



**Disclaimer:** Submitted to Dublin City University, School of Computing for module *CA545: Practicum*, 2005/2006. We hereby certify that the work presented and the material contained herein is our own except where explicitly stated references to other material are made.

## **Modelling Networks and Pathways in Systems Biology**

Jorma J. de Ronde jorma.deronde2@mail.dcu.ie	55217407
Cecile Petmi Ndjehan cecile.petmindjehan2@mail.dcu.ie	55218951

## **Abstract**

Systems biology takes the approach of modelling a (biological) system as a whole, instead of starting from the individual constituents and putting those together (the reductionist approach). In this thesis we model several biological systems using systems biology techniques and implement an automated simulation system. Out of several systems biology formalisms available we chose the Brane Calculus. The Brane Calculus offers an intuitive approach to modelling biological systems and focuses on membrane interactions. We describe the HIV life cycle, the Macrophage function and the HCV life cycle in the Brane Calculus. We then show how we successfully implemented the Brane Calculus and simulated several biological systems using this implementation. We also extended the Brane Calculus with a stochastic extension, allowing for quantification of molecules and membranes and stochastic decision making between applicable reactions. For the implementation we used Java in combination with the Tom plugin. Our stochastic extension makes use of the Gillespie algorithm.

## Table of contents

1	Introduction .....	3
2	Introduction to systems biology .....	3
	2.1 Advantages of systems biology models .....	5
	2.2 Applications.....	5
3	Introduction to the Brane Calculus.....	6
	3.1 Overview .....	6
	3.2 Basic framework .....	7
	3.3 Bitonal Interaction .....	8
	3.4 Molecules .....	9
	3.5 Explanation of the rules.....	9
	3.6 Brane molecule reaction.....	12
	3.7 Example: Viral Infection and Reproduction.....	12
4	Stochastic Extension to the Brane Calculus.....	13
	4.1 Stochasticity.....	14
	4.2 Gillespie's algorithm - overview .....	14
	4.3 Gillespie's algorithm - details.....	16
5	Detailed outline of the biological systems implemented in Brane calculi.....	17
	5.1 Macrophage function .....	17
	5.2 HIV life cycle .....	19
	5.3 HCV life cycle .....	21
6	Brane calculus implementation .....	23
	6.1 Introduction to Tom .....	23
	6.2 Implementation overview .....	24
7	Results.....	28
8	Conclusions and Future Work.....	30
	8.1 Conclusions .....	30
	8.2 Future work.....	31
9	References.....	32

# 1 Introduction

With more and more data becoming available about biological processes and pathways, greatly helped by the advent of new high-throughput technologies, it becomes possible to gain a better understanding of what is going on around us. With this better understanding it also becomes possible to come up with better simulation models. These models can then be used to make accurate predictions about how certain biological or chemical reagents will influence a system, thereby greatly reducing the need for wet-lab experiments and saving valuable time and money.

Biological processes can be immensely complex and it is not yet within our grasp to model them fully. However, certain elements are starting to come within reach. In order to model these processes a suitable modelling language must be present. Several experimental formalisms exist nowadays. One of these is the brane calculus, based on membrane interactions. Membranes are sites of major activity in a biological system and as such allow for a wide range of biological processes to be modelled. We chose to implement this formalism and extend it with a quantitative and stochastic extension.

This report starts off with an introduction to systems biology in general and the brane calculus in particular. Details of the stochastic extension and the actual implementation are then presented. We then summarise our results and finally conclude the report with our conclusion and discussion of future work.

## 2 Introduction to systems biology

From a historical point of view, biologists and chemists have usually tried to break a particular system under review down to its smallest parts and study those in hope of putting all these small parts together in the end. This was also due to practical limitations, since the available experimental procedures necessarily forced a 'one protein at a time' analysis during the middle of the 20th century. This approach is known as the reductionist approach [17].

However, the study of biological systems cannot be limited to simply listing its components (proteins, genes, cells, etc.). Just as a complete list of all the parts of a car gives a vague impression about that car, it does not necessarily mean that one will be

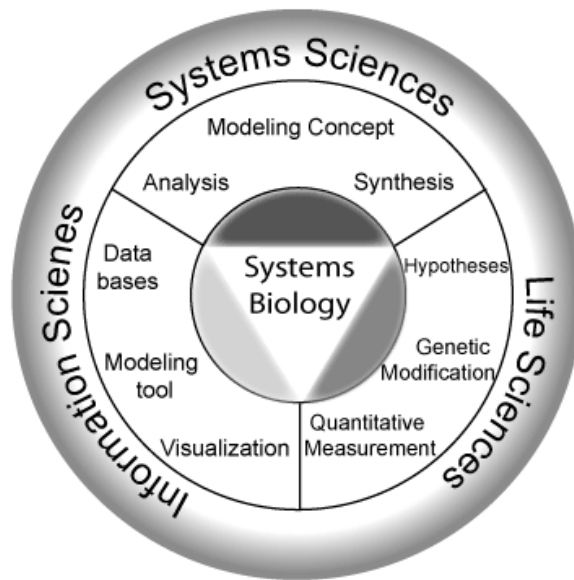
able to understand how the car functions. Similarly, a deeper understanding of biological systems can help to make clear how these parts work and interact with each other and with the environment they are in. In other words, a system-level understanding is required.

More recently, a new approach has emerged, where the aim is to model a system as a whole instead of trying to model all the constituents in detail, with an emphasis on developing a mathematical model of the system. Such a model can then be used to predict changes in a (biological) system and be iteratively tested to prove or disprove the model. The emergence of these types of model is largely due to the recent technological breakthroughs providing us with new data about biological processes [18].

DNA micro arrays, for example, allow us to collect data about global changes in gene expression and protein concentrations can be measured using mass spectrometry.

Using these techniques, the systems biologist can propose a hypothesis that explains the system's behavior. This hypothesis can then be used to mathematically model the system. Models are used to predict how changes in the system's environment affect the system and these models can be iteratively tested for their validity. New approaches are being developed by quantitative scientists, such as computational biologists, statisticians, mathematicians, computer scientists, engineers, and physicists, to improve our ability to make the high-throughput measurements needed to accurately develop a (simulation) model [19]. With more and better data, we will be able to refine and retest the models until the predicted behavior (predicted by the model) accurately reflects what is observed in real life biological processes [22].

To model systems biology, a process algebra is needed. Several mathematical modelling tools are available nowadays [1,21,22,23,24], each with its advantages and disadvantages. In this research we chose the Brane Calculus as our formalism. We also extended this formalism with a stochastic and quantitative part to complement the original.



*Figure 1. Schematic diagram of systems biology and the range of disciplines involved.  
Taken from [15]*

## **2.1 Advantages of systems biology models**

The goal of the various formalisms is to model living phenomena as accurately as possible. When this succeeds, a powerful tool becomes available to speed up drug development. When models are able to simulate biological processes with enough accuracy, less in vitro and in vivo experiments are necessary since the same experiments can be carried out on a computer. Ultimately, this will lead to greatly reduced development times for all sorts of biological products and medicines in particular. When systems biology advances, it will lead to a better understanding of all living matter around us [22].

## **2.2 Applications**

As we further our understanding of biological systems it becomes possible to accurately model biological pathways (both disease and healthy), we will have an

immensely powerful tool at our disposal to aid in target and drug discovery. Biological pathways are in general very complex. It is not easy to predict the effect changing one constituent of the system has on the system as a whole. With an accurate prediction model it becomes possible to predict these effects. Target discovery will benefit tremendously since lead selection will deliver much better leads, cutting down on time as well as money needed. When data about potential drugs is added to the pathway model (with potential targets already identified in the previous step) drug development will benefit as well, since interactions (between for example proteins) can be predicted to a certain degree so in silico simulation will suffice saving valuable lab time and money again [18].

### **3 Introduction to the Brane Calculus**

#### **3.1 Overview**

Biological membranes have multiple functions and play part in a number of cellular processes. Membranes act as selective barriers which give cells their outer boundaries (plasma membrane), their inner compartment (organelles), and allow an oriented exchange of molecules, energies and information between the cell and the environment, and between certain intracellular compartments. Biological membranes are, therefore, more than just an inert barrier or covering; they play an active part in the life of the cell [25].

The most important feature of a biomembrane is that they form a two dimensional fluid (lipid bilayer) composed of a double layer of lipid class molecules, specifically phospholipids, with occasional proteins intertwined, some of which function as channels [25]. The lipid bilayer is embedded in a three-dimensional fluid(water).

