Neural Computing and Applications (2023) 35:6335–6359
https://doi.org/10.1007/s00521-022-07935-6

S.I. : THEORY AND PRACTICE OF NATURAL COMPUTING

# Quantitative extensions of reaction systems based on SOS semantics

Linda Brodo[1] · Roberto Bruni[2] · Moreno Falaschi[3] · Roberta Gori[2] · Francesca Levi[2] · Paolo Milazzo[2]

Received: 26 March 2022 / Accepted: 7 October 2022 / Published online: 24 January 2023

**Abstract**
Reaction systems (RSs) are a successful natural computing framework inspired by chemical reaction networks. A RS consists of a set of entities and a set of reactions. Entities can enable or inhibit each reaction and are produced by reactions or provided by the environment. In this paper, we define two quantitative variants of RSs: the first one is along the time dimension, to specify delays for making available reactions products and durations to protract their permanency, while the second deals with the possibility to specify different concentration levels of a substance in order to enable or inhibit a reaction. Technically, both extensions are obtained by modifying in a modular way the Structural Operational Semantics (SOS) for RSs that was already defined in the literature. Our approach maintains several advantages of the original semantics definition that were: (1) providing a formal specification of the RS dynamics that enables the reuse of many formal analysis techniques and favours the implementation of tools, and (2) making the RS framework extensible, by adding or changing some of the SOS rules in a compositional way. We provide a prototype logic programming implementation and apply our tool to three different case studies: the tumour growth, the Th cell differentiation in the immune system and neural communication.

**Keywords** Bioinformatics · SOS rules · Reaction systems · Logic programming

## 1 Introduction

Inspired by natural phenomena, many new computational formalisms have been introduced to model different aspects of biology. Basic chemical reactions inspired Ehrenfeucht and Rozenberg's reaction systems (RSs) [1, 2]: a *qualitative* modelling formalism that is based on two opposite mechanisms: *facilitation* and *inhibition*. Facilitation means that a reaction can occur only if all its reactants are present, while inhibition means that the reaction cannot occur if any of its inhibitors is present. A reaction system is a set of *reactions*, each determined by its *reactants*, *inhibitors* and *products*, over a (finite) support set of biological entities. The theory of RSs is based on three principles: no permanency, any entity vanishes unless it is sustained by a reaction; no competition, an entity is either available for all reactions, or it is not available at all; and no counting, the exact concentration level of available entities is ignored, as if it was always high enough to activate all enabled reactions.

Dynamically, RS exploit a discrete time model, where each state collects the entities that are present at a given time unit. The computation of the next state is a deterministic procedure. However, the overall dynamics is

Roberto Bruni, Moreno Falaschi, Roberta Gori, Francesca Levi and Paolo Milazzo have contributed equally to this work.

✉ Linda Brodo
brodo@uniss.it

Roberto Bruni
roberto.bruni@unipi.it

Moreno Falaschi
moreno.falaschi@unisi.it

Roberta Gori
roberta.gori@unipi.it

Francesca Levi
francesca.levi@unipi.it

Paolo Milazzo
paolo.milazzo@unipi.it

[1] Dipartimento di Scienze Economiche e Aziendali, Università di Sassari, Via Muroni 25, Sassari, Italy

[2] Dipartimento di Informatica, Università di Pisa, Largo B. Pontecorvo 3, Pisa, Italy

[3] Dipartimento di Ingegneria dell'Informazione e Scienze Matematiche, Università di Siena, Via Roma 56, Siena, Italy

influenced by the entities received (nondeterministically) from the external environment, called contextual entities. Such entities join the current state of the system and participate to enabling and disabling reactions. The behaviour of a RS is hence defined as a discrete time interactive process consisting of a *context sequence* (the sets of entities received at each unit of time from the environment), a *result sequence* (the sets of entities produced at each unit of time by reactions) and a *state sequence* (the join of the context sequence with the result sequence). Since their introduction, RSs have shown to be a quite general computation model whose application ranges from the modelling of biological phenomena [3–6] and molecular chemistry [7] to theoretical foundations of computing [8–10].

### 1.1 Towards extensible reaction systems

As it is often the case for computational models, the study of RSs has to balance between simplicity and expressiveness. If more accurate models are required to precisely account for certain aspects of biological system, then RSs must be extended accordingly to increase their predictive power, possibly making their theory less straight or even cumbersome and more difficult to approach [11–17]. Moreover, different kinds of enhancements proposed in the literature do not necessarily agree.

Our long-term goal is to develop a convenient way to embed RSs in a modular and extensible formal framework, where new extensions can be accommodated with a friendly syntax, so to match simplicity with increased expressiveness. To this aim, we plan to build on a process algebraic representation of RSs, whose dynamics can be expressed as a labelled transition system (LTS) generated by a small set of inference rules defined by structural induction (SOS style) [18, 19]. We have already exploited this technique to explore and experiment with locally scoped entities and recursively defined nondeterministic contexts, where a single LTS accounts for different evolutions of the same system. Among the main advantages of our approach we mention: (1) *transparency*: each transition label conveys information about all the activities connected to the execution step it describes; (2) *compositionality*: the behaviour of a composite system is defined in terms of the behaviours of its constituents; and (3) *extensibility*: RS variants can be enhanced by modifying/adding language operators and inference rules in a modular fashion.

### 1.2 The problem

In this paper, motivated by three quite different case studies taken from the literature, we investigate the possibility to devise general-purpose extensions of our framework to tackle quantitative features of biological systems. The three case studies are concerned with a model of drug administration in tumour growth, a complex gene network that regulates the differentiation of T lymphocytes and the modelling of synaptic transmission between neurons, which we briefly introduce below.

*Drug administration in tumour growth* We develop a model for comparing the efficacy of drug administration strategies to block the tumour growth. Our model is inspired by the delay differential equation model presented in [20], where delays are added to differential equations to describe the duration of the different phases of the cell cycle. In particular, we model the immune system response and a phase-specific drug able to alter the natural course of action of the cell cycle of the tumour cells. While a general method for transforming differential equation models into RSs in such a way to preserve all properties is likely unfeasible, we use the case study to demonstrate that, for this particular example, it is possible to exploit delays and durations in order to rediscover some of the phenomena also present in the differential equation model. In this case, the advantage is of course the simplicity offered by a discrete time model and by the key feature of RSs (*no permanency*, *no competition*, *no counting*).

*Regulating differentiation in Th cells* We focus on the discrete dynamical model for differentiation in Th cells as proposed in [21], which was able to reproduce the most important dynamics aspects of the regulatory process. While a previous RS encoding had to classify different levels of the same entity in separate objects (see [22]), thus requiring some arbitrary ad hoc classification, we exploit this case study to show some advantages of using numerical (discrete) concentration levels instead of (distinct) object levels in RS models.

*Synaptic transmission* We introduce a simple functional model with a quantitative abstraction for synaptic transmission, that is the process that allows two neurons connected by a synapse to communicate. Communication consists in impulsive chemical signals that are sent from the first neuron to the second. Chemical signals take the form of neurotransmitters that are released by the first neuron and perceived by the second neuron, and they are stimulated by ionic currents. Mathematical continuous models were applied to model the dynamics of this synapse communication [23]. Here, we do not consider the kinetic rates of the different biological reactions, but we make a combined use of delays, durations and concentration levels to model different facets of this complex phenomenon. Although our model is very simple and deterministic, we show that the dynamics of the entire system is faithfully modelled and can be compared to more complex approaches such as [24], where a stochastic modelling method is used.

### 1.3 Contribution

The above case studies serve to witness that the ability to account for reactions with different speeds or to define different reactions according to the levels of concentration of certain entities can play a very important role in the study of biological phenomena. Therefore, in this paper we propose two conservative extensions of RSs that try to increase the expressiveness while preserving its simplicity as much as possible.

First, we add the possibility to express reaction delays and durations, which makes it possible to encode reactions with different speeds. A reaction with associated delay $n$ will deliver its products after $n$ time units. The value zero for $n$ represents fastest reactions (0 is the smallest delay, the product being immediately available) but delays can otherwise take any integer value $n > 0$ to model slower and slower reactions. For reactions, the delay is thus the time that elapses between the enabling of the reaction itself and its product being available in the system. Following this idea also a duration of 'permanency' can be specified for each reaction. A reaction $r$ that has delay $n$ and duration $m$ will deliver, if applicable, its products after $n$ time units and such products will remain available for the following $m$ time units before vanishing.

The second enhancement adds some quantitative information to each entity to model concentration levels that can influence both the facilitation and inhibition mechanisms of reactions. Each entity in a reaction comes with an approximated quantitative threshold that will be necessary for enabling the reaction. We note that we still maintain a qualitative perspective on the biological system, since the concentration levels will be used to determine the set of reactions that can be applied in any step, whereas competition between different enabled reactions to 'consume' available reactants will not be considered.

Most importantly, although we formalise the two features separately for ease of presentation, they can be integrated and used in combination without much efforts. Also, they are conservative extensions, meaning that the new features can be readily cancelled out by some default parameters whenever we just want to study their ordinary RS counterparts. This witnesses the flexibility, extensibility and modularity of an SOS-based approach.

The formal specification has been instrumental to develop a prototype implementation in Prolog that allows us to perform computational experiments and to compute and inspect the resulting LTS. Since the code follows the formal specification very closely (apart for minor optimisations), its soundness is easy to check by code inspection and the implementation will be easy to extend if new features must be added.

The tool has been fundamental to experiment with the case studies, because even if small- to medium-sized specifications were sufficient to encode the biological systems under scrutiny, it would have been very difficult and time consuming to analyse their behaviour without any automatic support.

*Structure of the paper* In Sect. 2, we recall the basics of RSs. In Sect. 3, we recall the syntax and operational semantics of our process algebra for RSs. The original contribution starts from Sect. 4, where we add new features to our framework: we introduce the concepts of delay and duration in Sect. 4.1 and linear patterns for expressing constraints on the concentration levels of the entities in Sect. 4.2. Section 5 describes the related work. The logic programming implementation of the new features is described in Sect. 6. In Sects. 7–9, we show how the extensions proposed in Sect. 4 can improve the study of the three biological phenomena we selected. For each case study, we report the key findings of our experimentation with the tool from Sect. 6. Section 10 discusses some related work and concludes the paper.

This article is the full version of the conference paper [25], here extended with a more detailed account of the theory behind our RS enhancements, many small examples to illustrate the syntax and semantics of our models, an in-depth inspection of the first two case studies and an entirely new example about synaptic transmission. We have also notably extended the implementation of our tool adding several features. We just mention a new parser for our extended syntax and an automated graphical representation of the computations, which we illustrate by including in the paper some automatically generated figures and graphics.

## 2 Reaction systems

The theory of reaction systems (RSs) [2] was born in the field of Natural Computing to model the behaviour of biochemical reactions in living cells. While our contribution builds on a process algebraic presentation of RSs, we recall here the main concepts as introduced in the classical set theoretic version. In the following, we use the term *entities* to denote generic molecular substances (e.g., atoms, ions, molecules) that may form some biochemical system.

Let $S$ be a (finite) set of entities. A reaction in $S$ is a triple $a = (R, I, P)$, where $R, I, P \subseteq S$ are finite sets[1] such that $R \cap I = \emptyset$. The sets $R$, $I$ and $P$ are the sets of *reactants*,

[1] Usually, $R$ and $I$ are required to be nonempty sets, but we prefer to relax this constraint to avoid the introduction of dummy entities and keep our models easier to read. We require instead that $R \cup I$ is not empty.

*inhibitors* and *products*, respectively. All reactants have to be present in the current state for the reaction to take place. The presence of any of the inhibitors blocks the reaction. Products are the outcome of the reaction, to be released in the next state. We denote with $rac(S)$ the set of all reactions over $S$. A reaction system is a pair $\mathcal{A} = (S, A)$ where $S$ is the set of entities, and $A \subseteq rac(S)$ is a finite set of reactions over $S$.

Given the current set of entities $W \subseteq S$, the result of a reaction $a = (R, I, P) \in rac(S)$ on $W$, denoted $res_a(W)$, is given by:

$$res_a(W) \triangleq \begin{cases} P & \text{if } en_a(W) \\ \emptyset & \text{otherwise} \end{cases} \qquad en_a(W) \triangleq R \subseteq W \ \wedge \ I \# W$$

where $en_a(W)$ is called the *enabling predicate* for the reaction $a$ in state $W$: it requires that all reactants $R$ are present in $W$ and that no inhibitor $I$ is present in $W$, here written $I \# W$ as a shorthand for $I \cap W = \emptyset$ (it reads as 'the sets $I$ and $W$ are disjoint'). Similarly, the result of the application of all reactions in $\mathcal{A}$ to $W$, denoted $res_A(W)$, is the obvious lifting of the above function, i.e.,

$$res_A(W) \triangleq \bigcup_{a \in A} res_a(W).$$

Since living cells are seen as open systems that react to environmental stimuli, the behaviour of a RS is formalised in terms of an *interactive process*. Let $\mathcal{A} = (S, A)$ be a RS and let $n \geq 0$. An $n$-step *interactive process* in $\mathcal{A}$ is a pair $\pi = (\gamma, \delta)$ s.t. $\gamma = \{C_i\}_{i \in [0,n]}$ is the *context sequence* and $\delta = \{D_i\}_{i \in [0,n]}$ is the *result sequence*, where $C_i, D_i \subseteq S$ for any $i \in [0, n]$, $D_0 = \emptyset$, and $D_{i+1} = res_A(C_i \cup D_i)$ for any $i \in [0, n-1]$. The context sequence $\gamma$ represents the environment, while the result sequence $\delta$ is entirely determined by $\gamma$ and $A$. We call $\tau = \{W_i\}_{i \in [0,n]}$ the *state sequence*, with $W_i \triangleq C_i \cup D_i$, for any $i \in [0, n]$. Note that each state $W_i$ in $\tau$ is the union of two sets: the context $C_i$ at step $i$ and the result set $D_i = res_A(W_{i-1})$ from the previous step.

***Example 1*** We consider a toy RS defined by

$$\mathcal{A} \triangleq (S, A) \qquad S \triangleq \{\mathsf{a}, \mathsf{b}, \mathsf{c}\} \qquad A \triangleq \{a_1\}$$

whose unique reaction is $a_1 \triangleq (\{\mathsf{a}, \mathsf{b}\}, \{\mathsf{c}\}, \{\mathsf{b}\})$, to be written more concisely as $(\mathsf{ab}, \mathsf{c}, \mathsf{b})$. Then, we consider a 3 - step interactive process $\pi \triangleq (\gamma, \delta)$, where

$$\gamma \triangleq \{C_0, C_1, C_2, C_3\}, \qquad \delta \triangleq \{D_0, D_1, D_2, D_3\},$$

with $C_0 \triangleq \{\mathsf{a}, \mathsf{b}\}$, $C_1 \triangleq \{\mathsf{a}\}$, $C_2 \triangleq \{\mathsf{c}\}$, and $C_3 \triangleq \{\mathsf{c}\}$; and $D_0 \triangleq \emptyset$, $D_1 \triangleq \{\mathsf{b}\}$, $D_2 \triangleq \{\mathsf{b}\}$, and $D_3 \triangleq \emptyset$. The context sequence can be written more concisely as $\gamma = \mathsf{ab}, \mathsf{a}, \mathsf{c}, \mathsf{c}$, and similarly, the result sequence can be shortened as $\delta = \emptyset, \mathsf{b}, \mathsf{b}, \emptyset$. Then, the resulting state sequence is $\tau = \{W_0, W_1, W_2, W_3\} = \mathsf{ab}, \mathsf{ab}, \mathsf{bc}, \mathsf{c}$. In fact, it is easy to check that, e.g., $W_0 = C_0$, $D_1 = res_A(W_0) = res_A(\{\mathsf{a}, \mathsf{b}\}) = \{\mathsf{b}\}$ because $en_{a_1}(W_0)$, and $W_1 = C_1 \cup D_1 = \{\mathsf{a}\} \cup \{\mathsf{b}\} = \{\mathsf{a}, \mathsf{b}\}$.

## 3 SOS rules for reaction systems

Inspired by process algebras such as CCS [26], in [18, 19, 27] the authors introduced an algebraic syntax for RSs and equipped it with SOS inference rules defining the behaviour of each operator. This made it possible to consider a LTS semantics for RSs, where states are terms of the algebra, each transition corresponds to a step of the RS and transition labels retain some information on the entities needed to perform each step. In this paper, we build on the approach in [19], which we briefly summarise below.

**Definition 2** (*RS processes*) Let $S$ be a set of entities. An *RS process* $\mathsf{P}$ is any term defined by the following grammar:

$$\mathsf{P} ::= [\mathsf{M}] \quad \mathsf{M} ::= (R, I, P) \,|\, D \,|\, \mathsf{K} \,|\, \mathsf{M}|\mathsf{M}$$
$$\mathsf{K} ::= \mathbf{0} \,|\, X \,|\, C.\mathsf{K} \,|\, \mathsf{K} + \mathsf{K} \,|\, \mathsf{rec}X.\mathsf{K}$$

where $R, I, P \subseteq S$ are nonempty sets of entities, $C, D \subseteq S$ are possibly empty set of entities, and $X$ is a process variable.

An RS process $\mathsf{P}$ embeds a *mixture* process $\mathsf{M}$ that is an arbitrary parallel composition of reactions $(R, I, P)$, (possibly empty) sets of currently present entities $D$, and *context* processes $\mathsf{K}$. We write $\prod_{i \in I} \mathsf{M}_i$ for the parallel composition of all $\mathsf{M}_i$ with $i \in I$. For example, we let $\prod_{i \in \{1,2\}} \mathsf{M}_i = \mathsf{M}_1 \mid \mathsf{M}_2$.

***Example 3*** A mixture process containing the reaction $a_1$ from Example 1 with initial entities $\mathsf{a}$ and $\mathsf{b}$ can be written $(\mathsf{ab}, \mathsf{c}, \mathsf{b}) \mid \mathsf{a} \mid \mathsf{b}$, and its RS process as $[(\mathsf{ab}, \mathsf{c}, \mathsf{b}) \mid \mathsf{a} \mid \mathsf{b}]$.

