Studying cancer-cell populations by programmable models of networks

Luca Bortolussi · Alberto Policriti

Received: 12 January 2012 / Revised: 1 May 2012 / Accepted: 8 May 2012 / Published online: 12 June 2012
© Springer-Verlag 2012

Abstract We draw the basic lines for an approach to build mathematical and *programmable* network models, to be applied in the study of populations of cancer-cells at different stages of disease development. The methodology we propose uses a stochastic Concurrent Constraint Programming language, a flexible stochastic modelling language employed to code networks of agents. It is applied to (and partially motivated by) the study of differently characterized populations of prostate cancer cells. In particular, we prove how our method is suitable to systematically reconstruct and compare different mathematical models of prostate cancer growth—together with interactions with different kinds of hormone therapy—at different levels of refinement. Moreover, we show our technique at work in analysing the nature of noise and in the possible presence of competing mechanisms in the models proposed.

Keywords Computational systems biology · Stochastic process algebras · Tumor growth modeling · Prostate tumor

1 Introduction

In general models are built for one of these tasks: qualitative, quantitative, or predictive analysis. The three tasks are not exclusive and often a study in one direction turns out to be useful in some of the others. The case of quantitative and predictive analyses is a clear example of this (positive) phenomenon. In this paper, we introduce the general infrastructure for a tool aiming at addressing, together with the above-mentioned goals, also a study of the *logic* and *nature* of the various components introduced in a (set of) model(s). In particular, we will carry out a study within a group of models of cancer-cell populations that will allow us to introduce, compare and evaluate different logics underlying competitive mechanisms among cancer-cell populations. This will be done in parallel with a study on the nature of noise observed analysing quantitative simulations.

The (long term) goal of this work is the development of a general framework for building programmable models to be used in the study of cells populations, with some emphasis on cancer cells. The models we have in mind are, in fact, models of populations of cells obeying specific growth and death laws and incorporating different stages of tumor evolution.

At the architectural level, we envisage three key features in our models:

- *programmability*, as the possibility to enter a high-level description of the full model in terms of networks of interacting agents;
- *hybridness*, as the possibility to carry out a complete description of the semantics of our networks of agents by a dynamical system with a controlled combination of both discrete and continuous evolution;

Electronic supplementary material The online version of this article (doi:[10.1007/s13721-012-0010-x](https://doi.org/10.1007/s13721-012-0010-x)) contains supplementary material, which is available to authorized users.

L. Bortolussi
Department of Maths and Geosciences,
University of Trieste, Trieste, Italy
e-mail: luca@dmi.units.it

A. Policriti (✉)
Department of Maths and Computer Science,
University of Udine, Udine, Italy
e-mail: alberto.policrit@uniud.it

- *stochasticity*, as the possibility of specifying the interaction mechanisms obeying to given stochastic laws.

The above-mentioned networks naturally implement the hybrid nature of the different phases through which the development of tumoral populations occurs. Moreover, they allow us to study both the logical and quantitative features of a model, at a higher level of description than classical models based on ordinary differential equations (ODE). As an example of such analyses, one can try to understand to what extent the observed *noise* depends on the structure of the internal/external interactions defining the model, as opposed to parameters variation. Finally, they allow to tune-up the level of discreteness to be used in the modelling activity.

Such kind of models, together with their simulation environment, should turn out particularly useful in those situations in which more therapies operate concurrently. As an example of such a situation considers locally advanced prostate cancer, a case in which the use of radiation therapy, together with androgen ablation, is usually recommended by the American Urology Association (2012). In fact, our concrete and motivating example is just the study of prostate cancer, with special emphasis on the predictiveness ability of models including external interactions (in the form of medicament dispensation) (Brawer 2006; Tanaka et al. 2010). For this reason we begin illustrating the various models of prostate cancer-cell evolution under different policies of chemical castration (see Sect. 2 and Tanaka et al. 2010). As a result of a multi-facet analysis of such policies, we will see that a set of natural questions will arise, such as the above-mentioned issue on the nature of noise (and consequent completeness of the model) or such as the presence of a competitive mechanism working “under the hood” and justifying the effectiveness of intermittent androgen ablation. The methodological aspects involved in tackling such issues should be an illustration of the benefits of our proposed approach.

Our modeling approach to describe prostate cancer is based on stochastic Concurrent Constraint Programming (sCCP, Bortolussi and Policriti 2008) a modeling language belonging to the class of stochastic Process Algebras (SPA) applied to Systems Biology (Ciocchetta and Hillston 2008).

SPA have been applied in modelling many aspects of biological systems, including tumor growth (Lecca et al. 2011; Mazza and Cavaliere 2009). In general terms, the advantages in using SPA in modelling come directly from their usability as programming languages: modularity and agent-centric view, being the most prominent among them. These features are often put in contrast with those of rule-based systems, widespread in systems biology as they fit more naturally with the formalism of biochemical networks (see, e.g., Danos et al. 2007; Kohn et al. 2006). sCCP 1 is,

in our opinion, a sort of natural compromise retaining the advantages of both approaches: it allows both an agent-based description/programming of the (asynchronous) interactions within networks, as well as the introduction of rules rendering specific biochemical laws.

The objectives of this work are twofold: first, we want to describe prostate cancer models and drug dispensation policies in sCCP. Models of prostate tumor in sCCP are presented in Sect. 4. Second, we want to clarify if noise observed in experimental data could be explained as a *structural* feature of the model or if should be ascribed to some external aspect, not modelled by the equations. In addition, we also put forward a simple model of an underlying competitive mechanism that, at a phenomenological level, explains the data. An experimental study supporting our conclusions and based on fitted data available in the literature, is presented in Sect. 5.

sCCP is particularly suitable for the proposed task because its semantics is naturally stochastic, it is easily programmable and extensible, and it also has a general semantics in terms of stochastic hybrid automata (Bortolussi and Policriti 2009b) (see Sects. 2.2, 2.3, and 2.4). A flexible semantics is widely recognized as significant in the literature on cancer modelling (see Deisboeck et al. 2011; Deisboeck and Couzin 2009; Fornari et al. 2011; De Matteis et al. 2012; Roose et al. 2007), in addition to this agents in sCCP have an *internal* computing power that enhances their modelling power when mechanisms at different semantic levels must be modelled.

Nevertheless, to properly use sCCP for cancer modelling in this application, we had to extend its set of primitives to describe agents behaviour. In particular, we had to introduce a means to describe actions triggered by conditions on model time and to give agents the ability of changing their environment (i.e. system variables) according to random laws. These extensions turned out to be very simple to introduce in the hybrid semantics framework of sCCP, cf. Sect. 3.

Part of the study described in this paper was previously presented in (Bortolussi and Policriti 2011).

2 Background

Below, we briefly discuss the two main objects of our work in this paper: prostate cancer modelling techniques and sCCP, the modelling language that we will use to build our agent network in the rest of the paper.

2.1 Prostate cancer modelling

Prostate is a gland of the male reproductive system responsible for the production of (part of the) seminal fluid. Prostate tumour is a very common (age related) disease,

consisting mainly in the development of a mass of tumoral cells whose growth is correlated with (dependent from) the presence of androgen hormones (e.g. testosterone) (Brawer 2006). Most effective therapies consist in androgen deprivation (castration) by surgical or chemical means (Brawer 2006). Under such therapies, one observes an initial fast decrease of tumoral masses which, however, after a variable interval of time, undergo a relapse phase. This is caused by the emergence of a line of androgen-independent tumoral cells, resistant to androgen deprivation (Abrahamsson 2010; Brawer 2006).

The above-biological behaviours have been modelled using phenomenological approaches (Ideta et al. 2008; Jackson 2004; Tanaka et al. 2010) based on ODEs. The variables of the model record the number of androgen-dependent/independent cells and are equipped with growth and death laws expressing their time evolution. The spatial structure of the tumour is ignored, only the cell number is recorded. The observable of the model is the serum Prostate Specific Antigen (Rao et al. 2008) (PSA)—a bio-marker whose concentration in serum is strictly related to the size of the tumour mass that is the number tumoral cells.

$$\begin{cases} \frac{dx}{dt} = G_x(x, z) - D_x(x, z) - M_{xy}(x, z) \\ \frac{dy}{dt} = G_y(y, z) - D_y(y, z) + M_{xy}(x, z) \\ \frac{dz}{dt} = P_z - D_z(z) \end{cases}$$

$$G_x(x, z) = \alpha_x \left(k_1 + (1 - k_1) \frac{z}{z + k_2} \right) x$$

$$D_x(x, z) = \beta_x \left(k_3 + (1 - k_3) \frac{z}{z + k_4} \right) x$$

$$G_y(y, z) = \alpha_y \left(1 - d \frac{z}{z_0} \right) y$$

$$D_y(y, z) = \beta_y y$$

$$M_{xy}(x, z) = m_1 \left(1 - \frac{z}{z_0} \right)$$

$$P_z = 0$$

$$D_z(z) = \frac{z}{\tau}$$

Model 1: ODE model of prostate tumor growth under Continuous Androgen Suppression taken from (Tanaka et al. 2010). P stands for *Production*, G for *Growth*, D for *Death*, and M for *Mutation*. To control P_z , the parameter α_z is set to 0 to represent the effect of androgen deprivation therapy. In absence of castration, such a rate would be $P_z = \frac{z_0}{\tau}$. Parameters are set according to (Tanaka et al. 2010): $\alpha_x = 0.0204$, $\alpha_y = 0.0242$, $\beta_x = 0.0076$, $\beta_y = 0.0168$, $k_1 = 0$, $k_2 = 2$, $k_3 = 8$, $k_4 = 0.5$, $m_1 = 5 \times 10^{-5}$, $z_0 = 20$, $d = 1$, $\tau = 62.5$. We agree that τ controls the speed of z dynamics, z_0 is the stationary value of androgen hormone in absence of chemical castration, and $d \in [0, 1]$ controls the effect of androgen hormone in the growth rate of AI cells (0 means that growth is independent from the hormone, 1 means that the growth is inhibited by the hormone). Note how both the growth and death terms for AD-cells x depend non-linearly on the amount of hormone z through a sigmoid-shaped function $\frac{z}{z+k}$.