The basic organization of the Brane Calculus is inspired by the fluid within fluid membrane structure; the Brane Calculi is characterised by two commutative monoides, each representing a kind of fluid.

In the Brane Calculus, the specific transformation selected are inspired by some biological constraints; however within the Brane calculus structure, such constraint can be refined or ignored. One of the constraints that may be adopted is the preservation of orientation which is related to Bitonality, which requires nested membrane to have opposite orientations as illustrated in the diagram below.

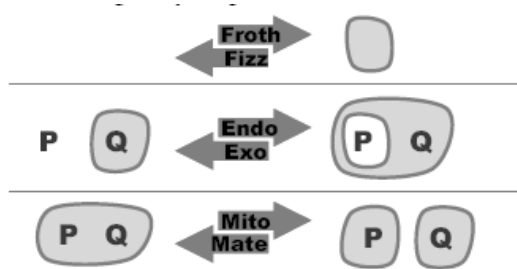


Figure 2. Examples of Bitonal reactions(from [1])

P and Q represent arbitrary subsystems; bitonality preservation means that the even/odd parity with which components are nested inside membranes must be preserved. P and Q remain on the same color background in each reaction; this means that in bitonal reactions there is never mixing of fluids from inside and outside any membrane.

### 3.2 Basic framework

#### Syntax and Reactions

##### Syntax

Systems	$P, Q ::= \diamond \mid P \circ Q \mid !P \mid \sigma \langle P \rangle$	nests of membranes
Branes	$\sigma, \tau ::= 0 \mid \sigma \tau \mid !\sigma \mid a.\sigma$	combinations of actions
Actions	$a, b ::= \dots$	(detailed later)

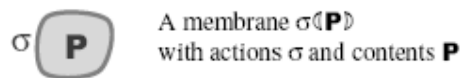


Figure 3. Brane Graphical Notation(from [1])

We abbreviate  $a.0$  as  $a$ , and  $0 \langle P \rangle$  as  $\langle P \rangle$ , and  $\sigma \langle \diamond \rangle$  as  $\sigma \langle \rangle$ .

Brane calculi Structures consist of two commutative monoids with replication:  $\circ$  is used for *systems* composition, with units  $(\diamond, \cdot)$  | is used for composition of *membranes*, with units  $(\circ, (!))$  for a model a “multitude” of components of the same kind.

As illustrated in the syntax below (from [1]), Systems consist of nested membranes; membranes consist of collections of actions.

Reactions happen only at the level of systems and are caused by action on membranes.

Actions may be bitonal actions of the membrane, bind and release, or molecular interactions.

### 3.3 Bitonal Interaction

#### Definition

Bitonal interactions are inspired by endocytosis and exocytosis. Endocytosis is a process whereby cells absorb material (molecules such as proteins) from outside by engulfing it in their cell membrane. Endocytosis is the opposite of exocytosis, and always involves the formation of a vesicle from part of the cell membrane

Endocytosis is divided into two processes: phagocytosis and pinocytosis.

Phagocytosis (literally, cell-eating) is the process by which cells ingest large objects, such as bacteria and viruses, and Pinocytosis (literally, cell-drinking) is the process concerned with the uptake of solutes and single molecules such as proteins by cells. Each action usually comes with a co-action that it is intended to interact with (pinocytosis does not have a co-action). (figure 4)

#### **Bitonal Action**

Actions  $a ::= \dots \vdash \vartheta_n \vdash \vartheta_n^\perp(\sigma) \vdash \vartheta_n \vdash \vartheta_n^\perp \vdash \otimes(\sigma)$  phago  $\vartheta$ , exo  $\vartheta$ , pino  $\otimes$

#### Derived Bitonal Interactions

Three bitonal operation can be derived from the previous phago/exo/pino, that is Mate Bud Drip; Mate cause irreversible membrane mixing, as in exo. Bud is similar to phago and Drip is similar to pino, but without co-action as illustrated in the diagram below.

.

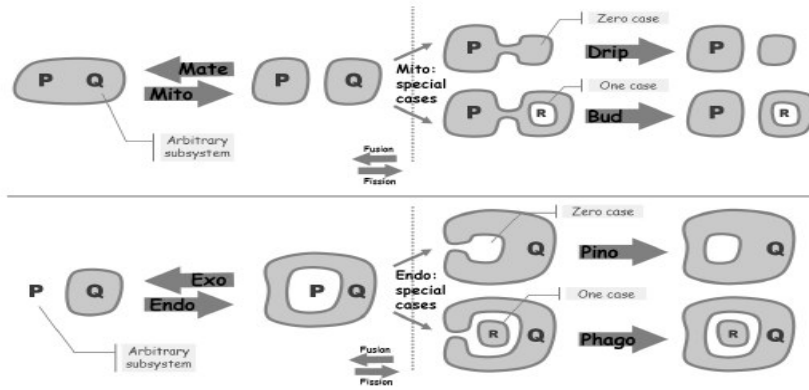


Figure 4. Representation of the principal membrane operation (from [2])

### 3.4 Molecules

Membranes can bind and release molecules on either sides of their surface as illustrated in the diagram below



Figure 5. Bind and Release (from [1])

Chemical reactions that are specific to a given compartment, molecular pumps and channels can be model by the bind and release action

### 3.5 Explanation of the rules

Luca Cardelli defined first a concrete syntax for membranes configuration (section 2.1). The syntax is factored by a congruence relation and a binary reduction relation is defined; the reduction relation includes the basic reductions that correspond to the basic operation of Brane Calculus. The transitive closure of the reduction relation describes the possible, non-deterministic, evolutions of a configuration.

Membrane structures that consist of a collection of nested membranes are formed of patches  $\sigma$ , where a patch can be a composition of sub-patches  $\sigma_1 / \sigma_2$ . An elementary patch consists of an action **a**. Actions come in complementary pairs that cause interaction between subsystems

Each Brane has a fixed collection of actions with a specific operational meaning in terms of reductions.

**Brane reactions :phago/exo/pino**

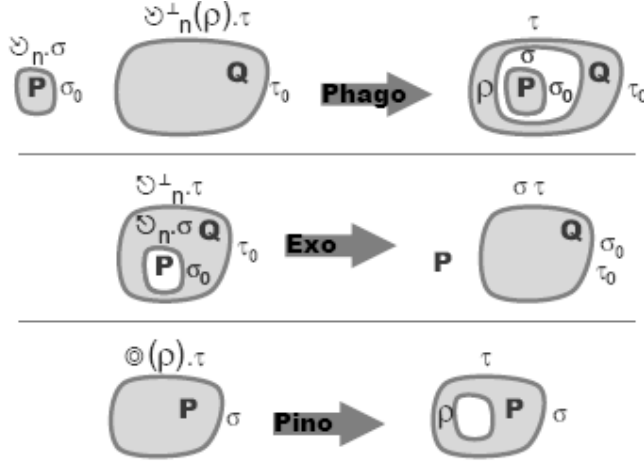


Figure 6. Phago, Exo, Pino actions operation (from [1])

$$\begin{aligned}
 \text{Phago} \quad & \vartheta_n.\sigma|\sigma_0\langle P \rangle \circ \vartheta_n^\perp(\rho).\tau|\tau_0\langle Q \rangle \longrightarrow \tau|\tau_0\langle \rho\langle \sigma|\sigma_0\langle P \rangle \rangle \circ Q \rangle \\
 \text{Exo} \quad & \vartheta_n^\perp.\tau|\tau_0\langle \vartheta_n.\sigma|\sigma_0\langle P \rangle \circ Q \rangle \longrightarrow P \circ \sigma|\sigma_0|\tau|\tau_0\langle Q \rangle \\
 \text{Pino} \quad & \theta(\rho).\sigma|\sigma_0\langle P \rangle \longrightarrow \sigma|\sigma_0\langle \rho\langle \diamond \rangle \circ P \rangle
 \end{aligned}$$

The phago action always comes with a complementary co-action

Here P; Q are arbitrary subsystems  $\sigma, \sigma_0, \tau, \tau_0, \rho$  are arbitrary patches, and the symbol  $\rightarrow$  means reduce to (a number of different reductions may be possible out of the same configuration).

The exo operation models the merging of two nested membranes, which starts with the membranes touching at a point. In this operation, the subsystem P gets expelled to the outside, and the residual patches are contiguous.

