THE UNIVERSITY of EDINBURGH

Edinburgh Research Explorer

Statistical Model Checking of Membrane Systems with Peripheral Proteins: Quantifying the Role of Estrogen in Cellular Mitosis and DNA Damage

Citation for published version:
Cavaliere, M, Mazza, T & Sedwards, S 2014, Statistical Model Checking of Membrane Systems with Peripheral Proteins: Quantifying the Role of Estrogen in Cellular Mitosis and DNA Damage. in P Frisco, M Gheorghe & MJ Pérez-Jiménez (eds), Applications of Membrane Computing in Systems and Synthetic Biology. Emergence, Complexity and Computation, vol. 7, Springer International Publishing, pp. 43-63. DOI: 10.1007/978-3-319-03191-0_2

Digital Object Identifier (DOI):
10.1007/978-3-319-03191-0_2

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
Applications of Membrane Computing in Systems and Synthetic Biology

OPEN ACCESS

Chapter 1

Statistical Model Checking of Membrane Systems with Peripheral Proteins: Quantifying the Role of Estrogen in Cellular Mitosis and DNA Damage

Matteo Cavaliere[1], Tommaso Mazza[2] and Sean Sedwards[3]

Abstract Systems biology is a natural application of membrane systems, allowing the analysis of biological systems using the formal technique of model checking. To overcome the intractable model size of typical biological systems, *statistical* model checking may be used to efficiently estimate the probability of properties of interest with arbitrary levels of confidence. In this chapter we analyse a biological system linked to breast cancer, using statistical model checking (SMC) applied to membrane systems. To do this, we have constructed a computational platform that integrates an SMC library with a stochastic simulator of membrane systems with peripheral proteins. We present the methodology to investigate the role of estrogen in cellular mitosis and DNA damage and we use our statistical model checker to find the most appropriate time-dependent dosage of antagonist that should be used to minimize the uncontrolled replication of abnormal cells.

1.1 Membrane Systems with Peripheral Proteins

Membrane systems are models of computation inspired by the structure and function of biological cells. The model was introduced in 1998 by Gh. Păun and since then many classes of membrane systems have been introduced and studied, with mathematical, computer science and biological motivations. An introductory guide to the field can be found in the recent handbook [1], while a review of applications of membrane computing to biology can be found in [2].

---

[1] Spanish National Center for Biotechnology, Spain
e-mail:mcavaliere@cnb.csic.es
[2] IRCCS Casa Sollievo della Sofferenza - Mendel laboratory, Rome, Italy
e-mail:t.mazza@css-mendel.it
[3] INRIA Rennes - Bretagne Atlantique, France
e-mail:sean.sedwards@inria.fr

According to the original definition [3], membrane systems comprise an hierarchical nesting of membranes that enclose regions (representing the *the cellular structure*), in which free-floating objects (representing *molecules*) exist. Each region can have associated rules, called *evolution rules*, for evolving the free-floating objects and modelling the biochemical reactions present in cellular compartments. Rules also exist for moving objects in a synchronized manners across membranes, *symport* and *antiport* rules, modelling cellular transport and more general communication rules [1].

In brane calculi, presented in [4], several operations (*pino*, *exo*, *phago*, *mate*, *drip*, *bud*) explicitly involving membranes with embedded proteins are considered and formalised in the framework of process calculi. An important difference between brane calculi and membrane computing is that with brane calculi the evolution of the system takes place *on* the membranes and not inside the compartments delimited by them. In [5] the operations of brane calculi are represented in the membrane computing framework and then studied by using tools from formal language theory. In [6] some of the membrane operations (pinocytosis and dripping) are considered in combination with the presence of free-floating objects and objects attached to the membranes, while in [7] objects (peripheral proteins) are attached to either side of a membrane, explicitly considering the inner and outer membrane surfaces. The motivation for this last model is to represent the cellular processes that are controlled by the presence of specific proteins on the appropriate side of and integral to the membrane: there is a constant interaction between floating chemicals and embedded proteins and between peripheral and integral proteins [8]. Essential receptor-mediated processes, such as endocytosis and signalling, are crucial to cell function and by definition are critically dependent on the presence of peripheral and integral membrane proteins.

The key features of the model considered in [7] are that in each region of the system there are floating objects (the floating chemicals) and, in addition, objects can be associated to each side of a membrane or integral to the membrane (the peripheral and integral membrane proteins). Systems constructed using this model can perform the following operations: (i) the floating objects can be processed/changed inside the regions of the system (emulating biochemical reactions) and (ii) the floating and attached objects can be processed/changed when they interact (modelling the interactions of the floating molecules with membrane proteins). A possible use of the model to study biological processes is shown in [9, 7], while related models are discussed in [10] and [11], where also the computational aspects are presented.

The use of a formal computational model such as membrane systems can be helpful for two reasons: to facilitate the implementation of an executable specification and allows the use of automatic methods to analyse and to discover features concerning the dynamics of the complex cellular systems, providing in this way algorithms that can mimic biological phenomena, [12].

In this book chapter we approach this second possibility by presenting model checking, an algorithmic technique to formally verify the performance of a system with respect to a property. The system is represented in a language with formal semantics (in our case, multiset rewriting) and the property is expressed in temporal

logics (e.g., *linear temporal logic*, LTL, and *computation tree logic*, CTL). The output of standard model checking algorithms for these logics is Boolean: the model either satisfies the property or it does not. Such algorithms have polynomial time complexity with respect to the size of the model, but this is generally exponentially related to the description of the model (i.e., the number of interacting components). See [13] for a recent historical overview of standard model checking techniques and [14] for comprehensive coverage.

Many real systems are modelled using some form of non-determinism to account for unknown interactions and environments. In particular, chemical systems are often modelled with the implicit assumption that molecules move randomly, with probabilities of interaction proportional to the total number of molecules (so-called *mass action* [15]). When non-determinism is expressed with probabilities, it is possible to quantify the probability of a property using *probabilistic* model checking [16], which uses probabilistic or stochastic temporal logics (e.g. *probabilistic computaion tree logic*, PCTL, and *continuous stochastic logic*, CSL) and numerical techniques to calculate the probability of a property. Probabilistic model-checking algorithms use the standard model checking algorithms to evaluate whether a formula is satisfied in a particular state, but incur additional computational cost (polynomial w.r.t. the model) to calculate the probability of being in the state. Although techniques and data structures exist to minimise the model [14, 16, 13], in the majority of real applications (especially biological applications) the model remains intractable. Notable successes of standard and probabilistic model checking applied to simplified biological systems include [17, 12, 18, 19]. In particular, in [17] the authors have shown the use of probabilistic model checking for the analysis of the cell cycle in eukaryotes, using a modelling language based on membrane systems and process algebra. Platforms based on membrane systems including model checking have been previously implemented, [20]. A review of the probabilistic models in membrane systems can be found in [1].

To overcome the state space explosion problem that afflicts most biological models, in this chapter we employ statistical model checking (SMC), which is an efficient, approximative, variety of probabilistic model checking. SMC has been applied to biology before (e.g., [21]), but here we present the first SMC investigation of a biological system using a membrane systems model that explicitly considers the role of membrane proteins. To achieve this, we present the first self-contained statistical model checker dedicated to membrane systems with peripheral and integral proteins.

There are important differences between probabilistic and statistical model checking. The characteristic feature of a statistical model checker is that it estimates the probability of a property by verifying the property against multiple independent executions (simulations) of the system. The confidence of the estimate can be guaranteed to arbitrary levels of confidence by standard statistical bounds (e.g. the Chernoff bound [22]) and in this way SMC trades certainty for tractability. In comparison to standard and probabilistic model checking, SMC does not require a finite state space, does not require decidable logics and is less strict about how the system is defined. Importantly, SMC is often significantly more efficient than probabilistic

model checking for a given level of precision. Fig. 1.1 compares the performance of probabilistic with statistical model checking applied to increasing size models of the probabilistic dining philosophers protocol, considering the property that if a philosopher is hungry, he will eventually be fed. The figure shows that the numerical model checker [23] scales exponentially with increasing numbers of philosophers (i.e., polynomially w.r.t. a model that increases exponentially), while the statistical model checker [24] scales linearly (proportional to the length of the property).
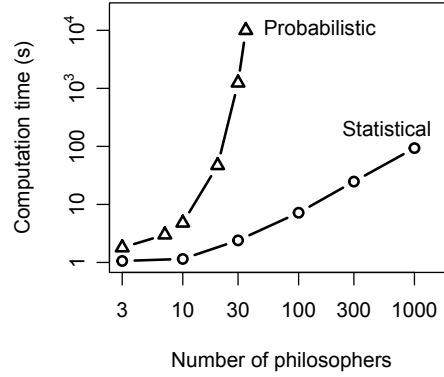
Fig. 1.1 Typical performance of probabilistic and statistical model checking applied to models of the probabilistic dining philosophers protocol.