A process context $\mathsf{K}$ is a possibly nondeterministic and recursive system: the nil context $\mathbf{0}$ halts the computation; the prefixed context $C.\mathsf{K}$ says that the entities in the (possibly empty) set $C$ are immediately available to be consumed by the reactions, and then, $\mathsf{K}$ is the context offered at the next step; the nondeterministic choice $\mathsf{K}_1 + \mathsf{K}_2$ allows the context to behave either as $\mathsf{K}_1$ or $\mathsf{K}_2$; $X$ is a process variable, and $\mathsf{rec}\, X.\, \mathsf{K}$ is the usual recursive operator of process algebras that intuitively corresponds to the recursive definition $X = \mathsf{K}$ (see Example 4). We write $\sum_{i \in I} \mathsf{K}_i$ for the nondeterministic choice between all $\mathsf{K}_i$ with $i \in I$. For example, we let $\sum_{i \in \{1,2\}} \mathsf{K}_i = \mathsf{K}_1 + \mathsf{K}_2$.

***Example 4*** The context process $\mathsf{a}.\mathsf{b}.\mathbf{0}$ represents a context sequence where $C_0 = \{\mathsf{a}\}$ and $C_1 = \{\mathsf{b}\}$, while $\mathsf{a}.(\mathsf{b}.\mathbf{0} + \mathsf{c}.\mathbf{0})$ represents a nondeterministic context that initially

**Fig. 1** SOS semantics of the RS processes

$$\frac{}{D \xrightarrow{\langle D\rhd\emptyset,\emptyset,\emptyset\rangle} \emptyset}\ (\mathit{Ent}) \qquad \frac{}{C.\mathsf{K} \xrightarrow{\langle C\rhd\emptyset,\emptyset,\emptyset\rangle} \mathsf{K}}\ (\mathit{Cxt})$$

$$\frac{\mathsf{K}_1 \xrightarrow{\ell} \mathsf{K}_1'}{\mathsf{K}_1 + \mathsf{K}_2 \xrightarrow{\ell} \mathsf{K}_1'}\ (\mathit{Suml}) \qquad \frac{\mathsf{K}_2 \xrightarrow{\ell} \mathsf{K}_2'}{\mathsf{K}_1 + \mathsf{K}_2 \xrightarrow{\ell} \mathsf{K}_2'}\ (\mathit{Sumr}) \qquad \frac{\mathsf{K}[^{\mathsf{rec}\ X.\mathsf{K}}/_{X}] \xrightarrow{\ell} \mathsf{K}'}{\mathsf{rec}\ X.\ \mathsf{K} \xrightarrow{\ell} \mathsf{K}'}\ (\mathit{Rec})$$

$$\frac{}{(R,I,P) \xrightarrow{\langle\emptyset\rhd R,I,P\rangle} (R,I,P)\ \mid\ P}\ (\mathit{Pro}) \qquad \frac{J\subseteq I \quad Q\subseteq R \quad J\cup Q\neq\emptyset}{(R,I,P) \xrightarrow{\langle\emptyset\rhd J,Q,\emptyset\rangle} (R,I,P)}\ (\mathit{Inh})$$

$$\frac{\mathsf{M}_1 \xrightarrow{\ell_1} \mathsf{M}_1' \quad \mathsf{M}_2 \xrightarrow{\ell_2} \mathsf{M}_2' \quad \ell_1 \frown \ell_2}{\mathsf{M}_1\ \mid\ \mathsf{M}_2 \xrightarrow{\ell_1\otimes\ell_2} \mathsf{M}_1'\ \mid\ \mathsf{M}_2'}\ (\mathit{Par}) \qquad \frac{\mathsf{M} \xrightarrow{\langle W\rhd R,I,P\rangle} \mathsf{M}' \quad R\subseteq W}{[\mathsf{M}] \xrightarrow{\langle W\rhd R,I,P\rangle} [\mathsf{M}']}\ (\mathit{Sys})$$

provides the entity $\mathsf{a}$ and then either the entity $\mathsf{b}$ or the entity $\mathsf{c}$ before stopping. The context process $\mathsf{rec}\ X.\ \mathsf{a}.\mathbf{0} + \mathsf{b}.X$ represents a nondeterministic context that, recursively, either provides the entity $\mathsf{a}$ and then halts, or the entity $\mathsf{b}$ and iterates.

The keen reader might have already noticed from the previous examples that there are different ways to represent the same concept. For example, why should we care to distinguish $[\mathsf{a} \mid (\mathsf{ab},\mathsf{c},\mathsf{b}) \mid \mathsf{b}]$ from $[(\mathsf{ab},\mathsf{c},\mathsf{b}) \mid \mathsf{ab}]$? Or $\mathsf{b}.\mathbf{0} + \mathsf{b}.\mathbf{0}$ from $\mathsf{b}.\mathbf{0}$? In fact, we prefer not to. In process algebras, this is typically achieved by defining a suitable notion of structural equivalence $\equiv$ and by taking processes up to such equivalence. Formally, we say that $\mathsf{P}$ and $\mathsf{P}'$ are structurally equivalent, written $\mathsf{P} \equiv \mathsf{P}'$, when they denote the same term up to the laws of commutative monoids (unit, associativity and commutativity) for parallel composition $\cdot \mid \cdot$, with $\emptyset$ as the unit, and the laws of idempotent and commutative monoids for choice $\cdot + \cdot$, with $\mathbf{0}$ as the unit. We also assume $D_1 \mid D_2 \equiv D_1 \cup D_2$ for any $D_1, D_2 \subseteq S$.

***Remark 5*** The processes $\emptyset$ and $\mathbf{0}$ are not interchangeable: as it will become clear from the operational semantics, the process $\emptyset$ can perform just a trivial transition to itself, while the process $\mathbf{0}$ cannot perform any transition and stops the computation.

**Definition 6** (*From RSs to RS processes*) Let $\mathcal{A} = (S, A)$ be a RS, and $\pi = (\gamma, \delta)$ an $n$-step interactive process in $\mathcal{A}$, with $\gamma = \{C_i\}_{i\in[0,n]}$ and $\delta = \{D_i\}_{i\in[0,n]}$. For any unit of time $i \in [0, n]$, the corresponding RS process $[\![\mathcal{A}, \pi]\!]_i$ is defined as follows:

$$[\![\mathcal{A},\pi]\!]_i \triangleq \left[\prod_{a\in A} a\ \mid\ D_i\ \mid\ \mathsf{K}_{\gamma^i}\right]$$

where the context process $\mathsf{K}_{\gamma^i} \triangleq C_i.C_{i+1}.\cdots.C_n.\mathbf{0}$ is the sequentialisation of the entities offered by $\gamma^i \triangleq \{C_j\}_{j\in[i,n]}$. We write $[\![\mathcal{A}, \pi]\!]$ as a shorthand for $[\![\mathcal{A}, \pi]\!]_0$.

***Example 7*** Here, we give the encoding of the reaction system $\mathcal{A}$ from Example 1. The resulting RS process is as follows:

$$\begin{aligned}\mathsf{P}\ &\triangleq [\![\mathcal{A},\pi]\!] = [\![(S,A),\pi]\!] = [\![(\{\mathsf{a},\mathsf{b},\mathsf{c}\},\{a_1\}),(\gamma,\delta)]\!]\\ &= [(\mathsf{ab},\mathsf{c},\mathsf{b})\ \mid\ \emptyset\ \mid\ \mathsf{K}_\gamma]\\ &\equiv [(\mathsf{ab},\mathsf{c},\mathsf{b})\ \mid\ \mathsf{K}_\gamma]\\ &\equiv [(\mathsf{ab},\mathsf{c},\mathsf{b})\ \mid\ \mathsf{ab}.\mathsf{a}.\mathsf{c}.\mathsf{c}.\mathbf{0}]\end{aligned}$$

Note that $D_0 = \emptyset$ is inessential and can be discarded thanks to structural congruence (because $\emptyset$ is the unit of parallel composition).

As already exemplified, our syntax allows for more general kinds of contexts than plain sequences. Nondeterministic contexts can be used to describe several alternative experimental conditions, while recursion can be exploited to extract some regularity in the longterm behaviour of a RS. Together, they can deal with a wide variety of in-breadth/in-depth behavioural analysis.

The behaviour of RS processes is defined as an LTS whose states are processes and whose transitions represent the possibility to move from one process configuration to another in a single unit of time. Transition labels are used to compose behaviours of separate components and to record some information about the entities involved in that move.

**Definition 8** (*Label*) A label is a tuple $\langle W \rhd R, I, P\rangle$ with $W, R, I, P \subseteq S$. The set of transition labels is ranged over by $\ell$.

In a transition label $\langle W \rhd R, I, P\rangle$, we record the set $W$ of entities currently in the system (produced in the previous step or provided by the context), the set $R$ of entities whose presence is assumed (either because they are needed as reactants on an applied reaction or because their presence prevents the application of some reaction); the set $I$ of entities whose absence is assumed (either because they

$$\dfrac{\dfrac{}{(\mathsf{ab},\mathsf{c},\mathsf{b}) \xrightarrow{\langle \emptyset \rhd \mathsf{ab},\mathsf{c},\mathsf{b}\rangle} (\mathsf{ab},\mathsf{c},\mathsf{b}) \mid \mathsf{b}}\,(\mathit{Pro}) \qquad \dfrac{\dfrac{\dfrac{}{\mathsf{ab}.\mathsf{K}_0 \xrightarrow{\langle \mathsf{ab} \rhd \emptyset,\emptyset,\emptyset\rangle} \mathsf{K}_0}\,(\mathit{Cxt})}{\mathsf{c}.\mathbf{0} + \mathsf{ab}.\mathsf{K}_0 \xrightarrow{\langle \mathsf{ab} \rhd \emptyset,\emptyset,\emptyset\rangle} \mathsf{K}_0}\,(\mathit{Sumr})}{\mathsf{K}_0 \xrightarrow{\langle \mathsf{ab} \rhd \emptyset,\emptyset,\emptyset\rangle} \mathsf{K}_0}\,(\mathit{Rec})}{\dfrac{(\mathsf{ab},\mathsf{c},\mathsf{b}) \mid \mathsf{K}_0 \xrightarrow{\langle \mathsf{ab} \rhd \mathsf{ab},\mathsf{c},\mathsf{b}\rangle} (\mathsf{ab},\mathsf{c},\mathsf{b}) \mid \mathsf{b} \mid \mathsf{K}_0}{[(\mathsf{ab},\mathsf{c},\mathsf{b}) \mid \mathsf{K}_0] \xrightarrow{\langle \mathsf{ab} \rhd \mathsf{ab},\mathsf{c},\mathsf{b}\rangle} [(\mathsf{ab},\mathsf{c},\mathsf{b}) \mid \mathsf{b} \mid \mathsf{K}_0]}\,(\mathit{Sys})}\,(\mathit{Par})$$

**Fig. 2** Full SOS derivation of a transition for process $\mathsf{P}_0$ (see Example 11)

$$\dfrac{\dfrac{}{(\mathsf{ab},\mathsf{c},\mathsf{b}) \xrightarrow{\langle \emptyset \rhd \mathsf{c},\emptyset,\emptyset\rangle} (\mathsf{ab},\mathsf{c},\mathsf{b})}\,(\mathit{Inh}) \qquad \dfrac{\dfrac{\dfrac{}{\mathsf{c}.\mathbf{0} \xrightarrow{\langle \mathsf{c} \rhd \emptyset,\emptyset,\emptyset\rangle} \mathbf{0}}\,(\mathit{Cxt})}{\mathsf{c}.\mathbf{0} + \mathsf{ab}.\mathsf{K}_0 \xrightarrow{\langle \mathsf{c} \rhd \emptyset,\emptyset,\emptyset\rangle} \mathbf{0}}\,(\mathit{Suml})}{\mathsf{K}_0 \xrightarrow{\langle \mathsf{c} \rhd \emptyset,\emptyset,\emptyset\rangle} \mathbf{0}}\,(\mathit{Rec})}{\dfrac{(\mathsf{ab},\mathsf{c},\mathsf{b}) \mid \mathsf{K}_0 \xrightarrow{\langle \mathsf{c} \rhd \mathsf{c},\emptyset,\emptyset\rangle} (\mathsf{ab},\mathsf{c},\mathsf{b}) \mid \mathbf{0}}{[(\mathsf{ab},\mathsf{c},\mathsf{b}) \mid \mathsf{K}_0] \xrightarrow{\langle \mathsf{c} \rhd \mathsf{c},\emptyset,\emptyset\rangle} [(\mathsf{ab},\mathsf{c},\mathsf{b}) \mid \mathsf{b} \mid \mathbf{0}]}\,(\mathit{Sys})}\,(\mathit{Par})$$

**Fig. 3** A second SOS derivation of a different transition for process $\mathsf{P}_0$ (see Example 11)

appear as inhibitors for an applied reaction or because their absence prevents the application of some reaction); and the set $P$ of products of all the applied reactions. Our LTS will be defined in such a way that any transition will carry a label $\langle W \rhd R, I, P\rangle$ such that $I$ and $W \cup R$ are disjoint, written $I \# (W \cup R)$.

As a convenient notation, we write $\ell_1 \otimes \ell_2$ for the component-wise union of labels. We also define a noninterference predicate over labels, written $\ell_1 \frown \ell_2$, that will be used to guarantee that there is no conflict between reactants and inhibitors of the reactions that take place in two separate parts of the system. Formally, we let:

$$\langle W_1 \rhd R_1, I_1, P_1\rangle \otimes \langle W_2 \rhd R_2, I_2, P_2\rangle \triangleq \langle W_1 \cup W_2 \rhd R_1 \cup R_2, I_1 \cup I_2, P_1 \cup P_2\rangle$$
$$\langle W_1 \rhd R_1, I_1, P_1\rangle \frown \langle W_2 \rhd R_2, I_2, P_2\rangle \triangleq (I_1 \cup I_2) \# (W_1 \cup W_2 \cup R_1 \cup R_2)$$

***Remark 9*** In Sect. 4.2, when we will present the extension with concentration levels, transition labels will be extended to include lower bounds on the availability of certain entities, and the operators $\otimes$ and $\frown$ will be updated accordingly.

**Definition 10** (*Operational semantics*) The operational semantics of processes is defined by the set of SOS inference rules in Fig.1.

The process $\mathbf{0}$ has no transition. The rule (*Ent*) makes available the entities in the (possibly empty) set $D$, then reduces to $\emptyset$. As a special instance of (*Ent*), we have, e.g., $\emptyset \xrightarrow{\langle \emptyset \rhd \emptyset,\emptyset,\emptyset\rangle} \emptyset$.

The rule (*Cxt*) says that a prefixed context process $C.\mathsf{K}$ makes available the entities in the set $C$ and then reduces to $\mathsf{K}$. The rule (*Rec*) is the classical rule for recursion. Here, $\mathsf{K}[^{\mathsf{rec}\, X.\, \mathsf{K}}/_{X}]$ denotes the process obtained by replacing in $\mathsf{K}$ every free occurrence of the variable $X$ with its recursive definition $\mathsf{rec}\ X.\ \mathsf{K}$. For example, we can use rule (*Rec*) to derive transitions such as $\mathsf{rec}\ X.\ \mathsf{a}.\mathsf{b}.X \xrightarrow{\langle \mathsf{a} \rhd \emptyset,\emptyset,\emptyset\rangle} \mathsf{b}.\mathsf{rec}\ X.\ \mathsf{a}.\mathsf{b}.X$. The rules (*Suml*) and (*Sumr*) select a move of either the left or the right component and discard the other process.

The rule (*Pro*) executes the reaction ($R$, $I$, $P$) (its reactants, inhibitors and products are recorded in the label), which remains available at the next step together with $P$. The rule (*Inh*) applies when the reaction ($R$, $I$, $P$) should not be executed; it records in the label the possible causes for which the reaction is disabled: possibly some inhibiting entities ($J \subseteq I$) are present or some reactants ($Q \subseteq R$) are missing, with $J \cup Q \neq \emptyset$, as at least one cause is needed for explaining why the reaction is not enabled. The rule (*Par*) puts two processes in parallel by pooling their labels and joining all the set components of the labels. The sanity check $\ell_1 \frown \ell_2$ is required to guarantee that there is no conflict between the two labels. The labels $\ell_1$ and $\ell_2$ are joined in the conclusion of $(Par)$, which carries the label $\ell_1 \otimes \ell_2$.

Finally, the rule $(Sys)$ requires that all the processes of the systems have been considered, and also checks that all the needed reactants are actually available in the system $(R \subseteq W)$. In fact, this constraint can only be met on top of all processes. The check that inhibitors are absent $(I \# W)$ is not necessary, as it is embedded in rule $(Par)$ by the premise $\ell_1 \frown \ell_2$.

***Example 11*** Let us consider the RS process $\mathsf{P}_0 \triangleq [(\mathsf{ab}, \mathsf{c}, \mathsf{b}) \mid \mathsf{rec}\ X.\ \mathsf{c}.\mathbf{0} + \mathsf{ab}.X]$ from Example 7. The process $\mathsf{P}_0$ has two outgoing transitions, depending on the entities provided by the context. The case where the context provides $\{\mathsf{a}, \mathsf{b}\}$ is detailed in Fig. 2, where we use the shorthand $\mathsf{K}_0 \triangleq \mathsf{rec}\ X.\ \mathsf{c}.\mathbf{0} + \mathsf{ab}.X$. Alternatively, the context $\mathsf{K}_0$ can provide $\{\mathsf{c}\}$, in which case we derive the transition in Fig. 3.

The following theorem (from [19]) shows that the rewrite steps of a RS exactly match the transitions of its corresponding RS process.

**Theorem 12** *Given a RS $\mathcal{A} = (S, A)$ and an n-step interactive process $\pi = (\gamma, \delta)$, with context sequence $\gamma = \{C_i\}_{i \in [0,n]}$, result sequence $\delta = \{D_i\}_{i \in [0,n]}$ and state sequence $\tau = \{W_i\}_{i \in [0,n]}$ (where, as usual, $W_i \triangleq C_i \cup D_i$ for any $i \in [0, n]$), let $\mathsf{P}_i \triangleq [\![\mathcal{A}, \pi]\!]_i$. Then, $\forall i \in [0, n-1]$:*

1. $\mathsf{P}_i \xrightarrow{\langle W \triangleright R,I,P \rangle} \mathsf{P}$ *implies $W = W_i$, $P = D_{i+1}$ and* $\mathsf{P} \equiv \mathsf{P}_{i+1}$;
2. *there exists $R, I \subseteq S$ such that* $\mathsf{P}_i \xrightarrow{\langle W_i \triangleright R,I,D_{i+1} \rangle} \mathsf{P}_{i+1}$.

## 4 Quantitative variants of reaction systems

In the following, we will introduce two different features in reaction systems.