The pino action creates an empty bubble within the membrane where the pino action resides; we can imagine that the original membrane buckles towards the inside and pinches off. The patch on the empty bubble so created  $\rho$  is a parameter to the pino action

**Derivable reaction: mate/bud/drip**

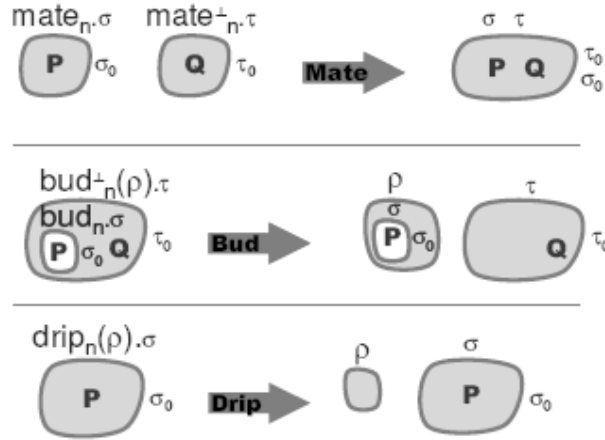
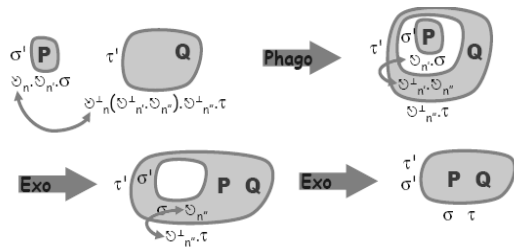


Figure 7. Mate, bud, drip operation (from [1])

$$\begin{aligned} \text{mate}_{n,\sigma} \circ \sigma \circ \sigma_0 \langle P \rangle &\circ \text{mate}_{n,\tau}^+ \circ \tau \circ \tau_0 \langle Q \rangle \longrightarrow * \sigma | \sigma_0 | \tau | \tau_0 \langle P \circ Q \rangle \\ \text{bud}_{n,\rho}^+ \circ \rho \circ \tau_0 \langle \text{bud}_{n,\sigma} \circ \sigma \circ \sigma_0 \langle P \rangle \circ Q \rangle &\longrightarrow * \rho \langle \sigma | \sigma_0 \langle P \rangle \rangle \circ \tau | \tau_0 \langle Q \rangle \\ \text{drip}_{n,\rho} \circ \rho \circ \sigma \circ \sigma_0 \langle P \rangle &\longrightarrow * \rho \langle \rangle \circ \sigma | \sigma_0 \langle P \rangle \end{aligned}$$

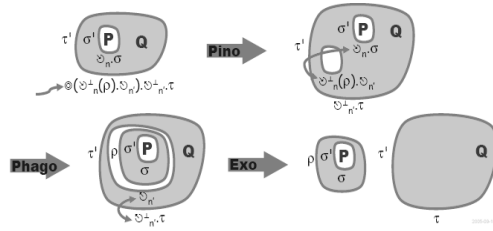
The mate merges two membranes like exo, but the membranes are not originally nested. The bud expels a membrane from inside another one the opposite of phago, but still wrapping an additional layer.

The drip operation produces an empty bubble like pino, but outside the membrane. The drip, mate, bud operations are expressible in the phago, exo, and pino by simple operations. All six operations have separate, direct, biological implementations, and they can all be considered as primitives.



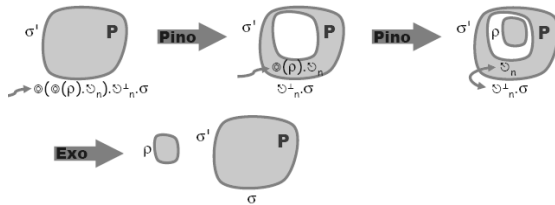
$$\begin{aligned} \text{mate}_{n,\sigma} \circ \sigma \circ \sigma_0 \langle P \rangle \circ \text{mate}_{n,\tau}^+ \circ \tau \circ \tau_0 \langle Q \rangle &= \\ \vartheta_n \circ \vartheta_{n'} \circ \sigma \circ \sigma_0 \langle P \rangle \circ \vartheta_n^+ \circ \vartheta_{n'}^+ \circ \tau \circ \tau_0 \langle Q \rangle &\xrightarrow{\text{Phago } n} \\ \vartheta_n^+ \circ \tau \circ \tau_0 \langle \vartheta_n^+ \circ \vartheta_{n'} \circ \sigma \circ \sigma_0 \langle P \rangle \rangle \circ Q &\xrightarrow{\text{Exo } n'} \\ \vartheta_n^+ \circ \tau \circ \tau_0 \langle \vartheta_n^+ \circ \sigma \circ \sigma_0 \langle P \rangle \rangle \circ P \circ Q &\xrightarrow{\text{Exo } n''} \\ \sigma | \sigma_0 | \tau | \tau_0 \langle P \circ Q \rangle & \end{aligned}$$

Figure 8. Abbreviation mate (from [1])



$$\begin{aligned} & \text{bud}_n^\perp(\rho). \tau \tau_0 \langle \text{bud}_n. \sigma | \sigma_0 \langle P \rangle \circ Q \rangle = \\ & \textcircled{\circ} (\vartheta_n^\perp(\rho). \vartheta_n). \vartheta_n^\perp. \tau \tau_0 \langle \vartheta_n. \sigma | \sigma_0 \langle P \rangle \circ Q \rangle \xrightarrow{\text{Pino}} \\ & \vartheta_n^\perp. \tau \tau_0 \langle \vartheta_n^\perp(\rho). \vartheta_n \langle \diamond \rangle \circ \vartheta_n. \sigma | \sigma_0 \langle P \rangle \circ Q \rangle \xrightarrow{\text{Phago}_n} \\ & \vartheta_n^\perp. \tau \tau_0 \langle \vartheta_n \langle \rho \langle \sigma | \sigma_0 \langle P \rangle \rangle \rangle \circ Q \rangle \xrightarrow{\text{Exo}_n} \\ & \rho \langle \sigma | \sigma_0 \langle P \rangle \rangle \circ \tau \tau_0 \langle Q \rangle \end{aligned}$$

Figure 9. Abbreviation bud (from [1])



$$\begin{aligned} & \text{drip}_n(\rho). \sigma | \sigma_0 \langle P \rangle = \\ & \textcircled{\circ} (\textcircled{\circ}(\rho). \vartheta_n). \vartheta_n^\perp. \sigma | \sigma_0 \langle P \rangle \xrightarrow{\text{Pino}} \\ & \vartheta_n^\perp. \sigma | \sigma_0 \langle \textcircled{\circ}(\rho). \vartheta_n \langle \diamond \rangle \rangle \circ P \rangle \xrightarrow{\text{Pino}} \\ & \vartheta_n^\perp. \sigma | \sigma_0 \langle \vartheta_n \langle \rho \langle \diamond \rangle \rangle \rangle \circ P \rangle \xrightarrow{\text{Exo}_n} \\ & \rho \langle \diamond \rangle \circ \sigma | \sigma_0 \langle P \rangle \end{aligned}$$

Figure 10. Abbreviation drip (from [1])

### 3.6 Brane molecule reaction

The operation here is the bind and release (B&R) and it essentially accounts for protein molecule including its interaction with membranes

systems	$P, Q ::= \dots \mid m$	$m \in M$ molecules
	$p, q ::= m_1 \circ \dots \circ m_k$	molecule multisets
actions	$\alpha ::= \dots \mid p_1(p_2) \Rightarrow q_1(q_2)$	bind&release



Figure 11. Bind and Release operation (from [1])

$$\text{B\&R} \quad p_1 \circ p_1(p_2) \Rightarrow q_1(q_2). \alpha | \sigma \langle p_2 \circ P \rangle \Rightarrow q_1 \circ \alpha | \sigma \langle q_2 \circ P \rangle$$

### 3.7 Example: Viral Infection and Reproduction

As a membrane computation, a virus infection can be summarized as followed

1. A virus approaches a cell, thus the cell “eat” the virus and form a vesicle from part of the membrane.

cell + virus phago cell (vesicle)

2. Inside the cell. The virus is approached by an endosome that tries to digest it.

Cell (endosome + vesicle) mate cell ((endosome + virus))

During the digestion, the virus is designed to interfere with the endosome and to release its RNA content into the cytoplasm of the cell.

Cell ((endosome + virus)) exo cell (endosome + vRNA)

4. Now the RNA of the virus is inside of the cell and it can be used for the proteins synthesis through cellular machineries.

Cell (endosome + vRNA) cellular machinery cell (viral protein + vRNA + viral envelope)

5. When the new virus components are build, they are assembled together and a new virus exits from the membrane of the cell.

