

Multiagent Modelling of a Bacterial Cell, a DnaA Titration Model Based Agent Model as an Example

Taivo Lints

Tallinn University of Technology, 19086 Tallinn, Estonia
taivo@vkg.werro.ee, <http://www.dcc.ttu.ee/taivo>

Abstract: In this paper it is proposed that two current trends – agent-oriented programming in computer science and whole organism models in biology – are likely to develop strong ties between each other. To back up this claim and to show the feasibility of using multiagent systems for modelling organisms, including the single-celled ones, a small multiagent model of a bacterial cell is developed and shown to give adequate results in simulations. The model is based on DnaA Titration Model from biology and deals with DNA replication and cell division.

1 Introduction

Computer science has progressed through several programming paradigms – procedural programming, data hiding using modules, data abstraction using abstract data types, and finally object-oriented programming – every new paradigm „embracing“ previous ones and adding something new. However, this progression, including object-orientedness, is well described already decades ago, e.g. by Stroustrup (1988). No new paradigms have gained enough weight yet to be added to this list.

Looking at these four programming approaches, it can be seen that the direction of advances is towards dividing the system into smaller, more autonomous pieces. One of the main reasons behind this trend is probably the ever-increasing complexity of software – it just can not be reasonably managed as a single huge chunk of code. The pressure to find even better paradigms than object-oriented programming is building up, both because the size of practical applications is growing fast and because more and more attention is paid to the interdisciplinary studies of complex adaptive systems, which rely heavily on concepts like emergent behavior, interactions, autonomy, etc. Luckily there already exists a programming approach that supports these ideas to some extent – the agent-oriented programming – and its popularity is on the rise.

Meanwhile in biology, the goals of computer modelling are slowly starting to shift from simulating a few isolated processes to creating the models of whole organisms. Even a new academic (sub)field, based on this trend, has emerged: systems biology.

The two trends – agent-oriented programming and whole organism models – are likely to develop strong ties between each other, because multiagent systems seem to be very suitable for modelling organisms. The purpose of this paper is to demonstrate that agent-based models are good not only for simulating large animals (where agents

could, for example, represent cells), but also for simulating unicellular organisms like bacteria. The paper is organized as follows: section 2 explains why agents are good for modelling intracellular processes, section 3 describes the relevant works, section 4 presents the DnaA Titration Model based agent model and the results from simulations, and section 5 concludes the paper.

2 Why use Agents?

Making models of intracellular processes is nothing new for biology and obviously a question immediately rises: why use agents when there already exists a well-developed mathematical apparatus for this?

2.1 Models of metabolism in biology

According to Gombert and Nielsen (2000), current mathematical models of cellular metabolism can be divided into two groups:

- *Stoichiometric models*, which are based on time invariant characteristics of metabolic networks and typically use metabolic flux analysis or metabolic network analysis. The predictive power of stoichiometric models is limited as they do not include information about many regulative processes that affect metabolism.
- *Kinetic models*, which add information about enzyme or microbial kinetics to stoichiometric models and therefore allow, at least in principle, quite detailed and predictive modelling of cell dynamics. The main problem with kinetic models is the need for very detailed information about the intracellular processes: kinetic models mostly consist of systems of differential equations which have many parameters that need fine-tuning for the model to work.

A good model of the whole bacterium should include the dynamical information but, as noted, typical kinetic models based on differential equations need extremely detailed information that we currently do not have. And even if it were available, the use of large systems of differential equations would need great skills in mathematics and the resulting model would hardly be intuitive and understandable to an average modeller. The agent-based approach would reduce these (and other) problems as explained in the following.

2.2 The advantages of multiagent systems

An agent, in general, is a system with the following properties (Wooldridge 1998):

- *Autonomy*: agents can make decisions about what to do without direct external intervention of other systems.
- *Reactivity*: agents are situated in an environment, can perceive it (at least to some extent) and are able to respond to the changes in it (i.e. are able to react).

- *Pro-activeness* (or *proactivity*): agents do not simply react to changes in the environment, but are also able to take the initiative.
- *Social ability*: agents can interact with other agents and participate in social activities.

In agent-oriented programming an agent is usually a piece of *software* possessing these four properties and many such agents acting together form a multiagent system. While it is possible to create systems with these properties also in object-oriented (OO) languages (and many agent development tools are indeed created this way), the OO paradigm does not *support*, but merely *enables* this. Using agent-oriented software development environments facilitates the creation of agent systems by letting the developer to concentrate mostly on the problem under study, eliminating or at least reducing the need to worry about how to implement agent communication, multi-threading, etc.