More specifically, a model can be defined by the system of differential equations of Model 1, which corresponds to the initial model used in Jackson (2004) and Tanaka et al. (2010). In the equations, we have three variables: x describes the population of androgen-dependent (AD) cancer cells, y describes the population of androgen-independent (AI) cancer cells, and z is the concentration of androgen hormone. The concentration of PSA is computed simply as $x + y$. This model is able to capture the relapse phase of the tumour due to androgen-independent cells growth (Ideta et al. 2008; Jackson 2004).

A (clever) clinical approach tackling the problem of relapse after a long interval of chemical castration is the so-called intermittent androgen suppression (IAS) policy, see (Abrahamsson 2010). The effectiveness of the IAS therapy is based on the fact that the underlying mechanism of the disease seems to involve a competition between androgen-dependent and androgen-independent cells. This implies that a complete reduction of the number of AD cells removes any obstacle to AI cells growth. Hence, a more effective strategy consists in maintaining a certain number of AD cells to inhibit AI cells growth. The overall effect of an intermittent androgen deprivation strategy is the delay—possibly for a very long time—of the development of an androgen-independent tumour.

Given the quantities involved and the presence of an external input—the medicament dispensation strategy—to keep into account as a time-controlled phase change, the mathematical modelling machinery naturally evolved into a hybrid model. To be precise, the system can switch between states representing normal and androgen deprivation modes (Ideta et al. 2008; Tanaka et al. 2010). These modes can be conveniently represented by a boolean variable u , where $u = 1$ describes the drug dispensation phase. The equations are obtained from the set of Model 1 by replacing P_z in the equation for the androgen hormone z by the u -dependent function $P_z(u) = \frac{z_0(1-u)}{\tau}$. Note that $P_z(1) = 0$, as in the previous model. A further evolution of the model consisted in the introduction of noise in the equation, thereby moving to a set of *stochastic* differential equations. This move was performed as an attempt to capture small fluctuations observed in the PSA data.

Our first motivation was to attempt to clarify whether the noise introduced in the model of Aihara et al. (Tanaka et al. 2010) is external or internal. In the former case, the explanation would call into play additional (noise) sources not related with growth and death laws of AD/AI cells. In the latter case, the noise could be explained in terms of fluctuations of such transitions, when considered as stochastic. Our approach is to begin designing a discrete and stochastic version of the above model, in terms of continuous time Markov chains (CTMC) (Gillespie 1977; Norris

1997). The construction will be given in Sect. 4 and the experimental results will be given in Sect. 5. Below, we briefly describe our agent's language that will act both as an intermediate layer in the translation to CTMC and as a computational counterpart of the model based on (stochastic) differential equations.

2.2 Stochastic concurrent constraint programming

We briefly introduce now (a simplified version of) sCCP, sketching the basic notions needed in the rest of the paper. More details on the language can be found in (Bortolussi and Policriti 2008). sCCP has two basic ingredients: *agents* and *constraints*. Agents are the main actors, interacting by asynchronously exchanging information, in form of *constraints*, through the *constraint store*. sCCP has been mainly applied as a modelling language for biological systems (Bortolussi and Policriti 2008), using the constraint store to describe the state of the system, e.g. number of molecular species. These quantities are described by a set of variables that can change value during computation, called *stream variables* (Bortolussi and Policriti 2008). At least for modelling simple biological scenarios, one needs very simple constraints, basically comparing and assigning new values to stream variables.

Example 2.1 Before providing a formal definition of sCCP, we introduce its main features with an example. Agents can be interpreted in different ways, depending on the meaning assigned to variables and transitions. At the lowest level, even a single cell can be represented by an agent. For instance, in the following piece of code, we model a cell that can be in two states (healthy or tumoral), depending on the number of internal mutations accumulated, and subject to duplication and apoptosis events.

$$\begin{aligned} \text{cell_H}(M): & -[M < \mu_0 \rightarrow M' = M + 1]_{\lambda_h(M)} \cdot \text{cell_H}(M) \\ & + [M = \mu_0 \rightarrow M' = M]_{k_m} \cdot \text{cell_T}(M) \\ \text{cell_T}(M): & -[M < \mu_1 \rightarrow M' = M + 1]_{\lambda_t(M)} \cdot \text{cell_T}(M) \\ & + [M = \mu_1 \rightarrow M' = M]_{\lambda_d} \cdot \mathbf{0} \end{aligned}$$

In the previous piece of code, we are describing the mutation from a cell in a healthy state (`cell_H`) to a cell in a tumoral state (`cell_T`), as a consequence of accumulation of (internal) genomic mutations. The number of mutations of a cell is represented by a (local) variable M , which is increased by one when a mutation event occurs. A mutation event in a healthy cell is represented by an action of the form $[M < \mu_0 \rightarrow M' = M + 1]_{\lambda_h(M)} \cdot \text{cell_H}(M)$. Here, $M < \mu_0$ is a guard, enabling the action only when the number of mutations M is less than a critical threshold μ_0 . The effect of the action is expressed by the reset $M' = M + 1$, which increased the mutation count by 1, and by specifying

the new state of the agent after the execution of the action, i.e. `cell_H(M)` (“.” is the sequencing operator). The label on the bottom right of the action, $\lambda_h(M)$, instead, represents the rate (i.e., the average speed) at which the action can be performed. In this case, $\lambda_h(M)$ is a function of the number of mutations M , and, in general, a rate can be a function of any subset of variables (thus allowing complex kinetics to be easily expressed). The operator $+$ represents the choice between two or more possible actions available to the agent. For instance, the healthy cell `cell_H` can either increase the number of internal mutations, or, when the critical threshold is reached be transformed into a tumoral state `cell_T`. A tumoral cell, instead, will survive until the number of genomic mutations remains below μ_1 (third action). When such a threshold is reached, the cell will die (fourth action). A dead cell is represented by the null agent $\mathbf{0}$, which is removed from the system.

The full program would run in parallel a number of agents equal to the number of cells modelled, each with its specific mutation-counting variable. If we want to enrich the model including events like cell duplication and cell apoptosis, we can change the cell agents by adding these new action abilities. For instance, `cell_H` may be described as follows:

$$\begin{aligned} \text{cell_H}(M): & -[M < \mu_0 \rightarrow M' = M + 1]_{\lambda_h(M)} \cdot \text{cell_H}(M) \\ & + [M = \mu_0 \rightarrow M' = M]_{k_m} \cdot \text{cell_T}(M) \\ & + [\text{true} \rightarrow \text{true}]_{\lambda_{dn}(M)} \cdot (\text{cell_H}(M) \parallel \text{cell_H}(M)) \\ & + [\text{true} \rightarrow \text{true}]_{\lambda_{an}(M)} \cdot \mathbf{0} \end{aligned}$$

Here, the cell duplication is encoded by the third action, expressing that cell duplication happens at a rate dependent on the number of mutations, and, once it fires, introduces two `cell_H` agents into the system, using the parallel construct. We remark that this very simple model has been introduced just for illustrative purposes, without any claim of biological correctness.

We present now a more formal treatment of sCCP.

Definition 2.1 An sCCP program is a tuple $\mathfrak{A} = (A, \mathfrak{D}, \mathbf{X}, \text{init}(\mathbf{X}))$, where

1. The *initial network of agents* A and the *set of definitions* $\mathfrak{D}$ are given by the following grammar:

$$\begin{aligned} A &= M | A \parallel A \quad M = \pi \cdot C | M + M \quad \pi = [G(\mathbf{X}) \rightarrow R(\mathbf{X}, \mathbf{X}')]_{\lambda(\mathbf{X})} \\ \mathfrak{D} &= \emptyset | \mathfrak{D} \cup \mathfrak{D} | \{C \stackrel{\text{def}}{=} M\} \end{aligned}$$

2. $\mathbf{X}$ is the set of stream variables of the store (with global scope), usually taking integer values;
3. $\text{init}(\mathbf{X})$ is a predicate on $\mathbf{X}$ of the form $\mathbf{X} = \mathbf{x}_0$, assigning an *initial value* to store variables.

In the previous definition, basic actions are *guarded updates* of (some of the) variables: $G(\mathbf{X})$ is a quantifier-free first-order formula whose atoms are inequality predicates on variables $\mathbf{X}$ and $R(\mathbf{X}, \mathbf{X}')$ is a predicate on $\mathbf{X}, \mathbf{X}'$ of the form $\mathbf{X}' = \mathbf{X} + \mathbf{k}(\mathbf{X}')$ denotes variables of $\mathbf{X}$ after the update),¹ for some vector $\mathbf{k} \in \mathbb{R}^n$. Each such action has a *stochastic duration*, specified by associating an exponentially distributed random variable to actions, whose rate depends on the state of the system through a function λ , with values $\lambda(\mathbf{X}) \in \mathbb{R}^+$. The semantics of sCCP is given by a Continuous Time Markov Chain (Norris 1997), deduced from the *labelled transition system* associated with an agent network by the operational semantics (Bortolussi and Policriti 2008).

All sCCP agents $C \stackrel{\text{def}}{=} M \in \mathfrak{D}$, according to Definition 2.1, are *sequential*, i.e. they do not contain any occurrence of the parallel operator, whose usage is restricted at the upper level of the network. sCCP sequential agents can be seen as automata synchronizing on store variables and they can be conveniently represented as labelled graphs, called *reduced transition systems* (RTS) (see Bortolussi and Policriti 2009a). More precisely, $\text{RTS}(C) = (S(C), E(C))$ is a multi-graph with vertices $S(C)$ corresponding to the different states of an agent and with edges $e \in E(C)$ corresponding to actions performable by the agent, labelled by the corresponding rate (λ_e), guard (G_e), and reset (R_e).