Cell (viral protein + vRNA + viral envelope) bud cell (endosome) + virus

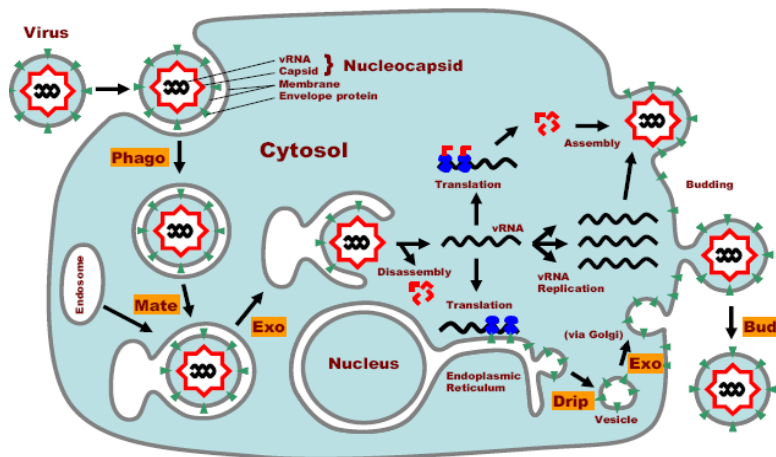


Figure 12. Viral infection and reproduction (from [1] p.279)

## 4 Stochastic Extension to the Brane Calculus

The original Brane Calculus as proposed by L. Cardelli [1] is a qualitative formalism. Since almost all biological processes involve some stochastics, an extension that includes this along with some quantitative aspects is desirable. We have attempted here to include these aspects. What follows is a short introduction to (general) stochastics and an overview of how we implemented this in our system.

## 4.1 Stochasticity

A stochastic process is a process whose behaviour is non-deterministic in the respect that the next state of the environment is partially but not fully determined by the previous state of the environment.

An example of a stochastic process in the natural world is pressure in a gas. Even though (theoretically speaking) each molecule is moving in a deterministic path, the motion of a collection of them is computationally and practically unpredictable. A large enough set of molecules will show stochastic characteristics, such as filling the container, exerting equal pressure, diffusing along concentration gradients, etc. These are observable properties of the system.

Most biological processes involve stochastics of some sort. For example, the process of proteins signalling cells is dependent on, among other things, the concentration of these proteins and cells. Just like the pressure of a gas it is unpractical to model this in a deterministic way. A good alternative in modelling this is to use stochastics.

In our research, stochastics are used to determine which reaction out of the possible reactions is going to occur in the next (time) step. This can be visualised as a number of membranes and proteins having the possibility to interact if they are in contact. Whether a specific reaction occurs depends on both the concentration and the reaction rate of this specific reaction. An algorithm is needed to computer the next reaction in line. We use Gillespie's algorithm which will be explained in the following section.

## 4.2 Gillespie's algorithm - overview

In 1977 Dan Gillespie published an algorithm [9] to compute the likelihood of reactions between molecules. The algorithm is particularly useful for simulating reactions within cells where the number of reactants (ligands on membranes, proteins, etc) is relatively low. Traditional continuous and deterministic biochemical rate equations are not able to accurately predict cellular reactions since they rely on bulk reactions that require the interactions of millions of molecules. These type of reactions are typically modeled as a set of coupled ordinary differential equations.

In contrast, the Gillespie algorithm allows a discrete and stochastic simulation of a system with few reactants because every reaction is explicitly simulated. When

simulated, the Gillespie represents a random walk that exactly represents the distribution of the Master Equation [10].

The physical basis of the algorithm is the collision of molecules within the environment where the reactions occur. It is assumed that collisions are frequent, but collisions with the orientation and energy needed for a reaction to occur are infrequent. Therefore, all reactions within the Gillespie framework involve at most two molecules. Reactions involving three molecules are assumed to be extremely rare (this is a very reasonable assumption since in real life processes these reactions seldom occur) and these reactions are modeled as a sequence of (binary) reactions.

Below is a summary of the steps to run the Gillespie algorithm:

1. **Initialization:** Initialize the number of molecules and membranes in the system, reaction rates and random number generators.
2. **Monte Carlo step:** Generate random numbers to determine the next reaction to occur as well as the time taken by this reaction.
3. **Update:** Increase the time by the randomly generated time in step 2. Update the molecule and membrane count based on the reaction selected in step 2.
4. **Iterate:** Go back to step 2 unless the number of reactants is zero or the simulation time has been exceeded.

So the likelihood of a certain reaction happening (this can be for example a Phagocytosis reaction or a Bind & Release action) depends on both the concentrations of the membranes and/or molecules involved and their corresponding reaction rates. A step by step analysis of the algorithm as implemented in our system is detailed in the next paragraph.

- 
1. For all  $x \in \hat{fn}(A)$  calculate  $a_x = \text{Act}_x(A) * \text{rate}(x)$
  2. Store non-zero values of  $a_x$  in a list  $(x_\mu, a_\mu)$ , where  $\mu \in 1 \dots M$ .
  3. Calculate  $a_0 = \sum_{\nu=0}^M a_\nu$
  4. Generate two random numbers  $n_1, n_2 \in [0, 1]$  and calculate  $\tau, \mu$  such that:

$$\tau = (1/a_0) \ln(1/n_1)$$

$$\sum_{\nu=1}^{\mu-1} a_\nu < n_2 a_0 \leq \sum_{\nu=1}^{\mu} a_\nu$$

5.  $\text{Next}(A) = x_\mu$  and  $\text{Delay}(A) = \tau$ .

**Definition 12.** *Calculating  $\text{Next}(A)$  and  $\text{Delay}(A)$  according to Gillespie [6].*

---

Figure 13. Overview of the Gillespie algorithm. Taken from [11]

### 4.3 Gillespie's algorithm - details

Figure 13 shows the algorithm as implemented in our Brane Calculus stochastic extension. In the first step all possible reactions are gathered (more details about how this is done can be found in the ‘Implementation’ section), including information about the quantities and reaction rates of the membranes and proteins involved.

$\text{Rate}(x)$  specifies the reaction rate of the reaction  $x$  and  $\text{Act}_x(A)$  the product of the quantities of both reactants (for example if there is a Phagocytosis reaction this product will be the amount of membranes containing the Phagocytosis action multiplied by the amount of membranes containing the Co-Phagocytosis action).

In step 2 the  $a$  value for each reaction is calculated. The  $a$  value of a reaction is the product of  $\text{Act}_x(A)$  times the  $\text{Rate}(x)$  value. This information is then stored in an easily accessible format. In step 3, a summation over all  $a$  values is computed, this gives us  $a_0$ . Now the actual computation of which reaction will happen next is performed. First two random numbers between 0 and 1 are generated:  $n_1$  and  $n_2$ . Now  $\tau$  represents the time it will take for the reaction to finish and iterating through all possible reactions, one of these is selected to be the next one to occur. As would be expected, reactions with a higher  $a$ -value (so either the reactants are present in higher concentrations, the reaction rate is higher, or both) have a higher chance of being selected. However, every possible reaction has a certain associated probability of being selected. In this way, it is ensured no dead-lock can occur, with one reaction completely taking over, for example.

In our extension, quantities are assigned to membranes and to molecules. Reaction rates are associated with the actions on the membranes.

In conventional brane calculus the Phagocytosis rule is as follows:

$$\nu_n \cdot \sigma \mid \sigma_0 \langle \mathbf{P} \rangle \circ \nu_n^\perp(\rho) \cdot \tau \mid \tau_0 \langle \mathbf{Q} \rangle \longrightarrow \tau \mid \tau_0 \langle \rho \langle \sigma \mid \sigma_0 \langle \mathbf{P} \rangle \rangle \circ \mathbf{Q} \rangle$$

*Equation 1a.*

Now with the stochastic extension, quantities and rates are added to this, resulting in the following rule (of course quantities and rates will vary):

$$\mathbf{1} \nu_n @ l \cdot \sigma \mid \sigma_0 \langle \mathbf{P} \rangle \circ \mathbf{1} \nu_n^\perp(\rho) @ l \cdot \tau \mid \tau_0 \langle \mathbf{Q} \rangle \longrightarrow \mathbf{1} \tau \mid \tau_0 \langle \rho \langle \sigma \mid \sigma_0 \langle \mathbf{P} \rangle \rangle \circ \mathbf{Q} \rangle$$

*Equation 1b.*

This represents the fact that one membrane with the Phagocytosis action and a membrane with the Co Phagocytosis action will result in a reaction with one membrane as shown on the right hand side of the arrow. Directly after the action symbol a rate has been added of the form '@rate'. Note that these rates are all relative, so a rate of one means one reaction occurs per given time unit. When two actions are involved in one reaction, the average rate is taken, but usually these rates will be the same for both actions.