The advantages of multiagent systems in the context of whole-cell models:

- The possibility to have a direct correspondence between the parts of the biological system and the parts of the model. For example a ribosome can be represented as an agent called Ribosome, with all the interesting reactive, proactive and “social” behavior of the real ribosome. This approach is much more intuitive than using complex mathematical apparatus and it boosts first the ease of construction of the model and then the understandability of the result.
- The model is more flexible. Due to the autonomous nature of agents it is relatively easy to add, remove and modify them, sometimes even without stopping the running simulation. In addition, multiagent systems can be distributed over many networked computers without extreme effort, making it easier to cope with the need for high computing power. Also the possibility to represent different parts of the cell with different levels of abstraction adds to the flexibility of the model (e.g. in the same simulation one agent may represent the whole cytoplasm while another may represent a single molecule floating around in it).
- As multiagent systems are based on the concepts of interactions and parallelism, it is quite easy to produce emergent behavior in the model (a behavior that is not explicitly written into the model's description, but instead appears during the simulation runs and is therefore of high interest to the modeller – if only prescribed processes were present in the simulation, then in principle all the behavior of the simulation would be known to the modeller in advance and computers would only help to conduct the otherwise tedious mathematical work and to visualize the system).

3 Relevant works

The idea of modelling a single biological cell as a multiagent system is not very widespread yet and only a fairly limited number of papers has been published. However, as agent-oriented programming is gaining popularity in different fields of science, it is likely that more and more cell biologists will get acquainted with it and the number of

relevant works will start to climb fast soon. The idea of one agent representing one cell has already gained some momentum in the research community, but it is not relevant enough in the context of this paper as there the emergent behavior will appear on the level of the population of cells, not on the level of a single cell. In the remainder of this section short descriptions of works dealing with cell-level multiagent models is given.

Somewhat surprisingly a more than a decade ago published paper by Paton (1993), which discusses several different interesting possibilities for describing a cell, already proposes the idea: “Cells can be described in this way, as open systems (Huberman and Hogg, 1988) which contain collections of autonomous computational agents interacting with each other. These open systems exhibit parallel distributed processing in that different parts of the cell do different things and the adaptive capabilities of the system are reflected in its data driven capacities, considerable degree of lack of global control, fault tolerance, high degree of communication and ability of multiple parts to carry out partial computations.” While there are no computer implementations offered in that paper, the later works of Paton and his colleagues are moving in this more practical direction, see e.g. Fisher *et al.* (1999) and Fisher *et al.* (2000).

A system called Cellulat developed by Gonzalez *et al.* (2003) represents proteins and other components participating in intracellular signalling as “internal autonomous agents” and the communication with external medium takes place through “interface autonomous agents”. All communication between agents is indirect, through the use of a shared “blackboard”. The blackboard has different levels corresponding to different static and nested cellular compartments, and the objects on the blackboard represent various intracellular signalling molecules and the like.

A paper by Alur *et al.* (2002) presents a hybrid system where each agent is characterized by a continuous state x and a collection of discrete modes. The changes of x are governed by the set of ordinary differential equations of the currently active mode. There are two types of agents: process agents or P-agents, which capture the dynamics involved in transcription, translation, protein binding, protein-protein interactions, and cell growth; and system agents or S-agents, which describe the accumulation or degradation of species (proteins, cells, DNA) in terms of concentration or simply numbers. P-agents take the quantities (concentrations or numbers) of different species relevant to the process from S-agents as inputs and calculate the rates of the reactions. S-agents take these rates and update the quantities of species.

Katara and Venkatasubramanian (2001) are studying the behavior of microbes in a binary substrate environment where both glucose and lactose are present and the microbes tend to eat the more nutritional substrate – glucose – first (the “glucose effect”). The cell in their model consists of one Nucleus agent, one Environment agent (which represents the protoplasm, i.e. the *inner* environment, of the cell) and of many Cellular-Organelle agents (all different organelles are generalized into one type of agent). The Cellular-Organelle agents can be either in mode A or mode B and, respectively, get the substrates S1 or S2 from the Environment and the time resources T1 or T2 from the Nucleus. Using these resources they produce biomass, which is essential for the survival of the organelles and the cell in general. Cellular-Organelles can divide into two new agents of the same type and either the same mode or, due to mutations, the opposite mode. In certain cases they can also change their mode by mimick-

ing the most productive organelles. The purpose of the model is to study the evolutionary emergence of the glucose effect.

Burleigh *et al.* (2003) use a swarm-based approach (basically an agent-based approach) for modelling the processes related to the *lac*-operon (a system of genes regulating the transport and metabolization of lactose). Every element in the simulation is an agent governed by simple rules of interaction. Dynamic elements in the system move randomly within the cell and execute specific actions when interacting with other agents. The main focus of the work is 3-dimensional visualization of intracellular processes, including compatibility with an immersive stereoscopic 3D environment called *The CAVE Automated Virtual Environment*.