In the following, we also need the notion of extended sCCP program $\mathfrak{A}^+$, in which we introduce a new set of variables $\mathbf{P} = \{P_C | C \in \mathfrak{D}\}$ taking integer values and recording how many copies of agent $C \in \mathfrak{D}$ are in parallel in the current sCCP-network. Rates, guards, and resets are modified to consistently treat $\mathbf{P}$ variables. With this trick, we can assume that in $\mathfrak{A}^+$, there is never more than one copy of the same agent running in parallel at the same time. Further details on these notions can be found in (Bortolussi and Policriti 2009a).

Definition 2.1 introduces some restrictions on the syntax, mainly to simplify the presentation. In particular, local variables and nesting of parallel composition are not allowed, thus forbidding the (low) level description of cells of Example 2.1. However, in the following, we will stick to a higher level approach, using variables to count the number of cells in each state. In Example 2.1, this means introducing variables H_m and T_m , counting how many healthy (H_m) or tumoral (T_m) cells with m mutations are in the system. Note that this set of variables is finite, as the number of possible mutations is bounded. This transformation corresponds to

“programming the dynamics” at a higher level of abstraction. We will discuss this usage in the next section. Here, we just stress that a conversion between these to modelling levels can be done automatically, as far as local variables can take a finite number of values, practically justifying the restrictions of Definition 2.1.

Even if the standard semantics of an sCCP program is given in terms of CTMC, sCCP programs have also different semantics, based on ODE (Bortolussi and Policriti 2009a) and hybrid automata (Bortolussi and Policriti 2009b), providing an additional degree of flexibility to the language. In particular, the hybrid semantics is parametric with respect to the degree of continuity introduced in the model, and it subsumes as special cases in both the CTMC and the ODE semantics. We will give now more details on this semantics, exploiting it in Sect. 3 to smoothly extend sCCP with additional primitives that we will need for modelling tumour-cells.²

2.3 Transition-driven stochastic hybrid automata

Transition-driven stochastic hybrid automata (TDSHAs) have been introduced in (Bortolussi and Policriti 2009b, Bortolussi and Policriti 2010) as a convenient formalism to associate a stochastic hybrid automaton to an sCCP program.

The emphasis of TDSHA is on *transitions* which, as always in hybrid automata, can be either discrete (corresponding to jumps) or continuous (representing flows acting on system’s variables). More specifically, there are two kinds of discrete transitions:

1. *instantaneous* or *forced* taking place as soon as their guard becomes true, and
2. *stochastic* occurring after an exponentially distributed delay.

Definition 2.2 A transition-driven stochastic hybrid automaton (TDSHA) is a tuple

$\mathfrak{T} = (Q, \mathbf{X}, \mathcal{C}, \mathcal{D}, S, \text{init})$, where

- Q is a finite set of *control modes* and $\mathbf{X} = \{X_1, \dots, X_n\}$ is a set of real valued *system’s variables*.³
- $\mathcal{C}$ is the set of *continuous transitions or flows*, containing triples $\tau = (q, \mathbf{s}, f)$, where $q \in Q$ is a mode, $\mathbf{s}$ is a real vector of size $|\mathbf{X}|$, and $f: \mathbb{R}^n \rightarrow \mathbb{R}$ is a (Lipschitz) function. They are indicated by $\mathbf{q}_\tau, \mathbf{s}_\tau$, and $\mathbf{f}_\tau$, respectively.
- $\mathcal{D}$ is the set of *discrete or instantaneous transitions*, whose elements are tuples $\eta = (q_1, q_2, G, R, w)$, where q_1 is the *exit-mode*, q_2 is the *enter-mode*, and

¹ The constraints that can be used to update the constraint store are rather limited, as they simply add a constant to some stream variables. This restriction, however, allows to interpret sCCP-actions as continuous fluxes, a required condition to define the hybrid semantics (see also Sect. 2.4).

² A software tool to model and analyse sCCP programs is under development. A preliminary version is available from the authors upon request.

³ Notation: the time derivative of X_j is denoted by $\dot{X}_j$, while the value of X_j after a change of mode is indicated by X'_j .

$w : \mathbb{R}^n \rightarrow \mathbb{R}^+$ is a weight function used to resolve non-determinism between two or more active transitions. Moreover, G is a quantifier-free first-order formula with free variables among $\mathbf{X}$ and R , which is a conjunction of atoms of the form $X' = r(\mathbf{X}, \mu)$, where $r : \mathbb{R}^{n+m} \rightarrow \mathbb{R}$, is the reset function of X , depending on system's variables $\mathbf{X}$ as well as on a vector of parameters μ . In particular, parameters in μ can be either constants or random values. The elements of a tuple η are indicated by $q_\eta^1, q_\eta^2, w_\eta, G_\eta$, and R_η , respectively.

- $\mathcal{S}$ is the set of *stochastic transitions*, whose elements are tuples $\eta = (q_1, q_2, G, R, \lambda)$, where q_1, q_2, G , and R are as for transitions in $\mathcal{D}$, while $\lambda : \mathbb{R}^n \rightarrow \mathbb{R}^+$ is a function giving the state-dependent rate of the transition. Such function is referred to by λ_η .
- $init$ is a pair $(q_0, \mathbf{X}_0) \in Q \times \mathbb{R}^n$, identifying the *initial state* of the system.

The dynamics of TDSHA can be formally defined (Bortolussi and Policriti 2009b) by associating with them a Piecewise Deterministic Markov Processes (Davis 1993). Intuitively, the TDSHA dynamics is given by periods of continuous evolution, interleaved by discrete jumps, as customary with hybrid systems.

Within each mode $q \in Q$, the TDSHA evolves following the solution of a set of ODE, constructed combining the different continuous transitions active in q . Each such $\tau \in \mathcal{C}$ contributes to the ODE with a flow given by $\mathbf{f}_\tau(\mathbf{X})$, whose effect on each variable is described by $\mathbf{s}_\tau$. Hence, in mode $q \in Q$, ODE are given by $\dot{\mathbf{X}} = \sum_{\tau \in \mathcal{C}, q_\tau^1 = q} \mathbf{s}_\tau \cdot \mathbf{f}_\tau(\mathbf{X})$.

Two kinds of discrete jumps are possible: stochastic transitions $\eta \in \mathcal{S}$ are fired according to their rate, while instantaneous transitions $\eta \in \mathcal{D}$ are fired as soon as their guard becomes true. In both cases, the state of the system is reset according to the policy specified by R_η . Choice among several active stochastic or instantaneous transitions is resolved probabilistically according to their rate or priority.

We stress that TDSHA has another source of stochasticity in addition to priorities and rates: resets. Resets function, in fact, can assign random values to variables, through their dependency on random variables (via the parameter's tuple μ). This feature of TDSHA will be exploited in Sect. 3.

Furthermore, a notion of asynchronous TDSHA product $\mathfrak{T} = \mathfrak{T} \otimes \mathfrak{T}$ can be defined in a simple way, see (Bortolussi and Policriti 2010). Essentially, the discrete state space of the product automaton is $Q_1 \times Q_2$, while transitions from state (q_1, q_2) are all those issuing from q_1 or q_2 .

2.4 Hybrid semantics of sCCP

In this section, we briefly recall the definition of the hybrid semantics for sCCP (Bortolussi and Policriti 2009b), which

is given by associating a TDSHA to an sCCP program. The mapping is compositional: first single sCCP components are converted to TDSHAs, then one TDSHA is obtained as their product.

A TDSHA is, ultimately, an automaton and the discrete skeleton of the TDSHA associated with an sCCP component C is directly derived from the RTS—which is a labelled graph—of C . The level of discreteness/continuity will be a parameter, which is fixed by partitioning RTS edges $E(C)$ into discrete edges $E_d(C)$ and continuous edges $E_c(C)$. The former ones will generate stochastic transitions and the latter ones continuous transitions. The edge partition is encoded by a boolean vector κ of size $E(C)$, where $\kappa[e] = 1$ means that $e \in E_c(C)$. The recipe for constructing the TDSHA $\mathfrak{T}_{C,\kappa} = (Q_C, \mathbf{Y}, \mathcal{C}_C, \mathcal{D}_C, \mathcal{S}_C, init_C)$ of component C of $\mathfrak{A}^+$ is sketched below.

Discrete modes Continuous edges in $E_c(C)$ can short-circuit different RTS states of $S(C)$. Hence, we collapse together RTS-states connected by an (undirected) path of continuous edges. Let $[C_i]_c$ be the set—equivalence class—of states containing $C_i \in S(C)$. Then, $Q_C = \{[C_i]_c | C_i \in S(C)\}$.

Variables The TDSHA uses exactly the variables $\mathbf{Y}$ of the extension $\mathfrak{A}^+$ system variables $\mathbf{X}$ and state variables $\mathbf{P}$.

Continuous transitions Each edge in $e \in E_c(C)$ becomes a transition in $(q, \mathbf{s}, f) \in \mathcal{C}_C$, where $q = [C_i]_c$, with C_i the source state of e , $\mathbf{s}$ is the update vector defining R_e (i.e. R_e equals $\mathbf{Y}' = \mathbf{Y} + \mathbf{s}$), and $f(\mathbf{Y}) = \lambda_e(\mathbf{Y})$.

Stochastic transitions Each edge $e \in E_d(C)$ produces a discrete stochastic transition $(q_1, q_2, G, R, \lambda) \in \mathcal{S}_C$, where q_1 is the equivalence class containing the source state C_1 of e , q_2 contains the target state C_2 of e , $G \stackrel{\text{def}}{=} G_e$, $\lambda \stackrel{\text{def}}{=} \lambda_e$, and $R \stackrel{\text{def}}{=} R_e \wedge \text{statevar_res}$. The second term of R is needed to correctly deal with classes of states of $S(C)$: $\text{statevar_res} \stackrel{\text{def}}{=} (\bigwedge_{C \in q_1} P'_C = 0) \wedge P'_{C_2} = 1$.

