Reasoning about Actions in Biophysical Systems

Chitta Baral and **Juraj Dzifcak** and **Nam Tran** and **Jicheng Zhao**
Department of Computer Science and Engineering,
Arizona State University,
Tempe, AZ 85233, USA.
{*chitta, juraj.dzifcak, namtran, jicheng*}*@asu.edu*

abstract

We develop a high level action description language to express knowledge about cellular processes and mechanisms. This involves representation and reasoning about both discrete properties and continuous processes. Both of them may be changed by exogenous actions or triggers. We use differential equations to represent continuous processes. We give syntax and semantics of the language and also present an approximate characterization. We present a temporal query language for such domains. We then illustrate the use of our language with respect to biological examples and discuss an implementation of the approximate characterization.

Introduction

Reasoning about actions and changes is a major cognitive task of a robot acting in the physical world. Rigorous formalisms for reasoning about actions have been developed and applied successfully to robotics (Grosskreutz & Lakemeyer 2000; Reiter 2001a; Thielscher 2000a; Shanahan 1998; Baral *et al.* 1998). Although the physical world is continuous, most of existing action languages are inherently discrete. The discrete modeling approach has been generally successful. Nevertheless, it has limitations in various physical domains where the modeling of continuous changes is essential. In this paper, we aim to develop a high level action language for modeling such physical domain, namely the domain of cellular processes and mechanisms. A cell is inherently a hybrid system that has both continuous properties such as concentrations of proteins and discrete properties such as whether two proteins are bounded or not. Besides, the constituents of a cell can be changed by either an exogenous action or an internally triggered action. An example of the former is the binding of a ligand to a receptor, and an example of the latter is the induction or stimulation of a biochemical reaction by some protein concentrations. Protein concentrations change continuously in time, and those concentrations determine the behavior of the cell and may trigger intra-cellular processes. On the other hand, the concentration of a particular protein is guided by concentrations of other proteins and some discrete properties of the cell. But it can be disrupted by extra-cellular events or newly triggered intra-cellular processes. The continuous changes are modeled in kinetic equations, for example, $A + B \rightleftharpoons AB$. This kinetic equation is translated into differential equations, such as

$$\dot{x}_A = \dot{x}_B = k_{\leftharpoondown} \times x_{AB} - k_{\rightharpoonup} \times x_A \times x_B$$
$$\dot{x}_{AB} = k_{\rightharpoonup} \times x_A \times x_B - k_{\leftharpoondown} \times x_{AB}$$

where $\dot{x} = dx/dt$ and x_A, x_B, x_{AB} are the concentration levels of proteins A, B and complex AB. Besides, $k_{\rightharpoonup}$ and $k_{\leftharpoondown}$ are rate constants for association and dissociation.

In this work, we design an action language to model cellular behaviors. With the goal of intuitiveness in mind, we choose to extend the high-level action language $\mathcal{A}$. To capture the continuous properties, we extend states to include continuous real-valued variables. Further new features are flow constraints and triggered actions. Flow constraints are based on differential equations and dictate the continuous change of the cell from states to states. Triggered actions are important elements in modeling the cell (Tran & Baral 2004; 2005). We adopt the semantics of hybrid automata for our language. A hybrid automata related semantics would allow for our utilization of state-of-the-art methods and techniques from hybrid automata and symbolic model checking community (Henzinger, Ho, & Wong-Toi 1997; R. Mateescu & H. Garavel. 1998; Cimatti *et al.* 2002; Gerard J. Holzmann 2003). This enables us in having an efficient implementation of reasoning about the cellular behaviors.