Querrec *et al.* (2003) propose a model of Mitogen-Activated Protein Kinase (MAPK) pathway where agents represent biochemical reactions. Agent-reactions have a three-stroke cycle of behavior which consists of perception, decision and action. During perception each agent reads from the environment the concentrations of all the elements intervening in this reaction. Then, in the decision phase, the agent calculates the quantity of each reagent which will appear or disappear in the reaction. And finally the action is taken, i.e. the concentrations of reagents and products are updated.

In the work of Khan *et al.* (2003) an agent corresponds to a molecular species and contains information about the state of every (group of) molecule(s) of that kind. In addition, the agent has a list of reactions this molecular species can participate in. Reactions are initiated by individual molecules (or groups of molecules) using a stochastic algorithm.

So, if there already are several papers demonstrating the applicability of agent-based approach to the modelling of intracellular processes, then why bother with producing yet another one? The first reason is: many of those mentioned works are not using the full range of benefits of the agent concept. In some cases the agents are tied together very rigidly, lacking the autonomy. In some systems the agents are required to execute strictly sequentially: no parallelism. Secondly, the choice to use agents to represent processes (instead of biological/physical objects), as is the case in several works, is definitely interesting, but may hinder the comprehensibility of the model. Also, the process-based models may, although do not necessarily have to, be more difficult to scale up to the whole-cell level where the structure of the cell and the locations of subcellular components become more important. And finally, all described systems are quite different from each other as well as from my model presented in the following section, which means that the search for best practices in the field of agent-oriented intracellular models is only starting and all contributions are very welcome.

4 DnaA Titration Model Based Agent Model

4.1 The DnaA Titration Model

The DnaA Titration Model, or more generally the Initiator Titration Model, is a model from biology describing the molecular level mechanisms that cause the initiation of

DNA replication. I chose to base my agent model on this model because it allows to connect some of the micro- and macro-level processes in the cell without involving too much details of many different intracellular processes. Basically the DnaA Titration Model explains how the initiation of DNA replication is regulated by the concentration of a protein called DnaA. So, by simulating this process we get the moments when DNA replication is initiated, and using some additional information from biology – the duration of DNA replication is constant; the time from the end of DNA replication to the start of cell division is constant; cell mass growth depends on environmental conditions, not on the DNA replication processes – we can “translate” these moments into the changes of cell mass, which is a macro-level variable for a bacterium. The details of the DnaA Titration Model relevant to this work can easily be inferred from the description of the agent model itself and therefore no separate biological description is given here. However, it should be kept in mind that, first of all, the DnaA Titration Model itself is not completely proved to be true yet and secondly the agent model makes a lot of simplifications.

4.2 JADE

JADE (Java Agent DEvelopment framework, <http://jade.tilab.com/>) was used for implementing the following model and the range of possibilities and features it offers has definitely affected the details of the model. JADE is a middleware for developing multiagent systems and includes a library of classes to be used for creating the agents, a (distributed) runtime environment and some graphical tools for administrating and monitoring this environment and the agents running in it.

4.3 The Agent Model

The agent model contains four types of agents: Environment, Bacterium, DnaA Factory and DNA. The values of the variables (time, cell mass, etc.) used in the model are not to be taken as directly comparable to the corresponding values in real bacteria, because this model is built to show only qualitatively, not quantitatively, adequate behavior.

Environment represents the environment around the bacterium.

- For the sake of simplicity, all environmental conditions are currently “reduced” into one single number: the bigger, the better for the bacterium.
- If receiving a question about environmental conditions, then the Environment will reply with a message containing the current value of the variable.
- Environmental conditions are not varying automatically, but can be changed by sending the Environment a request for a change (again a message – currently all agent communication happens through messages). Conditions from about 15 to 60 (arbitrary units) are handled adequately by the current prototype of the model.

Bacterium represents the whole cell of the bacterium, managing the agents that represent the parts of the cell (DNA(s), DnaA Factory) and simulating some of the behavior of those parts of the real cell that are not coded into agents yet.

- Cell mass increases according to environmental conditions. Currently a very simple linear growth function is used: the number representing environmental conditions is just added periodically (after every 100 milliseconds) to the cell mass.
- The Bacterium polls its DNAs to find out the total number of active DnaA promoters (an area on the DNA that carries the information necessary for producing DnaA protein), which influences the speed of DnaA production in the DnaA Factory – the Factory currently gets that number from DNA in order to reduce the message traffic (otherwise it should first ask the Bacterium about which DNAs belong to this cell and then poll all of them).
- The Bacterium stores free DnaA protein coming (in the form of messages) from DnaA Factory and sometimes also from DNAs, and distributes it to those who ask for it (DNAs). The larger the total amount of free DnaA, the bigger quantity is sent (to avoid the situation where DNA could bind a lot of DnaA and there really is a lot of free DnaA available in the cell, but sending them in small amounts takes a long time and unrealistically prolongs the binding process). No DnaA will be lost, because if somebody gets more of that protein than necessary, then the remainder is sent back to the Bacterium.
- If a DNA informs the Bacterium that its replication process has ended, then a pre-cell-division-timer (20 seconds) is initialized.
- When this timer expires, a new Bacterium is created and given half of the cell mass and free DnaA, and approximately half of the DNAs (*approximately*, because in cells with a fast cell cycle some DNAs may be in the middle of the replication process during cell division, meaning the daughter and parent DNA are physically connected and can not be separated). If the simulation is in the “single cell” mode, then the new Bacterium agent will be terminated to save computer resources.
- If the Bacterium is sent a question about cell mass, the amount of free DnaA, the number of DNAs and their connections with each other, or the number of active DnaA promoters in the cell, then it replies with a message containing the value of the corresponding variable.
- To help the modeller to observe the state of the Bacterium, the values of some variables are displayed in an information window (Fig. 1) and also logged into a text file.