Initial conditions $init_C$ is constructed by considering the initial state of the sCCP component in $\mathfrak{A}^+$.

Instantaneous transitions Instantaneous transitions are not needed at this level of the mapping. They have been used in (Bortolussi and Policriti 2009b) to describe dynamic partition policies between discrete and continuous transitions, and they will be used in the next section to give sCCP new functionalities.

Once we have constructed the TDSHA for each parallel component of $\mathfrak{A}^+$ corresponding to the initial network $A = C_1 \parallel \dots \parallel C_n$, we apply the (basically) standard product construction introduced at the end of Sect. 2.3 to obtain $\mathfrak{T}_{A, \kappa_1 \oplus \dots \oplus \kappa_n} = \mathfrak{T}_{C_1, \kappa_1} \otimes \dots \otimes \mathfrak{T}_{C_n, \kappa_n}$.

Lattice of TDSHA The reader can easily see that previous definition is parametric with respect to the degree of continuity introduced in the hybrid automata. We can arrange the different TDSHA obtained by different choices

of κ into a lattice, where the top element is the fully continuous and deterministic TDSHA, while the bottom element is the fully discrete and stochastic TDSHA. In particular, the fully continuous TDSHA has one single mode, no discrete transitions, and it corresponds to the set of ODE associated with an sCCP program by fluid-flow approximation (Bortolussi and Policriti 2009a). On the other hand, the fully discrete TDSHA is the CTMC associated with an sCCP by the standard semantics.

We stress the fact that sCCP is not a language designed to describe stochastic hybrid systems, but it is rather a programming language to model networks of (stochastically) interacting agents. The hybrid semantics has been defined as a computationally efficient approximation of the stochastic one. Consequently, the sCCP syntax has no way of specifying if an action has to be interpreted as discrete or continuous. This choice has to be performed at the RTS level. Practically, we are investigating the definition of general rules providing good automatic partitions (in terms of behaviour approximation and computational efficiency) and the external annotation of sCCP actions by naming them, similarly to BioPEPA (Ciocchetta and Hillston 2009). As a final remark, we point out that the asynchronous nature of sCCP allows the definition of the hybrid semantics in a relatively simple compositional way.

3 Extending sCCP: events and random updates

In this section, we will extend the sCCP language by equipping it with additional primitives that will be used, in our case study, to model prostate tumour dynamics. In particular, to model drug delivery policies, we need to describe SBML-like (Sbm; Ciocchetta 2009) events in sCCP. Furthermore, in the experiments, we will carry out in Sect. 5, we will also need to reset variables to random values, drawn according to given distributions. In the following, we will briefly introduce these extensions, both syntactically and semantically. More specifically, we will add new syntactic primitives to sCCP, framing their semantics within the skeleton of TDSHA, simply extending the hybrid semantics discussed in the previous section.

Events SBML-like events describe instantaneous modifications of the system, triggered by conditions on the system state or on the simulation time. Events allow us to model very easily the external influences on a system, such as, for instance, the hormone deprivation policies for prostate cancer. More precisely, following (Ciocchetta 2009), an SBML-like event is a conditional expression of the form $\text{if}(\text{trigger}) \text{ then}(\text{assignments}) \text{ with}(\text{delay})$,

where trigger is a condition involving system variables and the simulation time, assignment is an update of

(some) system variables, and delay is an optional time delay between the activation of the trigger and the consequent update of the variables.

We will first discuss how to describe events in sCCP with triggering actions depending on system's state only and with no delay. For this, it is sufficient to extend sCCP with instantaneous actions, i.e. actions taking *no time* to occur. Syntactically, this can be done simply by allowing rates to take infinity as value, i.e. we allow basic actions of the form $[G \rightarrow R]_\infty$. Even if instantaneous transitions are quite standard in stochastic process algebras (Hermanns et al. 2002; Bernardo and Gorrieri 1998; Ajmone Marsan et al. 1995), in the following we will take care to define their semantics within the TDSHA framework.

As opposed to instantaneous actions, time-controlled events are more delicate to express in stochastic process algebras, as they change the semantics of the language, from a CTMC to a generalized semi-Markov process (GSMP, German 2000; Glynn 1989), i.e. a stochastic process such that transition times can be sampled according to general distributions and that can keep memory of the remaining time to fire of any event. In particular, we are considering GSMPs mixing exponential and deterministic transition times. Such processes can be simulated using priority queues, as customary done in discrete event simulation. The introduction of time-controlled events in sCCP, however, is done by simply adding a special (*reserved*) variable, *Time*, to the language. The use of variable *Time* is restricted to infinite-rate actions, like $[Time = 10 \rightarrow R]_\infty$, and it *cannot* be reset. From a semantic point of view, the value of *Time* corresponds to the simulation time of the model. Of course, the guards of infinite-rate actions can combine in complex ways conditions on time and on system variables. Note that events with a delay d between the activation of the trigger and the consequent assignment to system's variables can be easily modelled within sCCP as a sequence of two instantaneous actions: $[\text{trigger} \rightarrow T' = \text{Time} + d]_\infty. [Time = T \rightarrow \text{assignment}]_\infty$.

Random updates The capability of updating system variables with random values can be useful in many applications, and it will be exploited in this paper to model uncertainty and intrinsic variability in (kinetic) parameters. For instance, we can model an action whose rate is a random variable rather than a number, abstracting in this way an (unknown) underlying mechanism giving rise to such randomness.

Alternatively, random resets can be used to incorporate uncertainty about a parameter in the model, according to the Bayesian point of view (Wilkinson 2006). In particular, we can select randomly the value of a parameter at the beginning of each simulation according to a given prior distribution (reflecting our knowledge on the parameter

value). Then, empirical distribution of the certain and the uncertain models is compared, assessing the sensitivity of the model with respect to the parameter.⁴

To formally add random resets in sCCP, we allow arbitrary reset functions in infinite-rate sCCP actions $[G \rightarrow R]_\infty$. Therefore, the reset R of such an actions can now be a conjunction of atoms of the form $X' = R(\mathbf{X}, \mathbf{P}, \mu)$, where μ is a vector of parameters, which can be constants or random variables, like in the definition of TDSHA of Sect. 2.3. We observe that there is no real obstacle in using general resets also in sCCP actions with finite rate, but in this case these actions cannot be approximated continuously in the hybrid semantics, but need to be kept discrete.

We can now formalise the extension of sCCP as follows:

Definition 3.1 An extended sCCP program is a tuple

$$\mathfrak{A} = (A, \mathfrak{D}_s, \mathfrak{D}_i, \mathbf{X}, \text{init}(\mathbf{X})), \text{ where}$$

1. $\mathbf{X}$ and $\text{init}(\mathbf{X})$ are as in definition 2.1.
2. The *initial network of agents* A , the *set of stochastic definitions* $\mathfrak{D}_s$ and the *set of instantaneous definitions* $\mathfrak{D}_i$ are given by the following grammar:

$$\begin{aligned} A &= M_s | M_i | A \parallel A \\ M_s &= \pi_s \cdot C_s | M_s + M_s \quad \pi_s = [G_s(\mathbf{X}) \rightarrow R_s(\mathbf{X}, \mathbf{X}')]_{\lambda(\mathbf{X})} \\ M_i &= \pi_i \cdot C_i | M_i + M_i \quad \pi_i = [G_i(\mathbf{X}) \rightarrow R_i(\mathbf{X}, \mathbf{X}', \mu)]_\infty \\ \mathfrak{D}_s &= \emptyset | \mathfrak{D}_s \cup \mathfrak{D}_s | \{C_s \stackrel{\text{def}}{=} M_s\} \\ \mathfrak{D}_i &= \emptyset | \mathfrak{D}_i \cup \mathfrak{D}_i | \{C_i \stackrel{\text{def}}{=} M_i\}, \end{aligned}$$

where G_i can contain also constraints on Time , while G_s cannot.

3.1 Mapping to TDSHA

To define the semantics of the extended version of sCCP, as anticipated, we will extend the semantics based on TDSHA. The formal definition of the TDSHA associated with an extended sCCP program follows the prescriptions of Sect. 2.4, with the following differences:

- Components with infinite rates $C \in \mathfrak{D}_i$ will generate a TDSHA $\mathfrak{T}_C$ in which each RTS-state C becomes a distinct mode of the TDSHA (i.e. all classes $[C]_c$ are singletons), and all its transitions are instantaneous. In particular, we will map an action $C = [G \rightarrow R]_\infty \cdot C_1 + M$ to instantaneous TDSHA transition $([C]_c, [C_1]_c, G, R, w)$, where the priority w is the constant function 1 (i.e. $w(\mathbf{X}) = 1$ for each $\mathbf{X}$).

⁴ An alternative way to incorporate lack of knowledge is to use imprecise probabilities, like in Imprecise Markov Chains (Skulj 2009).

- To properly capture the semantics of timed events, we add Time to the TDSHA's variables and we render its dynamics by adding a new automaton $\mathfrak{T}_{\text{time}}$ (sometimes called *time-monitor*) in parallel with the TDSHA obtained from the sCCP program. This automaton will have one single mode q , one variable (Time), initial state $(q, 0)$, and one single continuous transition, $(q, \mathbf{s}, f)$ —modeling the flow of time with rate $f = 1$ and stoichiometric vector $\mathbf{s} = \mathbf{1}$. As all actions containing Time have infinite rate, they will all become instantaneous transitions in the TDSHA. Therefore, an action with guard $\text{Time} = 10$ will activate as soon as TDSHA variable Time , which corresponds to the model time, will reach value 10.
- Random resets are dealt smoothly in this scheme: they will simply correspond to random resets of the TDSHA.