## 5 Detailed outline of the biological systems implemented in Brane calculi

### 5.1 Macrophage function

Macrophages are types of white blood cell, or leucocyte, found in all vertebrate animals. They are specialized in the removal of bacteria and other micro-organisms, or of cell debris after injury. Like phagocytes, when a macrophage ingests a pathogen, the pathogen becomes trapped in a food vacuole, which then fuses with a lysosome. Within the lysosome, enzymes and toxic oxygen compounds digest the invader

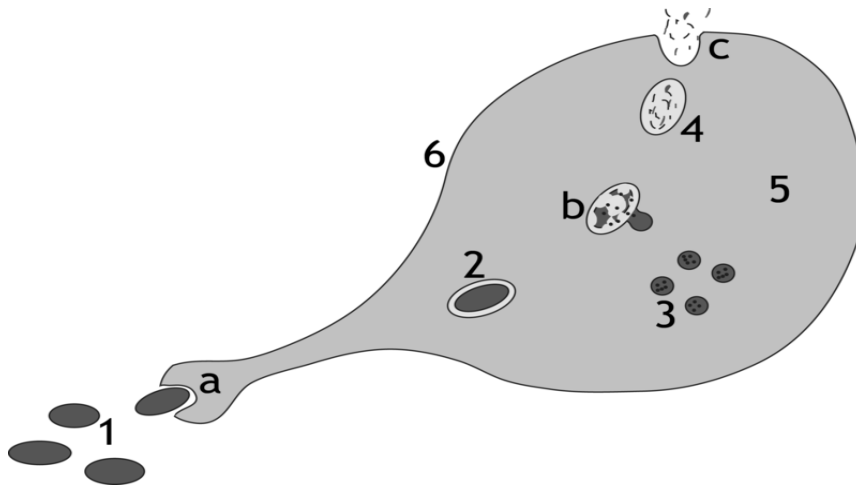


Figure 14. Macrophage ingesting a pathogen(from[3])

The process is as follows:

- a. Ingestion through phagocytosis; a phagosome is formed
- b. The fusion of lysosomes with the phagosome creates a phagosome; the pathogen is broken down by enzymes
- c. Waste material is expelled or assimilated (the latter not pictured)

**Parts:**

- 1.) Pathogens, 2.) Phagosomes, 3.) Lysosomes, 4.) Waste material, 5.) Cytoplasm
- 6.) Cell membrane

Pathogen  $\triangleq \exists n(x) \rightarrow$  a phago action on the membrane with content x  
 Membrane  $\triangleq !(mate)\exists^n! \exists^+ \rightarrow$  a phago co-action and an exo co-acton  
 Lysosome  $\triangleq !mate^+ \exists \langle \rangle \rightarrow$  membrane with a mate co-action and a phago action  
 Phagolysosome  $\triangleq !mate^+ \exists \langle \langle x \rangle \rangle$   
 X  $\triangleq$  wasted materials

The cell ingests the pathogen through the phago action and a phagosome is formed.

$$\exists n(x) \circ !(mate)\exists^n! \exists^+ \langle !mate^+ \exists \langle \rangle \circ Z \rangle \text{ phago}_n \rightarrow$$

In the cytoplasm the lysosome co-acts a mate action with the phagosome.

$$!(mate)\exists^n! \exists^+ \langle mate \langle \langle x \rangle \rangle \circ !mate^+ \exists \langle \rangle \circ Z \rangle \text{ mate} \rightarrow$$

After digestion by the lysosome enzymes, the wasted materials is coming out the cell by the exo action.

$$!(mate)\exists^n! \exists^+ \langle !mate^+ \exists \langle \langle x \rangle \rangle \circ Z \rangle \text{ exo} \rightarrow !(mate)\exists^n! \exists^+ \langle !mate^+ \exists \langle \rangle \circ Z \rangle \circ X$$

## 5.2 HIV life cycle

HIV stands for the Human Immunodeficiency Virus and is a Retrovirus. It is responsible for AIDS (Acquired Immune Deficiency Syndrome). It contains a special viral enzyme called Reverse Transcriptase, which allows the virus to convert its RNA to DNA and then integrate, and take over, a cell's own genetic material. Once taken over, the new cell - now HIV infected - begins to produce new HIV retroviruses. HIV replicates in and kills the helper T cells, which are the body's main defence against illness. The different stages in the HIV are illustrated in the following

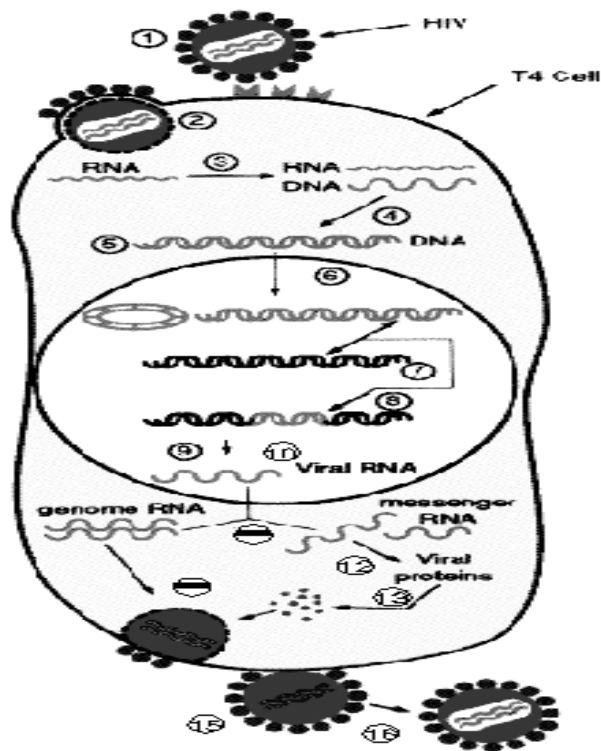


diagram.

Figure 15. HIV life cycle(from[4])

1. Attachment CD4-gp120 Interaction Gp120-Chemokine Receptor Interaction: virus binds to the cell at two receptors sites: CD4 and Chemokine receptor.
2. Viral Fusion/Uncoating: virus penetrates cell contents emptied into the cell
3. Reverse Transcription: single strands of virus RNA are converted into double-stranded DNA by the reverse transcriptase enzyme.
4. RNaseH Degradation
5. Second Strand Synthesis
6. Migration to Nucleus

7. Integration: viral DNA is combined with the cell's own DNA by the integrase enzyme
8. Latency
9. Early Transcription
10. Late Transcription: when the infected cell divides, the viral DNA is read and long chains of protein are made.
11. RNA Processing
12. Protein Synthesis
13. Protein Glycosylation
14. Assembly of Virion: sets of viral protein chains come together.
15. Viral Budding: immature virus pushes out the cell, taking some cell membrane with it.
16. Virion Maturation: The protein chains in the new viral particle are cut by the protease enzyme into individual proteins that combine to make a working virus.
17. Other: immature virus breaks free of the infected cell

Using the Brane formalism the overall cycle can be represented as follows

HIV virus  $\triangleq$   $\text{mate}_n(\text{mate}_{n'}(\mathcal{C}_n(\text{vRNA}))) \rightarrow$  three nested viral membranes: viral envelope with a mate action on it, nucleocapsid envelope with another mate action on it, and a supplementary membrane (with vRNA content) with an exo action on it.

T4 cells  $\triangleq$  membrane (nucleus o Z)  $\rightarrow$

Membrane  $\triangleq$   $!\text{mate}^+n!\mathcal{C}^+ \rightarrow$  a mate co-action and a phago action.

Nucleus  $\triangleq$   $!\text{mate}^+n'\mathcal{C}^+n(\mathcal{D}) \rightarrow$  membrane with two actions: a mate co-action and an exo action.

Viral-envelope  $\triangleq$   $\text{bud}^+(\text{mate}_{n2}) \rightarrow$

Nucap  $\triangleq$   $\text{capsid}(\text{mate}_{n2}(\mathcal{C}_n(\text{vRNA}))) \rightarrow$

Capsid  $\triangleq$   $!\text{bud}X \rightarrow$

### **Virus infection and replication**

The virus is going into the cell through a mate action that acts with a mate co-action in the cell's membrane.