DnaA Factory produces DnaA protein, depending on environmental conditions and on the number of active DnaA promoters in the cell: after every [6000 / conditions] milliseconds [the number of active DnaA promoters] molecules of DnaA is produced and sent to the Bacterium. Every Bacterium has one DnaA Factory.

DNA represents the double helix of DNA, either complete or under construction.

- If DnaA boxes (the binding sites of DnaA on DNA) are not full (of DnaA), then DNA asks for free DnaA from the Bacterium.
- The DnaA boxes outside of oriC (the origin of replication, an area on the DNA where the replication process starts) are filled with DnaA. There are 305 such boxes in the model.

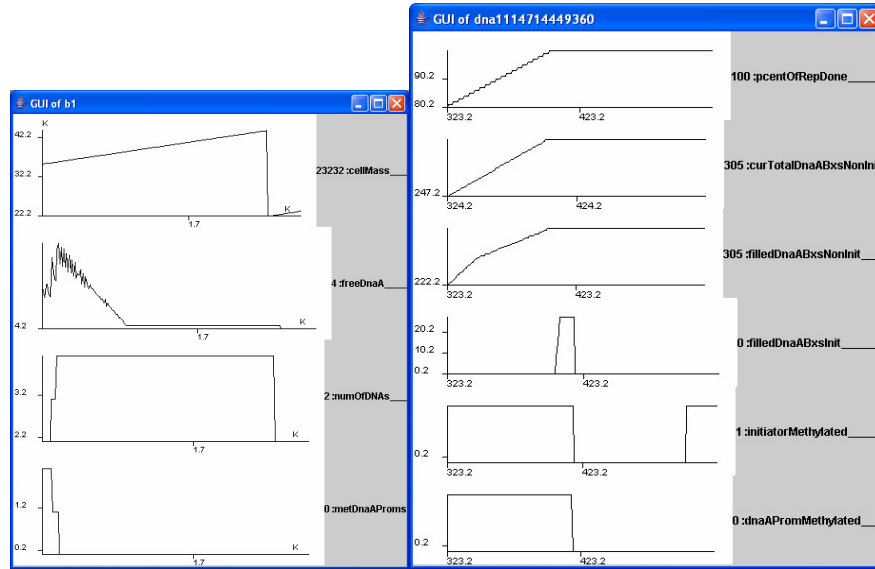


Fig. 1. Information windows of a Bacterium agent (on the left) and a DNA agent (on the right).

- If these boxes are full, oriC is not inhibited and there are over 40 free DnaA molecules in the cell, then the DnaA boxes in oriC are filled. There are 30 such boxes in the model. The requirement for 40 free DnaA molecules is introduced to approximately simulate the low affinity (i.e. low attraction) of some of the DnaA boxes in oriC that requires a higher DnaA concentration for the binding to happen.
- When the boxes in oriC are filled up, DNA replication begins: DnaA from oriC is thrown back to the Bacterium, oriC and DnaA promoter get inhibited for some time (oriC until 20% and DnaA promoter until 55% of replication is done, the same time intervals apply to the new DNA) and a new “non-complete” DNA agent is created.
- The speed of the replication process is 0.25 percentage points after every 100 milliseconds and the replication ends, obviously, when reaching 100%. Currently the DNA agent does not contain any genetic information, so there is actually no real copying taking place and only two other things happen in addition to the increase of percentage: the number of DnaA boxes outside of oriC on the new DNA increases from 0 to 305, and those DnaA boxes on the parent DNA that get “ridden over” by the “replication fork” are emptied and the freed DnaA is sent back to the Bacterium. The DnaA boxes outside of oriC have a uniform distribution over DNA in the model, although that is not exactly the case in reality.
- If the DNA is sent a question about the name of the Bacterium it belongs to, the name of its parent, or whether its DnaA promoter is inhibited, then it replies with a message containing the value of the corresponding variable.
- As with the Bacterium, the values of some variables of DNA are displayed in an information window (Fig. 1).