If we consider the TDSHA obtained by keeping all stochastic sCCP transitions discrete, then we can observe the following facts:

- The TDSHA associated with an extended sCCP program with instantaneous actions without conditions depending on Time in their guards remains a CTMC, provided the sCCP program cannot execute an infinite number of instantaneous actions in the same time instant (a well-known fact, cf. (Ajmone Marsan et al. 1995)). We call *loop-free* such sCCP programs. This holds because non-determinism between two or more active instantaneous transitions of TDSHA is resolved probabilistically by the use of priorities.
- The TDSHA associated with a generic (loop-free) extended sCCP program is stochastically equivalent to a generalised semi-Markov process.

We will now prove the second assertion (the first follows automatically). To do this, we first introduce the formal definition of a GSMP.

Definition 3.2 A GSMP is a tuple $\mathcal{G} = (S, s_0, \mathbb{E}, E, r, p, F)$, where

1. S is a countable set of states and $s_0 \in S$ is the initial state.
2. $\mathbb{E}$ is a countable set of events, and $E : S \rightarrow \wp(\mathbb{E})$ associates with each state s a subset of active events.
3. To each event $e \in \mathbb{E}$, we associate a clock taking values in the non-negative reals. The space of clock valuations is thus $\mathcal{C} = \mathbb{R}_+^\infty = \mathbb{R}_+ \times \mathbb{R}_+ \times \dots$, a clock valuation is denoted by $c \in \mathcal{C}$, and the value for the clock associated with e is c_e .
4. $r = S \times \mathbb{E} \rightarrow \mathbb{R}_+$ associates with each state and each event a rate, which is the rate at which the clock evolves. It satisfies $r(s, e) = 0$ if $e \notin E(s)$, and $r(s, e) > 0$ for at least one $e \in E(s)$.

5. $p(s', s, e)$ is the transition probability of jumping to state s' from state s , when event $e \in E(s)$ fired.
6. $F(\cdot, s, e)$ is the cumulative distribution function (CDF) of the clock associated with event $e \in \mathbb{E}$ in state s .

The dynamics of a GSMP process is informally defined in the following way. The initial state is s_0 (but a general distribution on the state space can be considered as well), and the initial valuation of clocks is defined by sampling the value of each active clock from the associated CDF in s_0 . Transitions from each state are executed according to the following policy: For each state, $s \in S$ and clock valuation c define the following quantities

- $t^* = t^*(s, c) = \inf\{t \geq 0 \mid \min_{e \in E(s)} \{c_e - t \cdot r(s, e)\} = 0\}$
- $c_e^* = c_e^*(s, c) = c_e - t^*(s, c)r(s, e)$
- $e^* = e^*(s, c) = \min\{e \in E(s) \mid c_e^*(s, c) = 0\}$ (assuming an ordering between events).

The event e^* is the event that fires in state s , where c represents the current clock valuation. t^* , instead, is the time elapsed in state s . After the firing of event e^* , the state is changed from s to s' with probability $p(s', s, e^*)$. Clock valuation c , instead, is changed as follows:

- The clocks of all events in $e \in R(s', s, e^*)$ are updated by sampling from their CDF $F(\cdot, s', e)$, where $R(s', s, e^*) = \{e^*\} \cup (E(s') \setminus E(s))$ is the set of newly enabled events, plus the event that fired.
- The clocks of all events $e \in W(s', s, e^*)$ are modified to take into account the elapsed time, i.e. they are set to the value $c_e^* \cdot W(s', s, e^*) = E(s') \cap (E(s) \setminus \{e^*\})$ is the set of all events remaining active after the transition.

The correctness of this construction is formally rendered in the following.

Theorem 3.1 The TDSHA $\mathfrak{T}$ obtained from an extended (loop-free) sCCP program, keeping all the non-instantaneous transitions stochastic, is stochastically equivalent to a GSMP.

Proof We just sketch the proof. We first need to define the GSMP associated with the fully discrete TDSHA. We will proceed in a similar way as in (Caravagna and Hillston 2010). The state space of the GSMP contains all the possible valuations of TDSHA variables, including *Time*. Hence, its state space is continuous. This is required to properly deal with events triggered by absolute conditions on time. Then, we associate a single clock c_{exp} with all stochastic transitions, and one clock c_η with each instantaneous transitions governed by a time constraint. All clocks advance at rate 1. The idea is that c_{exp} has an exponential CDF with rate in state s equal to the global exit time in s (i.e., the sum of all rates of all stochastic

transitions active in s), while the CDF of all other clocks is a deterministic function, constructed by properly inspecting the guard associated with the variable *Time*.⁵ The set of enabled transition in one mode depends on the guards predicates. The update kernel p , instead, is constructed taking into account resets, and possibly concatenating instantaneous transitions firing after a stochastic or a timed one, like in the mapping from TDSHA to PDMP (Bortolussi and Policriti 2012). In particular, it is defined in such a way to choose an active stochastic transition after the firing of the exponential event, with probability proportional to their rate, thus mimicking the SSA algorithm (Gillespie 1977). Furthermore, the update kernel has to properly update the time component of GSMP state space to mimic model time. The correctness is then proved by coupling the two stochastic processes (the PDMP obtained from TDSHA and the associated GSMP) (Billingsley 1979) and then reasoning inductively on the number of timed events fired.

We stress that the advantage of defining the semantics of these extensions in terms of TDSHA (instead of defining them at the operational semantics level, which is possible at least for random resets) is that they apply to all semantics of sCCP, including the CTMC and ODE-based semantics, which are special instances of the TDSHA semantics.

Equipped with these extensions to sCCP, we now focus on prostate tumour growth modelling. Our first goal is to build a discrete and stochastic (programmable) model, capable of rendering the prostate tumour-growth model(s) presented in Sect. 2.1. This approach will be discussed in the following section, while the effects of noise in this model will be discussed in Sect. 5.

4 sCCP model of prostate cancer growth

In this section, we describe how to go from a mathematical model like the one of Sect. 2 to sCCP programs, illustrating the technique directly on the prostate tumour case.

The basic principles in our approach consist in identifying from the differential equations the sCCP *variables* and the sCCP *interactions*. Variables will roughly correspond to variables in the differential equation model. Interactions will be obtained by disassembling the right-end sides of the differential equations, identifying explicitly the different actions modifying the populations. Interactions will be described by agents of the network.

⁵ For instance, if an instantaneous transition η has a guard $\text{Time} = t_0$, t_0 constant, and the corresponding event e_η is activated at time t (this value is stored in the state s'), then the associated CDF will be the Dirac delta $F(\cdot, s', e_\eta) = \delta_{t_0-t}(\cdot)$. Note that conditions like $\text{Time} = K$, with K a variable that can be changed by other events, can be dealt with by introducing vanishing states in which the clock c_η gets reset.

For prostate tumour cancer, following (Tanaka et al. 2010), we use four variables X , Y , Z , and V standing for the (numbers of) AD cells, AI cells, androgen hormone molecules, and PSA molecules, respectively.

It will be convenient to classify interactions into two classes: cellular interactions (i.e. growth, death, mutation) and molecular interactions (i.e. production, degradation). For prostate tumour cancer, we have five cellular interactions, corresponding to growth and death for AI and AD cells, in addition to an interaction modelling mutation of AD into AI cells. As far as molecular interactions are concerned, we consider four of them, i.e. production and degradation of androgen hormone and PSA. The differential equation model defines the PSA level as the sum of the number of AI and AD cells. We have chosen, instead, to treat PSA production and degradation explicitly, to give more internal flexibility to the model.

$$\begin{aligned}
&\text{growthAD} :- [X > 0 \rightarrow X' = X + 1]_{G_X(X,Z)} \cdot \text{growthAD} \\
&\text{deathAD} :- [X > 0 \rightarrow X' = X - 1]_{D_X(X,Z)} \cdot \text{deathAD} \\
&\text{growthAI} :- [Y > 0 \rightarrow Y' = Y + 1]_{G_Y(Y,Z)} \cdot \text{growthAI} \\
&\text{deathAI} :- [Y > 0 \rightarrow Y' = Y - 1]_{D_Y(Y,Z)} \cdot \text{deathAI} \\
&\text{produceANDH} :- [\text{true} \rightarrow Z' = Z + 1]_{P_Z} \cdot \text{produceANDH} \\
&\text{degradeANDH} :- [Z > 0 \rightarrow Z' = Z - 1]_{D_Z(Z)} \cdot \text{degradeANDH} \\
&\text{producePSA} :- [\text{true} \rightarrow V' = V + 1]_{P_V(X,Y)} \cdot \text{producePSA} \\
&\text{degradePSA} :- [V > 0 \rightarrow V' = V - 1]_{D_V(V)} \cdot \text{degradePSA} \\
&\text{mutateADtoAI} :- [X > 0 \rightarrow X' = X - 1 \wedge Y' = Y + 1]_{M_{XY}(X,Z)} \cdot \text{mutateADtoAI} \\
&G_X(X, Z) = \alpha_x \left(k_1 + (1 - k_1) \frac{Z}{Z + k_2 \Omega_Z} \right) X \\
&D_Y(Y, Z) = \beta_y Y \\
&D_Z(Z) = \frac{Z}{\tau} \\
&D_X(X, Z) = \beta_x \left(k_3 + (1 - k_3) \frac{Z}{Z + k_4 \Omega_Z} \right) X \\
&M_{XY}(X, Z) = m_1 \left(1 - \frac{Z}{z_0 \Omega_Z} \right) \\
&P_V(X, Y) = \frac{\Omega_V(X + Y)}{N_0} \\
&G_Y(Y, Z) = \alpha_y \left(1 - d \frac{Z}{z_0 \Omega_Z} \right) Y \\
&P_Z = 0 \\
&D_V(V) = V
\end{aligned}$$

Model 2: List of agents of the sCCP program modelling the prostate tumour growth. The initial configuration of the network consists of all the agents above, running in parallel. Rates are obtained from the ODE of Sect. 2 by suitably scaling parameters according to conversion factors, to be denoted Ω_Z, Ω_V , and $N_0 \cdot \Omega_V$ is the conversion factor between PSA concentration and PSA numerosity. It corresponds to the approximate number of molecules in a nanogram per millilitre of PSA. Analogously, Ω_Z is the conversion factor for Z , i.e. the number of molecules giving a concentration of a nano-mole per litre. N_0 , instead, is the reference number of cells, defined as the number of cells that on average produce $1 \frac{\text{ng}}{\text{ml}}$ units of PSA. The production and degradation rates of PSA are defined so that the average stationary number of PSA molecules is $\frac{\Omega_V(X+Y)}{N_0}$ corresponding to a concentration equal to $x + y$, which is the PSA value computed in the ODE model of Sect. 2.