$\text{mate}_n(\text{mate}_{n'}(\mathcal{C}_n(\text{vRNA}))) \circ !\text{mate}^+n!\mathcal{C}^+(\text{ o Z o } !\text{mate}^+n'\mathcal{C}^+n(\mathcal{D})) \text{ mate}n \rightarrow$

$!mate^+n \ !\vartheta^+ \ \llbracket \ mate_n \ \llbracket \vartheta_n \ \llbracket vRNA \rrbracket \rrbracket \circ Z \circ !mate^+n' \ | \vartheta^+n \ \llbracket \rrbracket \rrbracket$

$!\diamond(vRNA) \Rightarrow \diamond(vDNA)$  : reverse transcription in to the cytoplasm.

Migration to the cytoplasm, through a mate action.

$!mate^+n \ !\vartheta^+ \ \llbracket \ mate_n \ \llbracket \vartheta_n \ \llbracket vDNA \rrbracket \rrbracket \circ Z \circ !mate^+n' \ | \vartheta^+n \ \llbracket \rrbracket \rrbracket \ mate_n' \rightarrow$

$!mate^+n \ !\vartheta^+ \ \llbracket \ !mate^+n' \ | \vartheta^+n \ \llbracket \vartheta_n \ \llbracket vDNA \rrbracket \rrbracket \circ Z \rrbracket$

$!\diamond(vDNA) \Rightarrow \diamond(vRNA)$ : transcription in to the nucleus

$!mate^+n \ !\vartheta^+ \ \llbracket \ !mate^+n' \ | \vartheta^+n \ \llbracket \vartheta_n \ \llbracket vRNA \rrbracket \rrbracket \circ Z \rrbracket \ \text{exon} \rightarrow$

$!mate^+n \ !\vartheta^+ \ \llbracket \ !mate^+n' \ | \vartheta^+n \ \llbracket \rrbracket \circ vRNA \circ Z \rrbracket$

$vRNA \rightarrow vRNA \circ vRNA$ : viral replication.

Sythesis of viral proteins, envelope proteins through cellular machineries

$!vRNA(\diamond) \Rightarrow vRNA(\diamond).drip(\text{capsomers}) \ \llbracket \rrbracket$

$!vRNA(\diamond) \Rightarrow vRNA(\diamond).capsid$

$!vRNA(\diamond) \Rightarrow vRNA(\diamond).drip(\vartheta \ \text{viral-envelope}) \ \llbracket \ \text{nucleus} \rrbracket$

### Infected cell

Budding and liberation of new virus.

$!mate^+n2 \ !\vartheta^+ \ \llbracket \ \vartheta \ bud^+(mate_{n2}) \ \llbracket \rrbracket \circ !mate^+n2' \ | \ \vartheta_n \ \llbracket \rrbracket \circ !budIX \ \llbracket \ mate_{n2}' \ \llbracket \vartheta_{n2}(vRNA) \rrbracket \rrbracket \circ Z \rrbracket \ \text{exo} \rightarrow$

$!mate^+n2 \ !\vartheta^+ \ | \ bud^+ \ (mate_{n2}) \ \llbracket \ !bud \ | \ mate_{n2}' \ \llbracket \vartheta_{n2} \ \llbracket vRNA \rrbracket \rrbracket \rrbracket \circ !mate^+n2' \ | \ \vartheta_n \ \llbracket \rrbracket \circ Z \rrbracket$

$bud \rightarrow !mate^+n2 \ !\vartheta^+ \ \llbracket \rrbracket \circ Z \circ !mate^+n2' \ | \vartheta^+n \ \llbracket \rrbracket \rrbracket \circ mate_{n2} \ \llbracket \ mate_{n2}' \ \llbracket \vartheta_n \ \llbracket vRNA \rrbracket \rrbracket \rrbracket$

### 5.3 HCV life cycle

HCV (Hepatitis C Virus) is a hepatotropic virus that causes Hepatitis C, an infectious disease leading to liver inflammation. Chronic hepatitis can result later in cirrhosis and liver cancer. The

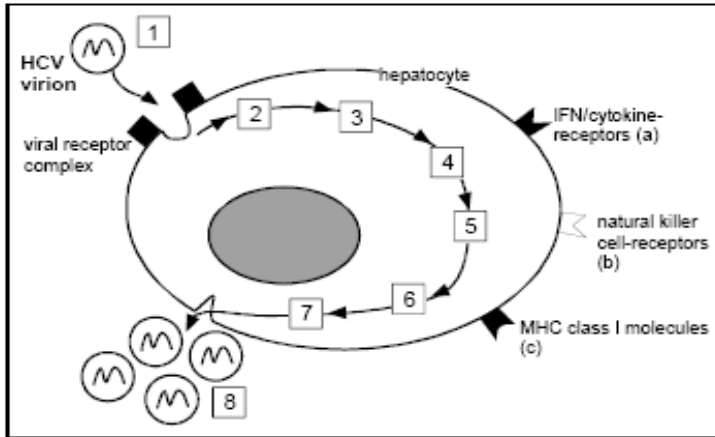


Figure 16. HCV life cycle (from [5])

The process is as follows:

Life cycle:

1. Binding of HCV to a cell surface receptor.
2. Cytoplasmic release and uncoating of the virus genome.
3. Translation
4. Polyprotein processing by cellular and viral proteases.
5. RNA replication
6. Packaging and assembly
7. Virion maturation.
8. Release from host cell.

Structures for defense

- a. Occupation of receptor leads to destruction leads to signal transduction (anti-viral status).
- b. Binding of NK cells leads to destruction of infected cell
- c. T cell epitopes of HCV presented on the MHC molecules target the infected cells for the attack by HCV-specific cytotoxic T-cells

9.

HCV virus	$\triangleq$	$\mathfrak{v}n\mathfrak{v}n(\mathfrak{v}RNA)$
Liver cell	$\triangleq$	membrane(Z)
Membrane	$\triangleq$	$! \mathfrak{v}^+n(\mathfrak{v}^+n)! \mathfrak{v}^+n'$
Vesicle	$\triangleq$	$\mathfrak{v}^+n (\mathfrak{v}n(\mathfrak{v}RNA))$
Viral-envelope	$\triangleq$	$bud^+ (\mathfrak{v}n\mathfrak{v}n)$

Nucap  $\triangleq$  capsid(vRNA)  
 Capsid  $\triangleq$  !budIX

### Virus infection and replication

The virus penetrates into the cell's cytoplasm through a phago action and a vesicle is formed.

$\vartheta_n \vartheta_n(vRNA) \circ !\vartheta^+_n(\vartheta^+_n)!!\vartheta^+_n'(\mathbb{Z}) \text{ phagon} \rightarrow$

The vesicle co-actio a exo action with virus to push the vRNA into the cytoplasm.

$!\vartheta^+_n(\vartheta^+_n)!!\vartheta^+_n'(\vartheta^+_n(\vartheta_n(vRNA)) \circ \mathbb{Z}) \text{ exon} \rightarrow$

$!\vartheta^+_n(\vartheta^+_n)!!\vartheta^+_n'(\vartheta^+_n(vRNA \circ \vartheta^+_n(\mathbb{D}) \circ \mathbb{Z}))$

$vRNA \rightarrow vRNA \circ vRNA$ : virus replication.

$!vRNA(\diamond) \Rightarrow vRNA(\diamond).drip(capsomers) (\mathbb{D})$

$!vRNA(\diamond) \Rightarrow vRNA(\diamond).capsid$

$!vRNA(\diamond) \Rightarrow vRNA(\diamond).drip(\vartheta \text{ viral-envelope}) (\mathbb{D} \text{ nucleus})$

### Infected cells

Budding and liberation of new virus ready to infect new cells

$!\vartheta^+_n2(\vartheta^+_n2)!!\vartheta^+_n'(\vartheta_n'.bud^+(\vartheta_n2\vartheta_n2) (\mathbb{D}) \circ !budIX (\vartheta_n2(vRNA) \circ \mathbb{Z}') \text{ exon}' \rightarrow$

$!\vartheta^+_n2(\vartheta^+_n2)!!\vartheta^+_n'(\vartheta_n2 \text{ bud}^+(\vartheta_n2\vartheta_n2) (\mathbb{D} !budIX (\vartheta_n2(vRNA) \circ \mathbb{Z}') \text{ bud} \rightarrow$

$!\vartheta^+_n2(\vartheta^+_n2)!!\vartheta^+_n'(\mathbb{Z}') \circ \vartheta_n2\vartheta_n2 (\vartheta_n2(vRNA))$

## 6 Brane calculus implementation

### 6.1 Introduction to Tom

Since our implementation makes extensive use of the Tom plugin [12], we will give a short introduction to Tom here.