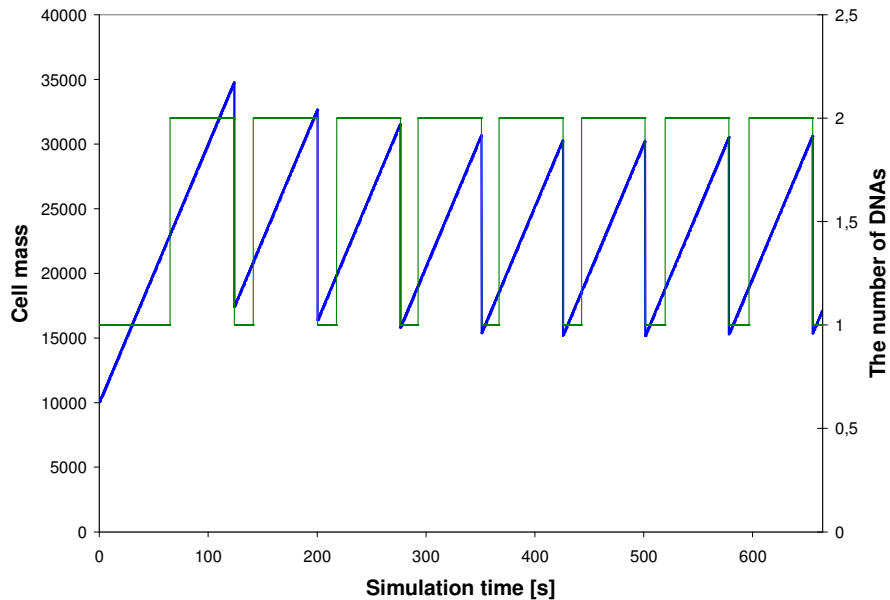


Fig. 2. Periodical cell division. Initial mass 10 000, environmental conditions 20. Sawtooth line is cell mass, square line is the number of DNAs in the cell.

4.4 The Results

By running the described model in a computer, its behavior can be found out and compared with the information about real bacteria.

The first question about this agent model is most certainly: “Will the bacterium start to replicate at all?” As it can be seen on Fig. 2, the cell will indeed start to divide and, in case of stable environmental conditions, a quite stable cell cycle will develop where both the time between divisions and the average cell mass are almost constant. The number of DNAs on this and the following figures is given according to the principle: step +1 will occur at the moment of initialization of the DNA replication, i.e. when a new DNA agent is created in the simulation. Biologically it would be more correct to label it “the number of oriCs in the cell”.

When the initial mass of the cell, given by the modeller, is somewhat lower or higher than normal, the cell should still stabilize into the same (oscillating) state as before. This is indeed the case in the model: see Figs. 3 and 4.

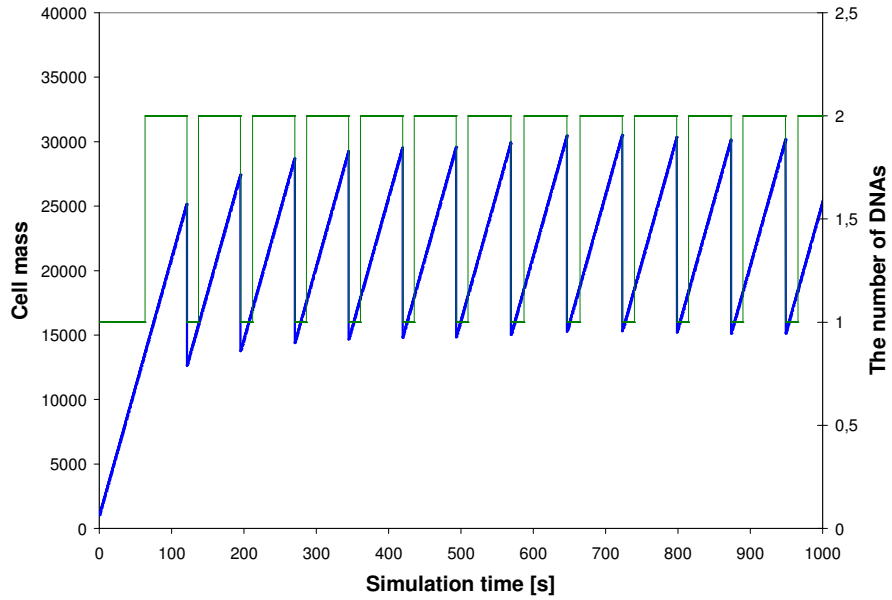


Fig. 3. Initial mass lower than normal: 1000. Environmental conditions 20. Sawtooth line is cell mass, square line is the number of DNAs in the cell.