In Model 2, the reader can find the full network of agents corresponding the equations presented in Sect. 2. To illustrate how agents interact, a precise definition of rates must be given. Consider, e.g., the agent corresponding to growth of AD cells:

$$\text{growthAD} :- [X > 0 \rightarrow X' = X + 1]_{G_X(X,Z)} \cdot \text{growthAD}$$

The agent represents a loop in which the number of AD cells grows by one at each iteration. The growth takes place only when the guard $X > 0$ is satisfied, and it has the effect of incrementing the value of X by one unit. The rate of the interaction is directly derived from the differential equation models:

$$G_X(X, Z) = \alpha_x \left(k_1 + (1 - k_1) \frac{Z}{Z + k_2 \Omega_Z} \right) X.$$

The parameter Ω_Z appearing in the above rate is not found in the differential equation model: it is a scaling parameter necessary to convert the concentration z of the hormones into the molecular count Z . Other conversion factors are required for the size of cell populations X and Y (N_0) and for PSA concentration (Ω_V), see caption of Model 2. The significance of these conversion factor in the dynamics of the entire model will be discussed in the Sect. 5.

4.1 Hybrid dynamics and drug dispensation

The sCCP program described above corresponds to the Continuous Androgen Suppression (CAS) policy (Brawer 2006; Jackson 2004). As said, more effective drug dispensation policies have been studied and, in particular, the IAS has been proven effective to control prostate tumour development (Abrahamsson 2010). The mathematical rendering of such a policy calls naturally into play a hybrid model describing the on/off modes of drug dispensation. In our stochastic program, this is done by introducing a variable U (corresponding to the variable u of the differential equation model) governing the switch between on/off androgen deprivation policy. The syntactic feature of U is that it turns out to appear only in guards and not in rates: it is purely a control variable. More specifically, it appears in the new agent corresponding to androgen hormone production:

$$\begin{aligned}
&\text{produceANDHc} :- [U = 0 \rightarrow Z' = Z + 1]_{\frac{\Omega_Z}{\tau}} \cdot \text{produceANDHc} \\
&\quad + [U = 1 \rightarrow Z' = Z + 1]_{0} \cdot \text{produceANDHc}
\end{aligned}$$

The above agent replaces the corresponding agent (produceANDH) in Model 2.

Typical drug dispensation policies for androgen deprivation control PSA concentration at fixed time intervals

(usually every 4 weeks) and determine whether dispensation should be resumed/suspended by checking whether specific threshold values are reached. To describe such a policy, we can use sCCP agents with infinite rate and guards on system time, as described in Sect. 3. More precisely, we just need a variable W , which will describe in which day the level of PSA will be controlled, initially set to $\delta_t = 28$, and add in parallel to the sCCP program the agent `checkPSA_on`, defined by:

$$\begin{aligned} \text{checkPSA_on: } & -[\text{Time} = W \wedge V < v_{\text{on}} \rightarrow W' = W + \delta_t \\ & \wedge U' = 0]_{\infty}.\text{checkPSA_off} \\ & +[\text{Time} = W \wedge V \geq v_{\text{on}} \rightarrow W' = W + \delta_t]_{\infty}.\text{checkPSA_on} \\ \text{checkPSA_off: } & -[\text{Time} = W \wedge V \geq v_{\text{off}} \rightarrow W' = W + \delta_t \\ & \wedge U' = 1]_{\infty}.\text{checkPSA_on} \\ & +[\text{Time} = W \wedge V < v_{\text{off}} \rightarrow W' = W + \delta_t]_{\infty}.\text{checkPSA_off} \end{aligned}$$

Note that, according to the semantics of Sect. 3, this is a proper hybrid model, as the explicit management of simulation time is a continuous ingredient. However, our semantics machinery in terms of TDSHA allows a much higher degree of flexibility, for instance treating (some of) the system variables as continuous to speed up simulation times.

4.2 Lotka–Volterra model of AD–AI competitiveness

In this section, we consider a variant of the model of Ahiara et al (Tanaka et al. 2010). Androgen-independent cells, as their name suggests, should be a line of tumour cells which grows independently in the presence of androgen hormones. However, in the model of Ahiara et al. (Tanaka et al. 2010), their growth rate depends on hormone concentration, more precisely it is large in absence of testosterone, and close to zero in presence of it. This dependence is purely phenomenological and tries to capture the competition between AD and AI cells.

Here, we consider a variant in which competition between AD and AI cells is modelled as a prey–predator interaction, in which AD cells “predate” AI cells, killing them. We want to stress that this interaction has to be interpreted in a purely phenomenological way. It is just an high level description of an unknown underlying mechanism. The rate of interaction is defined similarly to the Lotka–Volterra prey predator model, i.e. it is of the form $k_h XY$.

In addition to adding the predation of AI cells by AD cells, we modify the AI production rate, making it independent on the androgen hormone concentration. To keep the model simple, we assume a constant growth rate per AI

cell, hence with global rate $\alpha_Y Y$. Finally, we also modify the mutation rate, assuming that mutation can happen only unidirectionally from AD to AI cells, with rate inversely proportional to concentration of hormone, i.e. $\mu_{X,Y} \frac{1}{Z+k_{X,Y}}$.

The sCCP model is obtained in a straightforward way, modifying the rates of AI growth and mutation agents according to the discussion above, and introducing a new agent modelling the predation.

5 Experimental results

In this section, we present an experimental analysis of the models of the previous section. Additional figures are available in the Online Resource 1.

The parameters are fixed to the same values of the ODE model (see Model 1). The three scaling factors, N_0 , Ω_Z and Ω_V , have been fixed to values that we deemed meaningful and that were checked against real data, see the caption of Fig. 1.⁶

The first analysis that we consider is the comparison of the evolution of the agent-based stochastic model with the temporal evolution of ODE. In Fig. 1a, b, we compare the dynamics of PSA for the CAS and IAS drug dispensation policies, respectively. We show only the value of PSA, as it is the only observable quantity of the model. As we can see, the two models behave essentially in the same way. Even if a similar behaviour had to be expected, the fact that the stochastic systems basically show no noise at all may be seen as quite surprising. Actually, this phenomenon can be easily explained observing that the variables we are considering are all taking large values, i.e. they correspond to large populations, on the order of millions of cells or millions of molecules (in Fig. 1, a PSA level of 10 corresponds to 1 million molecules). In these circumstances, the relative magnitude of fluctuations, which is of the order of $\frac{1}{\sqrt{N}}$, is too small to produce significant effects (Gillespie 2000). This essentially means that the variability in behaviour between single cells is lost when we consider large populations: the differences cancel out and the observed behaviour essentially coincides with the average one.

Looking at the statistical behaviour at a fixed time ($t = 1,000$ for CAS and IAS; distribution for 100 runs is shown in Figs. S8d and S9d, we observe that the stochastic

⁶ Changing the value of the scaling factors can be seen as assuming a different granularity in counting. For instance, if we have N tumour cells, and we change the reference parameter N_0 from 1 to 10, it means that we count how many groups of 10 cells are present in the system.

Fig. 1 **a, b** Comparison of simulated trajectories of PSA in the sCCP programs and in the ODE models, both under CAS and IAS drug dispensation policies. Stochastic trajectories are generated by a Gillespie-like simulation algorithm (Gillespie 1977). The time unit is a day. Parameters of ODE are as in Model 1. Scaling parameters of the stochastic model are set to $N_0 = 10^5$, $\Omega_Z = 10^5$, $\Omega_V = 10^5$. PSA of the stochastic model is rescaled as $v = \frac{V}{\Omega_V}$ before being plotted, so to have the same scale in both modes. Initial values of scaled variables are $x = 15$, $y = 0.1$, $z = 12$. In the CAS policy, chemical castration is always in force. The IAS policy adopted is the following:

every 4 weeks (28 time units) the value of PSA is checked. Drug dispensation is interrupted if normalized PSA has dropped below 4, while it is resumed when PSA exceeds 10. The behaviour of the stochastic model is essentially indistinguishable from the behavior of the ODE one. This basically happens for scaling parameters greater than 10^3 , hence their precise values do not have a relevant impact in the dynamics, as long as they are large enough. **c** Comparison of single trajectories for CAS policy, and different values of N_0 . **d** Single trajectory for IAS policy in the presence of an external random disturbance source

processes are approximately normally distributed (Shapiro–Wilk test, null hypothesis cannot be rejected at 99 % confidence, p value 0.03123 for CAS, 0.7941 for IAS) with a very small dispersion (coefficient of variation: 0.017 for CAS, 0.012 for IAS), meaning that the distribution is almost deterministic. Furthermore, the mean of the stochastic process is very close to the value of the ODE (CAS, relative error of 0.8 %, IAS, relative error of 0.4 %).

