Simulation of genetic networks in multicellular context

Ute Platzer and Hans-Peter Meinzer

Deutsches Krebsforschungszentrum
Im Neuenheimer Feld 280
D-69120 Heidelberg, Germany
u.platzer@dkfz-heidelberg.de

Abstract

The aim of genetic network simulation and analysis so far has been the identi£cation of attractor states that represent differentiated cell types. In multicellular tissues state transitions depend not only on network topology, but also on cell-cell-contacts. We present Gene-O-Matic, a user-friendly software tool for the simulation of genetic networks and cell differentiation in multicellular organisms, and an example application.

Keywords: genetic network, simulation, cell differentiation, cell-cell interaction.

1 Introduction

Several groups have gathered genetic information for different organisms into so-called genetic networks and simulated the behaviour of these network. By doing so they reconstructed biological observations in the computer model and were able to make predictions about wet-lab experiments not yet done (for a recent review see [dJ02]).

Two main approaches can be distinguished: different types of differential equations, and logical formalisms. While the advocates of differential equations argue that discrete models do not offer enough precision, the people in favour of logical formalisms bring forward the argument that, to some approximation, genes do not take continuous states but are either on or off (or at least are con£ned to a discrete set of states, for reviews see for example [Kau91], [Kau93]).

Genetic network simulation and analysis often deals with the £nal states of state transitions. The aim is to identify attractor states and cycles that represent different types of differentiated cells. But cells in the process of differentiation change their state perpetuously. Our interest is in the process of differentiation itself, in recapitulating the state changes that have been observed biologically and that eventually lead to stable states. It is useful to reproduce the observed £nal states but even more so to accurately reconstruct the path taken to reach them. Only if we can model the whole process can we claim to understand it. Only then can we try to model mutants and make predictions about the results of experiments not yet done.

The models built so far focus on analysing single cells. Mechanisms such as cell division or inductive interactions have not been included. But many differentiation processes are based on intercellular signal transduction mechanisms. These cannot be modelled with the tools available today.

Here we present Gene-O-Matic, a genetic network simulation tool that combines molecular-level data with cell-level data. The simulation is based on a discrete-state genetic network (similar to a Boolean network), but also uses cell positions in space. The cells are represented as spheres, and each cell contains the genetic network. The cells can interact via *external* genes [JJN86] that enable communication between neighbouring cells. The simulation software was designed to be easy-to-use. As a test case a model of the embryonic development of the round worm *Caenorhabditis elegans* is discussed. The program can easily be adapted to simulate any cellular organism. To our knowledge, this is the £rst time that the simulation of genetic networks and three-dimensional cell models has been combined.

2 Material and Methods

The Gene-O-Matic software is written in Java (see http://java.sun.com) version 1.3. Some additional, freely available libraries were used (for a list see http://mbi.dkfz-heidelberg.de/mbi/research/cellsim/software/libraries.html). The software runs on Windows and Linux/Unix and should also run on Macintosh and any other operation system supporting java, though this has not been tested. It is available for download at http://mbi.dkfz-heidelberg.de/mbi/research/cellsim/software/geneomatic.html.

2.1 The network model

Each cell in an organism contains identical genetic information but different cells use this information in different ways. The use of genetic information can be regulated at three levels: DNA (transcription), mRNA (splicing and translation), and protein (activity and degradation). In our model all three levels are treated equally. Genes, mRNA, and proteins are jointly called biological information carriers (BICs). The three levels may be modelled explicitly by using three BICs: one BIC for the gene, one for the mRNA, and one for the protein, or implicitly by using one single BIC that represents gene, mRNA, and protein together.

2.2 Representation of the genetic state of a cell

Each BIC can have the state `1` (if the gene is transcribed, the mRNA translated, and the protein present and active) or `0` (if the gene is not transcribed, the mRNA not correctly processed, or the protein inactive or not present). The genetic state of a cell can thus be described by a vector with values in $\{0, 1\}$.