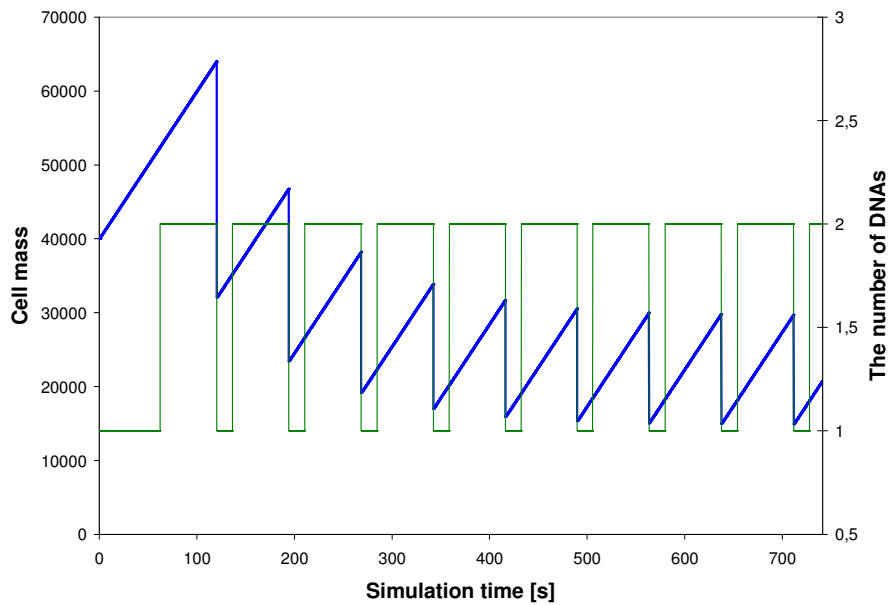


Fig. 4. Initial mass higher than normal: 40 000. Environmental conditions 20. Sawtooth line is cell mass, square line is the number of DNAs in the cell.

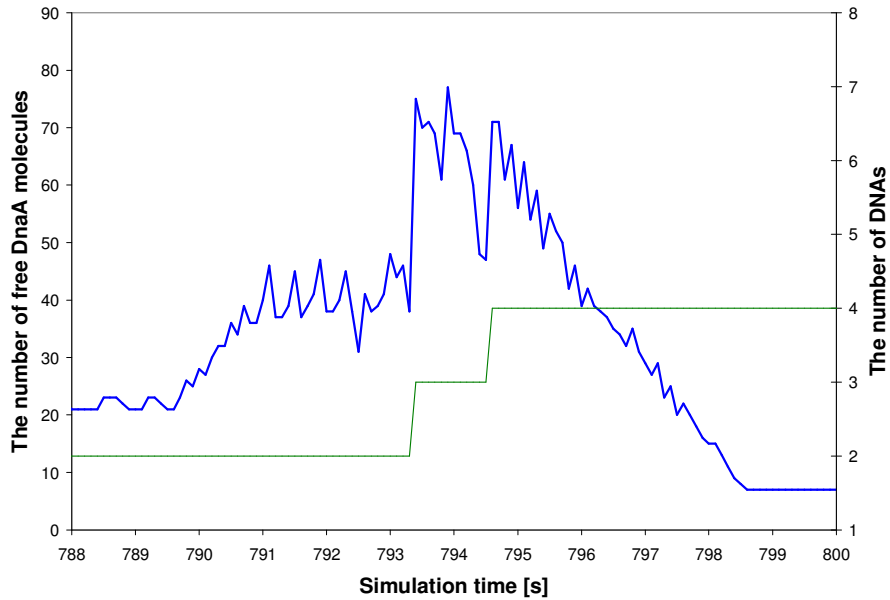


Fig. 5. Semisynchronous initiation of DNA replication. Environmental conditions 50. The irregular line is the amount of free DnaA in the cell, the two-stepped line is the number of DNA agents in the cell.

In very good environmental conditions bacteria grow fast and divide often. However, as the time of DNA replication and the delay between the end of DNA replication and the cell division initiated by this concrete DNA replication are constant, there is more than one DNA in a fast-growing bacterium and a new replication process is initiated before the previous one is finished. In normal bacteria the initiation happens on all oriCs almost, but not exactly, at the same moment. The phenomenon is called (semi)synchronous initiation and, as its mechanisms are described in the Initiator Titration Model, it should also occur in the agent model. Fig. 5 demonstrates the semisynchronous initiation of DNA replication in the simulation. What exactly happens there is described in the following:

- On the 791st simulation second the number of free DnaA molecules in the cell reaches the threshold of 40 and the DnaA boxes in oriCs of the two DNAs will start filling up.
- On the 793rd second all the DnaA boxes in the oriC of one of the DNAs get filled and the initiation of replication happens there. The DnaA molecules bound to this oriC are released and the amount of free DnaA in the cell increases sharply. The oriCs of this DNA and the new created DNA get inhibited to avoid a possible instantaneous reinitiation. The DnaA promoter, which is near the oriC area, gets inhibited as well and therefore the production of DnaA in the cell slows down.