Second, we compare the full stochastic model for $N_0 = 10^5$, as described in Model 2, with a reduced stochastic model in which PSA value is not subject to stochastic evolution, but it is set equal to $\frac{\Omega_V(X+Y)}{N_0}$, which is its equilibrium value at fixed numerosness of X and Y . In terms of the stochastic model, this essentially corresponds to assume that PSA is in Quasi-Steady State. See Fig. S1

for a comparison of trajectories. Looking at the distributions at time $t = 833$ for CAS and $t = 850$ for IAS (Fig. S2), we see that they are very similar. In particular, they are all normally distributed (Shapiro–Wilk, rejection of null hypothesis at 99 % confidence, p value 0.03123 for full CAS, 0.1463 for QSSA CAS, 0.7941 for full IAS, 0.4507 for QSSA IAS), and their variances cannot be distinguished (F test, p value 0.2838 for CAS, 0.6356 for IAS). This essentially means that the noise profile is the same both for the QSSA model and the full stochastic one. As for the average, they are distinguishable (t test, p value 0.003 for CAS, 3.157×10^{-6} for IAS). This fact is easily explained as in the full stochastic model the value of PSA is not perfectly synchronized with the number of cells (as in the QSSA one), due to the delay caused by the explicit

treatment in the dynamics of PSA. However, shifting right the distribution of the full stochastic model by the difference in means, the two distributions cannot be distinguished any longer (Kolmogorov–Smirnov test, t , p value 0.3667 for CAS, 0.967 for IAS).

Thanks to this fact, in the following we will mainly make use of the reduced stochastic model (which can be straightforwardly encoded in sCCP, letting PSA values be modified by the reset of growth and death agents for AI and AD cells), unless we are specifically considering variations in the stochastic production and degradation mechanisms for PSA.

5.1 Dependency on N_0

One of the advantages of having a discrete and stochastic model is that we can also study the behaviour of the model when the population of cells is small. One way to perform this experiment in this setting is to decrement the parameter N_0 , with the overall effect of reducing the total number of cancer cells. In Fig. 1c, we show the evolution of the number of tumour cells for different values of N_0 . As expected, as N_0 decreases the evolution becomes more noisy. Interestingly, for very small populations, the model changes its behaviour: there is a considerable probability that AI cells can extinguish before AD cells are eliminated by chemical castration, so that the tumour tends to be completely eliminated and no relapse can be observed. These low numbers may describe a situation during the initial stages of the tumour. Interestingly, the IAS therapy is less effective for low populations, as it reduces the chance of extinction of AD cells, so that the probability of tumour extinction is considerably lower.

Reducing N_0 essentially reduces the overall number of tumour cells, so we expect a more noisy behaviour. Note that we always look at the concentration of PSA divided by N_0 , so that models for different N_0 values are directly comparable. Note also that, since PSA is computed directly from number of AI and AD cells, we are essentially studying the dynamics of the latter.

First, we consider the CAS policy. As we can observe from Figs. S3 and S6, noise increases while decreasing N_0 . A similar behaviour can be observed for IAS policy (Figs. S4 and S7). In Fig. S10, we report some statistics of the behaviour of PSA at time $t = 1,000$. We measure noise with the coefficient of variation (CV), which is the ratio between standard deviation and average. The larger the coefficient of variation, the more noisy is the distribution. As we can see, CV is large for small N_0 , and becomes negligible for large values of N_0 . A visual comparison of the distributions can be seen in Figs. S8 and S9. Interestingly, the noise for small values of N_0 is

mainly caused by the fact that a relevant proportion of trajectories manifests tumour extinction. This is caused by the extinction of AI cells before the extinction of AD cells caused by chemical castration. More details are given in the next section.

In particular, in the IAS case, we can observe a large variety of behaviours for small values of N_0 (Fig. S5): the tumour can die, after a more or less long period of time, or it can remain controlled, or it can even exhibit a relapse phase. This happens because fluctuations of AI cells brings their number above a critical threshold, so that chemical castration is never interrupted.

5.2 Tumour extinction

Below, we look in more detail at the tumour extinction scenarios for small values of N_0 . We estimate the probability of extinction before time $t = 1,000$ both for CAS and IAS policies, from a sample of 1,000 trajectories. The interval estimates are done using the Wilson method at 95 % confidence. As we can see (Fig. 2), CS system has a much larger probability of extinction for values of N_0 between 1 and 40. This is because the intermittent androgen therapy often stops castration before all AD cells are extinct, even if AI cells are. Furthermore, extinction times (conditional to extinction happening before time $t = 1,000$) are much more narrowly distributed for CAS than for IAS (Fig. S12), again due to the effect of stopping chemical castration in the latter policy.

Looking at relapse probability in IAS therapy, we can observe that it happens with a very small probability for $N_0 > 10$ (see Fig. 2).

5.3 Internal noise source

The sCCP program model of tumor growth of Sect. 4 essentially does not show any noise in case of large cellular populations. Therefore, we considered possible modifications of the model to introduce some form of *internal* variability. This approach can be justified to check whether the variability in PSA concentration that is observed in real measurements can be explained by the simple structure of this model. First, we modified the model adding a random source of variation in the PSA production rate. We present an experiment in which the production rate of PSA is no longer a constant with respect to the total number of tumor cells, but it is variable. In particular, we assume that production rate of PSA is a random number, uniformly distributed in the interval $[0, 2\frac{\Omega_V(X+Y)}{N_0}]$. This is accomplished in the sCCP program by introducing a new variable K (the new production rate) and replacing the agent `producePSA` with the following one:

Fig. 2 Estimate of tumour extinction probability as a function of N_0 , with confidence intervals, for CAS and IAS policy. Estimates are based on 1,000 samples

$$\text{producePSA}' : -[true \rightarrow K' = \text{Unif}(0, 2)]_{\infty}.[true \rightarrow V' = V + 1]_{K_{N_0}^{\alpha_V(X+Y)}}.\text{producePSA}'.$$

Interestingly, even a large randomness in the PSA production rate does not result in a significant noisy behaviour: also in this case the fluctuations are averaged out.

We compared the standard stochastic model (with explicit PSA production and degradation) with the model in which PSA production rate is assumed to be a random value instead of a constant. A visual comparison of trajectories is provided in Fig. 3. As we can see, there is a change in the shape of the dynamics, both for CAS and for IAS (mainly for the latter). Moreover, IAS trajectories for the random production rate model show a bistable behaviour caused by the interface between noise and drug administration policy (cf. the caption of Fig. 3). If we look at noise behaviour, comparing the estimated profile of coefficient of variation (Fig. S13), we observe that the increase in CAS policy is not so relevant (at time $t = 833$, the CV for the standard model is 0.017, while for the other model is 0.02). As for IAS, the big increase is caused by the splitting of trajectories (Fig. S13b).

5.4 External noise source

We also considered a different modification of the model, in which we try to see whether noise may emerge as a consequence of the interaction with some *external* mechanism, possibly dependent on some global (physiological) condition. Specifically, we added a variable H (for “hidden”) that can assume just two possible values, namely 0 and 1, representing the presence/absence of an unspecified (physiological) condition. We assume that H switches from 0 to 1 and vice versa twice a day on average, and that the production rate of PSA is subject to a threefold increase when $H = 1$. Hence, we added to the model the following agent:

$$\text{hidden} : -[H = 0 \rightarrow H' = 1]_2.\text{hidden} + [H = 1 \rightarrow H' = 0]_2.\text{hidden},$$

and we replaced the PSA degradation rate by $(1 + 2H)$. Trajectories for this model are shown in Fig. 1d. In this case, we observed noise also in the case of large populations.

We compared the standard stochastic model, with explicit treatment of production and degradation of PSA, with the model in which the PSA degradation rate depends

Fig. 3 Comparison of 100 simulated trajectories, both for CAS and IAS policy, of the standard stochastic model and of the model with random production rate for PSA, cf. Sect. 5 of the main paper. For IAS, we observe a bifurcation in the trajectories. This happens because of the interaction between the increased variability of the

on an external factor, changing value twice a day on average. As can be seen from Figs. S14 and S15, the latter system manifests a remarkable increase in noise. In fact, looking at the statistics at time $t = 833$ for CAS policy and $t = 850$ for IAS policy, we observe a 18-fold increase in the coefficient of variation for CAS and an 56-fold increase in IAS (CV standard CAS: 0.017; CV CAS with external factor: 0.312; CV standard IAS: 0.012; CV IAS with external factor: 0.677).

The analysis we carried out is different from the one of (Tanaka et al. 2010). In this paper, noise is introduced in the model as an additional disturbance term in the ODEs (hence using Stochastic Differential Equations). In this case, however, the model does not provide any possible explanation of noise in terms of mechanisms intrinsic to the system under study. On the contrary, our study is aimed at better clarifying if the observed noise can be explained in terms of specific mechanisms, i.e. in terms of intrinsic properties of a given model. Note that some noise will inevitably be introduced by factors external to the system, for instance, by measurement errors.

What we understood from the analysis presented here is that the structure of phenomenological tumour cell growth models, like the one considered in this paper, may not be sufficiently rich to contain internal mechanisms for noise generation. If one is interested in these issues, then more complex models, taking into account more detailed biological mechanisms, should be considered. These models can also be easily described in our programming framework.

We stress that both internal and external noise models considered in the above two subsections, do not necessarily have any biologically significant interpretation. They are just illustrative examples of studies for possible sources of noise in systems with large populations. Moreover, our

process with a checkpoint for PSA value: this check is done near the deactivation threshold for chemical castration, giving rise to the splitting of trajectories. Note that, in any case, the randomness in the production rate changes the shape of the dynamics, particularly for the IAS model

experience shows that this kind of analysis is more easily carried out in a discrete and stochastic setting.

5.5 Experimental results on the competitive model

To conclude our experimental tour, we discuss the quantitative and qualitative behaviour of the model encoding some competitive AD–AI mechanism in its equations. To fix the parameters of the model presented in Sect. 4.2, we adopted the following strategy: the parameters of interactions that have not been modified from Ahiara original model are kept to the same value. The parameters of the newly introduced predation (k_{lv}) and of the two modified interactions (α_Y , $\mu_{X,Y}$, and $k_{X,Y}$) are fitted to the dynamics of the Ahiara models. Our goal in doing this is to show that the new model can match qualitatively and quantitatively the original model of Ahiara.

As a parameter fitting routine, we used the Evolutionary Algorithm strategy of COPASI (Hoops et al. 2006), running it for 1,000 generations, and obtained the following values: $k_{lv} = 0.0722$, $\alpha_Y = 0.02442$, $\mu_{X,Y} = 0.010$, and $k_{X,Y} = 0.0001$.