Further details of our SMC methodology are given in Section 1.2.1.

1.1.1 Formal Language Preliminaries

Membrane systems are based on *formal language theory* and *multiset rewriting* [25]. In this section we recall the theoretical notions and notations necessary in this chapter.

Given the set A we denote by $|A|$ its cardinality and by $\varnothing$ the empty set. We denote by $\mathbb{N}$ and by $\mathbb{R}$ the set of natural and real numbers, respectively.

As usual, an *alphabet V* is a finite set of symbols. By V^* we denote the set of all strings over V. By V^+ we denote the set of all strings over V excluding the empty string. The empty string is denoted by λ. The *length* of a string v is denoted by $|v|$. The concatenation of two strings $u, v \in V^*$ is written uv.

The number of occurrences of the symbol a in the string w is denoted by $|w|_a$.

A *multiset* is a set where each element may have a multiplicity. Formally, a multiset over a set V is a map $M : V \to \mathbb{N}$, where $M(a)$ denotes the multiplicity of the symbol $a \in V$ in the multiset M.

For multisets M and M' over V, we say that M is *included in* M' if $M(a) \leq M'(a)$ for all $a \in V$. Every multiset includes the *empty multiset*, defined as M where $M(a) = 0$ for all $a \in V$.

The *sum* of multisets M and M' over V is written as the multiset $(M+M')$, defined by $(M+M')(a) = M(a) + M'(a)$ for all $a \in V$. The *difference* between M and M' is written as $(M-M')$ and defined by $(M-M')(a) = max\{0, M(a) - M'(a)\}$ for all $a \in V$. We also say that $(M+M')$ is obtained by *adding M to M'* (or vice versa) while $(M-M')$ is obtained by *removing M' from M*. For example, given the multisets $M = \{a,b,b,b\}$ and $M' = \{b,b\}$, we can say that M' is included in M, that $(M+M') = \{a,b,b,b,b,b\}$ and that $(M-M') = \{a,b\}$.

If the set V is finite, e.g. $V = \{a_1,\ldots,a_n\}$, then the multiset M can be explicitly described as $\{(a_1,M(a_1)),(a_2,M(a_2)),\ldots,(a_n,M(a_n))\}$. The *support* of a multiset M is defined as the set $supp(M) = \{a \in V \mid M(a) > 0\}$. A multiset is empty (hence finite) when its support is empty (also finite).

A compact string notation can be used for finite multisets: if $M = \{(a_1,M(a_1)), (a_2,M(a_2)),\ldots,(a_n,M(a_n))\}$ is a multiset of finite support, then the string $w = a_1^{M(a_1)} a_2^{M(a_2)} \ldots a_n^{M(a_n)}$ (and all its permutations) precisely identify the symbols in M and their multiplicities. Hence, given a string $w \in V^*$, we can say that it identifies a finite multiset over V, written as $M(w)$, where $M(w) = \{a \in V \mid (a,|w|_a)\}$. For instance, the string *bab* represents the multiset $M(w) = \{(a,1),(b,2)\}$, that is the multiset $\{a,b,b\}$. The empty multiset is represented by the empty string λ.

1.1.2 Membrane Systems with Peripheral and Integral Proteins

Formal language theory can be used to provide a mathematical abstraction for the bidirectional interactions of floating molecules with cell membranes: biochemical rules and interactions between peripheral proteins and membranes can be formalised in terms of multiset rewriting rules.

In this section we introduce the main notions for membrane systems with peripheral and integral proteins.

As it is usual in the membrane systems field, a membrane is represented by a pair of square brackets, $[\,]$. A *membrane structure* is an *hierarchical nesting* of membranes enclosed by a main membrane called the *root membrane*. A label is associated to each membrane and it is written as a superscript of the membrane, e.g. $[\,]^1$. If a membrane has the label i we call it membrane i. Each membrane is identified by a unique label in an unique manner (there are no membranes with the same label).

A membrane structure is essentially that of a tree data structure, where the nodes are the membranes and the arcs represent the containment relation. Being a tree, a membrane structure can be represented by a string of matching square brackets, e.g., $[\,[\,[\,]^2\,]^1\,[\,]^3\,]^0$.

To each membrane there are associated three multisets, u, v and x over V, denoted by $[\ ]_{u|v|x}$, where V denotes a finite alphabet of *objects* (the symbol | is not part of the alphabet V).

Following the terminology used in [9] we say that the membrane is *marked* by u, v and x; x is called the *external marking*, u the *internal marking* and v the *integral marking* of the membrane. In general, we refer to them as *markings* of the membrane.

The internal, external and integral markings of a membrane model the proteins attached to the internal surface, to the external surface and integral to the membrane, respectively.

In a membrane structure, the region between membrane i and any enclosed membranes is called region i. To each region is associated a multiset of objects w called the *free objects* of the region. The free objects are written between the brackets enclosing the regions, e.g., $[\ aa\ [\ bb\ ]^1\ ]^0$. The free objects of a membrane model the floating chemicals within the regions of a cell.

We denote by $int(i)$, $ext(i)$ and $itgl(i)$ the internal, external and integral markings of membrane i, respectively. By $free(i)$ we denote the free objects of region i. For any membrane i, distinct from a root membrane, we denote by $out(i)$ the label of the membrane enclosing membrane i. The finite set of all possible labels is denoted by *Lab*.

The string

$$[\ ab\ [\ cc\ ]^2_{a|\ |}\ [\ abb\ ]^1_{bba|ab|c}\ ]^0$$

represents, for instance, a membrane structure, where to each membrane are associated markings and to each region are associated free objects. Membrane 1 is internally marked by bba (i.e., $int(1) = bba$), has integral marking ab (i.e., $itgl(1) = ab$) and is externally marked by c (i.e., $ext(1) = c$). To region 1 are associated the free objects abb (i.e., $free(1) = abb$). To region 0 are associated the free objects ab. Finally, $out(1) = out(2) = 0$. Membrane 0 is the root membrane. The string can also be depicted diagrammatically, as in Fig. 1.2.
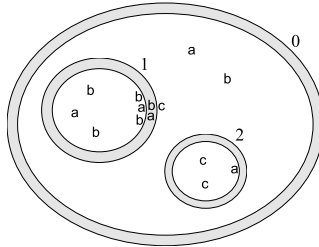
Fig. 1.2 Graphical representation of $[\ ab\ [\ cc\ ]^2_{a|\ |}\ [\ abb\ ]^1_{bba|ab|c}\ ]^0$

As in [9] we consider the rules *attach$_{in}$*, *attach$_{out}$*, *de-attach$_{in}$* and *de-attach$_{out}$*, defined in the following manner:

$$attach_{in} : [\ \alpha\ ]^i_{u|v|} \rightarrow [\ ]^i_{u'|v'|}\,,\ \ \alpha \in V^+, u, v, u', v' \in V^*, i \in Lab$$

$$attach_{out} : [\ ]^i_{|v|x}\ \alpha \rightarrow [\ ]^i_{|v'|x'}\,,\ \ \alpha \in V^+, v, x, v', x' \in V^*, i \in Lab$$

$$de\text{-}attach_{in} : [\ ]^i_{u|v|} \rightarrow [\ \alpha\ ]^i_{u'|v'|}\,,\ \ \alpha, u', v', u, v \in V^*, |uv| > 0, i \in Lab$$

$$de\text{-}attach_{out} : [\ ]^i_{|v|x} \rightarrow [\ ]^i_{|v'|x'}\alpha,\ \ \alpha, v', x', v, x \in V^*, |vx| > 0, i \in Lab$$

Using the notion of multiset presented earlier, we describe the formal semantics of the rules.