BICs can in¤uence other BICs in their own cells and in neighbouring cells. To model this BICs can have different ranges of activity (see table 1).
Prior to the calculation of the next state of a cell the states of the neighbouring cells have to be considered to identify potential intercellular interactions. This is done in several successive steps. First the external state of the cell is determined: the neighbouring cells are identi£ed and their states are combined into one `"external"` state vector. If in any of the neighbouring cells a BIC has state `1` its state in the external state vector is set to `1` as well, else it is `0`. In a second step the overall state of the cell is determined by combining external and internal state. A vector is created that contains for all BICs with *internal* range their state in the respective cell and for all *external* BICs their state from the external state vector. This technique of dividing the genetic network into

an internal and an external part has been described in [JJN86]. The resulting vector is used for calculation of the next state for the respective cell.

Table 1: Properties of BICs

Property	**Description**	**Used with**
Name	The name of the BIC, used in Boolean formulas.	Boolean network
Threshold	via the threshold you can specify how many BICs need to have activating in¤uence on this one to change its state to `1`.	Matrix
Range of Activity	This property can take one of these values: *internal* (meaning this BIC acts only within a single cell), *external* (meaning this BIC acts only on BICs in neighbouring cells), or *both*.	both
Location	Here you can specify which part of the cell the BIC is located in: *posterior*, *anterior*, *left*, *right*, *dorsal*, *ventral*, a combination of these, or none.	both
Function	The Boolean function regulating this BIC's activity.	Boolean network

Table 2: Properties of interactions

Property	**Description**
Strength	This value determines the strength and type of an interaction. Positive values indicate activation, negative values indicate inhibition. You can use it to specify that one interaction outweighs another, for example an inhibiting interaction with strength -1 can be outweighed by an activating interaction of strength +2.

2.3 Representation of the genetic network

From a calculation point of view the genetic network consists of a set of interactions between BICs. Gene-O-Matic offers two different methods to describe these interactions. Along with this two different calculation principles have been implemented.

2.3.1 Matrix

In this representation of the genetic network the interactions are represented as arrows connecting the BICs. Each interaction has a speci£c strength (see table 2). Activating interactions have a strength greater than zero. The strength of inhibiting interactions is below zero. All strength values are integers. The interactions can be formulated as a matrix where each component corresponds to the strength of the interaction between the BIC in the respective column and the BIC in the respective row.

The calculation algorithm for the matrix representation is based on a method used by Mendoza and Alvarez-Buylla [MAB98]. But while they use an asynchronous approach where the BICs change their states successively and not simultaneously, at the moment Gene-O-Matic offers only a synchronous approach. For the models we built so far the results of the asynchronous method already matched the observed data.

The next state of a cell is obtained by multiplying the interaction matrix with the state vector of a cell (after combining it with the states of the neighbouring cells). The resulting vector cannot be used for the new state vector directly as it may contain values other than `0` and `1` (e.g. if an interaction strength was larger than 1 or if two interactions activate the same gene). Therefore it is re-discretized by applying a step function, assigning a `1` whenever the result vector entry is larger than the respective BIC's threshold, and a `0` if the result vector entry is smaller than or equal to the BIC's threshold. The re-discretized vector represents the next genetic state of the cell.

2.3.2 Boolean network

In this representation interactions are formulated as Boolean functions. Each BIC is assigned a Boolean (or logic) function that combines all input BICs for that BIC using logical operators like `AND` (&), `OR` (|), and `NOT` (!). A BIC that is activated by BIC B and inhibited by BIC C could have a Boolean function that looks like this: $B\&(!\,C)$. Note that no explicit interactions ("arrows") are required here.

To calculate the next state of a cell in this model the Boolean functions for all BICs are evaluated, interpreting BICs with state `1` as `TRUE` and BICs with state `0` as `FALSE`. This is done synchronously, i.e. all functions are evaluated before any new states are actually set.

2.3.3 Setting the next state of the cells

Only if all cells know about their next states these states are set. Therefore all cells change their states synchronously and not successively. This is important to make sure the model retains its deterministic behaviour in the cell-cell interactions. If cells would change their states successively one would have to de£ne which cell changes its state £rst. In real living cells state changes do not occur as discrete steps but rather gradually, therefore it would not make sense to model them as successive events.