In Fig. 4, we compare the dynamics of our model with the one of Ahiara, in case of CAS and IAS policy. As can be seen, the qualitative behaviour is essentially preserved, and the agreement is good also from a quantitative point of view, a part from a small variation in the period of the PSA oscillations for the IAS policy.

6 Conclusion

The technique presented in this paper consists in showing how to step from a differential equation model to a

Fig. 4 Comparison on Ahiara et al. (Ideta et al. 2008) model with the Lotka–Volterra-based variant, both for CAS and IAS policies. Parameters are as in Model 1, and in the text of Sect. 5.5

“program” model, taking the form of a network of interacting agents. The specific programming language we used allows us also to introduce a stochastic element (internal to the model and) rendered as the *speed* at which any specific interaction takes place. In particular, we considered few extensions of sCCP programs to describe time-driven events and random updates. General sCCP programs, in addition, can easily model cell duplication events in the low-level model of Sect. 2, by an unrestricted usage of parallel composition and local variable declaration. In general, communication between agents representing single cells can be easily encoded in an asynchronous setting like the one of sCCP using dedicated variables, playing the role of communication channels, or modelling protein-mediated interaction. Exploiting the programmability of the shared memory (constraint store), one can easily introduce spatial information (Bortolussi and Policriti 2009c) or more complex cell interaction rules. Hence, sCCP programs allow us to model explicitly geometrically qualified interactions or complex competitive dynamics regulating cells growths and deaths. However, a satisfactory definition of the hybrid semantics for this larger class of sCCP programs is still an open issue.

A programming environment like sCCP allows the construction of a “wizard” for fast prototyping of (cancer) cell population dynamics. Among other things, this approach should allow us to easily address such basic questions as the effect and nature of noise, parameter dependencies, logical structure of the interactions, etc. More advanced analysis techniques, like statistical model checking (Jha et al. 2009), will further enhance the framework. Moreover, an additional advantage of such a

framework with respect to models based solely on ODEs is semantic flexibility: the modeller has many semantics at its disposal (including ODEs) and hence can easily compare stochastic, deterministic, and hybrid version of the model under study, and use a wider set of analysis tools, each specific to one semantic interpretation.

We presented here a quantitative analysis on the nature of noise for a differential equation model of prostate cancer based on the construction of an agent-based version of the model. Specifically, we showed that the phenomenological interactions of this model are not able to explain observed noise in data. We suggested that a more detailed description of interaction and regulation mechanisms involved is needed to better clarify the noise effects. We plan to further investigate this direction, taking also into account spatial organization of the tumour.

Our future work will also benefit from a comparison with experimental data,⁷ as well as from a further study of the usage of our tool in connection with the availability of data from current cancer genome sequencing project. An analysis of such kind of data for the design of interaction networks of the type presented here would provide an—entirely different and—interest application of our methodology. In such a study, the mechanisms governing different behaviours of the network could be related directly with such parameters as somatic mutations or structural variation to be found in specific cancer cells. We plan to work in this direction in the future.

⁷ This is not so relevant for the work presented here, given that the stochastic and the ODE model are essentially indistinguishable and given that a comparison with experimental data has been carried out in (Ideta et al. 2008) for the ODE model.

References

- Abrahamsson PA (2010) Potential benefits of intermittent androgen suppression therapy in the treatment of prostate cancer: a systematic review of literature. *Eur Urol* 57:49–59
- Ajmoné Marsan M, Balbo G, Conte G, Donatelli S, Franceschini G (1995) Modelling with generalized stochastic petri nets. Wiley, New York
- American urological association (2012). Best practice policy statement on cryosurgery for the treatment of localized prostate cancer
- Bernardo M, Gorrieri R (1998) A tutorial on empa: a theory of concurrent processes with nondeterminism, priorities, probabilities and time. *Theor Comput Sci* 202:1–54
- Billingsley P (1979) Probability and Measure. Wiley, New York
- Bortolussi L, Policriti A (2008) Modeling biological systems in concurrent constraint programming. *Constraints* 13(1):66–90
- Bortolussi L, Policriti A (2009a) Dynamical systems and stochastic programming—from ordinary differential equations and back. In: Transactions on computational systems biology XI. Lecture notes in computer science, vol 5750. Springer, Berlin, pp 216–267
- Bortolussi L, Policriti A (2009b) Hybrid semantics of stochastic programs with dynamic reconfiguration. In: Proceedings of the CompMod
- Bortolussi L, Policriti A (2009c) Tales of spatiality in stochastic concurrent constraint programming. In Proceedings of BioLogic09
- Bortolussi L, Policriti A (2010) Hybrid dynamics of stochastic programs. *Theor Comput Sci*
- Bortolussi L, Policriti A (2012) (hybrid) automata and (stochastic) programs, the hybrid automata lattice of a stochastic program. *J Log Comput*. doi:10.1093/logcom/exr045
- Bortolussi L, Policriti A (2011) Programmable models of growth and mutation of cancer-cell populations. In: Proceedings of the third international workshop on computational models for cell processes (COMPMOD 2011). EPTCS, vol 67, pp 19–33
- Brawer MK (2006) Hormonal therapy for prostate cancer. *Rev Urol* 8:S35–S47
- Caravagna G, Hillston J (2010) Modeling biological systems with delays in bio-pepa. In: Proceedings of the fourth workshop on membrane computing and biologically inspired process calculi 2010, Jena, Germany, 23 August 2010. Electronic Proceedings in Theoretical Computer Science, vol 40, pp 85–101
- Ciocchetta F (2009) Bio-PEPA with events. *T. Comput Syst Biol* 11:45–68
- Ciocchetta F, Hillston J (2008) Formal methods for computational systems biology. In: Process algebras in systems biology. Springer, Berlin, pp 265–312
- Ciocchetta F, Hillston J (2009) Bio-pepa: a framework for the modelling and analysis of biological systems. *Theor Comput Sci* 410(33–34):3065–3084
- Danos V, Feret J, Fontana W, Harmer R, Krivine J (2007) Rule-based modelling of cellular signalling. In Proceedings of the 18th international conference on concurrency theory, CONCUR 2007. Lecture Notes in Computer Science, vol 4703. Springer, Berlin, pp 17–41
- Davis MHA (1993) Markov Models and Optimization. Chapman & Hall, London
- Deisboeck TS, Wang Z, Macklin P, Cristini V (2011) Multiscale cancer modeling. *Annu Rev Biomed Eng* 13:127–155
- Deisboeck TS, Couzin ID (2009) Collective behavior in cancer cell populations. *BioEssays News Rev Mol Cell Dev Biol* 31(2):190–197
- De Matteis G, Graudenzi A, Antoniotti M (2012) A review of spatial computational models for multi-cellular systems, with regard to intestinal crypts and colorectal cancer development. *J Math Biol* (submitted)
- Fornari C, Cordero F, Manini D, Balbo G, Calogero R (2011) Mathematical approach to predict the drug effects on cancer stem cell models. *Electr Notes Theor Comput Sci* 277:29–39
- German R (2000) Performance analysis of communication systems with NonMarkovian stochastic petri nets. Wiley, New York
- Gillespie D (2000) The chemical langevin equation. *J Chem Phys* 113(1):297–306
- Gillespie DT (1977) Exact stochastic simulation of coupled chemical reactions. *J. Phys Chem* 81(25):2340–2361
- Glynn PW (1989) A gsmpl formalism for discrete-event systems. *Proc IEEE* 77:14–23
- Hermanns H, Herzog U, Katoen J-P (2002) Process algebra for performance evaluation. *Theor Comput Sci* 274(1–2):43–87
- Hoops S, Sahle S, Gauges R, Lee C, Pahle J, Simus N, Singhal M, Xu L, Mendes P, Kummer U (2006) COPASI: a COMplex PATHway SIMulator. *Bioinformatics* 22(24):3067–3074
- Ideta AM, Tanaka G, Takeuchi T, Aihara K (2008) A mathematical model of intermittent androgen suppression for prostate cancer. *Nonlinear Sci* 18:593–614
- Jackson TL (2004) A mathematical model of prostate tumor growth and androgen-independent relapse. *Disc Cont Dyn Sys B* 4:187–201
- Jha SK, Clarke EM, Langmead CJ, Legay A, Platzer A, Zuliani P (2009) A bayesian approach to model checking biological systems. In: Proceedings of the 7th international conference on computational methods in systems biology, CMSB 2009. Lecture Notes in Computer Science, vol 5688, pp 218–234
- Kohn KW, Aladjem MI, Kim S, Weinstein JN, Pommier Y (2006) Depicting combinatorial complexity with the molecular interaction map notation. *Mol Syst Biol* 2(51):1744–4292
- Lecca P, Kahramanogullari O, Morpurgo D, Priami C, Soo R (2011) Modelling the tumor shrinkage pharmacodynamics with blenx. In: 1st IEEE international conference on computational advances in bio and medical sciences (ICCABS)
- Mazza T, Cavaliere M (2009) Cell cycle and tumor growth in membrane systems with peripheral proteins. *Electr Notes Theor Comput Sci* 227:127–141
- Norris JR (1997) Markov chains. Cambridge University Press, Cambridge
- Rao AR, Motiwala HG, Karim OMA (2008) The discovery of prostate-specific antigen. *BJU Int* 101:5–10
- Roose T, Chapman SJ, Maini PK (2007) Mathematical models of avascular tumor growth. *SIAM Rev* 49(2):179–208
- Sbml website. <http://www.sbml.org>
- Skulj D (2009) Discrete time markov chains with interval probabilities. *Int J Approx Reason* 50(8):1314–1329
- Tanaka G, Hirata Y, Goldenberg SL, Bruchovsky N, Aihara K (2010) Mathematical modelling of prostate cancer growth and its application to hormone therapy. *Phyl Trans Royal Soc A* 368:5029–5044
- Wilkinson DJ (2006) Stochastic modelling for systems biology. Chapman & Hall, London