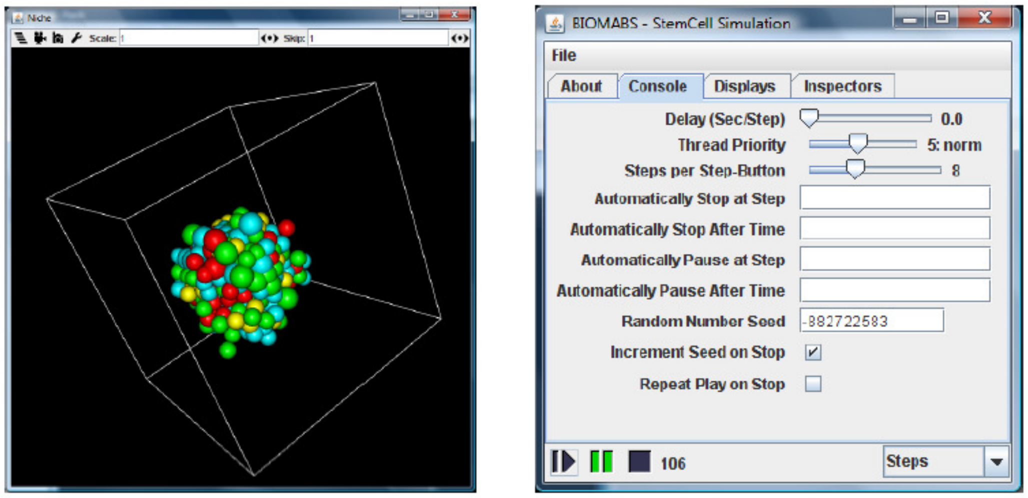


Figure 8. Screenshots of a simulation: (a) at the initial; (b) the middle; (c) and at the end of simulation. Note the exponential growth of the number of cells.

In another scale, each phase of cell life-cycle has a 2D graphical representation, presenting the state of the main entities involved in the process. These graphical representations, besides presenting a phase of the life-cycle, differentiate the capacity of differentiation of the cell by means of colors. As Micro scale we understand the state and behavior of each entity simulated individually. In the simulator tool it is possible to obtain the micro scale through the internal state of the cell selected and chronologically each internal interactions occurred by the selected cell. Fig. 9 shows a snapshot of the interface of the framework instance during a simulation. In the Macro scale, emergent behavior can be seen in the visualization area.

The GUI interface provides a console panel that contains useful settings for the simulation; an image and video capture; and 3D navigation tools. The specialists can speed, pause, play, stop or update some simulation settings. The console panel provided three slides: the first allows the insertion of some delay time between each time step (in case the simulation is too fast); the second increases the priority of the simulation thread; and the third allows execution of any number of steps upon pressing the play button (when the simulation is paused). Image and movie capture tools are helpful to visualize the results of the simulation, without having to repeat the processing. 3D navigation tools (e.g. Arcball [23]) and zoom controls permit customized visualization of the simulation.



Delay (Sec/Step) 0.0
Thread Priority 5: norm
Steps per Step-Button 8
Automatically Stop at Step
Automatically Stop After Time
Automatically Pause at Step
Automatically Pause After Time
Random Number Seed -882722583
Increment Seed on Stop ✓
Repeat Play on Stop ☐
106
Steps

(a) (b)

Figure 9. Screenshot of the user interface of the system: (a) it shows the display window and (b) it shows the console window.

4. The results

In order to execute the instantiation for the generation of neuron cells, a computer with the following features was used: Intel(R) Core (TM) 2 CPU 6300@ 1.86 GHz, 2GB of RAM. We execute the simulation in a 3D grid with dimensions 50×50×50 that allows visualizations of up to 125,000 cells.

During our sessions of test, the collaborators from the Neurogenesis and Cell Differentiation Lab (UFRJ) observed the first emergent phenomenon in the visualization tool, which was similar to the emergent phenomenon in vitro: the differentiated cells are located in the embryonic body's extremity while the specialized and stem cells are located at the embryonic body's center.

They evaluated the model and made some corrections, which we still need to work on. They requested inputting the initial number of stem cells in the simulator and the inputting of Retinoic Acid concentration. They also requested the building of some graphics showing the number of each cell variation over time and considering the Retinoic Acid concentration. Because optimizing the number of neurons during the differentiation process is an important issue for them, they need to follow these parameters during the simulation.

In addition, they want to investigate how aneuploidy[1] contributes to neuron generation. Hence we need to evolve the model in order to provide the internal cell visualization showing the chromosomes, and also to quantify the chromosomes in the cells and present this in the graphics.