The state changes occur in regular time intervals. The interval length as well as the time of the £rst change can be de£ned by the user. This makes it possible to co-ordinate state changes with cell positions in simulations where cell positions and times are taken from a database.

2.3.4 Converting between matrix and Boolean network

The matrix representation is a more intuitive description for genetic networks as one can create a network basically by drawing a picture of it. The Boolean model is available because there are some special cases that cannot be modelled accurately by the matrix representation without introducing additional BICs into the network and without using "Multi-BICs" that combine the functionality of several sequentially connected BICs (e.g. the Boolean operator `XOR` (^) that means that exactly one of two BICs is on). On the other hand different-strength interactions can be modelled only with the matrix but not with the Boolean network. Whether these special cases are really biologically relevant is a question not to be discussed here.

For convenience conversion mechanisms between the two models are provided. They do a near conversion (because as stated above an exact conversion may not be possible). The user can thus specify part of the network in the Boolean formalism and another part using the matrix formalism. The program can then combine the information to provide a complete model in both of these representations. It might also be interesting to see how these models differ in their behaviour.

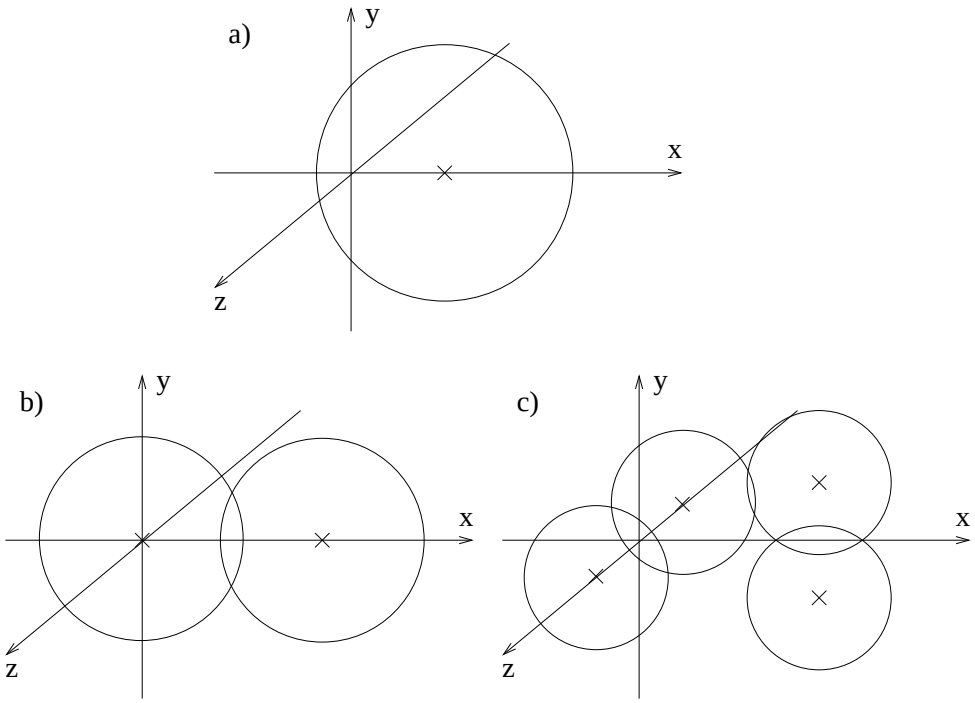

Figure 1: Simulated cell division directions.

A third calculation mechanism has been implemented that combines both Boolean and matrix features. For those BICs where a Boolean function is speci£ed the Boolean network model is used to calculate the next state, otherwise the matrix multiplication is used.

2.4 Representation of the cells

The simulation of the cells is based on the CellO software [GSM00]. The cells have a position and a size. The position is a point in 3D specifying the centre of a sphere. The size represents the radius of the sphere. Each cell is associated with a genetic state as described in section 2.2.

To calculate cell-cell contacts a restricted Delaunay triangulation is done. This is a mathematical method where neighbouring cells are connected by lines. The line lengths are restricted to a maximum of the sum of the radii of two cells (multiplied by a factor of 1.15 to allow for non-ideal spheres). Thereby connections between cells that are very far away from each other are prevented. They would result in non-spherical cell shapes.