The first extension is along the time dimension, to handle the concept of delay and durations/decay for reaction products. Instead of making the products of a reaction immediately available at the next time unit and then vanish in one step (as done in the original framework), we now allow the possibility to specify that a reaction will make available its products after a certain number of time units and that such products will not decay after just one step, but they can have a longer persistency. RS processes with delays and durations will be exploited to experiment with the first and third case studies.

The second extension adds some quantitative information for modelling concentration levels and linear constraints over them. RS processes with concentration levels will be exploited in the second and third case studies.

Both variants are obtained as simple modifications of the process algebraic framework presented in Sect. 3. For the sake of simplicity, both variations are described separately, as enhancements of the original SOS semantics, but it should be clear that they can be combined in a unique integrated framework.

### 4.1 Delays, durations and timed processes

In biology, it is well known that reactions occur with different frequencies. For example, since enzymes catalyse reactions, many reactions are more frequent when some enzymes are present, and less frequent when such enzymes are absent. Moreover, reactions describing complex transformations may require time before releasing their products. To capture these dynamical aspects in our framework by preserving the discrete and abstract nature of RS, we propose a discretisation of the *delay* between two occurrences of a reaction by using a scale of natural numbers, from 0 (smallest delay, highest frequency) up to $n$ (increasing delay, lower frequency).

Intuitively, the notation $D^n$ stands for making the entities $D$ available after $n$ time units, and we use the shorthand $D$ for $D^0$, meaning that the entities are immediately available. Similarly, we can associate a delay value with the product of each reaction by writing $(R, I, P)^n$ when the product of the reaction will be available after $n$ time units, and we write $(R, I, P)$ for $(R, I, P)^0$. The syntax for mixture processes is thus extended as below and the operational semantics is changed accordingly (see Fig. 4).

$$\mathsf{M} ::= (R, I, P)^n \mid D^n \mid \mathsf{K} \mid \mathsf{M}|\mathsf{M}$$

In Fig. 4, we only report the rules that are new and those that override the ones in Fig. 1. Note, e.g., that the semantics of context processes is unchanged. Rule $(Tick)$ represents the passing of one time unit, while rule $(Ent)$ notifies the availability of entities whose delay has expired. Rule $(Pro)$ attaches to the product of the reaction the same delay as the one of the reaction itself, while rule $(Inh)$ is used when the reaction is not enabled.

In the following, we use the name *timed processes* for processes with delays and durations. Our extension is conservative, i.e., it does not change the semantics of processes without delays and durations. Therefore, the encoding of standard RSs described in Def. 6 still applies.

**Proposition 13** *Timed processes are a conservative extension of RS processes.*

**Fig. 4** SOS semantics with delays and durations

$$\frac{}{D^0 \xrightarrow{\langle D\triangleright\emptyset,\emptyset,\emptyset\rangle} \emptyset}\ (Ent) \qquad \frac{}{D^{n+1} \xrightarrow{\langle \emptyset\triangleright\emptyset,\emptyset,\emptyset\rangle} D^n}\ (Tick)$$

$$\frac{}{(R,I,P)^n \xrightarrow{\langle \emptyset\triangleright R,I,P\rangle} (R,I,P)^n \ \mid\ P^n}\ (Pro) \qquad \frac{J\subseteq I \quad Q\subseteq R \quad J\cup Q\neq\emptyset}{(R,I,P)^n \xrightarrow{\langle \emptyset\triangleright J,Q,\emptyset\rangle} (R,I,P)^n}\ (Inh)$$

**Fig. 5** Two transition sequences of timed processes $\mathsf{P}_1$ and $\mathsf{P}_2$ (see Example 14)

$$\mathsf{P}_1 = [\mathsf{M}_1 \mid \mathsf{K}] \xrightarrow{\langle \mathsf{ac}\triangleright\mathsf{ac},\mathsf{bd},\mathsf{bd}\rangle} [\mathsf{M}_1 \mid \mathsf{b}^1 \mid \mathsf{d} \mid \emptyset.\mathbf{0}] \xrightarrow{\langle \mathsf{d}\triangleright\mathsf{d},\mathsf{abc},\mathsf{c}\rangle} [\mathsf{M}_1 \mid \mathsf{c}^1 \mid \mathsf{b} \mid \mathbf{0}]$$

$$\mathsf{P}_2 = [\mathsf{M}_2 \mid \mathsf{K}] \xrightarrow{\langle \mathsf{ac}\triangleright\mathsf{ac},\mathsf{bd},\mathsf{bd}\rangle} [\mathsf{M}_2 \mid \mathsf{b} \mid \mathsf{d}^1 \mid \emptyset.\mathbf{0}] \xrightarrow{\langle \mathsf{b}\triangleright\mathsf{b},\mathsf{acd},\mathsf{a}\rangle} [\mathsf{M}_2 \mid \mathsf{a}^1 \mid \mathsf{d} \mid \mathbf{0}]$$

***Example 14*** Let us consider two RSs sharing the same entity set $S = \{\mathsf{a},\mathsf{b},\mathsf{c},\mathsf{d}\}$ and the same reactions $a_1 = (\mathsf{a},\mathsf{b},\mathsf{b})$, $a_2 = (\mathsf{b},\mathsf{a},\mathsf{a})$, $a_3 = (\mathsf{ac},\mathsf{b},\mathsf{d})$, $a_4 = (\mathsf{d},\mathsf{a},\mathsf{c})$, but working with different reaction speeds. For simplicity, we assume only two speed levels are distinguished: 0 the fastest and 1 the slowest. The reaction system $\mathcal{A}_1$ provides the following speed assignment to its reactions: $\{a_1^1, a_2, a_3, a_4^1\}$. The reaction system $\mathcal{A}_2$ provides the following speed assignment to its reactions: $\{a_1, a_2^1, a_3^1, a_4\}$. We assume that the context process for both reaction systems is just $\mathsf{K}\triangleq\mathsf{ac}.\emptyset.\mathbf{0}$. The LTSs of their corresponding timed processes are in Fig. 5, where, for brevity we let:

$$\mathsf{M}_1\triangleq a_1^1 \mid a_2 \mid a_3 \mid a_4^1 \qquad \mathsf{M}_2\triangleq a_1 \mid a_2^1 \mid a_3^1 \mid a_4.$$

Additionally, inspired by [12], we can also provide entities with a duration, i.e., entities that last a finite number of steps. To this aim, we use the syntax $D^{[n,m]}$ to represent the availability of $D$ for $m > 0$ time units starting after $n$ time units from the current time. By assuming that each reaction only produces entities with the same duration, we can describe duration and delay also associated with reactions: $(R,I,P)^{[n,m]}$ means that all the entities in $P$ (the products) have a delay of $n$ but will last $m$ steps (once they appear in the state). While we could easily define the SOS rules for the above processes, we note that durations are just syntax sugar, because they can be encoded in timed processes as follows:

$$D^{[n,m]}\triangleq \prod_{k=n}^{n+m-1} D^k \qquad (R,I,P)^{[n,m]}\triangleq \prod_{k=n}^{n+m-1} (R,I,P)^k.$$

For example, we have $\mathsf{a}^{[2,3]} \equiv \mathsf{a}^2 \mid \mathsf{a}^3 \mid \mathsf{a}^4$ and $\mathsf{a}^{[0,1]} \equiv \mathsf{a}^0 \equiv \mathsf{a}$.

## 4.2 Concentration levels and linear processes

Quantitative modelling of chemical reaction requires taking molecule concentrations into account. An abstract representation of concentrations that is considered in many formalisms is based on *concentration levels*: rather than representing such quantities as real numbers, a finite classification is considered (e.g., *low/medium/high*) with a granularity that reflects the number of concentrations levels at which significant changes in the behaviour of the molecule are observed. In classical RSs, the modelling of concentration levels would require using different entities for the same molecule (e.g., $\mathsf{a}_\mathtt{l}$, $\mathsf{a}_\mathtt{m}$, and $\mathsf{a}_\mathtt{h}$ for low, medium and high concentration of $\mathsf{a}$, respectively). This may introduce some additional complexity due to the need of guaranteeing that only one of these entities is present at any time for the state to be consistent. Moreover, consistency would be put at risk whenever entities representing different levels of the same molecule (e.g., $\mathsf{a}_\mathtt{l}$ and $\mathsf{a}_\mathtt{h}$) could be provided by the context.

We now enhance RS process by adding some quantitative information associated with each entity of each reaction, so that levels are just natural numbers and the concentration levels of the products depend on the concentration levels of reactants. The idea is to associate linear expressions, such as $e = m\cdot x + n$ (where $m\in\mathbb{N}$ and $n\in\mathbb{N}^+$ are two constants and $x$ stands for a variable ranging over natural numbers),[2] to reactants and products of each reaction. In the following, we write $s(e)$ to state that expression $e$ is associated with entity $s$. Expressions associated with reactants are used as *patterns* to match the current levels of the entities involved in the reaction. Pattern matching allows to find the largest value for the variable $x$ (the same for all reactants) that is consistent with the current concentration levels. Then, linear expressions associated with products (that can contain, again, variable

[2] To ease the presentation, we require $n\in\mathbb{N}^+$ to guarantee that $e$ evaluates to a positive number, even when $x = 0$. Alternative choices are possible to relax this constraint.

$x$) can be evaluated to compute the concentration levels of those entities. Expressions can be associated also with reaction inhibitors in order to let such entities inhibit the reaction only when their concentration level is above a given threshold. However, we require inhibitor expressions to be *ground*; namely, they cannot contain the $m \cdot x$ term and simply correspond to a positive natural number $n$. Also the state of the system has to take into account concentration levels. Consequently, in the definition of states we will exploit again ground expressions: each entity in the state is paired with a natural number representing its concentration level.

***Example 15*** Assume that we want to write a reaction that produces $\mathsf{c}$ with a concentration level that corresponds to the current concentration level of $\mathsf{a}$ (but at least one occurrence of $\mathsf{a}$ must be present), and that requires $\mathsf{b}$ not to be present at a concentration level higher than 1. Such a reaction would be $r_1 = (R, I, P)$ where $R = \mathsf{a}(x+1)$, $I = \mathsf{b}(2)$ and $P = \mathsf{c}(x+1)$. In the state $\{\mathsf{a}(3), \mathsf{b}(1)\}$. Reaction $r_1$ is enabled by taking $x = 2$ (the maximum value for $x$ that satisfies $\mathsf{a}(x+1) \leq \mathsf{a}(3)$). Since $\mathsf{b}(1) < \mathsf{b}(2)$, entity $\mathsf{c}$ will be produced with concentration level $\mathsf{c}(x+1) = \mathsf{c}(2+1) = \mathsf{c}(3)$. On the contrary, in the state $\{\mathsf{a}(2), \mathsf{b}(2)\}$ the reaction $a$ is not enabled because the concentration of the inhibitor is too high ($\mathsf{b}(2) \not< \mathsf{b}(2)$).

To formalise the above linear constraints we introduce some notation and terminology. A *ground* linear expression is just a natural number, and $e[v/x]$ represents the *substitution* of variable $x$ with the value $v$ in $e$. A *pattern* $T = \{s_1(e_1), \ldots, s_k(e_k)\}$ is a set of associations of linear expressions to entities. We write $T(s_i)$ for the linear expression associated with $s_i$ in $T$. When $s$ is not present in $T$, we let $T(s) = 0$ by default, and we write $s \in T$ whenever $T(s) \neq 0$.

**Definition 16** (*Ground patterns*) A pattern $T = \{s_1(e_1), \ldots, s_k(e_k)\}$ is ground if $T(s_i) \in \mathbb{N}$ for any $s_i \in S$ and we write $\lfloor T \rfloor$ in this case. We denote by $T[v/x]$ the ground pattern such that $T[v/x](s_i) = e_i[v/x]$ for all $s_i \in S$.

A ground pattern $T$ is unitary if $T(s_i) \in \{0, 1\}$ for any $s_i \in S$. Given two ground patterns $T_1, T_2$, we write $T_1 \leq T_2$ if $T_1(s) \leq T_2(s)$ for all $s \in S$.

***Example 17*** Let us consider the pattern $T_1 = \{\mathsf{a}(x+1), \mathsf{b}(2x+1)\}$ and the ground pattern $T_2 = \{\mathsf{a}(3), \mathsf{b}(3), \mathsf{c}(2)\}$. We have $T_1(\mathsf{a}) = x+1$, $T_1(\mathsf{b}) = 2x+1$ and $T_1[2/x] = \{\mathsf{a}(3), \mathsf{b}(5)\}$. It is immediate to verify that $T_1[1/x] \leq T_2$, while $T_1[2/x] \not\leq T_2$.

We extend the syntax of reactions $r = (R, I, P)$ by taking $I$ as a ground pattern, and $R$ and $P$ as patterns such that if $R$ is ground then $P$ is ground. Formally, $r$ is valid iff $\lfloor I \rfloor$ and $\lfloor R \rfloor \Rightarrow \lfloor P \rfloor$.

For example, the triple $(\mathsf{a}(1), \mathsf{b}(x+1), \mathsf{c}(x+1))$ is not valid because its inhibitor pattern $\{\mathsf{b}(x+1)\}$ is not ground, and moreover, the product pattern $\{\mathsf{c}(x+1)\}$ is not ground while the reactant pattern $\{\mathsf{a}(1)\}$ is ground. Vice versa, the triple $(\mathsf{a}(x+1), \mathsf{b}(1), \mathsf{c}(x+1))$ is a valid reaction. We will see later that it makes sense to allow for reactants sets $R$ and inhibitors sets $I$ that are not disjoint (see Example 23). As a special case, when all patterns of a reaction $r = (R, I, P)$ are ground (respectively, unitary), we say $r$ is ground (respectively, unitary). Unitary reactions behave as reactions of ordinary RSs. A RS whose reactions are all ground (respectively, unitary) is also called ground (respectively, unitary).

A state $W$ is just a ground pattern. We write $I \# W$ and overload the previously used notation for denoting disjoint sets when the inhibitor pattern $I$ does not conflict with the state $W$, i.e., we let

$$I \# W \triangleq \forall s \in I.\ W(s) < I(s).$$

The definition states that whenever the entity $s$ is present in the inhibitor pattern $I$ (i.e., $I(s) > 0$), then the threshold required for $s$ to inhibit the reaction is strictly larger than the available concentration of $s$ in the current state (i.e., $W(s) < I(s)$).[3]

At each step, starting from a given state, the semantics verifies the enabled reactions using function $en()$, computes the *multiplicity* of each reaction application (the value of $x$ obtained by matching the current state $W$ against the pattern $R$) by function $mul()$, and computes the resulting state by function $res()$. Formally, given a reaction $a = (R, I, P)$ and a state $W$, we define:

- the function $en(a, W)$ returns 1 if the reaction is enabled, 0 otherwise

$$en(a, W) \triangleq \begin{cases} 1 & \text{if } R[0/x] \leq W \text{ and } I \# W \\ 0 & \text{otherwise} \end{cases}$$

- the function $mul(a, W)$ returns the value $v$ that will correctly bind $x$ when applied to state $W$

$$mul(a, W) \triangleq \begin{cases} 1 & \text{if } en(a, W) = 0 \text{ or } \lfloor R \rfloor \\ \max\{v \in \mathbb{N} \mid R[v/x] \leq W\} & \text{otherwise} \end{cases}$$

- the function $res(a, W)$ returns the product of the reaction $a$ on state $W$

$$res(a, W) \triangleq en(a, W) \cdot P[mul(a, W)/x]$$

[3] Note that the predicate $I \# W$ is no longer commutative: the first argument represents the inhibitors, while the second argument the available entities.

***Example 18*** Consider again the previous example, $r_1 = (R, I, P)$ with $R = \mathsf{a}(x+1)$, $I = \mathsf{b}(2)$ and $P = \mathsf{c}(x+1)$ and the state $W = \{\mathsf{a}(3), \mathsf{b}(1)\}$, we compute:

- $en(r_1, W) = 1$, as $R[0/x] = \mathsf{a}(1) \le \mathsf{a}(3)$ and $W(\mathsf{b}) = 1 < 2 = I(\mathsf{b})$.
- $mul(r_1, W) = 2$, as $\max\{x \in \mathbb{N} \mid R[v/x] = \mathsf{a}(v+1) \le \mathsf{a}(3)\mathsf{b}(1) = W\} = 2$.
- $res(r_1, W) = en(r_1, W) \cdot P[mul(r_1, W)/x] = 1 \cdot \mathsf{c}(2+1) = \mathsf{c}(3)$.

Once the product of each enabled reaction has been calculated, we need to compute the next state. We consider the operator $\sqcup$ that computes the maximum between two ground patterns by computing the point-wise maximum value of each entity. Analogously, to combine inhibitor constraints, we consider the operator $\sqcap$ that computes the minimum between two ground patterns. The two operators are defined as follows:[4]

$$(T_1 \sqcup T_2)(s) \triangleq \max\{T_1(s), T_2(s)\}$$

$$(T_1 \sqcap T_2)(s) \triangleq \begin{cases} T_1(s) & \text{if} \quad T_2(s) = 0 \\ T_2(s) & \text{if} \quad T_1(s) = 0 \\ \min\{T_1(s), T_2(s)\} & \text{otherwise} \end{cases}$$

***Example 19*** Assume we add a new reaction $r_2 = (R', I', P')$ to the previous example, where $R' = \mathsf{a}(x+2)\mathsf{b}(1)$, $I' = \emptyset$, $P' = \mathsf{c}(3x+2)$. The reaction $r_2$ is enabled in the state $W = \{\mathsf{a}(3), \mathsf{b}(1)\}$ and it produces

$$\begin{aligned} res(r_2, W) &= en(r_2, W) \cdot P'[mul(r_2, W)/x] \\ &= 1 \cdot \mathsf{c}(3x+2)[1/x] = \mathsf{c}(5). \end{aligned}$$

Since we already observed that $res(r_1, W) = \mathsf{c}(3)$, we conclude that the reactions $r_1$ and $r_2$ in the state $W$ produce the entities $\mathsf{c}(3) \sqcup \mathsf{c}(5) = \mathsf{c}(5)$.