- High concentration of free DnaA considerably speeds up the binding process on the other DNA (as already mentioned, and also visible on Fig. 5, there were two DNAs in the cell in the beginning of the time frame under study here).
- In the other half of the 794th second the DnaA boxes in the oriC of this other DNA also get filled and the DNA replication starts. Here, too, the DnaA molecules are freed from oriC and the amount of free DnaA in the cell skyrockets once again. oriC and DnaA promoter of both the parent DNA and the new created DNA get inhibited and the production of DnaA in the cell stops.
- During the construction of new DNA double helices the number of empty high affinity DnaA boxes outside the oriC increases. The free DnaA binds to those boxes and the total amount of free DnaA in the cell drops below the threshold of 40 molecules long before the oriCs get uninhibited again.

The best indicator of the adequacy of the created agent model would be the emergence of the initiation mass. According to the generally accepted Cooper-Helmstetter-Donachie model, the initiation of DNA replication occurs only when the cell has a certain mass, called initiation mass. This mass depends on the number of oriCs in the cell. When environmental conditions get better and bacterium's growth speed increases, the initiation mass stays constant until the growth speed reaches a certain value. Then the initiation mass usually doubles (due to the semisynchronous initiation of DNA replication, which causes the number of oriCs in the cell to be a power of two for most of the time) and stays constant for another range of growth speeds, etc. The concept of initiation mass is not explicitly built into the Initiator Titration Model nor into the agent model. Therefore, its presence in simulation results would indicate the adequacy of the Initiator Titration Model and the ability of the multiagent system to allow its emergence from micro-level behavior (however, it should be noted that the presence of initiation mass in the results would not be a *proof* of model's correctness). As shown on Fig. 6, the initiation mass indeed emerges during simulations. It is not absolutely constant, though: on the second plateau on the graph there seems to be a slight increase in the initiation mass when the growth speed increases. The roots of this increase are to be studied more thoroughly in the future, because the current model prototype and / or JADE environment are not stable enough to allow very long simulation runs and therefore the increase may also be just an error due to the smallness of the dataset used for creating this graph.

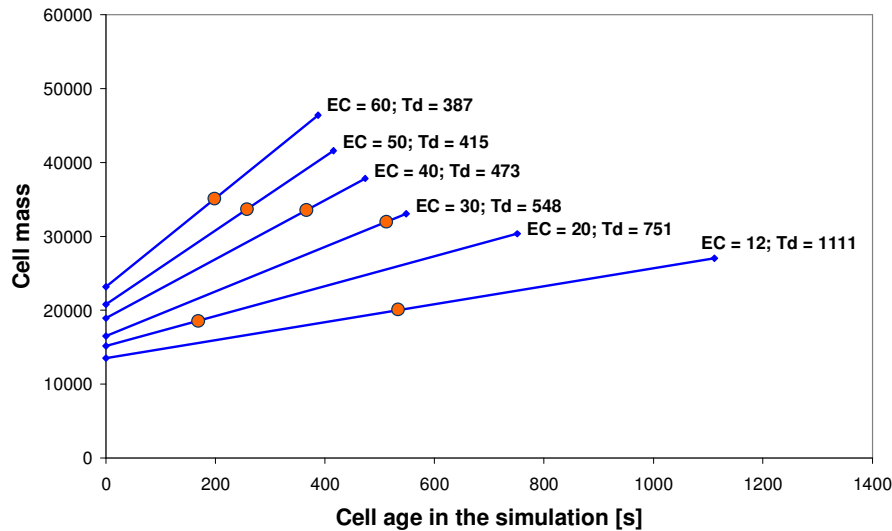


Fig. 6. The presence of the initiation mass in the results of simulations. Every line represents the life cycle of a cell from one division to another. The circles show the moments of DNA replication initiation. EC = Environmental Conditions; Td = the Time of Division (relative to the previous division), i.e. the length of the cell cycle.

5 Conclusions

The agent model was relatively easy to construct and its description is quite understandable even for a non-biologist. In spite of its simplicity, the model gives results that seem to be reasonable. Although only a few intracellular processes were included in this prototype, the other processes are unlikely to be fundamentally different and therefore the agent models should be generally appropriate for modelling a biological cell.

The presented model was, as mentioned, just a small prototype and a lot of further work lies ahead to create a practical agent-based cell modelling tool and / or methodology:

- The choice of agent development environment should get more attention – JADE was chosen as it offered enough support to create the prototype, but there may be better tools available.
- Current model depends on astronomical time due to the way it uses timers. It needs manual reconfiguration when moving from one computer to another with different computing power. Better solutions should be found for event timings and to ensure

the freshness and / or reliability of data received through message passing (especially when the simulation will be distributed over many computers).