Two methods exist to model cell behaviour. First the cell positions can be taken from a database of recorded (or pre-calculated) positions. The recorded cell positions for the *C. elegans* embryo [SHMS97] are an example of such a database. The user does not need to specify any cell information in the genetic network editor except which cells are present at the beginning of the simulation and what their genetic states are.
The second possibility is to simulate cell movement and divisions. Here the user has to specify position and size of the initial cells. Additionally some information about the division direction of the cells has to be provided. The division direction of a cell is the direction of the mitotic spindle, a vector orthogonal to the actual plane along which the daughter cells separate. The simulation environment assumes that cells divide in a direction orthogonal to the division direction of their

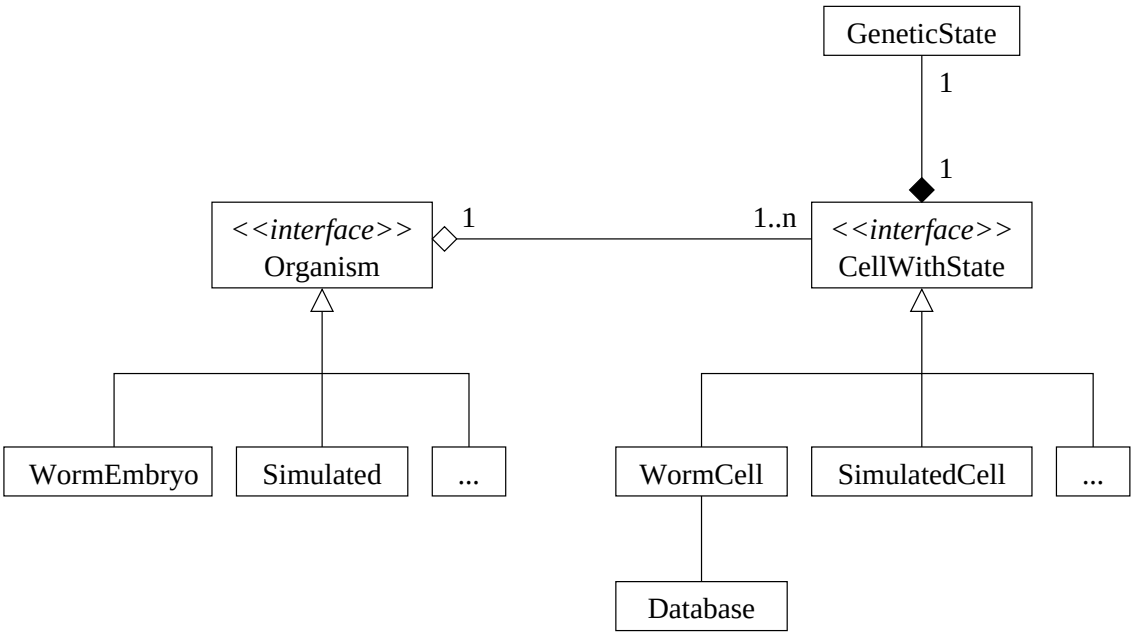

Figure 2: Interface and class structure of organisms, cells, and genetic state.

mother, parallel to the division plane of the mother cell. So two random vectors parallel to the mother cell's division plane and orthogonal to each other are calculated as the division directions of the two daughter cells. An example is shown in £gure 1. Crosses indicate cell centres. The mother cell (a) divides along the x-axis. The two daughter cells (b) therefore divide in directions orthogonal to the x-axis (c), in our example along the y-axis (right daughter) and z-axis (left daughter).
The division direction vectors may be rotated if the cell is in a speci£c genetic state. The rotation angle and the Boolean function describing the genetic state can be speci£ed by the user. Also the time of division can be regulated via the genetic network by selecting one or several BICs to trigger cell division. This corresponds to a very basic implementation of a cell cycle.

2.5 Extensibility of the software

The software is written in Java and follows an object-oriented approach. This allows for simple extension and adaptation of the software to other problems, for example to simulate another type of organism or to provide another method to calculate next states.