In the SOS style, the hypotheses under which a reaction is applied or inhibited are recorded in the label and their consistency is verified by rule (*Par*) and (*Sys*). We stretch here the fact that such hypotheses consist of constraints over concentration levels. If we assume that a reaction $a = (R, I, P)$ is enabled with multiplicity $v$, it means that it must be $\forall s \in I.\ W(s) < I(s)$ and $\forall s \in S.\ R[v/x](s) \le W(s)$ but also that $R[v+1/x] \not\le W$. The first two constraints can be already represented in the ordinary labels. Instead, we note that the property $R[v+1/x] \not\le W$ is quantified existentially over the entities, i.e., it is equivalent to $\exists s \in S.\ R[v+1/x](s) > W(s)$. Thus, in general, different constraints $R_1 \not\le W$ and $R_2 \not\le W$ due to the applications of different reactions cannot be combined in a single expression of the form $R \not\le W$. To account for such constraints, we need to extend labels with a set of bounds $B = \{R_1, ..., R_n\}$ for which we shall require that in the current state $W$ we have $\forall i \in [1, n].\ R_i \not\le W$. To this aim, for $B = \{R_1, ..., R_n\}$ and $\ell = \langle W \triangleright R, I, P \rangle$, we write $B \not\le \ell$ iff $\forall i \in [1, n].\ R_i \not\le W$.

**Definition 20** (*Bounded Labels*) A bound is a set $B = \{R_1, ..., R_n\}$ of ground patterns. A bounded label is a pair $B@\ell$, where $B = \{R_1, ..., R_n\}$ is a set of bounds and $\ell = \langle W \triangleright R, I, P \rangle$ is an ordinary label. As a special case, we abbreviate $\emptyset@\ell$ as $\ell$.

Our LTS will be defined in such a way that any transition will carry a bounded label $B@\langle W \triangleright R, I, P \rangle$ such that $I \# (W \sqcup R)$ and $B \not\le \ell$.

To define the bound related to the application of a reaction when the rule (*Pro*) is applied, we define the function $bnd()$ as follows:

$$bnd(R, v) \triangleq \begin{cases} \emptyset & \text{if } \lfloor R \rfloor \\ \{R[v+1/x]\} & \text{otherwise} \end{cases}$$

To handle the presence of bounds, we update the operation $\otimes$ and $\frown$, to combine and to compare extended labels, as follows:

$$\begin{aligned} &(B_1@\ell_1) \otimes (B_2@\ell_2) \triangleq (B_1 \cup B_2)@(\ell_1 \otimes \ell_2) \\ &\langle W_1 \triangleright R_1, I_1, P_1 \rangle \otimes \langle W_2 \triangleright R_2, I_2, P_2 \rangle \\ &\quad \triangleq \langle W_1 \sqcup W_2 \triangleright R_1 \sqcup R_2, I_1 \sqcap I_2, P_1 \sqcup P_2 \rangle \\ &(B_1@\ell_1) \frown (B_2@\ell_2) \\ &\quad \triangleq \ell_1 \frown \ell_2 \ \wedge \ (B_1 \cup B_2) \not\le (\ell_1 \otimes \ell_2) \\ &\langle W_1 \triangleright R_1, I_1, P_1 \rangle \frown \langle W_2 \triangleright R_2, I_2, P_2 \rangle \\ &\quad \triangleq (I_1 \sqcap I_2) \# (W_1 \sqcup W_2 \sqcup R_1 \sqcup R_2) \end{aligned}$$

Apparently, the syntax for *linear* processes is the same as the ordinary one presented in Sect. 3.

$$\begin{aligned} \mathsf{M} &::= (R, I, P) \mid D \mid \mathsf{K} \mid \mathsf{M}|\mathsf{M} \\ \mathsf{K} &::= \mathbf{0} \mid X \mid C.\mathsf{K} \mid \mathsf{K} + \mathsf{K} \mid \mathsf{rec}\ X.\ \mathsf{K} \end{aligned}$$

The difference is that now $C$ and $D$ are ground patterns, and in any reaction $(R, I, P)$, we require that both $\lfloor I \rfloor$ and $\lfloor R \rfloor \Rightarrow \lfloor P \rfloor$ hold. The operational semantics for linear

$$\frac{v \in \mathbb{N} \quad R_v = R[v/x] \quad P_v = P[v/x] \quad B_v = bnd(R, v)}{(R, I, P) \xrightarrow{B_v@\langle \emptyset \triangleright R_v, I, P_v \rangle} (R, I, P) \mid P_v} \ (Pro)$$

$$\frac{J \le I \quad Q \le R[0/x] \quad J \sqcup Q \ne \emptyset}{(R, I, P) \xrightarrow{\emptyset@\langle \emptyset \triangleright J, Q, \emptyset \rangle} (R, I, P)} \ (Inh)$$

$$\frac{\mathsf{M}_1 \xrightarrow{B_1@\ell_1} \mathsf{M}_1' \quad \mathsf{M}_2 \xrightarrow{B_2@\ell_2} \mathsf{M}_2' \quad (B_1@\ell_1) \frown (B_2@\ell_2)}{\mathsf{M}_1 \mid \mathsf{M}_2 \xrightarrow{(B_1@\ell_1) \otimes (B_2@\ell_2)} \mathsf{M}_1' \mid \mathsf{M}_2'} \ (Par)$$

$$\frac{\mathsf{M} \xrightarrow{B@\langle W \triangleright R, I, P \rangle} \mathsf{M}' \quad R \le W}{[\mathsf{M}] \xrightarrow{B@\langle W \triangleright R, I, P \rangle} [\mathsf{M}']} \ (Sys)$$

**Fig. 6** SOS semantics for concentration levels

[4] We recall that $T(s) = 0$ means that $s$ is not mentioned in $T$.

processes is changed accordingly (see Fig. 6, where we only report the rules that are modified).

A linear process is ground if it contains only ground reactions. It is immediate to observe that if the reaction $(R, I, P)$ is ground, any application of rule $(Pro)$ will produce a label of the form $\emptyset@\ell$, because $bnd(R, v) = \emptyset$ for any $v$ when $\lfloor R \rfloor$. Since nonempty bounds $B$ can only be produced by rule $(Pro)$, it follows that any transition of a ground RS process will also have the form $\emptyset@\ell$, i.e., only ordinary labels are generated by ground linear processes.

A (ground) linear process is unitary if it contains only unitary patterns. It is not difficult to see that unitary processes behave as the ordinary RS processes in Sect. 3. Hence, likewise timed processes, we have the following result.

**Proposition 21** *Linear processes are a conservative extension of RS processes.*

***Example 22*** The unary process $[(\mathsf{a}(1)\mathsf{b}(1), \mathsf{c}(1), \mathsf{b}(1)) \mid \mathsf{rec}\ X.\ \mathsf{c}(1).\mathbf{0} + \mathsf{a}(1)\mathsf{b}(1).X]$ corresponds to the RS process $\mathsf{P}_0 \triangleq [(\mathsf{ab}, \mathsf{c}, \mathsf{b}) \mid \mathsf{rec}\ X.\ \mathsf{c}.\mathbf{0} + \mathsf{ab}.X]$ from Example 11.

***Example 23*** Let us consider the linear process

$$\mathsf{P}_1 \triangleq [(\mathsf{a}(x+1), \mathsf{a}(4), \mathsf{a}(x+2)) \mid \mathsf{K}]$$

where $\mathsf{K} \triangleq \mathsf{rec}\ X.\ \mathsf{a}(3).X$ We remark that the reaction $a = (\mathsf{a}(x+1), \mathsf{a}(4), \mathsf{a}(x+2))$ contained in $\mathsf{P}_1$ has nondisjoint reactants and inhibitors. This makes sense because the inhibitor pattern $\mathsf{a}(4)$ can be used to fix a boundary on the maximum concentration level of $\mathsf{a}$ where the reaction is still enabled. Letting $R = \mathsf{a}(x+1)$, $I = \mathsf{a}(4)$, $P = \mathsf{a}(x+2)$ and $W = \mathsf{a}(3)$, we have:

$$R[0/x] = \mathsf{a}(1) \le \mathsf{a}(3) = W \qquad R[2/x] = \mathsf{a}(3) \le \mathsf{a}(3) = W$$
$$R[1/x] = \mathsf{a}(2) \le \mathsf{a}(3) = W \qquad R[3/x] = \mathsf{a}(4) \not\le \mathsf{a}(3) = W$$

Moreover $I \# W$ holds, because the condition $\forall s \in I.\ W(s) < I(s)$ amounts to $W(\mathsf{a}) = 3 < 4 = I(\mathsf{a})$. Therefore, $en(a, W) = 1$, because $R[0/x] \le W$ and $I \# W$, $mul(a, W) = \max\{v \in \mathbb{N} \mid R[v/x] \le W\} = 2$ and

$$res(a, W) = en(a, W) \cdot P[mul(a, W)/x] = P[2/x] = \mathsf{a}(4).$$

Consequently, we can derive the transition $\mathsf{P}_1 \xrightarrow{B@\ell} \mathsf{P}_2$ as shown in Fig. 7. Note that, at the next time unit, the reaction $a$ will not be enabled, because for $W' = \mathsf{a}(4) \sqcup \mathsf{a}(3) = \mathsf{a}(4)$ we have that $I \# W'$ is false (the condition $\forall s \in I.\ W'(s) < I(s)$ amounts to $W'(\mathsf{a}) = 4 \not< 4 = I(\mathsf{a})$). Thus, by composing in parallel three transitions (derived using rule (Inh), (Ent) and (Rec), respectively).

$$\frac{}{a \xrightarrow{\emptyset@\langle \emptyset \rhd \mathsf{a}(4), \emptyset, \emptyset \rangle} a}\ (Inh) \qquad \frac{}{\mathsf{a}(4) \xrightarrow{\emptyset@\langle \mathsf{a}(4) \rhd \emptyset, \emptyset, \emptyset \rangle} \emptyset}\ (Ent)$$

$$\frac{\dots}{\mathsf{K} \xrightarrow{\emptyset@\langle \mathsf{a}(3) \rhd \emptyset, \emptyset, \emptyset \rangle} \mathsf{K}}\ (Rec)$$

Since $\mathsf{a}(3) \sqcup \mathsf{a}(4) = \mathsf{a}(4)$, we can conclude that $\mathsf{P}_2 \xrightarrow{\emptyset@\langle \mathsf{a}(4) \rhd \mathsf{a}(4), \emptyset, \emptyset \rangle} \mathsf{P}_1$.

## 5 Related work

The model of RSs is qualitative as there is no direct representation of the number of molecules involved in biochemical reactions as well as of rate parameters influencing the frequency of reactions. In [13], the authors introduce an extension with discrete concentrations allowing for quantitative modelling. They demonstrate that although RSs with discrete concentrations are semantically equivalent to the original qualitative RSs, they provide much more succinct representations in terms of the number of molecules being used. They then define the problem of reachability for RSs with discrete concentrations and provide its suitable encoding in satisfiability modulo theory, together with a verification method (bounded model checking) for reachability properties. Experimental results show that verifying RSs with discrete concentrations instead of the corresponding basic RS is more efficient.

A crucial feature of a RS is that (unless introduced from outside the system) an entity from the current state will belong also to the next state only if it is in the product set of an enabled reaction. In other words, an entity vanishes unless it is sustained by a reaction. In [12], it is introduced an extension where such a property is mitigated; indeed, they provide each entity $x$ with a duration $d(x)$, which guarantees that $x$ will last through at least $d(x)$ consecutive states. The authors demonstrate that duration/decay is a result of an interaction with a 'structured environment', and they also investigate fundamental properties of state sequences of reaction systems with duration.

Each of the above enhancements of the RS framework requires complex changes in the syntax and semantics of the original framework, and they cannot be easily combined together. Our formal framework for RSs is more flexible, since it allows us to define extensions by simply playing with the defined SOS rules. We have shown this

**Fig. 7** A transition for the linear process $\mathsf{P}_1$ from Example 23

$$\dfrac{\dfrac{}{a \xrightarrow{\mathsf{a}(4)@\langle\emptyset\rhd\mathsf{a}(3),\mathsf{a}(4),\mathsf{a}(4)\rangle} a \mid \mathsf{a}(4)}\,(\mathit{Pro}) \qquad \dfrac{\dfrac{}{\mathsf{a}(3).\mathsf{K} \xrightarrow{\emptyset@\langle\mathsf{a}(3)\rhd\emptyset,\emptyset,\emptyset\rangle} \mathsf{K}}\,(\mathit{Cxt})}{\mathsf{K} \xrightarrow{\emptyset@\langle\mathsf{a}(3)\rhd\emptyset,\emptyset,\emptyset\rangle} \mathsf{K}}\,(\mathit{Rec})}{\dfrac{a \mid \mathsf{K} \xrightarrow{\mathsf{a}(4)@\langle\mathsf{a}(3)\rhd\mathsf{a}(3),\mathsf{a}(4),\mathsf{a}(4)\rangle} a \mid \mathsf{a}(4) \mid \mathsf{K}}{\mathsf{P}_1 = [a \mid \mathsf{K}] \xrightarrow{\mathsf{a}(4)@\langle\mathsf{a}(3)\rhd\mathsf{a}(3),\mathsf{a}(4),\mathsf{a}(4)\rangle} [a \mid \mathsf{a}(4) \mid \mathsf{K}] = \mathsf{P}_2}\,(\mathit{Sys})}\,(\mathit{Par})$$

possibility by defining extensions with reaction delays and durations in Sect. 4.1, and with concentration levels in Sect. 4.2. Also adapting our prototype tool for RS execution as we discuss in Sect. 6 is made easier by the SOS formalisation. It is worth noting that these and other extensions can be combined and integrated in our framework by following the same approach. There are several approaches using process algebras for modelling biological systems which are based on SOS formalisations (cf. the survey [28]), but we are the first to combine the expressiveness and flexibility of process algebras and RSs.

There are some similarities between our mechanism for delays and lazy transition systems introduced in [29] for modelling asynchronous circuits. Indeed the work [29] introduces a methodology to optimise asynchronous circuits by making the assumption that a gate introduces a noninstantaneous delay and that two gate delays have always a bigger delay than a single gate. This allows to determine whether an event in the graph of the states happens before another one, or at the same time. In order to model this behaviour, lazy transition systems distinguish between the enabling and the firing of an event in a transition system. This looks similar to the delay that we impose on some entities in our framework. The methodology in [29] allows to show that some states (due to precedence between events) can never be reached and the state graph can be optimised. We believe that optimisation of asynchronous circuits could be an interesting challenge for applying our framework. We also note that the work in [10] shows a tight relationship of reaction systems and synchronous circuits, while our extension with delay and duration might open the way to show a relationship of our extension with asynchronous circuits.

## 6 The tool

A preliminary implementation of RSs in a logic programming language (Prolog) was already presented in [19] where the intended aim was rapid prototyping. In this paper, we enrich such implementation by introducing delays/durations and the concept of concentration levels in RSs as they are formally defined in the previous sections. In particular, we will describe how such extensions have been integrated in the prototype resulting in a new tool available for download.[5] Thanks to the modular nature of the SOS formalisation, the integration of new features into the existing tool is facilitated. The use of a declarative programming language reduces the distance between the implementation and the mathematical specification given in Sects. 3–4 in a significant way, which is important to reduce the presence of bugs in the tool and thus offers a convenient tradeoff between efficiency and correctness.

Our interpreter allows the combined use of delay and duration with linear patterns in RS specifications and exploits DCG (Declarative Clause Grammars) rules to offer a friendly syntax to users. Internally, quantitative entities are encoded in two ways as either the term `e(Entity,Delay,Level)` or the term `e(Entity,Delay,M,N)`, where the parameters `M` and `N` are the coefficients of a linear expression $\mathtt{M}x + \mathtt{N}$. The first format is used for ground instances, while the second for linear patterns. For efficiency reasons, sets of quantitative entities are implemented as ordered lists and RS processes are represented as tuples of the form `sys(Delta,Es,Ks,Rs)` where `Delta` is the environment that collects all recursive definitions exploited in contextual processes, `Es` is the set of currently available entities, `Ks` represents the parallel composition of all contextual processes and `Rs` represents the parallel composition of all reactions.

To experiment with the tool, the user can write a separate specification file, say, e.g., `myspec.pl`, and then change the directive for importing the specification in the main file of our tool to something like `:- [<path>/myspec.pl"].` where `\verb|<path>|` is the global path of `myspec.pl` in her file system. The specification file requires the definition of four predicates, one for each of the components in `sys(Delta, Es,Ks,Rs)`. All predicates expect a single string.

To briefly account for the syntax defined by our DCG rules, a sample specification is shown in Fig. 8, together with usage instructions. Roughly, an entity `a` has default delay 0 and level 1, when we write `a(2)` it means `a` has delay 2 but still level 1 and when we write `a(1,2)` we declare that `a` has delay 1 and level 2. For nonground

[5] http://www.di.unipi.it/~bruni/LTSRS/.

```
/* environment */
myDeltaQ('[ x1 = ({a}.nil + {b(0,2)}.x2) ,
            x2 = {c,d}.x1
          ]').

/* initial entities, if any */
myEQ('{ a(1) , c , b(2,3) }').

/* context processes */
myKsQ('[ x1 , {a(0,3)}.{c}.nil ]').

/* reactions (the keyword react can be omitted)*/
myRsQ('[      ( {a(0,2x+1)}       , {b(0,3)}    , {c(x+10)} ),
         react( {a(x+1),c,b(0,2)} , {e(0,3),d} , {f,f(1),f(4),g(3)} )
       ]').

/* to check that your whole specification is correctly parsed try
?- mysystemQ(S).

to check single parts of the specification try one of the following
?- myparsedenvironmentQ(DeltaQ).
?- myparsedentitiesQ(EQ).
?- myparsedcontextQ(KsQ).
?- myparsedreactionsQ(RsQ).

to save the LTS on the file system in .dot format, run:
?- main_do_Q(digraph).

to save a single maximal trace on the file system in .txt format, run:
?- main_do_Q(run).

to save a single maximal trace on the file system as a .csv file, run:
?- main_do_Q(csv).
*/
```

$$( \mathsf{a}(2x+1)\ ,\ \mathsf{b}(3)\ ,\ \mathsf{c}(x+10)\ )\ \mid\ (\ \mathsf{a}(x+1)\ \mathsf{c}\ \mathsf{b}(2)\ ,\ \mathsf{e}(3)\ \mathsf{d}\ ,\ \mathsf{f}\ \mathsf{f}^1\ \mathsf{f}^4\ \mathsf{g}^3\ )\ \mid$$
$$\mathsf{a}^1\ \mid\ \mathsf{c}\ \mid\ \mathsf{b}(3)^2\ \mid\ \mathsf{rec}\ x_1.(\mathsf{a.nil} + \mathsf{b}(2).\mathsf{rec}\ x_2.\mathsf{c}\ \mathsf{d}.x_1)\mid \mathsf{a}(3).\mathsf{c.nil}$$

**Fig. 8** A sample RS specification and its process like syntax

patterns, we use either the syntax `a(2x+1)` with implicit delay 0, or `a(1,2x+1)` when an explicit delay must be considered. We believe the rest of the syntax is self-explanatory. Note that, in the product set of the same reaction, we can specify different delays for each entity, as well as multiple delays for the same entity, even noncontiguous ones. When writing RS specifications, this adds a little more flexibility w.r.t. to assigning the same delay and duration to a whole reaction. (Otherwise, we would need to repeat different instances of reactions with the same reactants and inhibitors but different quantitative product sets.)