The *attach_{in}* rule is *applicable to membrane i* if $free(i)$ includes α, $int(i)$ includes u and $itgl(i)$ includes v. When the rule *is applied to membrane i*, α is removed from $free(i)$, u is removed from $int(i)$, v is removed from $itgl(i)$, u' is added to $int(i)$ and v' is added to $itgl(i)$. The objects not involved in the application of the rule are left unchanged in their original positions.

The *attach_{out}* rule is applicable to membrane i if $free(out(i))$ includes α, $itgl(i)$ includes v, $ext(i)$ includes x. When the rule is applied to membrane i, α is removed from $free(out(i))$, v is removed from $itgl(i)$, x is removed from $ext(i)$, v' is added to $itgl(i)$ and x' is added to $ext(i)$. The objects not involved in the application of the rule are left unchanged in their original positions.

The *de-attach_{in}* rule is applicable to membrane i if $int(i)$ includes u and $itgl(i)$ includes v. When the rule is applied to membrane i, u is removed from $int(i)$, v is removed from $itgl(i)$, u' is added to $int(i)$, v' is added to $itgl(i)$ and α is added to $free(i)$. The objects not involved in the application of the rule are left unchanged in their original positions.

The *de-attach_{out}* rule is applicable to membrane i if $itgl(i)$ includes v and $ext(i)$ includes x. When the rule is applied to membrane i, v is removed from $itgl(i)$, x is removed from $ext(i)$, v' is added to $itgl(i)$, x' is added to $ext(i)$ and α is added to $free(out(i))$. The objects not involved in the application of the rule are left unchanged in their original positions.
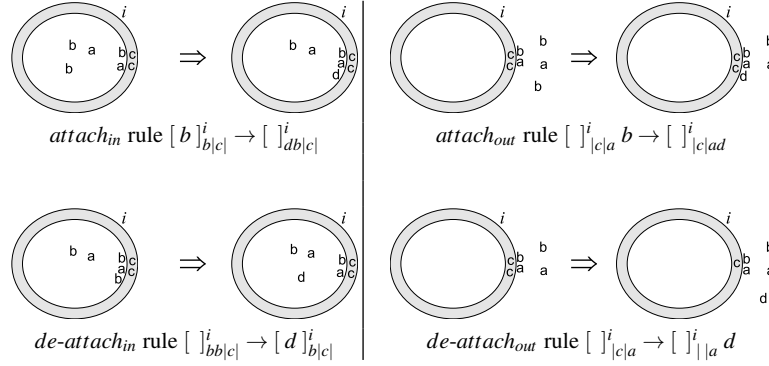
Instances of *attach_{in}*, *attach_{out}*, *de-attach_{in}* and *de-attach_{out}* rules are depicted in Fig. 1.3.

Extending the model in [9], we also consider rules that model the shuttling and translocation of proteins across membranes, as those considered in [10] and [11]. In this case, floating objects can cross membranes depending on the proteins present on the membrane (proteins may also change during the translocation).

$$shuttle_{out} : [\ \alpha\ ]^i_{u|v|} \rightarrow [\ ]^i_{u'|v'|}\ \alpha,\ \ \alpha \in V^+, u, v, u', v' \in V^*, i \in Lab$$

$$shuttle_{in} : [\ ]^i_{|v|x}\ \alpha \rightarrow [\ \alpha]^i_{|v'|x'}\,,\ \ \alpha \in V^+, v, x, v', x' \in V^*, i \in Lab$$

The operation of translocation and shuttling can be envisaged as an instantaneous combination of attach and de-attach rules described above. The *shuttle_{in}* rule is

$attach_{in}$ rule $[\,b\,]^i_{b|c|} \rightarrow [\ ]^i_{db|c|}$ $attach_{out}$ rule $[\ ]^i_{|c|a}\, b \rightarrow [\ ]^i_{|c|ad}$

$de\text{-}attach_{in}$ rule $[\ ]^i_{bb|c|} \rightarrow [\,d\,]^i_{b|c|}$ $de\text{-}attach_{out}$ rule $[\ ]^i_{|c|a} \rightarrow [\ ]^i_{|\,|a}\, d$

Fig. 1.3 Examples of $attach_{in}$, $attach_{out}$, $de\text{-}attach_{in}$ and $de\text{-}attach_{out}$ rules, showing how free and attached objects may be rewritten. For example, in the $attach_{in}$ rule one of the two free instances of b is rewritten to d and added to the membrane's internal marking.

applicable to membrane i if $itgl(i)$ includes v, $ext(i)$ includes x and $free(out(i))$ includes α. When the rule is applied to membrane i, v is removed from $itgl(i)$, x is removed from $ext(i)$, v' is added to $itgl(i)$, x' is added to $ext(i)$ and α is added to $free(i)$. The objects not involved in the application of the rule are left unchanged in their original positions.

The $shuttle_{out}$ rule is applicable to membrane i if $itgl(i)$ includes v, $int(i)$ includes u and $free(i)$ includes α. When the rule is applied to membrane i, v is removed from $itgl(i)$, u is removed from $int(i)$, v' is added to $itgl(i)$, u' is added to $int(i)$ and α is added to $free(out(i))$. The objects not involved in the application of the rule are left unchanged in their original positions.

$shuttle_{in}$ rule $b[\,]^i_{|c|} \rightarrow [\,b\,]^i_{|c|}$ $shuttle_{out}$ rule $[\,b\,]^i_{|c|} \rightarrow [\ ]^i_{|c|}\, b$
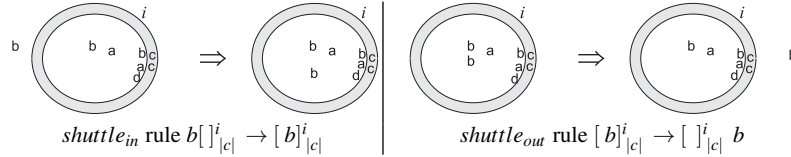
Fig. 1.4 Examples of $shuttle_{in}$ and $shuttle_{out}$ rules, showing how free objects can cross membranes with the help of peripheral and integral proteins.

We denote by $\mathscr{R}^{att}_{V,Lab}$ the set of all possible *attach*, *de-attach* and *shuttle* rules over the alphabet V and set of labels Lab.

As in [9], we also consider *evolution rules* that replace the free objects contained in a region conditional on the markings of the enclosing membrane. These rules represent the biochemical reactions that take place within the cytoplasm of a cell. An evolution rule has the following syntax:

$$evol : [\,\alpha \rightarrow \beta\,]^i_{u|v|}$$

where $u, v, \beta \in V^*$, $\alpha \in V^+$, and $i \in Lab$.

The semantics of the rule is as follows. The rule is applicable to region i if $free(i)$ includes α, $int(i)$ includes u and $itgl(i)$ includes v. When the rule is applied to region i, α is removed from $free(i)$ and β is added to $free(i)$. The membrane markings and the objects not involved in the application of the rule are left unchanged in their original positions.

We denote by $\mathscr{R}^{ev}_{V,Lab}$ the set of all evolution rules over the alphabet V and set of labels Lab. An instance of an evolution rule is represented in Fig. 1.5.
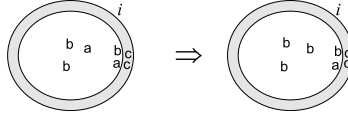
Fig. 1.5 *evol* rule $[\,a \rightarrow b\,]^i_{b|c|}$. Free objects can be rewritten inside the region and the rewriting can depend on the integral and internal markings of the enclosing membrane.

In general, when a rule has label i we say that a rule is associated to *membrane i* (in the case of *attach* and *de-attach* rules) or is associated to *region i* (in the case of *evol* rules). For instance, in Fig. 1.3 the $attach_{in}$ is associated to membrane i.

The objects of α, u and v for $attach_{in}$/$evol$ rules, of α, v and x for $attach_{out}$ rules, of u and v for de-$attach_{in}$ rules and of v and x for de-$attach_{out}$ rules are the *reactants* of the corresponding rules. E.g., in the attach rule $[\,b\,]_{a|c|} \rightarrow [\,]_{d|c|}$, the reactants are a, b and c.