Figure 2 shows an UML diagram explaining the relationship between the organism, the cells it contains, and their genetic state. For different types of organisms, different cell types may be used. Eventually cells may be associated with a database that contains their positions and other information (for example when they divide and what their daughter cells are). Each cell implements the interface "CellWithState" that allows the simulation environment to access and modify the cell's genetic state as required during the simulation. Two types of organisms and their associated cell types are shown: worm embryos which take cell information from recorded data, and simulated organisms where cell positions and division parameters are calculated during the simulation based on the genetic state of the cells. It is also indicated by $\boxed{\ldots}$ which classes have to be implemented to extend the program to simulate another organism.

2.6 Graphical user interface

The user interface of the genetic network editor was designed to be clear and easy-to-use. It consists of a window with four areas. The most important is the network area where the user can enter the genetic network to simulate. BICs and interactions can be added by mouse clicking or by dragging them from the database area. The database area is present only if a database of BICs and interactions exists; the user may create a genetic network to use as a database. BICs and interactions are represented by boxes and lines respectively. The properties of BICs and interactions are easily accessible via the right mouse button.
The other two areas are panels that provide ways to add cells and parameters to the simulation. The user must specify which cells exist at the beginning of the simulation and what their initial genetic states are. This can be done in the cells panel. The parameter panel is necessary only for simulated positions. It allows speci£cation of division parameters such as which BICs trigger cell division or how the division direction is altered if speci£c BICs are active. Via a menu bar additional simulation options, e.g. the intervals between the state changes, are accessible.

2.6.1 User interface during simulation

During the simulation two types of windows are present. The "Run" window allows the user to control the simulation (to go through it step-wise or to run up to a speci£ed time and to decide which views are to be shown). The view windows show the simulated organism from different perspectives. There can be up to four different views, three projections from different directions in two dimensions, and one three-dimensional view. In the three-dimensional view the user can click on cells to obtain additional information about them. The colours of the cells in the view windows are determined according to the cells' genetic states.

2.6.2 Display of simulation results

The results of a simulation, i.e. the genetic states of all cells over time, can be displayed in a cell lineage tree. Different colours represent different genetic states of the cells. The user can choose the colours and the BICs to be displayed. This offers a good overview of the development of the whole organism at once. Additionally the genetic states of all cells are stored in a log £le for later reference.

2.6.3 Help

For most functions of the Gene-O-Matic program context-sensitive help is available. Furthermore an online user guide is provided that describes how to create a new network and start a simulation. It is also available in the internet at http://mbi.dkfz-heidelberg.de/mbi/research/cellsim/net/help.

3 Results and discussion

3.1 Proof of concept

To test Gene-O-Matic, the round worm *Caenorhabditis elegans* was chosen as a model organism. Its embryo develops inside a transparent egg shell and is easily observable. Therefore a lot of

data has been collected about embryonic development, on the molecular level as well as on the cellular level. Thus *C. elegans* is well suited for testing a program that combines these two levels. Cell positions were taken from recorded data [SHMS97]. Only the positions of the cell nuclei are known, therefore we made the assumption that the nucleus is exactly the centre of the cell.
With the genetic network we compiled from the literature Gene-O-Matic is capable of simulating embryonic development correctly up to the 22-cell stage, thereby generating seven genetically different cell types (unpublished data). The simulated protein activities match the expression patterns observed in the embryo. Via externally acting BICs all inductive interactions occurring in the embryo up to that stage are reconstructed. Therefore the program can be used to simulate cell differentiation in multicellular organisms.

Some aspects of the modelling process are worthy of a more detailed discussion. They will be explained in the next paragraphs.

3.1.1 Representation of the cells

In one aspect the software is not yet capable of representing biological facts accurately: the modelling of inhomogeneous molecule distributions within single cells. For example in the worm embryo a signal is sent to a cell and induces an asymmetric activation of a protein. Upon division of the now asymmetric cell two different daughter cells are generated, one where the protein is active and one where it is inactive. A workaround had to be invented to reproduce this effect in the simulation: the signal is sent after the division of the cell to only one of the daughter cells. The other daughter cell is not in contact with the signalling cell and therefore does not receive the signal.