Tom stands for 'ToOne Matching'. This stems from the original idea of adding pattern matching facilities and the fact that when solving a matching problem, only one subject is considered. It is a pattern matching compiler developed at INRIA [13]. It is

especially well-suited for programs involving transformations on trees and/or terms. It can also be used to parse XML documents.

Tom adds a new matching primitive to Java (we used Tom in combination with Java; it can also work as a plugin to C): `%match`. This construct is similar to the `match` primitive found in functional languages: given a term (called a subject) and a list of pairs of the form ‘pattern-action’, the `match` primitive selects a pattern that matches the subject and performs the corresponding action. This construct can be seen as an extension of the `switch/case` construct as used in Java. The main difference is that the discrimination occurs on the term level and not on the atomic values like characters or integers. These patterns are used to discriminate and retrieve information from an algebraic data structure. Tom also allows the definition of new, algebraic data structures, making it a very flexible extension. These aspects allow for programming by pattern matching. Since the brane calculus can be seen as a term rewriting language Tom has proved to be a very useful tool in implementing this formalism.

```
String who = "World";
%match(String who) {
    "World" -> { System.out.println("Hello " + who); }
    _       -> { System.out.println("Don't panic"); }
}
```

Figure 17. Example ‘`%match`’ construct in the Tom language

Consider figure 17: Tom will look for a match of the string ‘who’ by checking the patterns. The first pattern is “World”, so it will only match the ‘who’ input string if it is exactly that string. The ‘\_’ pattern is used as a wildcard, so it will match anything. This example is a very simple one, but much more complex patterns can be created as we will show later on in this chapter.

## 6.2 Implementation overview

For the implementation we made extensive use of the Tom plugin (described above). This allowed the code to be similarly structured as the brane calculus formal rules. Tom allows the definition of new algebraic data structures and that is what we used to define the brane calculus constituents. The definition follows the brane calculus

definitions as defined by L. Cardelli [1] closely, with a few exceptions for implementation convenience.

A brane calculus system consists of a membrane, a molecule or a composition of systems. A difference with the brane calculus definition here is that we implemented the composition construct ('comp') as a list instead of a binary relationship. This does not influence the way the formalism works, given that the original brane calculus has associative congruence rules stating that 'P o (Q o R) = (P o Q) o R'.

In our stochastic extension of the brane calculus. molecules and membranes have quantities associated with them. A membrane can be further broken down into a brane, upon which patches with actions reside, and the contents of the membrane. The contents consist of a further brane calculus system.

A brane can hold any number of patches and a single patch can hold a single action. Each action has a reaction rate associated with it, which will be used in the Gillespie algorithm. Using these constructs, a full brane calculus system can be built which then can be subjected to the program. The core of the program is the rule matching part.

Each rule in the brane calculus corresponds to a (Tom) pattern in the program against which a system can be matched. Sometimes more than one pattern is used, but this is only for symmetry reasons. That is, two rules might be explicitly defined to allow the matching of two associated actions (for example a Phagocytosis and a Co Phagocytosis action): one for the case where one action is before the other action in the list and one for the reverse. It might be possible to do this in just one rule but we feel this would reduce code readability, which was deemed more important here, especially considering the fact that this implementation might be used in further research.

An example of the Phagocytosis rule is given below:

$$\nu_n.\sigma \mid \sigma_0 \langle \mathbf{P} \rangle \circ \nu_n^\perp(\rho).\tau \mid \tau_0 \langle \mathbf{Q} \rangle \longrightarrow \tau \mid \tau_0 \langle \rho \langle \sigma \mid \sigma_0 \langle \mathbf{P} \rangle \rangle \circ \mathbf{Q} \rangle$$

Equation 2a.

In the implementation notation this rule is represented as shown in figure 18.

```

comp (A*,
  memb(brane(b1*, patch(a1@action(p1, r1, Phago(), col, Active()),
    id1), b2*), contents1, q1),
  B*,

```

```

memb(brane(b3*, patch(a2@action(p2, r2, Phago_co()), co2,
Active()), id2), b4*), contents2, q2),
C*) ->
{
  bSystem successorSystem = `comp(
  memb(brane(patch(processAction(a2), id2), b3*, b4*),
  comp(injectNil(memb(brane(co2),
  memb(brane(patch(processAction(a1), id1), b1*, b2*),
  contents1, newQuantity1), newQuantity2)),
  contents2),
  newQuantity1),
  memb(brane(b1*, patch(a1, id1), b2*), contents1,
  OldQuantity1),
  memb(brane(b3*, patch(a2, id2), b4*), contents2,
  OldQuantity2),
  A*, B*, C*);
}

```

Figure 18. Phagocytosis rule in the implementation

The different quantities (newQuantity1, newQuantity2, OldQuantity1, OldQuantity2) all depend on the quantities and rates as matched by r1, q1, r2 and q2. The left hand side of the rule is the part that will be matched against a system. Only the general structure and the Phago() and Phago\_co() are constraints; the rest are variables that will be instantiated during matching.

When a system is evaluated, this is done in a recursive way. So first the components of the system at the top level are investigated, then the systems that are inside membranes at this top level are examined and so forth.

The steps that the program goes through are as follows:

1. Read in the brane calculus system and add it to the stack
2. Evaluate the system on top of the stack and see if any rule can be applied
 

Two situations can occur:

  - I. A rule can be applied
    - a. The successor state and associated reaction rate and quantities of involved membranes are returned.
    - b. Since the system is a recursive one, it is only possible to accurately return

one system at a time. So the successor state is returned, but since more rules might be applicable, the system must be examined again with the rule that was just applied deactivated. This necessitates at least one and possible two more runs of the evaluating function since the rule might involve two associated actions (for example Phagocytosis and Co Phagocytosis) and both of these might be involved in another reaction.

So one or two new systems are added to the stack (depending on how many deactivated actions have to be considered).

- c. Remove the system currently under investigation from the stack
- d. Go back to the start of step 2

**II. No rule can be applied**

- a. Remove the system currently under investigation from the stack
- b. Go back to the start of step 2

3. When there are no more systems are on the stack, all rules that can possibly be applied to the current system have been gathered. The next step is to decide which reaction is going to happen. To do this an implementation of the Gillespie algorithm is used. A detailed explanation of this algorithm can be found in the section ‘Gillespie’s algorithm – details’.

Step 3 therefore consists of selecting which reaction is going to be applied.

4. When a reaction has been selected, the corresponding successor state is added to the, now empty, stack and the whole process is repeated from step 2 onwards.
5. The system halts when either no more successor states can be computed or the pre-configured iteration or time limit has been exceeded.

An example iteration of a simple brane calculus system can be found in appendix A.

## 7 Results

We were able to accurately implement the brane calculus as described in [1]. The following rules were implemented: Phagocytosis, Exocytosis, Pinocytosis, Mate, Bud, Drip and Bind & Release. With these rules, it was possible to implement the biological systems that we propose to simulate. We also implemented the viral infection example system as described in [1], an adaption of the Semliki Forest virus system[14]. Using the system with stochastics disabled and applying the appropriate rules, a viral infection loop (i.e. cell gets infected by virus, virus uses cell machinery to replicate, virus buds from cell, the cycle repeats itself).

Since multiple rules can be applied at various steps in the process, the appropriate rule needs to be selected by hand. This seems to be a drawback to the brane calculus in its current form, since there is no way to choose between applicable rules. This can be a cause for problems since sometimes the premature application of a specific rule can prevent the proper completion of a biological process. An example of this would be the usage of viral RNA to build a nucleocapsid before the viral RNA is replicated, thereby preventing the production of a viral envelope by the Endoplasmic Reticulum (viral RNA is needed to produce a viral envelope). The virus would not have any way to bud from the cell and the whole process halts (until new viral RNA is inserted into the cell).