A membrane system with peripheral and integral protein is a mathematical model that considers membranes to which can be associated peripheral proteins, integral proteins, free objects and using the operations described above. The rules presented here can be implemented in the computational tool using the appropriate syntax [26], and present redundancies whose purpose is to allow flexibility during the modelling processes. Several specific restricted variants of the proposed rules have been investigated and a review can be found in [10].

Here, following [9], we consider the stochastic extension of the model.

Definition 1. A *stochastic membrane system with peripheral and integral proteins* and n membranes is a construct

$$\Pi = (V_\Pi, \mu_\Pi, (u_0, v_0, x_0)_\Pi, \ldots, (u_{n-1}, v_{n-1}, x_{n-1})_\Pi, w_{0,\Pi}, \ldots, w_{n-1,\Pi}, R_\Pi,$$
$$t_{in,\Pi}, t_{fin,\Pi}, rate_\Pi)$$

- V_Π is a non-empty alphabet of objects.
- μ_Π is a membrane structure with $n \geq 1$ membranes injectively labelled by labels in $Lab_\Pi = \{0, 1, \cdots, n-1\}$, where 0 is the label of the root membrane.

- $(u_0, v_0, x_0)_\Pi = (\lambda, \lambda, \lambda), (u_1, v_1, x_1)_\Pi, \cdots, (u_{n-1}, v_{n-1}, x_{n-1})_\Pi \in V^* \times V^* \times V^*$ are called *initial markings of the membranes*.
- $w_{0,\Pi}, w_{1,\Pi}, \cdots, w_{n-1,\Pi} \in V^*$ are called *initial free objects of the regions*.
- $R_\Pi \subseteq \mathscr{R}^{att}_{V, Lab_\Pi - \{0\}} \cup \mathscr{R}^{ev}_{V, Lab_\Pi}$ is a finite set of evolution rules, attach/de-attach and shuttle rules.
- $t_{in,\Pi}, t_{fin,\Pi} \in \mathbb{R}$ are called the *initial time* and the *final time*, respectively.
- $rate_\Pi : R_\Pi \longmapsto \mathbb{R}$ is the *rate mapping*. It associates to each rule a *reaction rate*.

An *instantaneous description I* of Π consists of the membrane structure μ_Π with markings associated to the membranes and free objects associated to the regions. We denote by $\mathbb{I}(\Pi)$ the set of *all instantaneous descriptions* of Π. We say *membrane (region) i of I* to denote the membrane (region, respectively) i present in I.

Let I be an arbitrary instantaneous description from $\mathbb{I}(\Pi)$ and r an arbitrary rule from R_Π. Suppose that r is associated to membrane $i \in Lab_\Pi$ if $r \in \mathscr{R}^{att}_{V, Lab_\Pi - \{0\}}$ (or to region $i \in Lab_\Pi$ if $r \in \mathscr{R}^{ev}_{V, Lab_\Pi}$).

If r is applicable to membrane i (or to region i, accordingly) of I, then we say that *r is applicable to I*. We denote by $r(I) \in \mathbb{I}(\Pi)$ the instantaneous description of Π obtained when the rule r is applied to membrane i (or to region i, accordingly) of I (in short, we say *r is applied to I*).

The *initial instantaneous description* of Π, $I_{in,\Pi} \in \mathbb{I}(\Pi)$, consists of the membrane structure μ_Π with membrane i marked by $(u_i, v_i, x_i)_\Pi$ for all $i \in Lab_\Pi - \{0\}$ and free objects $w_{i,\Pi}$ associated to region i for all $i \in Lab_\Pi$.

A *configuration* of Π is a pair (I, t) where $I \in \mathbb{I}(\Pi)$ and $t \in \mathbb{R}$; t is called the *time* of the configuration. We denote by $\mathscr{C}(\Pi)$ the set of all configurations of Π. The *initial configuration* of Π is $C_{in,\Pi} = (I_{in,\Pi}, t_{in,\Pi})$.

Suppose that $R_\Pi = \{rule^1, rule^2, \ldots, rule^m\}$ and let S be an arbitrary sequence of configurations $\langle C_0, C_1, \cdots, C_j, C_{j+1}, \cdots, C_h \rangle$, where $C_j = (I_j, t_j) \in \mathscr{C}(\Pi)$ for $0 \leq j \leq h$. Let $a_j = \sum_{i=1}^{m} p_j^i$, $0 \leq j \leq h$, where p_j^i is the product of $rate(rule^i)$ and the *mass action* combinatorial factor [27] for $rule^i$ and I_j.

The sequence S is an *evolution* of Π if

- for $j = 0$, $C_j = C_{in,\Pi}$
- for $0 \leq j \leq h-1$, $a_j > 0$, $C_{j+1} = (r_j(I_j), t_j + dt_j)$ with r_j, dt_j as in [27]:

    - $r_j = rule^k$, $k \in \{1, \cdots, m\}$ and k satisfies $\sum_{i=1}^{k-1} p_j^i < ran'_j \cdot a_j \leq \sum_{i=1}^{k} p_j^i$

    - $dt_j = (-1/a_j) ln(ran''_j)$

    where ran'_j, ran''_j are two random variables over the sample space $(0, 1]$, uniformly distributed.
- for $j = h$, $a_j = 0$ or $t_j \geq t_{fin,\Pi}$.

In other words, an evolution of Π is a sequence of configurations $\langle C_0, C_1, \cdots, C_j, C_{j+1}, \cdots, C_h \rangle$, starting from the initial configuration of Π, where,

given the current configuration $C_j = (I_j, t_j)$, the next one $C_{j+1} = (I_{j+1}, t_{j+1})$, is obtained by applying the rule r_j to the current instantaneous description I_j and adding dt_j to the current time t_j. The rule r_j and the associated dt_j are determined by Gillespie's theory of chemically reacting systems [15] applied to the current instantaneous description I_j (i.e., effectively, the rule with the shortest waiting time is selected to be executed). The evolution of the system halts when all rules have probability zero to be executed (following from the fact that $a_j = 0$) or when the current time is greater or equal to the specified final time. The detailed explanation of the application of Gillespie's stochastic simulation algorithm [15] to a membrane system with peripheral proteins can be found in [9] and [28]. Similar implementations of the Gillespie algorithm have been already proposed in different models of membrane systems where the algorithm has been specifically adapted and optimized to run in multiple compartments and, for these reasons, referred to as multi-compartment in [17] and [20].

1.2 Statistical Model Checking for Membrane Systems with Peripheral Proteins

A stochastic membrane system with peripheral proteins can capture the essential dynamics of a cellular system. It may be used to address questions concerning the interplay between the biochemical processes present in the various compartments and the proteins associated to the cellular membranes. Sometimes, these questions can be resolved in an analytical manner [29] or by executing the model on a computer, [9, 30, 28]. In this chapter we use this latter approach (for an analytical approach see the review [10]). As defined in Section 1.1.2, a single evolution of the system produces an outcome that represents the quantities of the involved entities, floating molecules, peripheral proteins and compartments. However, because of the stochastic applications of the rules, each evolution of the system may lead to (a possibly very large number of) different outcomes.

In what follows, we describe the use of statistical model checking to investigate biological systems, where properties of interest are specified using temporal logic.

1.2.1 Temporal logic as a query language

Many useful properties of systems can be expressed in terms of maxima, minima or averages of system variables. With more complex reactive systems, such as biological systems, it is often desirable to investigate properties that comprise sequences of events and events dependent on time. Temporal logic provides a formal means to express these properties and remains reasonably intuitive for moderately complex properties. In this chapter we use temporal logic as a query language to investigate temporal properties of a biological system. We have developed a statistical

model checker based on the logic of PLASMA [24]. This logic is also similar to the *bounded linear temporal logic* (BLTL) of [21]. Specifically, we have constructed a tool using PLASMA-lab [31, 32], a statistical model checking library that works with an external simulator. We have thus created a simulator that implements a language of stochastic membrane systems with peripheral and integral proteins [33].