There have been various works in action languages dealing with continuous changes and triggered actions. Logic and differential equations were combined to model piecewise continuous systems in (Sandewall 1989; Kolen & Zhao 1996). Natural actions - triggered actions in disguise - and continuous changes have been studied in situation calculus (Fusaoka 1996; Reiter 1996; 2001b; Pinto 1998; Grosskreutz & Lakemeyer 2000). Event calculus based formulations of discrete-continuous systems were provided in (Shanahan 1990; Miller & Shanahan 1996). Various other formalisms for continuous changes and processes have been proposed (Herrmann & Thielscher 1996; Chintabathina, Gelfond, & Watson 2005; Thielscher 2000b; Baral, Son, & Tuan 2002; Thielscher 2001). Nevertheless, none of the existing works has all the key features: the intuitiveness of high-level action language $\mathcal{A}$, combination of continuous changes and discrete changes, triggered actions, and a large body of support for efficient reasoning (Henzinger, Ho, & Wong-Toi 1997; R. Mateescu & H. Garavel. 1998; Cimatti *et al.* 2002; Gerard J. Holzmann 2003). Moreover, the only language which has been applied to biological domains (as far as we know), does not capture the reasoning about continuous changes (Tran & Baral 2004).

With the focus on formulating reasoning about actions, our work also differs from the array of biological modeling frameworks. Those include qualitative reasoning systems (Heidtke & Schulze-Kremer 1998; Alur *et al.* 2001; Antoniotti *et al.* 2003a; 2003b; Batt *et al.* 2005), Petri nets (Reddy, Liebman, & Mavrovouniotis 1996; Peleg, Yeh, & Altman 2002), π-calculus (Regev, Silverman, & Shapiro 2001), process algebra (Funahashi *et al.* 2003; Calder, Hillston, & Gilmore. 2004), rewriting logic (Eker *et al.* 2002; Talcott *et al.* 2004), and symbolic modeling checking (Peres & Comet 2003; Chabrier *et al.* 2004; Fages, Soliman, & Chabrier-Rivier). By extending action language $\mathcal{A}$, our work enables us to reason in problems such as planning, explanation, prediction, etc.

Action Language for Continuous Processes

Syntax

The alphabet of **ALCP** consists of the following parts.

- a finite set $\mathbf{A}$ of actions: there are two distinguished types of actions: *exogenous* actions and *triggered* actions.
- a finite set $\mathbf{B}$ of Boolean variables.
- a finite set $\mathbf{X}$ of continuous (real) variables and the set of dotted variables $\dot{\mathbf{X}} = \{ \dot{x} \mid x \in \mathbf{X} \}$: the variables represent *processes* that change continuously through time and the dotted variables represent their first derivatives.
- a set $\mathbf{F}$ of functions from $dom(x_1) \times dom(x_2) \times \ldots \times dom(x_N)$ to $\mathbb{R}$, where $x_1, x_2, \ldots x_N$ are the continuous variables and $dom(x_i)$ is the domain of x_i.

A *term* is an algebraic expression of variables in $\mathbf{X}$, functions in $\mathbf{F}$, constants in $\mathbb{R}$ and operators in $\{+, -, \times, \div\}$. An *atom* is of the form $f = \top$ or $f = \bot$ where $f \in \mathbf{B}$; or is of the form E_1 **op** E_2, where E_1 and E_2 are terms and **op** $\in \{<, \leq, =, \geq, >\}$. Atoms $f = \top$ and $f = \bot$ are usually shorten to f and $\neg f$, respectively. A *flow constraint* is of the form $\dot{x} \equiv \psi$, where $\dot{x} \in \dot{\mathbf{X}}$ and ψ is a term.

Domain description A domain description is a collection of propositions of the form

$$f \textbf{ constrains } \dot{x} \equiv \psi \quad (1)$$
$$a \textbf{ causes } v = \varphi \textbf{ if } g \quad (2)$$
$$h \textbf{ triggers } b \quad (3)$$

We assume that f is a conjunction of atoms of Boolean variables, while g and h are conjunctions of atoms. $\dot{x} \equiv \psi$ is a flow constraint, $v \in \mathbf{X} \cup \mathbf{B}$ is a variable. φ is a term if $v \in \mathbf{X}$ and φ is either $\top$ or $\bot$ when $v \in mathbfB$. Finally, we assume that a is an action, and b is a triggered action. Propositions of the form (1), (2) and (3) are respectively called *flow rules*, *dynamic rules* and *trigger rules*.

Intuitively, a flow constraint $\dot{x} \equiv \psi$ indicates that continuous variable x changes at a speed ψ. To illustrate the syntax, let us consider a hypothetical example about the cell.