5. Conclusions and Future Work

We have being developing a stem cell computational model. If the model is sufficiently detailed and accurate, it serves as a reference, a guide for interpreting experimental results and a powerful means of suggesting new hypotheses supporting the physicians and researchers. Moreover, the simulation will let physicians test experimentally unfeasible scenarios and can potentially reduce experimental costs and time (experiments in vitro can last weeks; the same experiments can take just a few minutes if they are done *in silico*). Furthermore, such a technique avoids ethical problems about the use of embryonic stem cells. The building of this model is inserted in the Second Grand Challenge in Computer Science which is to create, evaluate, modify, compose, manage and exploit computational models for a variety of domains and applications, ranging from experiments on artificial systems through to those concerning natural or social interactions and the stem cell behavior is a natural interaction system.

The stem cell computational model developed is based on an agent-based framework that can be reused for simulating different kinds of cells and different cellular processes rather than only stem cell behavior. Not only can we reuse all the already developed modeling but also we are able to model learning, adaptive behavior and open systems. There is a 3D visualization tool and the framework can be instantiated to different differentiation processes rather than only to neuron generation.

The stem cell researchers' collaborators were very excited about the first results. We already are working on the issues they raised and presented in last section.

Although the number of cells running together and the time of execution in the simulation were satisfactory, we need to increase this number and to achieve this goal we are distributing the framework and application in a grid architecture with ten processors. If we have around 20,000 cells we can reach more refined self-organizing mechanisms that might occur in such systems.

Besides the indicated ongoing and future work, an important fundamental engineering issue is to achieve a macroscopic behavior that meets the requirements and emerges only from the behavior of locally interacting agents when designing self-organizing emergent multi-agent systems. To date, agent-oriented methodologies are mainly focused on engineering the microscopic issues, i.e. the agents, their rules, the protocols, how they interact, etc, without explicit support for engineering the required macroscopic behavior. As a consequence, the macroscopic behavior is achieved in an ad-hoc manner. A fundamental problem is the lack of a method that allows us to systematically specify desirable macroscopic properties, map them to the behavior of individual agents, build the system, and validate the required macroscopic properties. So we are working on a method and a representation model to handle this issue.

[1] An aneuploid is an individual organism whose chromosome number differs from the wild type by part of a chromosome set. Generally, the aneuploid chromosome set differs from wild type by only one or a small number of chromosomes.

Also, as an ongoing work, we have being defining a verification method to apply in this project supported by a tool. This tool will be a framework that would be used not only for this application domain but to self-organizing systems in general. Through this tool, we also plan to optimize the emerging behavior generated by the self-organizing stem cell represented by agents. By optimization we mean the establishment of optimum differentiation or proliferation rates, for instance through the addition and removal of some specific factors in the niche. Hence, the challenge would be to define macro properties and, starting from local interactions, to integrate a specialized online search planner to optimize the behavior so that the macro properties can be satisfied. Therefore, the simulator might allow more interactions with the simulation environment, increasing the tool usability and dependability, as well as helping the validation process vis-à-vis the in vitro process.

Considering the three milestone levels of achievement presented in first section, the results already achieved are in the first milestone - exhibiting the phenomena used as data to build the model (good performance on the training set). So, we are still working on refinements on this level and also on the second level (exhibiting other stem cell behavior phenomena known to stem cell researchers).

We plan as a long term research result, exhibiting new phenomena not presently known to stem cell researchers, as a disease (good performance on the unknown set), and which is the targeted and main difficult goal.

From this first year of experience, we noticed how important is the multidisciplinary that has to be fostered not only between Computer Science and the stem cell research, but also within Computer Science itself. The software engineers need to cooperate with researchers in human-machine interaction, computer graphics, scientific visualization, artificial intelligence, grid computing, and other areas necessary to solve the challenges. All of these people must themselves interact continuously with researchers also.

Acknowledgments. We wish to acknowledge the team of collaborators in this project, entitled LANDIC, which is the Neurogenesis and Cell Differentiation Lab, including stem cell researcher Stevens Rehen. The team helped us in the conceptual model validation, with some discussions around the research hypothesis, and will continue to work with us in the future. This work was supported by MCT/**CNPq** through the "*Grandes Desafios da Computação no Brasil: 2006-2016*" (Main Computation Challenges in Brazil: 2006-2016) Project (Proc. CNPq 550865/2007-1).

References

[1] Gatti, M., de Vasconcellos, J.E., Lucena, C.J.P.; An Agent Oriented Software Engineering Approach for the Adult Stem-Cell Modeling, Simulation and Visualization. Workshop SEAS, João Pessoa, 2007.