For the purposes of exposition, a logical property ϕ of our model checker is constructed using the following abstract syntax:

$$\phi = \phi \vee \phi \mid \phi \wedge \phi \mid \neg\phi \mid F_{\leq b}\phi \mid G_{\leq b}\phi \mid \phi\, U_{\leq b}\phi \mid X\phi \mid \alpha$$
$$\alpha = numeric\,(\,>\mid\geq\mid=\mid\leq\mid<\,)\,numeric$$

$\vee, \wedge$ and $\neg$ are the standard logical connectives *or*, *and* and *not*. α is an atomic proposition constructed from numeric expressions of constants and system variables using the standard relational operators. X is the *next* temporal operator ($X\phi$ is true iff ϕ is true on the next step). F, G and U are temporal operators bounded by a closed interval $[0, b]$, where b may refer to steps or time. We use the notation ϕ_t and ψ_t to denote the value of the propositions ϕ and ψ at step or time t. F is the *finally* or *eventually* operator ($F_{\leq b}\phi$ is true iff $\exists t \in [0, b] : \phi_t$ is true). G is the *globally* or *always* operator ($G_{\leq b}\phi$ is true iff $\forall t \in [0, b] : \phi_t$ is true). U is the *until* operator ($\psi U_{\leq b}\phi$ is true iff $\exists t \in [0, b] : \phi_t$ is true $\wedge (t = 0 \vee \forall t' \in [0, t[: \psi_{t'}$ is true). In the case of nested temporal operators, the time bound of an inner temporal operator is relative to the time bound of its directly enclosing operator. Hence, e.g., $F_{\leq 3}G_{\leq 4}\phi$ is true iff $\exists t \in [0, 3], \forall t' \in [t, t+4] : \phi_{t'}$ is true.

Statistical model checking works by verifying a property ϕ against $N \in \mathbb{N}$ independent simulation runs. Each simulation run evaluates to true or false and the probability that ϕ is true on an arbitrary execution of the system is estimated by the standard Monte Carlo estimator $\frac{1}{N}\sum_{i=1}^{N} 1(\phi)$, where $1(\cdot)$ is an indicator function that returns 1 if its argument is true and 0 otherwise. To quantify the confidence of the estimate, in this chapter we use a Chernoff bound [22] that guarantees, for given N, that the absolute error of the estimate is less than ε with probability δ, where $2\varepsilon N = \ln(2/\delta)$.

1.3 A Case Study: The Role of Estrogen in Cellular Mitosis and DNA Damage

At a cellular level, life is punctuated by the recurrence of four major phases: `Gap 1` `(G1)`, `S`, `Gap 2 (G2)`, and `M`. `G1` is in between mitosis and DNA replication and is responsible for cell growth. The transition occurring at the restriction point (called `R`) during the `G1` phase commits a cell to the proliferative cycle. If the conditions that enforce this transition are not present, the cell exits the cell cycle and enters a non-proliferative phase (called `G0`) during which cell growth, segregation and apoptosis occur. Replication of DNA takes place during the synthesis phase (called `S`). It is followed by a second gap phase responsible for cell growth and preparation for di-

vision. Mitosis and production of two daughter cells occur in the `M` phase. Switches from one phase to another are canonically regulated by a family of Cyclins that act as regulatory subunits for the Cyclin-Dependent Kinases (CDKs). According to the actual phase of the cell cycle, a disparate number of chemicals interact with the Cyclin-CDKs complexes to prevent or favour their move into the nucleus and, consequently, to block or promote the next phase transition [34].

Here we focus on the pre-mitotis `G2` phase and model the contention on the Mitosis Promoting Factor (MPF) (i.e., the Cyclin B1/2-CDKs complex), after the occurrence of DNA damage, by two key contributors: p53, the "guardian of the genome" [35], and the estrogens. The predominance of one's function over the other results in a proliferative rather than in a quiescent state of the cell.

It is known that p53 is a crucial protein in multicellular organisms, where it regulates the cell cycle and functions as a tumour suppressor. p53 has many mechanisms of anticancer function. It can (i) activate DNA repair proteins when DNA has sustained damage; (ii) induce growth arrest by holding the cell cycle at the `G1/S` regulation point on DNA damage recognition; (iii) initiate apoptosis, the programmed cell death, if DNA damage proves to be irreparable. In `G2` phase and after DNA damage, activated p53 binds DNA and induces expression of 14-3-3-σ (a.k.a. Stratifin) [36]. Stratifin mRNA exits the nucleus and, after translation, obstructs cell cycle entry by sequestering MPF, thereby preventing its shuttling to the nucleus [37].

Conversely, estrogens, the primary female sex hormones, promote cell cycle progression. They are intracellular proteins present both on the cell surface membrane and in the cytosol. Their actions are assumed to be mediated by estrogen receptors (`ER`s) which are found in different ratios in the different tissues of the body:

- `ER`$_\alpha$: endometrium, breast cancer cells, ovarian stroma cells and hypothalamus.
- `ER`$_\beta$: kidney, brain, bone, heart, lungs, intestinal mucosa, prostate and endothelial cells.

`ER`*s* actions can be selectively enhanced or disabled by some estrogen receptor *modulators*, in accordance with the binding affinity level of each estrogenic compound. In the classic model, the *estrogen 17 beta estradiol* binds to the `ER`, causing displacement of chaperone proteins. Dimers of the estrogen-`ER` complexes can then act as transcription factors by binding to specific estrogen response element (ERE) sequences in the promoters of target genes, evoking a wide range of transcriptional responses.

Efp, a RING-finger-dependent ubiquitin ligase, is a relevant target gene product of `ER`$_\alpha$. It is predominantly expressed in various female organs and is responsible for the proteolysis of 14-3-3-σ and then it is essential for estrogen-dependent cell proliferation. Its transcription is mediated by the estrogen-`ER`$_\alpha$ complex which enters the nucleus and binds to its ERE. The Efp mRNA can exit the nucleus, translate and eventually bind to the complex stratifin-MPF, floating in the cytoplasm in an inactive form. The newly formed complex dissociates into a macromolecule of ubiquitinated stratifin and one active MPF complex. While the former is targeted for death by proteolysis, the latter can enter the nucleus and promote mitosis progression.

Fig. 1.6 SBGN diagram of the considered model.

The described pathway is collected from the Biocarta Pathway Database and redrawn using Systems Biology Graphical Notation (SBGN) glyphs in Fig.1.6.

The described cell cycle pathway is crucial to the understanding of cancer because one of the hallmarks of cancer is the uncontrolled proliferation of abnormal or damaged cells. The focus of many treatments is therefore to prevent cell division (*mitosis*) of such cells. Estrogen is implicated in certain important types of breast cancer (e.g., estrogen receptor positive cancer), so in this chapter we present the methodology to study the role of estrogen in cellular mitosis, analysing the scenario in which DNA damage occurs. Sources of damage are not discussed in depth here,

although they are commonly ascribed to a prolonged exposure to ionizing or ultra-violet irradiation, or to sporadic mistakes made by the mitotis machinery. In view of this, we investigate the relationship between damage, estrogen, its receptor, and a typical *antagonist* that acts to prevent mitosis.

There are many synthetic molecules on the market that play the role of antagonist to cancer. Tamoxifen, Raloxifene and Anastrozole are some of the most representative. Their antagonism makes sense when coupled with true agonists, which are molecules that typically bind to receptors of a cell and trigger a response by that cell. Agonists often mimic the action of a naturally occurring substance, whereas antagonists block the action of the agonists or cause an action opposite to that of the agonists. The current accepted definition of receptor antagonist is based on the *receptor occupancy* model: agonists are thought to turn on a single cellular response by binding to the receptor, thus initiating a biochemical mechanism for change within a cell. Antagonists are thought to turn off that response by blocking the receptor from the agonist. Whenever the action of the antagonist results to be irreversible, that is its effect lasts throughout the lifetime of the antagonist itself, its dynamics can be described as a key broken off in the lock that prevents any other key from being inserted.

These antagonists are essentially *prodrugs*, and we have added them, in an appropriate way, to the described pathway (see Fig. 1.6). In particular, Tamoxifen, an estrogen blocker that belongs to the class of non-steroidal anti-estrogens, causes cells to remain in the G0 and G1 phases of the cell cycle. In particular, it fights breast cancer by competing with estrogen for space on the receptors of the tumour tissue. Each molecule of Tamoxifen that binds to a receptor prevents an estrogen molecule from engaging at the same place. This can facilitate the treatment of cancer because without a continuous supply of estrogen, cancer cells do not develop and the ability of the tumour to spread is reduced.