Example 1. *Ligand binding to Receptor activates Protein. The concentration x of active Protein increases in time according to the differential equation $\dot{x} = 1$. The concentration of inactive Protein decreases according to $\dot{x} = -0.1x$. Besides, whenever the concentration of active Protein is greater than 20, the degradation of Protein (by some other protein) is triggered, which keeps the concentration of Protein stable at $x = 20$. The domain description consists of the following propositions.*

$$\begin{aligned}
active \wedge \neg stable \ &\textbf{constrains}\ \dot{x} \equiv 1 \\
\neg active \ &\textbf{constrains}\ \dot{x} \equiv -0.1x \\
active \wedge stable \ &\textbf{constrains}\ \dot{x} \equiv 0 \\
bind \ &\textbf{causes}\ active \\
degrade \ &\textbf{causes}\ stable \textbf{ if } active \wedge (x \geq 20) \\
\neg stable \wedge x \geq 20 \ &\textbf{triggers}\ degrade
\end{aligned}$$

Here, the alphabet consists of the components: $\mathbf{A} = \{bind, degrade\}$*;* $\mathbf{B} = \{active, stable\}$, $\mathbf{X} = \{x\}$, *and* $\mathbf{F} = \{f_{\alpha,\beta}(x) \equiv \alpha x + \beta \mid \alpha, \beta \in \mathbb{R}\}$. □

An *action theory* is a pair $(\mathcal{D}, \mathcal{O})$ where $\mathcal{D}$ is a domain description and $\mathcal{O}$ is a set of observations of the form

$$\textbf{initially}\ v = c$$

where $v \in \mathbf{X} \cup \mathbf{B}$, c is a constant in the domain of x when $v \in \mathbf{X}$, and c is either $\top$ or $\bot$ when $v \in \mathbf{B}$.

Queries An action sequence is a sequence $A_1 : t_1, \ldots A_n : t_n$, where A_is are sets of exogenous actions and t_is are continuous time points ($0 \leq t_1 < \ldots < t_n$). A policy is a set of statements of the form **do** a **if** f, where a is an exogenous action and f is a conjunction of atoms.

A query in **ALCP** is of the forms

$$Q_1 = f_1 \textbf{ after } S$$
$$Q_2 = f_2 \textbf{ given } P$$

where f_1, f_2 are temporal goal formulas about properties of a sequence of states, S is an action sequence and P is a policy. Intuitively, Q_1 asks if f_1 will be true after the action sequence S occurs, and Q_2 asks if f_2 is true with respect to the policy P.

Temporal logics such as LTL, CTL* (Emerson & Clarke 1982), and their extensions have been used to express discrete properties of temporal goals. We define a temporal goal formula to be built on atoms, the connectives $\neg$, $\wedge$, $\vee$ and operators $\Box$, $\Diamond$, $\bigcirc[\delta]$ and $\cup$. The operators $\Box$, $\Diamond$, $\bigcirc[\delta]$ and $\cup$ are a generalization of linear temporal logic operators, whose semantics shall be defined later.

Definition 1 (Temporal goal)**.** *Let $\langle p \rangle$ be an atom, $\langle f \rangle$ be a goal formula, δ be a constant in $\mathbb{R}$.*

$$\begin{aligned}
\langle f \rangle \quad ::= \quad & \langle p \rangle \mid \langle f \rangle \wedge \langle f \rangle \mid \langle f \rangle \vee \langle f \rangle \mid \neg \langle f \rangle \\
& \mid \bigcirc [\delta] \langle f \rangle \mid \Box \langle f \rangle \mid \Diamond \langle f \rangle \mid \langle f \rangle \cup \langle f \rangle
\end{aligned}$$

Semantics

Given a domain description $\mathcal{D}$, an *interpretation* I of $\mathcal{D}$ is a value assignment of variables in $\mathcal{D}$. Given an interpretation I, we evaluate the value $I(\varphi)$ of a function φ (or the truth value $I(f)$ of a propositional logical formula f) in the usual way. An interpretation I satisfies a logical formula f, written as $I \models f$, if $I(f) = \top$. $I \models \varphi = c$ if $I(\varphi) = c$ where $c \in \mathbb{R}$. Similarly, $I \models \varphi > c$ if $I(\varphi) > c$ and $I \models \varphi >< c$ if $I(\varphi) < c$.