In the following three sections, we will exploit our tool to study three models of biological systems, in which the new features of reaction systems that we have introduced in this paper can play an important role. In more details the first biological system need timed reactions to be faithfully modelled, while the second one requires the ability to handle different levels of concentrations, which is accomplished by linear reactions. Finally, both features are used to simulate neural transmission as the third case study.

All figures representing the LTS of the case studies have been generated using the primitive `main_do_Q(digraph)`. There are many available options to define different colours of nodes based on their textual descriptions and to select which information is shown. Our default choice is to print the entities provided by the context as transition labels and just the entities currently present in the system inside the nodes of the LTS. Whenever nonrecursive contexts are considered, a single maximal run can be generated using the primitive `main_do_Q(run)`. The neuron activity figures in Sect. 9 have been obtained by generating automatically the `csv` description of the concentration levels of all entities in the states of a run through the directive `main_do_Q(csv)` and then importing such raw data in a spreadsheet.

## 7 Drug administration in tumour growth

The case study presented in this section is concerned with a delay differential equation model of tumour growth as proposed in [20]. We first will model the system using timed processes and then will execute some simulations to compare different drug administration strategies.

### 7.1 Biological phenomenon

The cell cycle is a series of sequential events leading to cell duplication. It consists of four phases: $G_1$, S, $G_2$ and M. The $G_1$ phase is a resting phase (or gap period) called presynthetic phase. $G_1$ could last as long as 48 h and is the longest phase of the cycle. The next phase is the *S* phase or synthetic period, where the replication of DNA occurs. This phase may last between 8 and 20 h. The cells complete the DNA replication and enter another gap period $G_2$ called the postsynthetic phase. $G_2$ is a preparation phase for mitosis. The first three phases ($G_1$, S and $G_2$) are called interphase (I). The last phase is mitosis *M* in which the cells segregate the duplicated sets of chromosomes between daughter cells. Mitosis is the shortest phase of all, lasting up to 1 h. The duration of the cell cycle is very much dependent on the type of cell and their growth conditions. The most typical (human) normal cell will have a cell cycle duration of approximately 24 h, with various exceptions (e.g., liver cells can take up to a year to complete their cycle).

There are many checkpoints throughout the cell cycle that prevent the cell from completing the cycle if it detects an abnormality. A cancerous cell does not necessarily divide more rapidly than their normal counterparts, but they lose the ability to regulate the cell cycle, thus proliferation of these cells is not controlled. Once mitosis is completed, each daughter cell can enter the cycle again or shift into a quiescent phase during which cells do not divide for long periods. Phase-specific drugs alter the natural course of action for the active or cycling cells. Many chemotherapeutic agents acting on the *S* phase aim to suppress mitosis and therefore have no visible effect until the M phase.

In [20], a delay differential equation model of tumour growth has been proposed that includes the immune system response and a phase-specific drug able to alter the natural course of action of the cell cycle of the tumour cells. A delay is used to model the duration of the interphase.

### 7.2 On encoding drug effects on cell cycles using timed processes

Inspired from [20], we define a RS model of tumour growth using delays and durations. We consider two populations of tumour cells: those in the interphase of the cell cycle ($\mathsf{T_I}$) and those in mitosis phase ($\mathsf{T_M}$). We assume that cells reside in the interphase for $\sigma$ time units. Moreover, we represent the drug with entity $\mathsf{D}$ and assume that, once received from the environment, it takes an active form $\mathsf{D_a}$ and disappears after a delay of $\delta$ time units. The reactions of the model are the following:

$$\mathsf{a}_1 \triangleq (\mathsf{T_I}, \mathsf{D_a}, \mathsf{T_M})^{\sigma} \qquad \mathsf{a}_2 \triangleq (\mathsf{T_M}, \emptyset, \mathsf{T_I}) \qquad \mathsf{a}_3 \triangleq (\mathsf{D}, \emptyset, \mathsf{D_a})^{[0,\delta]}.$$

Let us assume that the system starts from a configuration in which tumour cells are in the interphase. Hence, the corresponding timed process is

$$\mathsf{P} \triangleq [\mathsf{K} \mid \mathsf{T_I} \mid \mathsf{A}]$$

where $\mathsf{A} \triangleq \mathsf{a}_1 \mid \mathsf{a}_2 \mid \mathsf{a}_3$ and $\mathsf{K}$ is a suitable context process. Now, different drug administration strategies can be simulated by providing different definitions for $\mathsf{K}$. As a first experiment, let us consider $\sigma = 1$ and $\delta = 2$ and two different context processes $\mathsf{K_0} \triangleq \mathsf{rec}\ X.\,\emptyset.X$ and $\mathsf{K_3} \triangleq \mathsf{rec}\ X.\,\mathsf{D}.\emptyset.\emptyset.\emptyset.X$.

The context $\mathsf{K_0}$ describes the case when no drug is administered; in this case, tumour cells execute the cell cycle infinitely:

$$\begin{aligned}\mathsf{P}[{}^{\mathsf{K_0}}/_{\mathsf{K}}] = & [\mathsf{K_0} \mid \mathsf{T_I} \mid \mathsf{A}] \xrightarrow{\langle \mathsf{T_I} \rhd \mathsf{T_I}, \mathsf{D_a T_M D}, \mathsf{T_M} \rangle} [\mathsf{K_0} \mid \mathsf{T_M^1} \mid \mathsf{A}] \xrightarrow{\langle \emptyset \rhd \emptyset, \mathsf{T_I T_M D}, \emptyset \rangle} \\ & [\mathsf{K_0} \mid \mathsf{T_M} \mid \mathsf{A}] \xrightarrow{\langle \mathsf{T_M} \rhd \mathsf{T_M}, \mathsf{T_I} \& \mathsf{D}, \mathsf{T_I} \rangle} [\mathsf{K_0} \mid \mathsf{T_I} \mid \mathsf{A}] \xrightarrow{\langle \mathsf{T_I} \rhd \mathsf{T_I}, \mathsf{D_a T_M D}, \mathsf{T_M} \rangle} \ldots\end{aligned}$$

The context $\mathsf{K_3}$ describes the case when the drug is administered every 4 time units. Here we observe that the cell cycle is interrupted after two interphase–mitosis phases and the cell dies (note that last state cannot evolve in a state describing the cell at any phase):

$$\begin{aligned}\mathsf{P}[{}^{\mathsf{K_3}}/_{\mathsf{K}}] = & [\mathsf{K_3} \mid \mathsf{T_I} \mid \mathsf{A}] \xrightarrow{\langle \mathsf{T_I D} \rhd \mathsf{T_I D}, \mathsf{D_a T_M}, \mathsf{T_M D_a} \rangle} \\ & [\emptyset.\emptyset.\emptyset.\mathsf{K_3} \mid \mathsf{D_a^1} \mid \mathsf{D_a} \mid \mathsf{T_M^1} \mid \mathsf{A}] \xrightarrow{\langle \mathsf{D_a} \rhd \mathsf{D_a}, \mathsf{T_I T_M D}, \emptyset \rangle} \\ & [\emptyset.\emptyset.\mathsf{K_3} \mid \mathsf{D_a} \mid \mathsf{T_M} \mid \mathsf{A}] \xrightarrow{\langle \mathsf{D_a T_M} \rhd \mathsf{D_a T_M}, \mathsf{T_I D}, \mathsf{T_I} \rangle} \\ & [\emptyset.\mathsf{K_3} \mid \mathsf{T_I} \mid \mathsf{A}] \xrightarrow{\langle \mathsf{T_I} \rhd \mathsf{T_I}, \mathsf{D_a T_M D}, \mathsf{T_M} \rangle} \\ & [\mathsf{K_3} \mid \mathsf{T_M^1} \mid \mathsf{A}] \xrightarrow{\langle \mathsf{D} \rhd \mathsf{D}, \mathsf{T_I T_M}, \mathsf{D_a} \rangle} \\ & [\emptyset.\emptyset.\emptyset.\mathsf{K_3} \mid \mathsf{D_a^1} \mid \mathsf{D_a} \mid \mathsf{T_M} \mid \mathsf{A}] \xrightarrow{\langle \mathsf{T_M D_a} \rhd \mathsf{D_a T_M}, \mathsf{T_I D}, \mathsf{T_I} \rangle} \\ & [\emptyset.\emptyset.\mathsf{K_3} \mid \mathsf{D_a} \mid \mathsf{T_I} \mid \mathsf{A}] \xrightarrow{\langle \mathsf{D_a T_I} \rhd \mathsf{D_a}, \mathsf{T_M D}, \emptyset \rangle} [\emptyset.\mathsf{K_3} \mid \mathsf{A}] \ldots\end{aligned}$$

We have performed several experiments dealing with different drug administration strategies (including the two that we have just discussed) using our tool where the reaction system $a_1, a_2, a_3$ is run in parallel with $\mathsf{T_I}$ and with

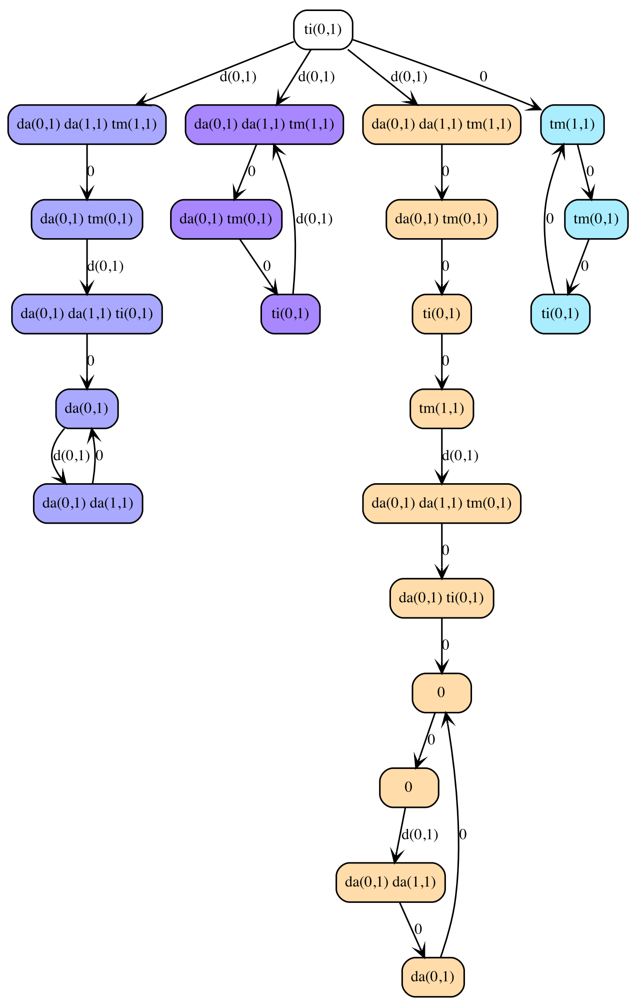


**Fig. 9** Different drug administration strategies with delay $\sigma = 1$ and duration $\delta = 2$

nondeterministic context $\sum_{i\in[0,3]} \mathsf{K}_i$, where $\mathsf{K}_1 = \mathsf{rec}\ X.\,\mathsf{D}.\emptyset.X$ and $\mathsf{K}_2 = \mathsf{rec}\ X.\,\mathsf{D}.\emptyset.\emptyset.X$. Each branch of the tree of Fig. 9 depicts the evolution of the system driven by a different context: from left to right, we see the effect of $\mathsf{K}_1$, then $\mathsf{K}_2$, followed by $\mathsf{K}_3$ and finally $\mathsf{K}_0$. To improve readability, each transition is labelled with just the entities provided by the context (a transition labelled with 0 indicates that no entity is provided by the context at that step) and the states are labelled using just the available entities using the format `Entity(Delay,Level)` (context processes and reactions are hidden). For example, `da(0,1) da(1,1) tm(1,1)` stands for $\mathsf{Da} \mid \mathsf{Da}^1 \mid \mathsf{T_M}^1$. A state labelled with 0 indicates that no entity is present in such a state.

As expected, in the evolution with the context $\mathsf{K}_3$ the cell dies after performing two cycles of interphase and mitosis (observe that in the final cycle only the drug remains in circle). More interestingly, a drug strategy that gives the drug as described by context $\mathsf{K}_2$ (one drug administration followed by two consecutive pauses) does not lead to the death of the cell. Finally, when drug is administrated as described by context $\mathsf{K}_1$ the result is the death of the cell after just one complete cycle of interphase and mitosis.

The parameter $\delta$ can be varied to test alternative drug activation times. Of course, if the drug remains in circulation for a longer time its effects will be stronger. For example, assuming a duration $\delta = 3$ the experiment with the context $\mathsf{K}_3$ describes the case in which the drug remains active between two consecutive administrations, and this leads the cell to die at an earlier stage (after just one cycle):

$$\mathsf{P}[{}^{\mathsf{K}_3}/_{\mathsf{K}}] = [\mathsf{K}_3 \mid \mathsf{T_I} \mid \mathsf{A}] \xrightarrow{\langle \mathsf{T_I D} \rhd \mathsf{T_I D}, \mathsf{D_a T_M}, \mathsf{T_M D_a} \rangle}$$
$$[\emptyset.\emptyset.\emptyset.\mathsf{K}_3 \mid \mathsf{D_a}^{[0,3]} \mid \mathsf{T_M^1} \mid \mathsf{A}] \xrightarrow{\langle \mathsf{D_a} \rhd \mathsf{D_a}, \mathsf{T_I T_M D}, \emptyset \rangle}$$
$$[\emptyset.\emptyset.\mathsf{K}_3 \mid \mathsf{D_a}^{[0,2]} \mid \mathsf{T_M} \mid \mathsf{A}] \xrightarrow{\langle \mathsf{D_a T_M} \rhd \mathsf{D_a T_M}, \mathsf{T_I D}, \mathsf{T_I} \rangle}$$
$$[\emptyset.\mathsf{K}_3 \mid \mathsf{D_a}^{[0,1]} \mid \mathsf{T_I} \mid \mathsf{A}] \xrightarrow{\langle \mathsf{D_a T_I} \rhd \mathsf{D_a}, \mathsf{T_M D}, \emptyset \rangle}$$
$$[\mathsf{K}_3 \mid \mathsf{A}] \ldots$$

Repeating also the other set of experiments with all the other different administration strategies represented by nondeterministic context $\sum_{i\in[0,3]} \mathsf{K}_i$, we depict the results in Fig. 10. They show how the effect of the drug is much stronger when $\delta = 3$: as expected, both drug administration strategies $\mathsf{K}_1$ and $\mathsf{K}_3$ stop the cell cycle but this time after just one cycle of interphase and mitosis. Moreover, now even the strategy $\mathsf{K}_2$ leads to the death of the cell, as desired.

Finally, one may wonder which are the effects of increasing the value for $\sigma$ that represents a slower mitosis phase. The result of such experiments are reported in Fig. 11, showing that the administration strategies $\mathsf{K}_1$, $\mathsf{K}_2$ and $\mathsf{K}_3$ are still successful in killing the cell.

## 8 Regulating differentiation in Th cells

The case study presented in this section is concerned with a Boolean network model of the regulatory process for differentiation in Th cells as proposed in [21] and recently translated in a RS model [22]. While the RS model from [22] is able to reproduce the most important dynamics aspects of the regulatory process, it must encode different levels of the same entity as separate objects. Here we show that, using linear processes, the ability to directly deal with concentration levels offers a more natural and simple way to represent this biological phenomenon.

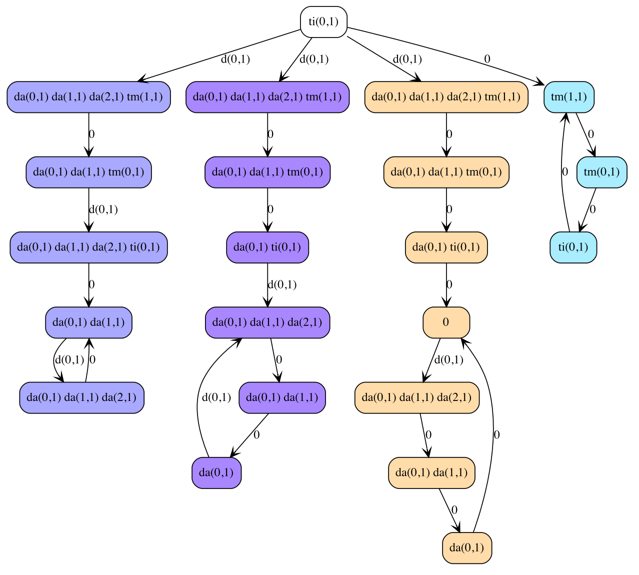


**Fig. 10** Different drug administration strategies with delay $\sigma = 1$ and duration $\delta = 3$

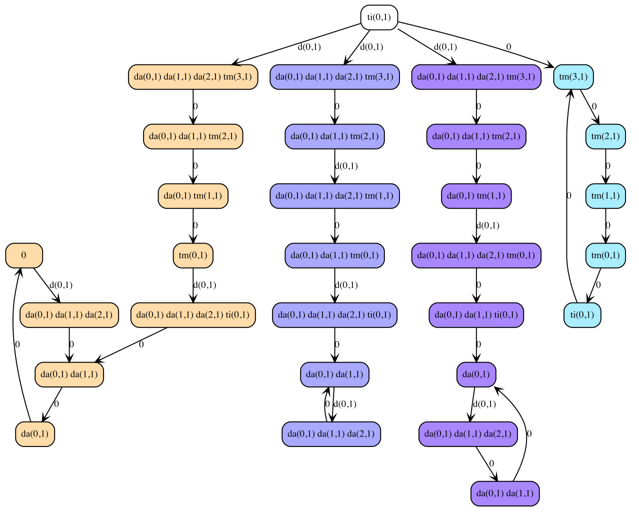


**Fig. 11** Different drug administration strategies with delay $\sigma = 3$ and duration $\delta = 3$

## 8.1 Biological phenomenon

The immune system is composed by various cell types, including antigen cells and B and T lymphocytes. Among the latter, T cells can be further sub-classified into T helper 1 (*Th1*) or T helper 2 (*Th2*) cells, originating from a common precursor *Th0*. The molecules secreted by *Th1* cells lead to an inflammatory immune responses, while those secreted by *Th2* cells intervene in humoral immune responses. Importantly, molecules produced by mature *Th* cells promote their own differentiation and at the same time inhibit the differentiation of cells of the other type.