Recent experimental work has recognised that drug therapies may be more effective when linked to specific phases of the cell cycle [38] - so-called *chronoparmaceuticals* used in *chronotherapy* - so we specifically consider the time (delay) of the damage with respect to availability of cyclins (the molecules that control mitosis). In this chapter we propose a preliminary study on the interplay between the time of the DNA damage, the amount of the damage and the presence of antagonists.

1.4 Methodology and Results

Using PLASMA-lab [32], we have implemented a statistical model checker that allows us to analyse the evolutions of a stochastic membrane systems with peripheral proteins (defined in Section 1.1.2), specified in an appropriate syntax and using queries expressed in temporal logic (defined in Section 1.2.1). The integrated computational platform can be found at the web-page [26].

The biological model used in our investigation (discussed in Section 1.3) is described in terms of a membrane system in Fig. 1.7, where the corresponding simu-

lator script can also be found. In our study, we explicitly consider as parameters the amount of damage, the delay time of the damage and the amount of antagonist. We model the existence of damage by the production of $p53$, a well known indicator, that results from a damage signal that is instantiated as a quantity of molecules denoted as *damage*. Our modelling language allows us to inject an amount of damage at a specific time, using the syntax *amount damage@delay*, where "*amount*" is a number of *damage* molecules and "*delay*" is the value of the time delay from the start of the simulation. As an indicator of mitosis, we define a molecular species denoted *MITOSIS*, that is produced by the cyclins in the nucleus. For the purposes of our investigation, we set an arbitrary minimum of 300 molecules of *MITOSIS* to indicate that mitosis will proceed. Our results remain qualitatively similar if this value is changed.

To estimate the probability of mitosis we consider the following temporal logic property:

$$F_{\leq delay}\ damage = 0 \land (X\ damage = amount \land (F_{\leq 20}\ MITOSIS > 300)) \quad (1.1)$$

This property states that in the interval $[0, delay]$ there will be a time when *damage* = 0 and in the next state *damage* = *amount* and within 20 time units *MITOSIS* > 300. In our experiments, the value of *delay* is set to the time at which we inject damage and *amount* to the amount of *damage* injected. Hence, the first part of the property is guaranteed to be satisfied and is used to detect the precise step that damage occurs[1]. This allows the remainder of the property to be timed relative to the damage event. The value of 20 time units is chosen to be sufficiently long to capture all interesting behaviour following the damage.

We performed statistical model checking on the model with our model checking tool using the property (1.1). Fig. 1.8 illustrates the results of considering amounts of damage in the range $[300, 1000]$ with increments of 25 and delays in the range $[0, 5]$ with increments of 0.2. Each point is the result of 37 simulations, which is sufficient (according to the Chernoff bound defined in Section 1.2.1) to give a confidence of ± 0.1 with probability 0.95. 0.95 (95%) is a standard high level of confidence, while ± 0.1 is sufficiently precise to resolve the detail in the figure. Fig. 1.9 is the result of the same experiments, but with 1000 *antagonist*. The results for each figure were generated on a single machine (Intel Core i7 2.8Ghz, 4 GB RAM) in less than an hour. The general trends are clear in both figures: the probability of mitosis increases with increasing delay and decreases with increasing damage; above a delay of about 4 time units, mitosis is guaranteed, independent of amount of damage. This confirms our expectation that cell damage causes cell cycle arrest and that if the damage occurs after mitosis has begun, it will be too late to have an effect. Comparing Fig. 1.8 and Fig. 1.9, we see that the addition of antagonist increases the region of the figure where the cell cycle is arrested. In particular, the system is more sensitive to lower amounts of damage, confirming our understanding of the effect of an antagonist. We also note that the addition of antagonist has much less effect with respect to delay.

---

[1] For zero delay we use the simpler property *damage* = *amount* $\land (F_{\leq 20}\ MITOSIS > 300)$

Π system *estr*

$V_{estr} = \{damage, p53, ERalpha\_estrogen, Efp,$

$stratifin, CDKs\_Cyclin\_B1\_2, MITOSIS,$
$estrogen, ERalpha, CDKs\_Cyclin\_B1\_2\_stratifin,$
$Ub\_ligase, CDKs\_Cyclin\_B1\_2\_stratifin\_ubiquitin,$
$stratifin\_ubiquitin\}$

$r_1 : [\,damage \rightarrow p53\,]_{||}^{nucleopl}$

$r_2 : [\,]_{||}^{nucleopl} ERalpha\_estrogen \rightarrow [ERalpha\_estrogen]_{||}^{nucleopl}$

$r_3 : [\,ERalpha\_estrogen\,]_{||}^{nucleopl} \rightarrow [\,]_{||}^{nucleopl} ERalpha\_estrogen$

$r_4 : [\,ERalpha\_estrogen \rightarrow ERalpha\_estrogen + Efp\,]_{||}^{nucleopl}$

$r_5 : [\,Efp\,]_{||}^{nucleopl} \rightarrow [\,]_{||}^{nucleopl} Efp$

$r_6 : [\,p53 \rightarrow p53 + stratifin\,]_{||}^{nucleopl}$

$r_7 : [\,stratifin\,]_{||}^{nucleopl} \rightarrow [\,]_{||}^{nucleopl} stratifin$

$r_8 : [\,]_{||}^{nucleopl} CDKs\_Cyclin\_B1\_2 \rightarrow [CDKs\_Cyclin\_B1\_2]_{||}^{nucleopl}$

$r_9 : [\,CDKs\_Cyclin\_B1\_2\,]_{||}^{nucleopl} \rightarrow [\,]_{||}^{nucleopl} CDKs\_Cyclin\_B1\_2$

$r_{10} : [\,CDKs\_Cyclin\_B1\_2 \rightarrow MITOSIS\,]_{||}^{nucleopl}$

$r_{11} : [\,]_{||ERalpha}^{cyto} estrogen \rightarrow [\,]_{||ERalpha\_estrogen}^{cyto}$

$r_{12} : [\,]_{||ERalpha\_estrogen}^{cyto} \rightarrow [\,ERalpha\_estrogen\,]_{||}^{cyto}$

$r_{13} : [\,CDKs\_Cyclin\_B1\_2 + stratifin \rightarrow$
$\quad CDKs\_Cyclin\_B1\_2\_stratifin\,]_{||}^{cyto}$

$r_{14} : [\,Efp + Ub\_ligase + CDKs\_Cyclin\_B1\_2\_stratifin \rightarrow$
$\quad CDKs\_Cyclin\_B1\_2\_stratifin\_ubiquitin + Ub\_ligase\,]_{||}^{cyto}$

$r_{15} : [\,CDKs\_Cyclin\_B1\_2\_stratifin\_ubiquitin \rightarrow$
$\quad CDKs\_Cyclin\_B1\_2 + stratifin\_ubiquitin\,]_{||}^{cyto}$

$r_{16} : [\,stratifin\_ubiquitin \rightarrow \lambda\,]_{||}^{cyto}$

$r_{17} : [\,]_{||ERalpha}^{cyto} antagonist \rightarrow [\,]_{||ERalpha\_antagonist}^{cyto}$

$\mu_{estr} = [\,[\,]^{nucleopl}\,]^{cyto}$

$w_{cyto} = CDKs\_Cyclin\_B1\_2^{1000}, Ub\_ligase^{1000}$
$w_{nucleopl} = \lambda$

$w_0 = estrogen^{10000}$
$(u_{nucleopl} = \lambda, v_{nucleopl} = \lambda, x_{nucleopl} = \lambda)$
$(u_{cyto} = \lambda, v_{cyto} = \lambda, x_{cyto} = ERalpha^{1000})$
$t_{in} = 0, t_{fin} = 20.000$
$rate(r_3) = 0.3, \; rate(r_8) = 0.06,$
$rate(r_9) = 0.03, \; rate(r_{11}) = 0.1$