- The spatial structure of the cell should be taken into account.
- It is likely that the new tool would first be used for modelling the most interesting processes. These may, however, not be immediately integratable with each other in order to create the whole-cell model – some other processes may be necessary that “glue” them together. Therefore it would be useful to define beforehand the requirements for the small simulations that would allow an easy integration later, when all necessary process models are available.
- The increasing complexity of models requires better graphical user interfaces to keep the behavior of the system understandable for the modeller.
- As the number of parts of a bacterial cell is huge – about 10^9 molecules, many of which are tied to each other, many participate in various biochemical reactions, etc.
 - it would be sensible to allow the user to change the abstraction levels of the simulation. The abstraction level of (visual) representation would determine, how many details the user sees. The abstraction level of the simulation itself would determine, how detailed (sub)models are used inside the simulation, e.g. whether a certain type of objects inside the cell is represented as one agent, or each object of that type has its own representative agent.

6 Acknowledgements

The initial ideas of this work originate from the discussions between professors of Tallinn University of Technology Leo Mõtus (a computer scientist) and Raivo Vilu (a biochemist).

The work has been financially supported by:

- Department of Computer Control, Tallinn University of Technology, where I work as an engineer.
- Estonian Science Foundation: in 2004 I was one of the investigators under the research grant 4860 “Agents and Their Behaviour in Real World“ and in 2005 I am one of the investigators under the research grant 6182 “Studying Multiagent Systems in Heterogeneous Environment with Dynamically Changing Structure (The Hopad hoc project)“.
- Ministry of Education and Research of Estonia: I am one of the investigators under the research grant 0142509s03 “Intelligent Components and Their Integration Problems”.

7 References

- Alur, R., Belta, C., Kumar, V., Mintz, M., Pappas, G. J., Rubin, H. and Schug, J. 2002, “Modeling And Analyzing Biomolecular networks”, Computing in Science & Engineering, vol. 4, no. 1, pp. 20-31, database: IEEE Xplore.
- Burleigh, I., Suen, G. and Jacob, C. 2003, “DNA in Action! A 3D Swarm-based Model of a Gene Regulatory System”, Proceedings of the First Australian Conference on Artificial Life.

- Lecture Notes in Computer Science., Springer-Verlag: Berlin, 15p.
<http://pages.cpsc.ucalgary.ca/~jacob/ESD/Research/LacOperon/3D/DNA-Action.pdf>
- Gombert, A. K. and Nielsen, J. 2000, "Mathematical modelling of metabolism", *Current Opinion in Biotechnology*, vol. 11, no. 2, pp. 180-186, database: ScienceDirect.
- Fisher, M. J., Malcolm, G. and Paton, R. C. 2000, "Spatio-logical processes in intra-cellular signalling", *Biosystems*, vol. 55, no. 1-3, pp. 83-92, database: ScienceDirect.
- Fisher, M. J., Paton, R. C. and Matsuno, K. 1999, "Intracellular signalling proteins as 'smart' agents in parallel distributed processes", *Biosystems*, vol. 50, no. 3, pp. 159-171, database: ScienceDirect.
- Gonzalez, P. P., Cardenas, M., Camacho, D., Franyuti, A., Rosas, O. and Lagunez-Otero, J. 2003, "Cellulat: an agent-based intracellular signalling model", *Biosystems*, vol. 68, no. 2-3, pp. 171-185, database: ScienceDirect.
- Katara, S. and Venkatasubramanian, V. 2001, "An agent-based learning framework for modeling microbial growth", *Engineering Applications of Artificial Intelligence*, vol. 14, no. 6, pp. 715-726, database: ScienceDirect.
- Khan, S., Makkena, R., McGeary, F., Decker, K., Gillis, W. and Schmidt, C. 2003, "A Multi-Agent System for the Quantitative Simulation of Biological Networks", *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pp. 385-392, database: ACM Digital Library.
- Paton, R. C. 1993, "Some Computational Models at the Cellular Level", *Biosystems*, vol. 29, no. 2-3, pp. 63-75, <http://citeseer.ist.psu.edu/paton93some.html>
- Querrec, G., Rodin, V., Abgrall, J. F., Kerdelo, S. and Tisseau, J. 2003, "Uses of Multiagents Systems for simulation of MAPK pathway", *Third IEEE Symposium on Bioinformatics and Bioengineering, Proceedings*, pp. 421-425, database: IEEE Xplore.
- Stroustrup, B. 1988, "What is Object-Oriented Programming?", *IEEE Software*, vol. 5, no. 3, pp. 10-20, database: IEEE Xplore.
- Wooldridge, M. 1998, "Agents and Software Engineering", *AI*IA Notizie*, vol. 3, pp. 31-37, <http://www.csc.liv.ac.uk/~mjw/pubs/aiia98.pdf>