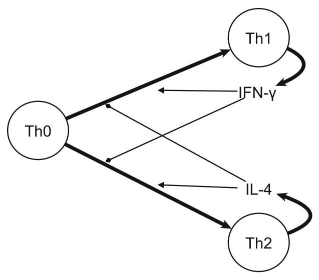


**Fig. 12** Differentiation of Th cells

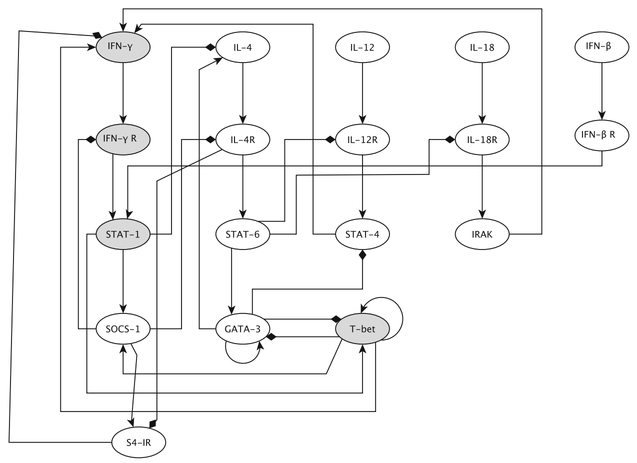


**Fig. 13** Graphical representation of the Boolean network

This is illustrated in Fig. 12, where the $Th1$ differentiation has as principal promoter IFN-$\gamma$ (such positive relation is represented by a standard arrow from IFN-$\gamma$ to $Th1$) and IL-4 as inhibitor (such negative relation is represented by an arrows ending with a rhombus directed from IL-4 to $Th1$), while, on the contrary, the $Th2$ differentiation has as principal promoter IL-4 and IFN-$\gamma$ as inhibitor.

A complex gene network regulates the differentiation of *Th0* cells. Studying the molecular mechanisms of this differentiation process is relevant since enhanced *Th1* and *Th2* responses may cause autoimmune and allergic diseases, respectively.

While a number of molecules were known to participate in this process, before [21], it was not clearly understood how they regulate each other to ensure differentiation. Finally, in [21] a Boolean network model of such a regulatory process has been conceived from the large amount of molecular data available in the literature. The proposed network includes 17 nodes regulating the differentiation of the *Th0* precursor [30, 31].

The structure of the Boolean network which describes how substances influence each other is depicted in Fig. 13, where, as before, standard arrows describe a promoting

$$
\begin{array}{lcl}
\text{IFN-}\gamma\text{-m}(t+1) & = & (\text{STAT-4}(t) \wedge \neg\text{IRAK}(t) \wedge \neg\text{T-bet-h}(t)) \vee \\
 & & (\text{T-bet-m}(t) \wedge (\neg\text{STAT-4}(t) \vee \neg\text{IRAK}(t))) \\
\text{IFN-}\gamma\text{-h}(t+1) & = & (\text{STAT-4}(t) \wedge \text{IRAK}(t)) \vee \text{T-bet-h}(t) \\
\text{IFN-}\gamma\text{R-m}(t+1) & = & \text{IFN-}\gamma\text{-m}(t) \vee (\text{IFN-}\gamma\text{-h}(t) \wedge \text{SOCS-1}(t)) \\
\text{IFN-}\gamma\text{R-h}(t+1) & = & \text{IFN-}\gamma\text{-h}(t) \wedge \neg\text{SOCS-1}(t) \\
\text{STAT-1-m}(t+1) & = & (\text{IFN-}\beta\text{R}(t) \wedge \neg\text{IFN-}\gamma\text{R-h}(t)) \vee \text{IFN-}\gamma\text{R-m}(t) \\
\text{STAT-1-h}(t+1) & = & \text{IFN-}\gamma\text{R-h}(t) \\
\text{T-bet-m}(t+1) & = & (\text{STAT-1-m}(t) \vee \text{T-bet-m}(t)) \wedge \\
 & & \neg(\text{STAT-1-h}(t) \vee \text{T-bet-h}(t) \vee \text{GATA-3}(t)) \\
\text{T-bet-h}(t+1) & = & (\text{STAT-1-h}(t) \vee \text{T-bet-h}(t)) \wedge \neg\text{GATA-3}(t) \\
\text{IL-4}(t+1) & = & \text{GATA-3}(t) \wedge \neg\text{STAT-1-m}(t) \wedge \neg\text{STAT-1-h}(t) \\
\text{S4-IR}(t+1) & = & \text{IL-4}(t) \wedge \neg\text{SOCS-1}(t) \\
\text{IL-12R}(t+1) & = & \text{IL-12}(t) \wedge \neg\text{STAT-6}(t) \\
\text{IL-18R}(t+1) & = & \text{IL-18}(t) \wedge \neg\text{STAT-6}(t) \\
\text{IFN-}\beta\text{R}(t+1) & = & \text{IFN-}\beta(t) \\
\text{GATA-3}(t+1) & = & (\text{STAT-6}(t) \vee \text{GATA-3}(t)) \wedge \neg(\text{T-bet-h}(t) \vee \text{T-bet-m}(t)) \\
\text{SOCS-1}(t+1) & = & \text{T-bet-m}(t) \vee \text{T-bet-h}(t) \vee \text{STAT-1-m}(t) \vee \text{STAT-1-h}(t) \\
\text{IRAK}(t+1) & = & \text{IL-18R}(t) \\
\text{STAT-4}(t+1) & = & \text{IL-12R}(t) \wedge \neg\text{GATA-3}(t) \\
\text{STAT-6}(t+1) & = & \text{S4-IR}(t)
\end{array}
$$

**Fig. 14** Boolean functions modelling the differentiation of Th cells from time $t$ to time $t+1$

relation while arrows ending with a rhombus represent an inhibiting relation.

The update functions that described how each influenced substance changes at the following step are summed up in Fig. 14. Without going into all details, the particularity of this system is that some substances (the ones coloured in grey in Fig. 13) admit different concentration levels of activation: an object can be inactive, activated at the medium level of concentration or activated at the high (maximum) level of concentration. This is the reason why, in Fig. 14, different update functions are used to describe the behaviour of a single object. Indeed, the different update functions describe how the different concentration level of the substance influence the other entities. For example, the behaviour of the object IFN-$\gamma$ (coloured grey in Fig. 13) is described by two update functions in Fig. 14: the one that updates IFN-$\gamma$ at the medium level (IFN-$\gamma$-m) and a second one that updates IFN-$\gamma$ at the high level (IFN-$\gamma$-h). The same holds for all the entities coloured in grey in Fig. 13 since they admit different levels of concentrations.

The above model identifies two key pathways involving IFN-$\gamma$ and IL-4. In the pathway involving IFN-$\gamma$, *Th1* cells produce IFN-$\gamma$ which acts on a membrane receptor (IFN-$\gamma$R). The transduction of the IFN-$\gamma$/IFN-$\gamma$R signal acts via STAT-1, which can be activated also by IFN-$\beta$ via IFN-$\beta$R. STAT-1 cannot be activated by IL-4, but STAT-1 itself modulates IL-4 signal through other molecules. Further down the IFN-$\gamma$ signal transduction pathway is SOCS-1, a molecule that is highly expressed in *Th1* cells, but not in *Th0* and *Th2* cells. IFN-$\gamma$ strongly induces SOCS-1 via a STAT-1-dependent pathway. SOCS-1, in turn, influences both the IFN-$\gamma$ and IL-4 pathways. Finally, it is known that SOCS-1 is able to block the capacity of IL-4R to generate a signalling in response to IL-4. T-bet is a transcription factor detected in *Th1*, but not in *Th0* or *Th2* cells. Its expression is up-regulated by IFN-$\gamma$, via STAT-1. In turn, T-bet is an IFN-$\gamma$ activator, thus creating an indirect positive feedback. Furthermore, it has been shown that T-bet is able to induce the transcription of its own gene.

The second pathway, involving IL-4, starts by the binding of IL-4 to its receptor, IL-4R, which is highly expressed in *Th2* cells. The IL-4R signal is transduced by STAT-6, which in turn activates GATA-3. GATA-3 is capable of inducing IL-4, thus establishing a positive feedback loop. The influence of the IL-4 pathway on the IFN-$\gamma$ pathway is mediated by GATA-3, since T-bet is down-regulated by GATA-3 expression. Conversely, T-bet is capable of inhibiting GATA-3. This mutual inhibition ensures that *Th1* and *Th2* cells express either one or the other molecule (T-bet in *Th1* and GATA-3 in *Th2*), but not both.

Apart from previous two key pathways, there are other molecules which affect the differentiation of *Th0* cells and we do not describe here.

## 8.2 On replacing distinct objects by concentration levels in linear processes

A standard closed RS (that is a RS with empty environment) that uses different entities to model different levels for the grey nodes in Fig. 13 was already defined in [22], where the authors translated the Boolean network into a RS able to reproduce the dynamics of the update functions in Fig. 14. However, this encoding was ad hoc, because it required the introduction of different objects to deal with different levels of the same entity. For example, we needed two different objects IFN-$\gamma$-m and IFN-$\gamma$-h to represent IFN-$\gamma$ at the medium and high level. The use of two different objects to model different level of activation required a particular care when considering sets of entities. Indeed, in this case not all subsets can have a meaning since a substance can either be activated at the medium or at the high level but not both. The artificial concept of *valid state* was introduced to avoid entities representing different levels of the same object to be present at the same time.

We aim to show that linear processes (from Sect. 4.2) allows us to seamlessly model the level of concentrations that are the key feature of this biological system. We express different concentrations levels with concentration values $\{1, 2\}$, where 1 stands for medium and 2 for high. For example, the state $\{\text{STAT-1}(1), \text{T-bet}(2)\}$ states that we have a medium concentration of STAT-1 and a high concentration of T-bet. The resulting linear RS contains the 26 reactions in Fig. 15 that model the system described in Fig. 14. Since durations are not needed, we exploit the

**Fig. 15** Reactions with concentration levels

| **Legenda:** ( $R$ , | $I$ , | $P$ ) |
|---|---|---|
| ( STAT-4(1) , | IRAK(1) T-bet(2) , | IFN-$\gamma$(1) ) |
| ( T-bet(1) , | IRAK(1) , | IFN-$\gamma$(1) ) |
| ( T-bet(1) , | S4-IR(1) , | IFN-$\gamma$(1) ) |
| ( STAT-4(1) IRAK(1) , | $\emptyset$ , | IFN-$\gamma$(2) ) |
| ( T-bet(2) , | $\emptyset$ , | IFN-$\gamma$(2) ) |
| ( GATA-3(1) , | STAT-1(1) , | IL4(1) ) |
| ( IFN-$\gamma$(1) , | $\emptyset$ , | IFN-$\gamma$R(1) ) |
| ( IFN-$\gamma$(2) SOCS-1(1) , | $\emptyset$ , | IFN-$\gamma$R(1) ) |
| ( IFN-$\gamma$(2) , | SOCS-1(1) , | IFN-$\gamma$R(2) ) |
| ( IL-4(1) , | SOCS-1(1) , | IL-4R(1) ) |
| ( IL-12(1) , | STAT-6(1) , | IL-12R(1) ) |
| ( IL-18(1) , | STAT-6(1) , | IL-18R(1) ) |
| ( IFN-$\beta$(1) , | $\emptyset$ , | IFN-$\beta$R(1) ) |
| ( IFN-$\beta$R(1) , | IFN-$\gamma$R(2) , | STAT-1(1) ) |
| ( IFN-$\gamma$R($x+1$) , | $\emptyset$ , | STAT-1($x+1$) ) |
| ( IL-4(1) , | $\emptyset$ , | STAT-6(1) ) |
| ( IL-12R(1) , | GATA-3(1) , | STAT-4(1) ) |
| ( IL-18R(1) , | $\emptyset$ , | IRAK(1) ) |
| ( STAT-1(1) , | $\emptyset$ , | SOCS-1(1) ) |
| ( T-bet(1) , | $\emptyset$ , | SOCS-1(1) ) |
| ( STAT-6(1) , | T-bet(1) , | GATA-3(1) ) |
| ( GATA-3(1) , | T-bet(1) , | GATA-3(1) ) |
| ( T-bet(1) , | GATA-3(1) STAT-1(2) , | T-bet(1) ) |
| ( T-bet(2) , | GATA-3(1) , | T-bet(2) ) |
| ( STAT-1($x+1$) , | GATA-3(1) , | T-bet($x+1$) ) |
| ( IL-12R(1) IL-18R(1) , | GATA-3(1) , | S4-IR(1) ) |

syntax from Sect. 4.2: a linear pattern is written as $\mathsf{a}(m \cdot x + n)$, while ground patterns just as $\mathsf{a}(n)$.[6]

We now show that using the linear patterns framework has many advantages compared to approaches where the different levels are modelled using different objects. In the following, we assume that the medium level of concentration of an entity is represented by a new entity whose name is obtained postponing the suffix -m to the name of the original entity while the high level of concentration of the entity is represented by a second new entity whose name is obtained postponing the suffix -h.

For example, when using two objects such as STAT-1-h and STAT-1-m to model the different levels of STAT-1, a reaction like $(\{\text{GATA-3}\}, \{\text{STAT-1-h}, \text{STAT-1-m}\}, \{\text{IL-4}\})$ needs to be included in the reaction system to describe the production of IL-4, which is inhibited by STAT-1 *at any* concentration. Such rule must include two objects as inhibitors (one for each level of STAT-1, using suffixes -h and -m), which seems somehow artificial, because both objects refer to the same entity, although they carry different names. On the contrary, in our framework based on linear patterns the very same constraint can be modelled by the following unary reaction: ( GATA-3(1) , STAT-1(1) , IL-4(1) ).

Indeed, such reaction is enabled in any state containing GATA-3, but no STAT-1 at any level of concentration, as desired. Moreover, even it emerged the necessity to differentiate among other levels of STAT-1 (e.g., low, or very high) the above reaction would still remain valid. Similarly, let us consider the reaction for the production of SOCS-1 that is enabled when T-bet is present at any level of concentration. When the different levels are modelled using different objects, we need to write two different reactions, one for each level, namely $(\{\text{T-bet-h}\}\ ,\ \emptyset\ ,\ \{\text{SOCS-1}\})$ and $(\{\text{T-bet-m}\}\ ,\ \emptyset\ ,\ \{\text{SOCS-1}\})$. Instead, using concentration levels, the production of SOCS-1 can be expressed by the single unary reaction ( T-bet(1) , $\emptyset$ , SOCS-1(1) ). Likewise the previous rule, even it emerged the necessity to differentiate among other levels of T-bet the above reaction would still remain unchanged.

But there are even more interesting consequences in replacing different objects by different concentration levels. In fact, it is very frequent that the level of one entity that is produced is dependent upon the level of some reactant. For example, using different objects, we need two reactions such as $(\{\text{IFN-}\gamma\text{R-m}\}\ ,\ \emptyset\ ,\ \{\text{STAT-1-m}\})$ and $(\{\text{IFN-}\gamma\text{R-h}\}\ ,\ \emptyset\ ,\ \{\text{STAT-1-h}\})$ to express the fact that the levels of IFN-$\gamma$R and STAT-1 are correlated. Using linear patterns, one reaction suffices: ( IFN-$\gamma$R($x+1$) , $\emptyset$ , STAT-1($x+1$) ).

As RS specifications can grow very large and complex, having the ability to keep the number of reactions as small as possible has many advantages, because reactions will be easier to write, to maintain, to change, to study and to extend and they will also be more flexible to experiment with. For example, imagine a situation where one wants to compare models based on different levels of some entities, without having the knowledge for fixing in advance how many levels is more convenient to use: building on the above examples it should be evident that using linear processes the comparison may be conducted even without rewriting a new specification for each possible combination of levels.

[6] The input for the tool should include a delay 0 as a first parameter, like in $\mathsf{a}(0, n)$.

Our prototype implementation allows us to compute the LTS. We performed one in silico experiment that show all the paths leading to *Th1* differentiation (characterised by the presence of T-bet as the marker of the differentiation of the cell into *Th1* form) and *Th2* (characterised by the presence of GATA-3 as the marker of the differentiation of the cell into *Th2* form).

Note that there are two possible paths that lead to the expression of to *Th1*. The first one is driven by the up-regulation of IFN-$\gamma$ that is expressed at the maximal level at the initial state: IFN-$\gamma(2)$. The second path leading to the expression of *Th1* is driven by the initial expression of both IL-12 and IL-18. The initial state in this case is IL-$12(1) \mid$ IL-$18(1)$, and after 9 steps, the system reaches the *same* stable state. Instead, the evolution leading to *Th*2 is activated by the initial state IL-$4(1)$ and after 6 evolution steps reaches the stable state IL-$4(1) \mid$ GATA-$3(1)$ $\mid$ STAT-$6(1) \mid$ IL-4R$(1)$.

Our tool allows us to inspect all different evolution paths by starting with the reaction system in parallel with the initial context

$$\mathsf{K} \triangleq \text{IFN-}\gamma(2).\mathsf{K}_\emptyset \; + \; \text{IL-12}(1)\,\text{IL-18}(1).\mathsf{K}_\emptyset \; + \; \text{IL-4}(1).\mathsf{K}_\emptyset$$

where $\mathsf{K}_\emptyset \triangleq \mathsf{rec}\; X.\; \emptyset.X$ is the trivial recursive process that provides an empty context at any step. The corresponding LTS is given in Fig. 16, where it is possible to observe that the evolution path triggered by the context IL-$12(1)$ IL-$18(1)$ takes a few steps before joining the path driven by the up-regulation of IFN-$\gamma$. For the differentiation leading to *Th1*, it can be observed that IL-4 needs to be inactive; on the contrary, for the differentiation leading to *Th2*, IFN-$\gamma$ needs to be inactive.

## 9 Synaptic transmission

The case study presented in this section requires the combined use of duration and concentration levels in a single RS. The case study is concerned with the modelling of synaptic transmission in neural networks communication. Our goal is to show that with RSs it is possibile to approximate spiking behaviours obtained by kinetic and stochastic models such as those defined in [24, 32].