Simulator script

```
object damage,p53,ERalpha_estrogen,Efp,
stratifin,CDKs_Cyclin_B1_2,MITOSIS,
estrogen,ERalpha,CDKs_Cyclin_B1_2_stratifin,
Ub_ligase,CDKs_Cyclin_B1_2_stratifin_ubiquitin,
stratifin_ubiquitin

rule nucleoplasm_rules
{
    damage → p53
    || + ERalpha_estrogen− > ERalpha_estrogen + ||
    ERalpha_estrogen + || 0.3 → || + ERalpha_estrogen
    ERalpha_estrogen− > ERalpha_estrogen + Efp
    Efp + || → || + Efp
    p53 → p53 + stratifin
    stratifin + || − > || + stratifin
    || + CDKs_Cyclin_B1_2  0.06  → CDKs_Cyclin_B1_2 + ||
    CDKs_Cyclin_B1_2 + ||  0.03  → || + CDKs_Cyclin_B1_2
    CDKs_Cyclin_B1_2 → MITOSIS
}
rule cytoplasm_rules
{
    ||ERalpha + estrogen 0.1 → ||ERalpha_estrogen
    ||ERalpha_estrogen− > ERalpha_estrogen + ||
    CDKs_Cyclin_B1_2 + stratifin →
        CDKs_Cyclin_B1_2_stratifin
    Efp + Ub_ligase + CDKs_Cyclin_B1_2_stratifin →
        CDKs_Cyclin_B1_2_stratifin_ubiquitin + Ub_ligase
    CDKs_Cyclin_B1_2_stratifin_ubiquitin →
        CDKs_Cyclin_B1_2 + stratifin_ubiquitin
    stratifin_ubiquitin → *
    ||ERalpha + antagonist → ||ERalpha_antagonist
}
compartment
    nucleoplasm[1000 damage@500,nucleoplasm_rules]
compartment cytoplasm[1000 CDKs_Cyclin_B1_2,
    1000 Ub_ligase,nucleoplasm,
    cytoplasm_rules : ||1000ERalpha]
system cytoplasm, 10000 estrogen

evolve  0 − 20000
```

Fig. 1.7 A stochastic membrane systems with peripheral and integral proteins that presents the P53-independent G2/M cell cycle arrest pathway, with data collected from the Biocarta Pathways Database, described in Fig. 1.6 and ranges of proportionality between the coefficients obtained using preliminary western-blotting experiments. On the left side we present the formal language model following the formal syntax presented in Section 1.1.2 (the membrane labels cyto and nucleopl stands for cytoplasm and nucleoplasm, respectively); on the right side the equivalent simulator script using the appropriate syntax. The complete description of the simulator syntax can be found in [28], [9] and at the platform webpage [26].

Having observed that the presence of antagonist increases the sensitivity of the system to damage, we investigated the amount of antagonist required to cause cell cycle arrest. The results are plotted in Fig. 1.10 and were achieved in the following way. We estimated the probability of mitosis, as defined by the property (1.1), considering three parameters: amount of damage in the range $[300, 900]$ with increments of 25, delay of damage in the range $[0, 4]$ with increments of 0.2, and amount of antagonist in the range $[0, 2100]$ with increments of 100. Thus, for each combination of amount and delay of damage, we constructed a sequence of probabilities corresponding to the amounts of antagonist, each estimated with the same level of confidence used for Figs. 1.8 and 1.9. The results were generated on a single machine (Intel Core i7 2.8Ghz, 4 GB RAM) in less than 14 hours. Three qualitatively distinct sequences of probabilities emerged: (i) probabilities consistently below 0.5 for all considered values of antagonist (black area in Fig. 1.10); (ii) probabilities consistently above 0.5 for all considered values of antagonist (yellow area marked with '+' symbols) and (iii) probabilities decreasing from values above 0.5 to values below 0.5 with increasing antagonist. Sequences of type (i) correspond to a range of parameters where cell cycle arrest is inevitable, regardless of antagonist. Sequences of type (ii) correspond to a range of parameters where either mitosis is inevitable or more than the maximum amount of antagonist that we tried is required to cause cell cycle arrest. From sequences of type (iii) we were able to estimate the amount of antagonist corresponding to the transition value of 0.5, by interpolating between two adjacent points or from a single point if, by chance, its value was exactly 0.5.
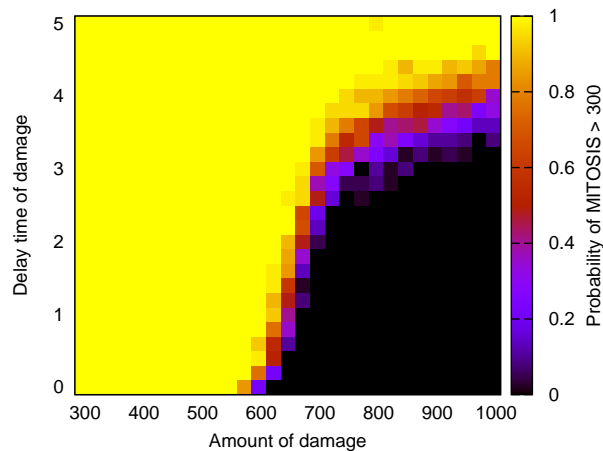
Fig. 1.8 The effect of amount and delay of damage on mitosis: the probability of *MITOSIS* > 300 for various amounts and time delay of damage. The figure illustrates the general trend that the probability of mitosis increases with increasing delay and decreasing damage.
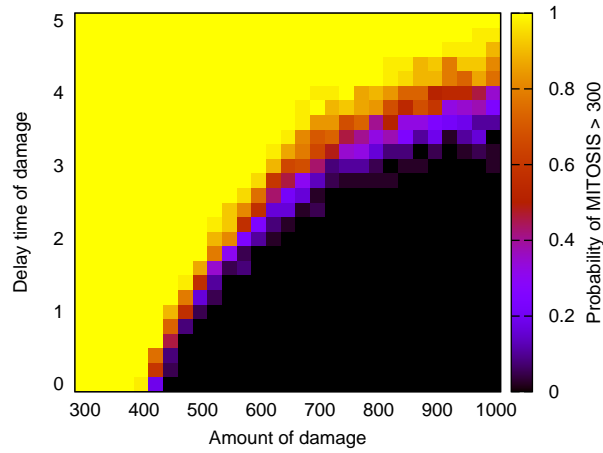
Fig. 1.9 The effect of antagonist on the response to damage: the probability of *MITOSIS* > 300 for various amounts and delay of damage in the presence of 1000 *antagonist*. In comparison with Fig. 1.8, the figure demonstrates that antagonist increases the sensitivity of the system to damage.
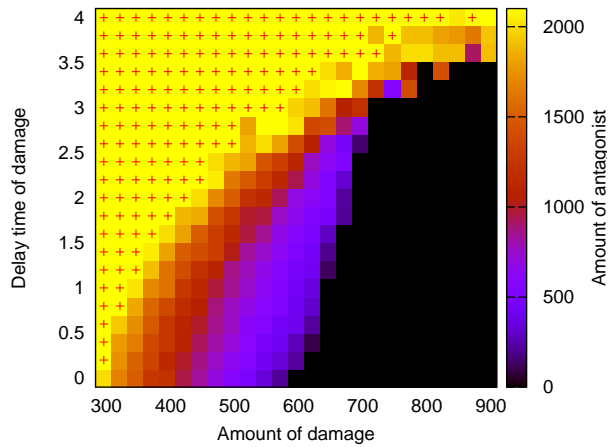
Fig. 1.10 The required dosage of antagonist to prevent mitosis: the amount of antagonist required to make the probability of *MITOSIS* > 300 less than 0.5 when previously it was greater than 0.5. In the black areas the probability of *MITOSIS* > 300 is less than 0.5, independent of antagonist. The '+' symbol indicates that at least 2100 antagonist would be required to prevent mitosis.

1.5 Conclusions

Even though the life cycle of cells is not generally synchronized, we recall here that replication is subject to the presence of cyclins, that are released in a timely way. Our results then show that it is possible to target cells *on time*, that is, to occupy estrogen receptors (ERs) at time points that are maximally effective. This would convey a double advantage: minimizing the necessary quantity of antagonist and making its effect optimal. The definition of the most effective dosage curve is indeed not a simple task, especially in cases where antagonists cannot be specific for particular cells. For example, in the case of ER+ breast cancer, Tamoxifen is currently taken once or twice a day and it is usually prescribed at 20 mg for 5 years. This dosage is due to the unavoidable ineffectiveness of the drug when reaching cells in unfavorable time points, as well as to the fact that Tamoxifen does not specifically target breast cancer cells; its molecules circulate within the body and target any cell that contains an available ER. The consequence of this is that while Tamoxifen works as an anti-estrogen for the breast, it acts as estrogen (i.e., agonist) in the uterus and, to a lesser extent, in the heart, blood vessels and bones. In cases like this, a tuned *chronotherapy* cannot eliminate the risk of side effects, but can drastically reduce it. The presented results show that the use of statistical model checking in membrane systems could be helpful to individuate the appropriate time-dependent dosage of antagonists in cancer treatments, [39].