It is planned to extend Gene-O-Matic to include asymmetric molecule distributions within single cells.

3.1.2 Robustness

For the simulation of *C. elegans* to produce the correct results it is very important to tune the times of state changes to the recorded cell positions, i.e. a state change has to occur exactly when the cells make the correct contacts. This is further complicated by the fact that due to the spherical representation of the cells the exact shape of a cell cannot always be correctly modelled. As the position of a cell is based on the measured position of the nucleus further errors may occur for cells where the nucleus is not located in the middle of the cell.

These problematic effects are less grave for simulations with simulated rather than recorded cell positions because here the user is more free in the choice of time and place of the interactions. To demonstrate this we are working on a simulation of *Arabidopsis thaliana* ¤ower development.

3.2 Con£rmation of experimental results and predictive power

Construction of a model of a well-known biological process, simulation of its behaviour, and comparison of the simulation results with experimental observations allows con£rmation of the existing data. If the simulation results equal the experimental results one can conclude that everything about that process is known. If the simulation gives other results however one may assume that some elements crucial to the investigated process are misplaced in the model or even still missing.

Furthermore simulation enables scientists to quickly try out different hypotheses about a process yet unknown and enables them to make predictions about the underlying biological mechanism. In our simulation we succeeded in sending a speci£c intercellular signal although the exact biological mechanism is still unknown. For this purpose two hypothetical proteins were introduced into the model that had not been detected in biological experiments. Later critical reading of the respective literature revealed that indeed the existence of similar proteins had already been suggested (though they still have not been found experimentally).

This clearly demonstrates that Gene-O-Matic can indeed be used not only to recapitulate long-known facts but also to make predictions about unknown processes.

Even for mechanisms that seem to be understood completely Gene-O-Matic can give new insights. Imagine one protein being deactivated by the activity of another. Upon cell division the inhibiting protein is separated from the £rst one, as happens in the *C. elegans* embryo on several occasions. No biological data exist about a possible re-activation of the £rst protein now that it is no longer inhibited. In the simulation both possibilities can easily be modelled. This is a good example of how a computer model may be used to suggest experiments biologists could make. If experimental data were available here more precise statements about the underlying molecular mechanisms of protein activation could be made.

3.3 Conclusion

We present Gene-O-Matic, a computer tool for the qualitative simulation of genetic networks in multicellular context. It is easy-to-use for inexperienced users. The tool was applied to simulate the development of the *C. elegans* embryo. The results compare well with the observed features, indicating that this simulation method is suitable for simulating development and cell differentiation in multicellular organisms.

References

[dJ02] H. de Jong. Modeling and simulation of genetic regulatory systems: a literature review. *J Comp Biol*, 2002. (to appear).

[GSM00] M. Gumbel, R. Schnabel, and H.P. Meinzer. Analysis of cell migrations in *C. elegans* using computer simulations. *Proceedings of the 14th European Simulation Multiconference (SCS Publications)*, pages 605–611, 2000.

[JJN86] E.R. Jackson, D. Johnson, and W.G. Nash. Gene networks in development. *J Theor Biol*, 119(4):379–96, Apr 21 1986.

[Kau91] S.A. Kauffman. Antichaos and adaptation. *Sci Am*, 265(2):78–84, Aug 1991.

[Kau93] S.A. Kauffman. *The origins of order: Self-organization and selection in evolution.* Oxford University Press, New York, 1993.

[MAB98] L. Mendoza and E.R. Alvarez-Buylla. Dynamics of the genetic regulatory network for *Arabidopsis thaliana* ¤ower morphogenesis. *J. Theor. Biol.*, 193(2):307–319, Jul 1998.

[SHMS97] R. Schnabel, H. Hutter, D. Moerman, and H. Schnabel. Assessing normal embryogenesis in *Caenorhabditis elegans* using a 4d microscope: variability of development and regional speci£cation. *Dev. Biol.*, 184(2):234–65, Apr 15 1997.