### 9.1 Biological phenomenon

Synaptic transmission is the process that allows two neurons connected by a synapse to communicate. Communication consists in impulsive chemical signals that are sent from the first neuron to the second. Chemical signals take the form of neurotransmitters that are released from the

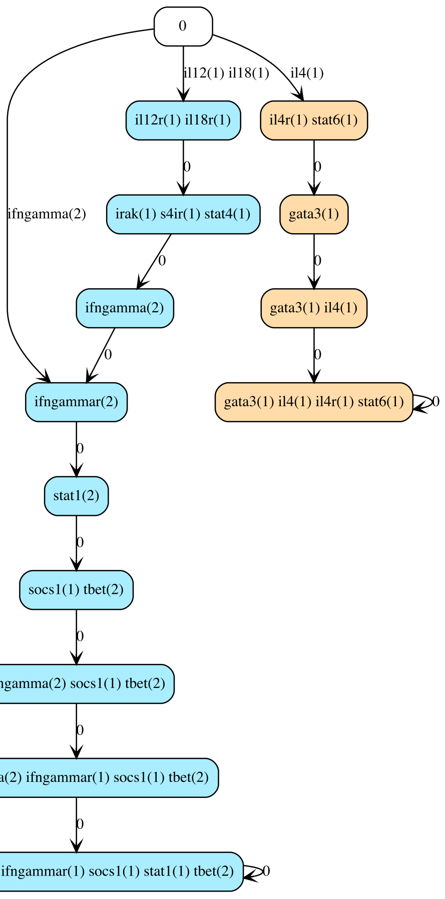


**Fig. 16** Evolution of $Th_0$ cell

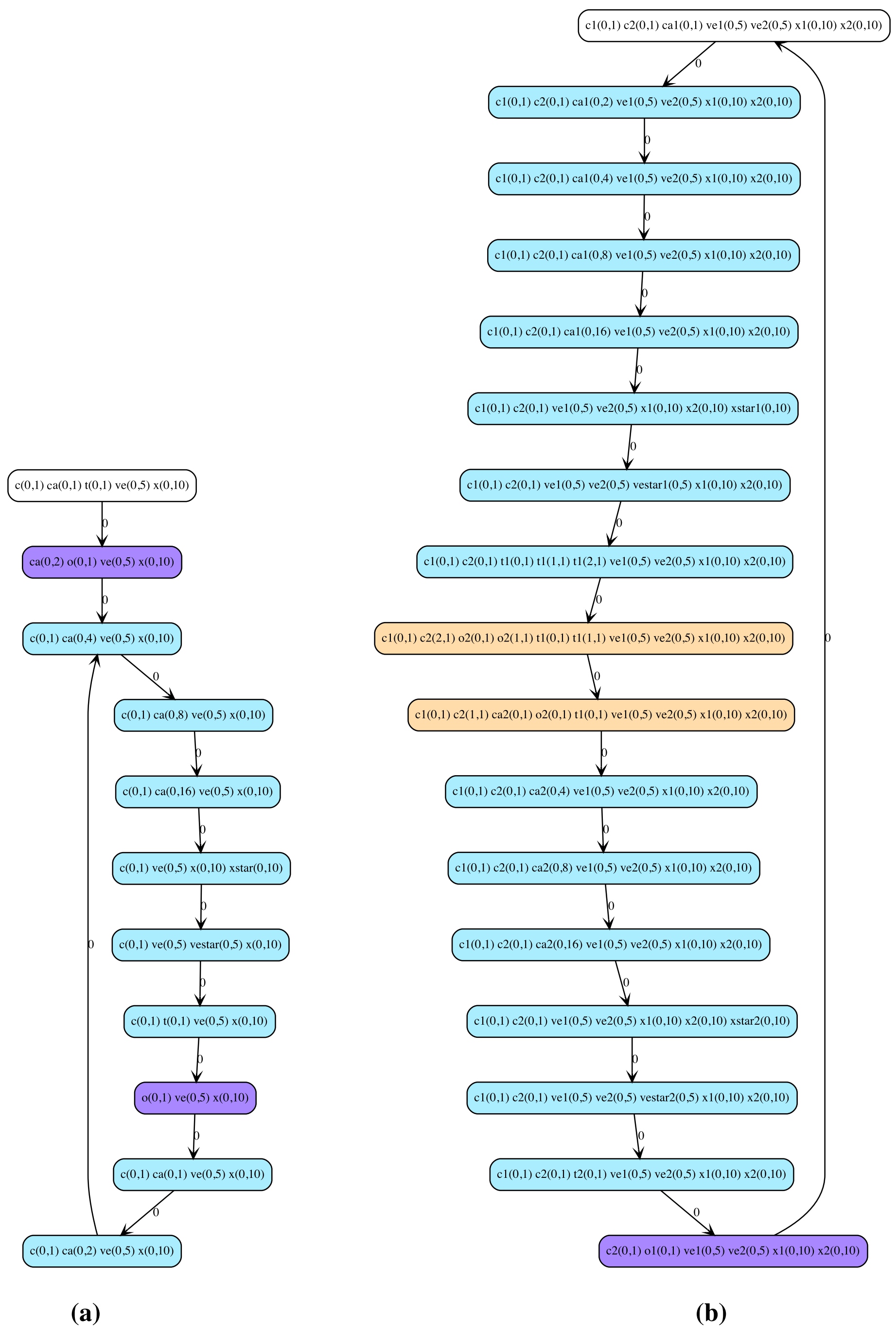
| **Legenda:** | $( R\,,$ | $I\,,$ | $P\,)$ |
|---|---|---|---|
| pre-synaptic | | | |
| $(r1)$ | $(\ \mathrm{Ca}(x+1)\ ,$ | $\mathrm{Ca}(10)\ ,$ | $\mathrm{Ca}(2x+2)\ )$ |
| $(r2)$ | $(\ \mathrm{X}(x+1)\ ,$ | $\mathrm{X}^*(1)\ ,$ | $\mathrm{X}(x+1)\ )$ |
| $(r3)$ | $(\ \mathrm{Ve}(x+1)\ ,$ | $\mathrm{Ve}^*(1)\ ,$ | $\mathrm{Ve}(x+1)\ )$ |
| $(r4)$ | $(\ \mathrm{Ca}(x+10)\ \mathrm{X}(x+10)\ ,$ | $\emptyset\ ,$ | $\mathrm{X}^*(x+10)\ )$ |
| $(r5)$ | $(\ \mathrm{X}^*(10)\ \mathrm{Ve}(x+1)\ ,$ | $\emptyset\ ,$ | $\mathrm{X}(10)\ \mathrm{Ve}^*(x+1)\ )$ |
| $(r6)$ | $(\ \mathrm{Ve}^*(x+1)\ ,$ | $\emptyset\ ,$ | $\mathrm{T}(1)\ \mathrm{Ve}(x+1)\ )$ |
| post-synaptic | | | |
| $(r7)$ | $(\ \mathrm{c}(x+1)\ ,$ | $\mathrm{T}(1)\ ,$ | $\mathrm{c}(x+1)\ )$ |
| $(r8)$ | $(\ \mathrm{T}(1)\ ,$ | $\emptyset\ ,$ | $\mathrm{o}(1)\ )$ |
| $(r9)$ | $(\ \mathrm{o}(1)\ ,$ | $\emptyset\ ,$ | $\mathrm{Ca}(1)\ \mathrm{c}(1)\ )$ |

**Fig. 17** Rules for the pre- and the postsynaptic sides activity of a neuron

first neuron and perceived by the second one, and they are stimulated by ionic currents.

The macroscopic dynamics of the currents involved in synaptic transmission can be described by means of kinetic models in which all the essential processes are expressed in terms of reactions. Synaptic transmission can be described as a two-phase phenomenon. The first phase (presynaptic release) is the release of neurotransmitters by the first neuron. It is stimulated by a calcium current that promotes the release of neurotransmitters from vesicles in which they are contained into the synaptic cleft. The second phase

**Fig. 18** Neuron activities: **a** one neuron; **b** synaptic transmission between two neurons


c(0,1) ca(0,1) t(0,1) ve(0,5) x(0,10)
ca(0,2) o(0,1) ve(0,5) x(0,10)
c(0,1) ca(0,4) ve(0,5) x(0,10)
c(0,1) ca(0,8) ve(0,5) x(0,10)
c(0,1) ca(0,16) ve(0,5) x(0,10)
c(0,1) ve(0,5) x(0,10) xstar(0,10)
c(0,1) ve(0,5) vestar(0,5) x(0,10)
c(0,1) t(0,1) ve(0,5) x(0,10)
o(0,1) ve(0,5) x(0,10)
c(0,1) ca(0,1) ve(0,5) x(0,10)
c(0,1) ca(0,2) ve(0,5) x(0,10)
0
(a)
c1(0,1) c2(0,1) ca1(0,1) ve1(0,5) ve2(0,5) x1(0,10) x2(0,10)
c1(0,1) c2(0,1) ca1(0,2) ve1(0,5) ve2(0,5) x1(0,10) x2(0,10)
c1(0,1) c2(0,1) ca1(0,4) ve1(0,5) ve2(0,5) x1(0,10) x2(0,10)
c1(0,1) c2(0,1) ca1(0,8) ve1(0,5) ve2(0,5) x1(0,10) x2(0,10)
c1(0,1) c2(0,1) ca1(0,16) ve1(0,5) ve2(0,5) x1(0,10) x2(0,10)
c1(0,1) c2(0,1) ve1(0,5) ve2(0,5) x1(0,10) x2(0,10) xstar1(0,10)
c1(0,1) c2(0,1) ve1(0,5) ve2(0,5) vestar1(0,5) x1(0,10) x2(0,10)
c1(0,1) c2(0,1) t1(0,1) t1(1,1) t1(2,1) ve1(0,5) ve2(0,5) x1(0,10) x2(0,10)
c1(0,1) c2(2,1) o2(0,1) o2(1,1) t1(0,1) t1(1,1) ve1(0,5) ve2(0,5) x1(0,10) x2(0,10)
c1(0,1) c2(1,1) ca2(0,1) o2(0,1) t1(0,1) ve1(0,5) ve2(0,5) x1(0,10) x2(0,10)
c1(0,1) c2(0,1) ca2(0,4) ve1(0,5) ve2(0,5) x1(0,10) x2(0,10)
c1(0,1) c2(0,1) ca2(0,8) ve1(0,5) ve2(0,5) x1(0,10) x2(0,10)
c1(0,1) c2(0,1) ca2(0,16) ve1(0,5) ve2(0,5) x1(0,10) x2(0,10)
c1(0,1) c2(0,1) ve1(0,5) ve2(0,5) x1(0,10) x2(0,10) xstar2(0,10)
c1(0,1) c2(0,1) ve1(0,5) ve2(0,5) vestar2(0,5) x1(0,10) x2(0,10)
c1(0,1) c2(0,1) t2(0,1) ve1(0,5) ve2(0,5) x1(0,10) x2(0,10)
c2(0,1) o1(0,1) ve1(0,5) ve2(0,5) x1(0,10) x2(0,10)
0
(b)

(postsynaptic uptake) involves different receptors (e.g., AMPA, NMDA, $\mathrm{GABA}_A$ and $\mathrm{GABA}_B$) that react to the availability of the transmitters with the creation of new currents in the second neuron. Different receptors generate currents with different intensities and rise/decay times.

As an example of synaptic transmission we mention the one mediated by the calyx of Held, which is a large synapse in the mammalian auditory brainstem circuit that synapses onto the cell body of the principal neuron of the medial nucleus of the trapezoid body (MNTB). The functional communication between the active sites of the calyx of Held and the principal neuron of MNTB is implemented by the release of a large number of synaptic vesicles containing glutamate.

Roughly, the vesicle release (exocytosis) depends on the amount of calcium, $Ca2^+$, in the presynaptic site. After the exocytosis has took place, the vesicles release their content that in turn bind the receptors of the postsynaptic sides. Here, the AMPA receptors bind the neurotransmitter released by the vesicle, the glutamate, and causes a chain

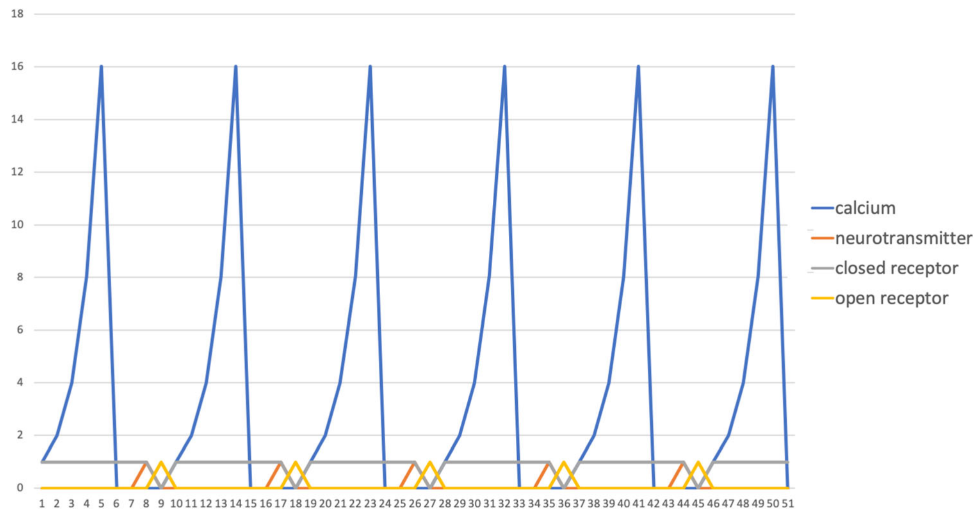


**Fig. 19** Activity chart of one neuron

| **Legenda:** | ( $R$ , | $I$ , | $P$ ) |
|---|---|---|---|
| | | $\dots$ | |
| $(r_1 6)$ | ( $\mathrm{Ve}_1^*(x+1)$ , | $\emptyset$ , | $\mathrm{T}_1(1)^{[0,3]}\ \mathrm{Ve}_1(x+1)$ ) |
| $(r_1 7)$ | ( $\mathrm{c}_1(x+1)$ , | $\mathrm{T}_2(1)$ , | $\mathrm{c}_1(x+1)$ ) |
| $(r_1 8)$ | ( $\mathrm{T}_2(1)$ , | $\emptyset$ , | $\mathrm{o}_1(1)$ ) |
| | | $\dots$ | |
| $(r_2 7)$ | ( $\mathrm{c}_2(x+1)$ , | $\mathrm{T}_1(1)$ , | $\mathrm{c}_2(x+1)$ ) |
| $(r_2 8)$ | ( $\mathrm{T}_1(1)$ , | $\mathrm{o}_2(1)$ , | $\mathrm{o}_2(1)^{[0,2]}\ \mathrm{c}_2(1)^2$ ) |
| $(r_2 9a)$ | ( $\mathrm{o}_2(1)\ \mathrm{T}_1(1)$ , | $\emptyset$ , | $\mathrm{Ca}_2(1)$ ) |
| $(r_2 9b)$ | ( $\mathrm{o}_2(1)\ \mathrm{Ca}_2(x+1)\ \mathrm{T}_1(1)$ , | $\emptyset$ , | $\mathrm{Ca}_2(x+4)$ ) |

**Fig. 20** Model of two interacting neurons: receptor of neuron two is slower in closing

of reactions inside the postsynaptic side that ends with the changing of the receptor configuration in its open form.

For a more detailed description of the biological phenomenon and of mathematical models of the dynamics of synaptic transmission, we refer to [23, 24, 32].

## 9.2 A discrete model of neural communication

In this paper, we present a simple functional model with a quantitative abstraction that does not consider the kinetic rates of the different biological reactions. We describe three neural examples, consisting of one, two and three neurons, respectively. The dynamics described by our model can be qualitatively compared with those described in [24, 32].

*A single-neuron model*

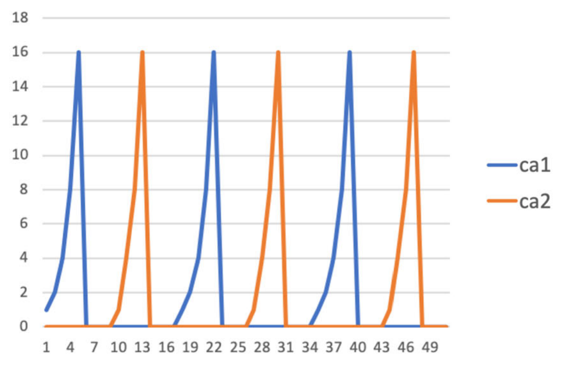
The synaptic activity concerning the presynaptic side is described by the reactions in Fig. 17, together with the one of the membrane receptor in the postsynaptic side. The presynaptic activity is characterised by the growth of calcium by doubling its quantity at each step, until the threshold 10 is reached. This is modelled by reaction ($r1$). Reactions ($r4$) and ($r5$) model the formation of the vesicles, Ve$^*$, that are then ejected via exocytosis releasing their content T by reaction ($r7$). Reaction ($r8$) models the binding of an external neurotransmitter T, opening of the neural receptor, which changes its own state from c, closed, to o, open, and ($r9$) models the closure of the receptor. The remaining reactions ($r2$), ($r3$) and ($r7$) model the permanency of vesicles, Ve, the calcium ligand, X, and the closed receptor c. By abuse of notation, we indicate as T both the whole content of the vesicle released by the presynaptic side and the neurotransmitter that binds the receptors on the postsynaptic membrane (causing the neuron to send the T signal to itself).

Given the initial state $\mathrm{c}(1) \mid \mathrm{Ca}(1) \mid \mathrm{X}(10) \mid \mathrm{Ve}(5) \mid \mathrm{T}(1)$, the LTS showing the cyclic behaviour of the neuron is shown in Fig. 18a, where the colour of each node depends on the status (closed/open) of the receptor.

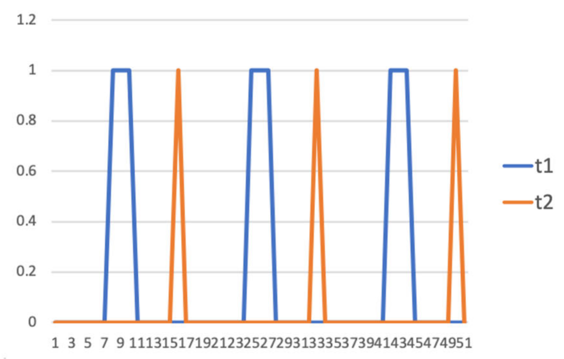
Figure 19 shows the peaks of the calcium quantity that activate the release of the neurotransmitter that in turn causes the opening of the receptor; then, the amount of the calcium restarts to rise again.