References

1. Gh. Păun, G. Rozenberg, and A. Salomaa. *The Oxford Handbook of Membrane Computing*. Oxford University Press, Inc., New York, NY, USA, 2010.
2. M. Pérez-Jiménez and F. Romero-Campero. P systems, a new computational modelling tool for systems biology. *T. Comp. Sys. Biology*, 4220:176–197, 2006.
3. Gh. Păun. Computing with membranes. *J. Comput. Syst. Sci.*, 61(1):108–143, 2000.
4. L. Cardelli. Brane calculi. In Danos V. and Schachter V., editors, *Computational Methods in Systems Biology*, volume 3082 of *Lecture Notes in Computer Science*, pages 257–278. Springer Berlin / Heidelberg, 2005.
5. L. Cardelli and Gh. Păun. An universality result for a (mem)brane calculus based on mate/drip operations. *Int. J. Found. Comput. Sci.*, 17(1):49–68, 2006.
6. R. Brijder, M. Cavaliere, A. Riscos-Núñez, G. Rozenberg, and D. Sburlan. Membrane systems with marked membranes. *Electronic Notes in Theoretical Computer Science*, 171(2):25 – 36, 2007. Proceedings of the First Workshop on Membrane Computing and Biologically Inspired Process Calculi (MeCBIC 2006).
7. M. Cavaliere and S. Sedwards. Membrane systems with peripheral proteins: Transport and evolution. *Electronic Notes in Theoretical Computer Science*, 171(2):37 – 53, 2007. Proceedings of the First Workshop on Membrane Computing and Biologically Inspired Process Calculi (MeCBIC 2006).
8. D. Bray, A. Johnson, J. Lewis, M. Raff, K. Robert, P. Walter, and B. Alberts. *Essential Cell Biology: An Introduction to the Molecular Biology of the Cell*, volume 1. Garland Pub., 1998.
9. M. Cavaliere and S. Sedwards. Modelling cellular processes using membrane systems with peripheral and integral proteins. In *Proceedings of the 2006 international conference on Com-*

putational Methods in Systems Biology, CMSB'06, pages 108–126, Berlin, Heidelberg, 2006. Springer Berlin / Heidelberg.

10. M. Cavaliere, S. N. Krishna, Gh. Păun, and A. Păun. P systems with objects on membranes. In Gh. Păun, G. Rozenberg, and A. Salomaa, editors, *The Oxford Handbook of Membrane Computing*, pages 363–388. Oxford University Press, 2010.

11. A. Păun, M. Păun, A. Rodríguez-Patón, and M. Sidoroff. P systems with proteins on membranes: a survey. *Int. J. Found. Comput. Sci.*, 22(1):39–53, 2011.

12. J. Fisher and Th. A. Henzinger. Executable cell biology. *Nature Biotechnology*, 25:1239–1249, Nov. 2007.

13. E.M. Clarke, E. A. Emerson, and J. Sifakis. Model checking: algorithmic verification and debugging. *Commun. ACM*, 52(11):74–84, November 2009.

14. E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. The MIT Press, January 1999.

15. D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361, 1977.

16. C. Baier and J.-P. Katoen. *Principles of Model Checking*. MIT Press, May 2008.

17. F. J. Romero-Campero, M. Gheorghe, L. Bianco, D. Pescini, M. J. Pérez-Jiménez, and R. Ceterchi. Towards Probabilistic Model Checking on P Systems Using PRISM. In *Proceedings of the 7th international conference on Membrane Computing*, WMC'06, pages 477–495, Berlin, Heidelberg, 2006. Springer Berlin / Heidelberg.

18. J. Heath and et al. Probabilistic model checking of complex biological pathways. *Theoretical Computer Science*, 391:239–257, 2008.

19. M. Kwiatkowska, G. Norman, and D. Parker. Using probabilistic model checking in systems biology. *SIGMETRICS Perform. Eval. Rev.*, 35(4):14–21, March 2008.

20. J. Blakes, J. Twycross, F. J. Romero-Campero, and N. Krasnogor. The infobiotics workbench: an integrated in silico modelling platform for systems and synthetic biology. *Bioinformatics*, 27(23):3323–3324, December 2011.

21. E. M. Clarke, J. R. Faeder, C. J. Langmead, L. A. Harris, S. Jha, and A. Legay. Statistical model checking in biolab: Applications to the automated analysis of t-cell receptor signaling pathway. In M. Heiner and A.M Uhrmacher, editors, *Computational Methods in Systems Biology*, volume 5307 of *Lecture Notes in Computer Science*, pages 231–250. Springer Berlin / Heidelberg, 2008.

22. H. Chernoff. A Measure of Asymptotic Efficiency for Tests of a Hypothesis Based on the sum of Observations. *Ann. Math. Statist.*, 23(4):493–507, 1952.

23. The PRISM website. www.prismmodelchecker.org.

24. C. Jegourel, A. Legay, and S. Sedwards. A platform for high performance statistical model checking: Plasma. In Cormac Flanagan and Barbara König, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, volume 7214 of *Lecture Notes in Computer Science*, pages 498–503. Springer Berlin / Heidelberg, 2012. 10.1007/978-3-642-28756-5_37.

25. J. E. Hopcroft, R. Motwani, and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2006.

26. Platform webpage. https://sites.google.com/site/cytosim/smc.

27. D. T. Gillespie. Stochastic simulation of chemical kinetics. *Annual Review of Physical Chemistry*, 58(1):35–55, 2007.

28. S. Sedwards and T. Mazza. Cyto-sim: a formal language model and stochastic simulator of membrane-enclosed biochemical processes. *Bioinformatics*, 23(20):2800–2802, 2007.

29. M. Cavaliere and S. Sedwards. Decision problems in membrane systems with peripheral proteins, transport and evolution. *Theor. Comput. Sci.*, 404(1-2):40–51, September 2008.

30. M. Cavaliere and T. Mazza. A (natural) computing perspective on cellular processes. In Lodhi H.M. and Muggleton S.H., editors, *Elements of Computational Systems Biology*, pages 115–138. Wiley, 2010.

31. B. Boyer, K. Corre, A. Legay, and S. Sedwards. PLASMA-lab : a flexible, distributable statistical model checking library. In *Quantitative Evaluation of Systems*, 2013.

32. PLASMA-lab project webpage. https://project.inria.fr/plasma-lab.

33. Cyto-Sim language webpage. https://sites.google.com/site/cytosim/language.
34. P. Ballarini, T. Mazza, A. Palmisano, and A. Csikasz-Nagy. Studying irreversible transitions in a model of cell cycle regulation. *Electron. Notes Theor. Comput. Sci.*, 232:39–53, March 2009.
35. D. P. Lane. p53, guardian of the genome. *Nature*, 358(6381):15–16, 1992.
36. H. Hermeking, C. Lengauer, K. Polyak, T. C. He, L. Zhang, S. Thiagalingam, K. W. Kinzler, and B. Vogelstein. 14-3-3 sigma is a p53-regulated inhibitor of g2/m progression. *Mol. Cell.*, 1(1):3–11, 1997.
37. C. Laronga, H.-Y. Yang, C. Neal, and M.-H. Lee. Association of the cyclin-dependent kinases and 14-3-3 sigma negatively regulates cell cycle progression. *The Journal of Biological Chemistry*, 275(30):23106–23112, 2000.
38. K. Ross. Circadian rhythms play role in cancer research. *Journal of the National Cancer Institute*, 98(12):806–807, 2006.
39. H. Kitano. A robustness-based approach to systems-oriented drug design. *Nature Reviews Drug Discovery*, 5:202–210, 2007.