Our stochastic extension solves this particular problem to some extent by associating a stochastic rate and a quantity to each membrane and molecule. By setting the appropriate rates and quantities, the system can be simulated without any user intervention, letting the Gillespie algorithm choose the rule to apply at each time step and setting the different quantities accordingly. This still results in some rules being applied prematurely but setting the rates appropriately minimizes this, resulting in an accurate simulation of the process. In our previous example this would mean that part of the viral RNA would be involved in the formation of nucleocapsids but a stochastically selected other part of the viral RNA would be involved in viral RNA reproduction and the production of viral envelopes thereafter. These rates usually are not given in the biological literature. Obviously these are all relative rates as well, so they must be calculated on a per-case basis. We used rates and quantities based on our own estimates. Setting these values is an arbitrary process but this is common to most stochastic systems, especially in biology.

The macrophage system is, relatively speaking, the simplest system. Simulating it in our implementation proved to be no problem and the results were as expected, mirroring the intermediate systems and resulting system as calculated by hand.

The HIV system is a more complex system and thus more error prone. The non-stochastic simulation with user guidance delivers the desired simulation results, going through the infection, replication and budding steps exactly as was calculated by hand. A trace of this system in action can be found in appendix A.

One disadvantage with the Brane calculus is that the rules can not be directly applied to all kind of biological systems. That is what happened with the HIV system; using the rules, we had to model something slightly different to the actual biological system. We modelled the virus as  $\text{mate}_n(\text{mate}_n(\text{v}_n(\text{vRNA})))$ , adding in a supplementary membrane, to make it work. To accurately reflect the HIV structure it should have been something like:  $\text{mate}_n(\text{mate}_n(\text{v}_n \text{vRNA}))$ .

The stochastic version without user guidance results in some non-functional membranes and molecules being produced but tweaking the initial reaction rates and quantities minimizes this and the main pathway delivers similar results as the non-stochastic one, with the added benefit of having quantities associated with each membrane and molecule. The most important part in the virus life cycle is the reaction catalysed by the reverse transcriptase (RT), (ie is the reverse transcription of vRNA in the cytoplasm of the T4 cell); each virus has two copies of RNA and the rate of infection is about  $10^{10}$  virions in 24 hours per virus, and that is if the two copies of vRNA have been transcribed. So to know if the infection succeeds or not depend on the the quantity of the vRNA and their ability to be transcribed.

The HCV system yields similar results as the simulation of the example viral infection system as introduced by L.Cardelli [1]. This is not surprising as large parts of the system are quite similar.

## 8 Conclusions and Future Work

### 8.1 Conclusions

We have implemented and extended the brane calculus [1]. Our implementation was used to simulate a number of biological processes. The simulation results were accurate and mirrored the examples given in the literature. The stochastic extension allows simulation of biological processes without user intervention and adds a quantitative element to the previously qualitative simulation. The added complexity to the system with the stochastic extension is the fact that quantities and reaction rates must be set. This is often an arbitrary process since the rates and quantities are not directly deducible from the biological description of a process. Quantification of the biological constituents in a reaction are still difficult to perform but would give good estimates to set the rates and quantities in the simulation model.

The brane calculus, with the stochastic extension, is a useful formalism when biological processes need to be described that involve membrane interactions. Other reactions can also be modelled, but a different formalism probably would be more appropriate. For systems like the ones we modelled the rules and syntax of the brane calculus are very suitable and intuitive. It can be quite complicated however, to build up a more complex system such as the HCV system. The fact that all reactions occur on the membrane level means a single brane on a membrane can become quite cluttered with patches. Of course, in real life the membranes are even more complex and littered with proteins sticking out so it is not necessarily a drawback that things can become complicated. A graphical notation of the system in addition to the brane calculus notation greatly improves readability especially since the location of each patch greatly influences the outcome of a system because only membranes that are orientated in the right direction can interact.

In conclusion our stochastic extension and implementation of the brane calculus form a step forward in the development and implementation of this formalism and in the modelling and automated analysis of biological systems. The biological processes we implemented show that the brane calculus has good potential in modelling and simulating real life biological pathways and systems.

## 8.2 Future work

The current representation of our system in code was kept as clean as possible but with larger systems it can become complicated to see what goes where. A user interface with some sort of graphical input and output would greatly improve the setting up of a system. Currently the system status (including all systems present and their quantities) is shown after each time step, the tracking of a specific type of system or a single system is not explicitly possible. A graph of quantities over time would also greatly enhance result checking after a simulation run.

The syntax and rules are also open to extension with some possible ones already mentioned in [1]. New rules should be straightforward to add to the already existing ones since no rule depends on another one, allowing for easy addition or removal of a particular rule.

It would also be interesting to see more biological systems simulated and see whether similar results to the ones we got in this research can be obtained.

Lastly it remains a challenge to simulate biological systems with accurate quantities and reaction rates. Improved data about these systems, obtained from wet lab experiments, will make simulations more accurate and increase their usefulness.

Ultimately, it is our hope that these models will help in drug discovery (through the prediction of target proteins and molecules) and drug development (through prediction of pathways and the effect specific proteins and medicinal agents will have on these pathways).

## 9 References

- [1] Luca Cardelli; Brane Calculi: Interactions of Biological Membranes draft 2005
- [2] Luca Cardelli; Abstract Machine of System Biology, 2005
- [3] <http://en.wikipedia.org/wiki/Macrophage>
- [4] [http://www.thebody.com/niaid/hiv\\_lifecycle/virpage.html](http://www.thebody.com/niaid/hiv_lifecycle/virpage.html)
- [5] [http://www.vhpb.org/files/html/Meetings\\_and\\_publications/VHPB\\_Meetings/geneva2002/S2AP1%](http://www.vhpb.org/files/html/Meetings_and_publications/VHPB_Meetings/geneva2002/S2AP1%20Grob.pdf) [6] Grob.pdf
- [7] Luca Cardelli. Bitonal membrane systems. Draft, 2003.
- [8] L.Cardelli, Brane calculi. Interactions of biological membranes, in: Proc. Computational
- [9] DT Gillespie, "Exact Stochastic Simulation of Coupled Chemical Reactions". The Journal of Physical Chemistry, Vol. 81, No. 25, 1977.
- [10] DT Gillespie, "A General Method for Numerically Simulating the Stochastic Time Evolution of Coupled Chemical Reactions", Journal of Computational Physics 2, 403-434 (1976).
- [11] Artificial Biochemistry slides, <http://www.luca.demon.co.uk/ArtificialBiochemistry.htm>
- [12] Tom: a software environment for defining transformations, <http://tom.loria.fr>
- [13] The French National Institute for Research in Computer Science and Control, <http://www.inria.fr/>
- [14] The Semliki Forest virus, [http://en.wikipedia.org/wiki/Semliki\\_forest\\_virus](http://en.wikipedia.org/wiki/Semliki_forest_virus)
- [15] Systems Biology Group IST/ISYS University of Stuttgart website. <http://www.sysbio.de>
- [16] Luca Cardelli. Brane calculi (slides). Slides., 2003.
- [17] H Kitano. Foundations of Systems Biology. MIT Press: 2001. ISBN 0262112663
- [18] E Klipp, R Herwig, A Kowald, C Wierling, and H Lehrach. Systems Biology in Practice. Wiley-VCH: 2005. ISBN 3527310789
- [19] G Bock and JA Goode (eds). "In Silico" Simulation of Biological Processes, Novartis Foundation Symposium 247. John Wiley & Sons: 2002. ISBN 0-470-84480-9

- [20] C. Priami. The Stochastic pi-calculus. *The Computer Journal* 38: 578-589, 1995
- [21] A. Regev, E.M. Panina, W. Silverman, L. Cardelli, E. Shapiro. Bioambients: An Abstraction for Biological Compartments. *Theoretical Computer Science*, Volume 325, Issue 1 , 28 September 2004, Pages 141-167.
- [22] Werner, E., "The Future and Limits of Systems Biology", *Science STKE* 2005, pe16 (2005).
- [23] C. Priami and P. Quaglia. Beta binders for biological interactions. In V. Danos and V. Vincent Schächter, editors, *Proc. 2nd Int. Workshop on Computational Methods in Systems Biology, CMSB '04, Lecture Notes in BioInformatics*. Springer, 2005.
- [24] V. Danos and J. Krivine. Formal molecular biology done in CCS-R. In *BioConcur '03, Workshop on Concurrent Models in Molecular Biology*, 2003.
- [25] B.Alberts, D.Bray, J.Lewis, M.Raff, K.Roberts, J.D.Watson. *Molecular Biology of the Cell*. Third Edition, Garland.

## **Acknowledgements**

We would like to thank Geoff Hamilton, our supervisor, for his help and advice throughout this project. Cecile would like to thank Gaston Tcheutchoua and their children (Aisling, Nelda and Linda). Jorma would like to thank his family and all his friends from LG27.