*A two-neuron model* Here, we consider two neurons such that the neurotransmitter released by the first neuron activates the receptor of the second one and vice versa. In this model, we assume the two neurons have different speeds: the receptor of the second neuron is slower to close. This implies that it remains open for a longer time and this allows a greater quantity of calcium to be produced. In Fig. 20, we only present the modified rules for the two neurons. Positive delays are represented as superscripts. We use subscripts 1 and 2 to distinguish between the entities belonging to neuron one and two, respectively.

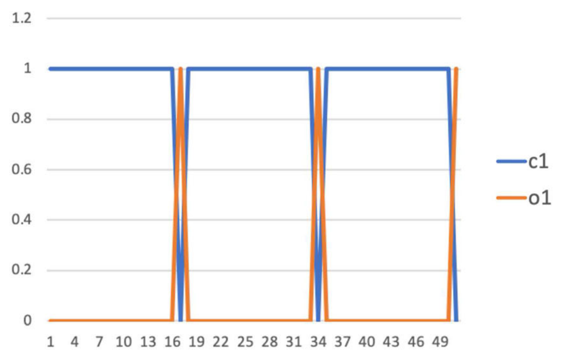
The opening of the receptor of neuron one is modelled by reaction $(r_1 8)$, and the opening of the receptor of neuron two is described by reaction $(r_2 8)$. Reactions $(r_2 9a)$ and $(r_2 9b)$ model the increase of calcium stimulated by the open receptor, whose closure also depends on reaction $(r_2 8)$ (using delay 2). Given the initial state



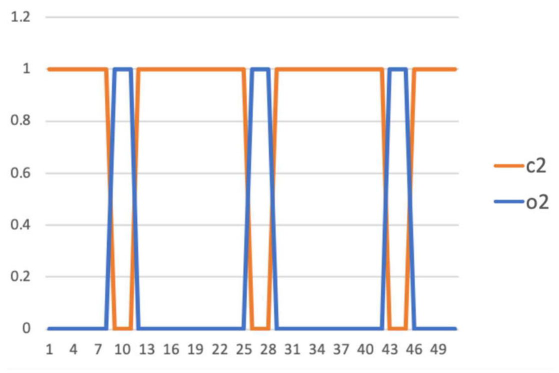
(1) levels of calcium



(2) presence of neurotransmitters



(3) activity of receptor 1



(4) activity of receptor 2

**Fig. 21** Charts for the activity of two neurons

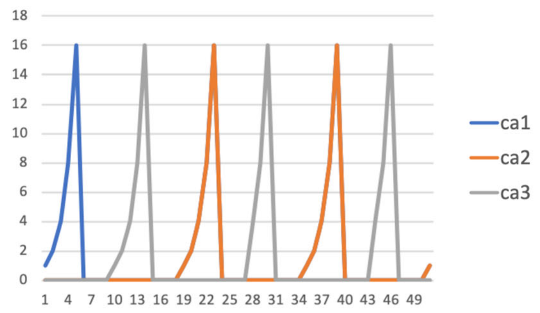
c(1) | Ca(1) | X(10) | Ve(5) | T(1), the LTS showing the cyclic behaviour of the neuron is shown in Fig. 18b, where states in which the receptors of each of the two neurons are open are coloured differently.

Finally, the effect of the interaction between the two neurons is shown by charts (1)–(4) of Fig. 21: in (1), the calcium in the second neuron, ca2, grows more quickly than ca1; in (2), the neurotransmitter $T_2$ remains active longer than $T_1$; and consequently, in (4) the receptor of neuron two remains open longer than the receptor of neuron one, in chart (3).

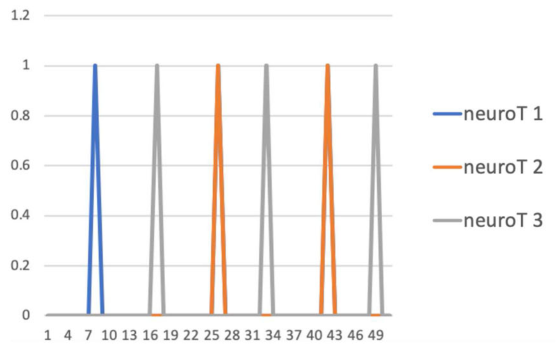
*A three-neuron model* The three neurons in this example are assumed to form a network with a `ypsilon` structure: the neurotransmitters of neuron one and two, $T_1$ and $T_2$, respectively, interact with the two receptors in neuron three, c31, and c32, respectively. Then, the neurotransmitter $T_3$ of neuron three interacts with the receptors of neurons one and two, $c_1$ and $c_2$, respectively. As done for the case of two neurons, we use subscripts 1, 2 and 3 to denote entities belonging to each of the three neurons, and in Fig. 22, we only give the rules that change with respect to the previous cases. In particular, reactions $r_3 9a$, $r_3 9b$ and $r_3 9c$ manage the opening of one or both of the two receptors in neuron three. The charts in Fig. 23 show that when only neurons one releases a neurotransmitter, $T_1$ in chart (2), only receptor (a) of neuron three will be open, as shown by chart (5). Chart (1) shows that when only one receptor in neuron three is open, the increase of calcium in neuron three is slower with respect to when both the receptors are open. Please note that in chart (2), the second and the third activation of neurotransmitter $T_1$ (the blue one) is overlaid by the activation of neurotransmitter $T_2$ (the orange one).

| **Legenda:** | ( $R$ , | $I$ , | $P$ ) |
|---|---|---|---|
| | | ... | |
| $(r_1 7)$ | ( $c_1(x+1)$ , | $T_3(1)$ , | $c_1(x+1)$ ) |
| $(r_1 8)$ | ( $T_3(1)$ , | $\emptyset$ , | $o_1(1)$ ) |
| | | ... | |
| $(r_2 7)$ | ( $c_2(x+1)$ , | $T_3(1)$ , | $c_2(x+1)$ ) |
| $(r_2 8)$ | ( $T_3(1)$, | $\emptyset$ , | $o_2(1)$ ) |
| | | ... | |
| $(r_3 7a)$ | ( $c_{31}(x+1)$ , | $T_1(1)$ , | $c_{31}(x+1)$ ) |
| $(r_3 8a)$ | ( $T_1(1)$ , | $\emptyset$ , | $o_{31}(1)$ ) |
| $(r_3 9a)$ | ( $o_{31}(1)$ , | $o_{32}(1)$ , | $c_{31}(1)\ Ca_3(1)$ ) |
| $(r_3 7b)$ | ( $c_{32}(x+1)$ , | $T_2(1)$ , | $c_{32}(x+1)$ ) |
| $(r_3 8b)$ | ( $T_2(1)$ , | $\emptyset$ , | $o_{32}(1)$ ) |
| $(r_3 9b)$ | ( $o_{32}(1)$ , | $o_{31}(1)$ , | $c_{32}(1)\ Ca_3(1)$ ) |
| $(r_3 9c)$ | ( $o_{32}(1)\ o_{31}(1)$ , | $\emptyset$ , | $c_{31}(1) c_{32}(1)\ Ca_3(4)$ ) |

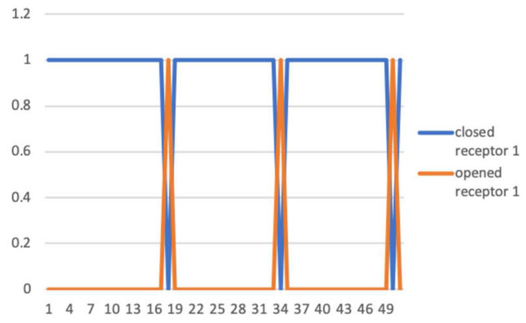
**Fig. 22** Rules for three interacting neurons connected to form a `ypsilon`



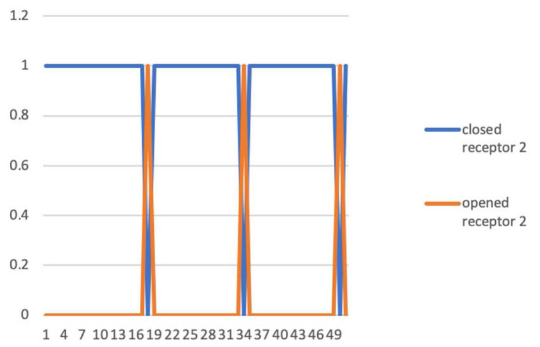
(1) levels of calcium



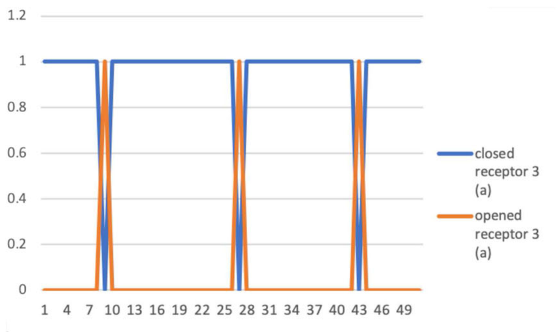
(2) presence of neurotransmitters



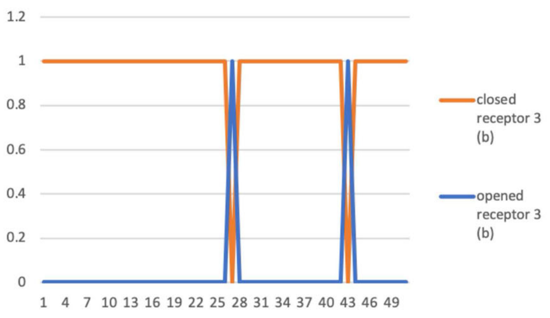
(3) activity of receptor 1



(4) activity of receptor 2



(5) activity of receptor 3



(6) activity of receptor 2

**Fig. 23** Charts for the activity of three interacting neurons

## 10 Conclusions and future work

In this paper, we presented the formal theory of timed and linear RS processes, we developed a tool where both extensions are integrated and we used this tool to investigate some biological pathways, gene regulation and neural networks.

As future work, we plan to exploit our framework to deepen the study of quantitative extensions of RSs without abandoning the discrete and abstract nature of RSs. Moreover, the availability of a formal semantics will allow us to study and apply formal analysis techniques aimed at assessing dynamical properties of the modelled biological systems, like logical properties and behavioural equivalences.

Finally, we plan to investigate the applicability of abstract interpretation techniques [33–35] to study properties of quantitative reaction systems by exploiting under- and over-approximations of current states, which is useful to make the analysis of large systems tractable.

**Acknowledgements** Research supported by University of Pisa PRA_2020_26 *Metodi Informatici Integrati per la Biomedica*, by MIUR PRIN Project 201784YSZ5 *ASPRA–Analysis of Program Analyses*, and by University of Sassari *Fondo di Ateneo per la ricerca 2020*. We thank the reviewers for their useful suggestions which helped us to improve our paper.

**Funding** Open access funding provided by Università degli Studi di Sassari within the CRUI-CARE Agreement.

## Declarations

**Conflict of interest** The authors have no competing interests to declare that are relevant to the content of this article.

## References

1. Ehrenfeucht A, Rozenberg G (2007) Reaction systems. Fundamenta informaticae 75(1–4):263–280
2. Brijder R, Ehrenfeucht A, Main M, Rozenberg G (2011) A tour of reaction systems. Int J Found Comput Sci 22(07):1499–1517. https://doi.org/10.1142/S0129054111008842
3. Azimi S, Iancu B, Petre I (2014) Reaction system models for the heat shock response. Fundam. Informaticae 131(3–4):299–312. https://doi.org/10.3233/FI-2014-1016
4. Corolli L, Maj C, Marinia F, Besozzi D, Mauri G (2012) An excursion in reaction systems: from computer science to biology. Theor Comput Sci 454:95–108. https://doi.org/10.1016/j.tcs.2012.04.003
5. Azimi S (2017) Steady states of constrained reaction systems. Theor Comput Sci 701(C):20–26. https://doi.org/10.1016/j.tcs.2017.03.047
6. Barbuti R, Gori R, Levi F, Milazzo P (2016) Investigating dynamic causalities in reaction systems. Theor Comput Sci 623:114–145. https://doi.org/10.1016/j.tcs.2015.11.041
7. Okubo F, Yokomori T (2016) The computational capability of chemical reaction automata. Nat Comput 15(2):215–224. https://doi.org/10.1007/s11047-015-9504-7
8. Ehrenfeucht A, Main MG, Rozenberg G (2010) Combinatorics of life and death for reaction systems. Int J Found Comput Sci 21(3):345–356. https://doi.org/10.1142/S0129054110007295
9. Ehrenfeucht A, Main MG, Rozenberg G (2011) Functions defined by reaction systems. Int J Found Comput Sci 22(1):167–178. https://doi.org/10.1142/S0129054111007927
10. Shang Z, Verlan S, Petre I, Zhang G (2019) Reaction systems and synchronous digital circuits. Molecules 24(10):1–13. https://doi.org/10.3390/molecules24101961
11. Ehrenfeucht A, Rozenberg G (2007) Events and modules in reaction systems. Theor Comput Sci 376(1–2):3–16. https://doi.org/10.1016/j.tcs.2007.01.008
12. Brijder R, Ehrenfeucht A, Rozenberg G (2011) Reaction systems with duration. In: Computation, cooperation, and life: essays dedicated to Gheorghe Păun on the occasion of his 60th birthday. Springer, Berlin, Heidelberg, pp 191–202. https://doi.org/10.1007/978-3-642-20000-7_16
13. Meski A, Koutny M, Penczek W (2016) Towards quantitative verification of reaction systems. In: Amos M, Condon A (eds) Unconventional computation and natural computation. Springer, Cham, pp 142–154. https://doi.org/10.1007/978-3-319-41312-9_12
14. Ehrenfeucht A, Kleijn J, Koutny M, Rozenberg G (2017) Evolving reaction systems. Theor Comput Sci 682:79–99. https://doi.org/10.1016/j.tcs.2016.12.031
15. Bottoni P, Labella A, Rozenberg G (2019) Reaction systems with influence on environment. J Membr Comput 1(1):3–19. https://doi.org/10.1007/s41965-018-00005-8
16. Bottoni P, Labella A, Rozenberg G (2020) Networks of reaction systems. Int J Found Comput Sci 31:53–71. https://doi.org/10.1142/S0129054120400043
17. Koutny M, Pietkiewicz-Koutny M, Yakovlev A (2021) Asynchrony and persistence in reaction systems. Theor Comput Sci 881:97–110. https://doi.org/10.1016/j.tcs.2020.11.040
18. Pardini G, Barbuti R, Maggiolo-Schettini A, Milazzo P, Tini S (2014) Compositional semantics and behavioural equivalences for reaction systems with restriction. Theor Comput Sci 551:1–21. https://doi.org/10.1016/j.tcs.2014.04.010
19. Brodo L, Bruni R, Falaschi M (2021) A logical and graphical framework for reaction systems. Theor Comput Sci 875:1–27. https://doi.org/10.1016/j.tcs.2021.03.024
20. Villasana M, Radunskaya A (2003) A delay differential equation model for tumor growth. J Math Biol 47(3):270–294. https://doi.org/10.1007/s00285-003-0211-0
21. Mendoza L (2006) A network model for the control of the differentiation process in th cells. Biosystems 84(2):101–114. https://doi.org/10.1016/j.biosystems.2005.10.004
22. Barbuti R, Gori R, Milazzo P (2021) Encoding boolean networks into reaction systems for investigating causal dependencies in gene regulation. Theor Comput Sci. https://doi.org/10.1016/j.tcs.2020.07.031
23. Schneggenburger R, Neher E (2000) Intracellular calcium dependence of transmitter release rates at a fast central synapse. Nature 406:889–893. https://doi.org/10.1038/35022702
24. Bracciali A, Brunelli M, Cataldo E, Degano P (2008) Stochastic models for the in silico simulation of synaptic processes. BMC Bioinform. https://doi.org/10.1186/1471-2105-9-S4-S7
25. Brodo L, Bruni R, Falaschi M, Gori R, Levi F, Milazzo P (2021) Exploiting modularity of SOS semantics to define quantitative extensions of reaction systems. In: Aranha C, Martín-Vide C, Vega-Rodríguez MA (eds) Proceedings of TPNC 2021. Lecture Notes in Computer Science, vol 13082. Springer, Cham, pp 15–32. https://doi.org/10.1007/978-3-030-90425-8_2
26. Milner R (1980) A calculus of communicating systems, vol 92. Lecture Notes in Computer Science. Springer, Heidelberg
27. Brodo L, Bruni R, Falaschi M (2019) Enhancing reaction systems: a process algebraic approach. In: Alvim M, Chatzikokolakis K, Olarte C, Valencia F (eds) The art of modelling computational systems. LNCS, vol 11760. Springer, Berlin, pp 68–85. https://doi.org/10.1007/978-3-030-31175-9_5
28. Bernini A, Brodo L, Degano P, Falaschi M, Hermith D (2018) Process calculi for biological processes. Nat Comput 17(2):345–373. https://doi.org/10.1007/s11047-018-9673-2
29. Cortadella J, Kishinevsky M, Kondratyev A, Lavagno L, Taubin A, Yakovlev A (1998) Lazy transition systems: application to timing optimization of asynchronous circuits. In: 1998 IEEE/ACM international conference on computer-aided design. Digest of Technical Papers (IEEE Cat. No.98CB36287), pp 324–331 . https://doi.org/10.1145/288548.288633
30. Murphy KM, Reiner SL (2002) Decision making in the immune system: the lineage decisions of helper t cells. Nat Rev Immunol 2:933–944. https://doi.org/10.1038/nri954

31. Agnello D, Lankford CSR, Bream J, Morinobu A, Gadina M, O'Shea JJ, Frucht DM (2003) Cytokines and transcription factors that regulate t helper cell differentiation: new players and new insights. J Clin Immunol 23(3):147–161. https://doi.org/10.1023/A:1023381027062
32. Destexhe A, Sejnowski ZFMTJ (1998) Kinetic models of synaptic transmission. Methods Neuronal Model 1–25
33. Cousot P, Cousot R (1977) Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In: Proceedings of the ACM POPL'77. ACM, New York, pp 238–252. https://doi.org/10.1145/512950.512973
34. Cousot P (2021) Principles of abstract interpretation. MIT Press, Cambridge
35. Bodei C, Brodo L, Gori R, Levi F, Bernini A, Hermith D (2017) A static analysis for Brane Calculi providing global occurrence counting information. Theoret Comput Sci 696:11–51. https://doi.org/10.1016/j.tcs.2017.07.008