[2] Gatti, M., von Staa, A., Lucena, C.; AUML-BP: A Basic Agent Oriented Software Development Process Model Using AUML; Monografias em Ciência da Computação, Departamento de Informática, PUC-Rio, No. 21/07, 25 pg., 2007.

[3] Luke, S., Cioffi-Revilla, C., Panait, L. and Sullivan, K. (2004), 'MASON: A New Multi-Agent Simulation Toolkit', SwarmFest 2004, Eighth Annual Swarm Users/Researchers Conference, University of Michigan, Ann Arbor, Michigan USA.

[4] Gamma, E., Helm, R., Johnson, R., Vlissides, J. "Design Patterns: Elements of Reusable Object-Oriented Software." Reading, MA: Addison Wesley, 1995.

[5] Lei, J., Mackey, M.C. (2007). "Stochastic differential delay equation, moment stability, and application to hematopoietic stem cell regulation system," SIAM J. Appl. Math. (2007), 67(2), 387–407.

[6] Agur, Z., Daniel, Y. and Ginosar, Y. (2002). The universal properties of stem cells as pinpointed by a simple discrete model. Mathematical Biology, 44:79–86, 2002.

[7] Axelrod, R. (1997) The Complexity of Cooperation (Princeton Univ. Press, Princeton).

[8] Epstein, E. M. (2002) Proc. Natl. Acad. Sci. USA 99, 7243–7250

[9] Helbing, D., Farkas, I. & Vicsek, T. (2000) Nature 407, 487–490.

[10] Jennings, N., R. (2000). On Agent-Based Software Engineering. Artificial Intelligence Journal, 117 (2) 277-296, 2000.

[11] d'Inverno, M. and Prophet, J. (2004). Modeling, simulation and visualisation of adult stem cells. In P. Gonzalez, E. Merelli, and A. Omicini, editors, Fourth International Workshop on Network Tools and Applications, NETTAB, pages 105–116, 2004.

[12] d'Inverno, M. and Saunders, R. (2005). Agent-based modeling of stem cell organisation in a niche. In Engineering Self-Organising Systems, volume 3464 of LNAI. Springer, 2005.

[13] d'Inverno, M., Theise, N. D. and Prophet, J. (2005). Mathematical modeling of stem cells: a complexity primer for the stem cell biologist. In C. Potten, J. Watson, R. Clarke, and A. Renehan, ed., Tissue Stem Cells: Biology and Applications. Marcel Dekker.

[14] Scadden, D. T. (2006). The stem-cell niche as an entity of action. Nature 441, 1075-1079 (29 June 2006) | doi:10.1038/nature04957; Published online 28 June 2006.

[15] Vogel, H., Niewisch, H., and Matioli, G.; J. Theor. Biol., 22(2):249-270, 1969.

[16] Loeffler, M. and Grossmann, B.; J. Theor. Biol., 150(2):175-191, 1991.

[17] Loeffler, M. and Roeder, I. Cells Tissues Organs, 171(1):8-26, 2002.

[18] Lord, B.I.; in Stem cells, Cambridge Academic Press, 1997, pp 401-422.

[19] Wichmann, H. E. and Loeffler, M.; Mathematical modeling of cell proliferation: Stem cell regulation in hemopoiesis, CRC Press, Boca Raton, 1985.

[20] Metropolis, N. and Ulam, S. (1949). The Monte Carlo Method. J. Amer. Stat. Assoc. 44, 335-341, 1949.

[21] Theise, N. D. and d'Inverno, M. (2003). Understanding cell lineages as complex adaptive systems. Blood, Cells, Molecules and Diseases, 32:17–20, 2003.

[22] Weiss, G.; Multi-Agent Systems: A Modern Approach to Distributed Artificial Intelligence. (ed). MIT Press, 2001. Chapter 4, pp. 165-199.

[23] Ken Shoemake. Arcball rotation control, pages 175{192. Academic Press Professional, Inc., 1994.

[24] M. Spivey. The Z Notation (second edition). Prentice Hall International: Hemel Hempstead, England, 1992.

[25] T. De Wolf, T. Holvoet; Towards a Methodology for Engineering Self-Organising Emergent Systems; Self-Organization and Autonomic Informatics (I), Volume 135 of Frontiers in Artificial Intelligence and Applications; H. Czap et al. (Eds.); IOS Press, 2005.

[26] Workshop Report on Grand Challenges in Computer Science Research in Brazil: 2006-2016. Brazilian Computer Society (SBC), 2006.

[27] UK Computing Research Committee and the British Computer Society. iViS - in Vivo - in Silico: The Virtual Worm, Weed and Bug Breathing Life into the Biological DataMountain. A Grand Challenge For Computational Systems Biology., 2004.