A *state* of $\mathcal{D}$ is an interpretation s if for each $x \in \mathbf{X}$, there exists at most one flow rule (f **constrains** $\dot{x} \equiv \psi$) such that $s \models f$. If such a flow rule exists, then the flow constraint $\dot{x} \equiv \psi$ is said to be entailed by s, written as $s \models \dot{x} \equiv \psi$. If for a continuous process x, no flow constraint f **constrains** $\dot{x} \equiv \psi$ is in the domain description or $s \not\models f$, then we consider that $s \models \dot{x} \equiv 0$, meaning that the value of x is not changed.

An action a is triggered by a state s, if there exists a trigger (f **triggers** a) such that $s \models f$.

The following example illustrates the notion of interpretations and states.

Example 2. *Let us consider the domain description in Example 1. Let $s = \{active, \neg stable, x := 20.1\}$ be an interpretation. That is: $s(active) = \top$, $s(stable) = \bot$, and $s(x) = 20.1$. Let $\varphi(x) = 0.5x + 1$, then $s(\varphi) = 0.5{*}20.1{+}1 = 11.05$. Let $g = \neg stable \wedge (\varphi \leq 10 \vee x > 20)$. Then $s(g) = \top$, thus $s \models g$.*

s entails the flow constraint $\dot{x} \equiv 1$. Since it is the only flow constraint for variable x, s is a state. We also have that $s \models \neg stable \wedge x \geq 20$, thus s triggers the action degrade. □

Transition functions Given a domain $\mathcal{D}$, the effect of an action a in a state s is the set of assignments $E(a, s) = \{v := s(\varphi) \mid (a$ **causes** $v = \varphi$ **if** $f) \in \mathcal{D}$ and $s \models f\}$. The effect of a set A of actions in a state s is the set $E(A, s) = \bigcup_{a \in A} E(a, s)$. The set $E(A, s)$ is consistent if there exist no continuous variable v and constants $c \neq c'$ such that $v := c \in E(A, s)$ and $v := c' \in E(A, s)$, and there exist no Boolean variable v such that both $v := \top$ and $v := \bot$ are in $E(A, s)$.

In the following definitions, we assume that a domain description $\mathcal{D}$ is given.

Definition 2 (Discrete state transition). *Let A be a set of actions and s is a state such that $E(A, s)$ is consistent. Then A transforms s to a state s' (denoted ($\Phi(s, A)$)) if:*

- *$s' = s \setminus \overline{E(A, s)} \cup E(A, s)$, where $\overline{E(A, s)} = \{v := c \mid v := c' \in E(A, s) \wedge c \neq c'\}$.*
- *and there exists at most one flow rule "f **constrains** $\dot{x} \equiv \psi$" for any continuous variable x given $s' \models f$.* □

Definition 3 (Continuous state transition). *A state s changes continuously after a time interval $\delta > 0$ to a new state s_δ, if:*

- *For all $x \in \mathbf{X}$, if s entails a flow constraint $\dot{x} \equiv \psi$, then there exists a differentiable function $F_x : [0, \delta] \to \mathbb{R}$ such that $s(x) = F_x(0)$ and for all $\epsilon \in (0, \delta)$, the equality $\dot{x} \equiv \psi$ is satisfied by the set of assignments $\{ x := F_x(\epsilon), \dot{x} := \dot{F}_x(\epsilon) \mid x \in \mathbf{X} \}$. Here, $\dot{F}_x : (0, \delta) \to \mathbb{R}$ is the first derivative of F_x. $s_\delta(x) = F_x(\delta)$.*
- *If s entails no flow constraint of the form $\dot{x} \equiv \psi$, then $s_\delta(x) = s(x)$.*
- *For all $\epsilon \in [0, \delta)$, no action is triggered by the state $s_\epsilon = s_{|B} \cup \{x := s(x) \mid \nexists F_x\} \cup \{x := F_x(\epsilon) \mid \exists F_x\}$. Here, $s_{|B}$ is the subset of Boolean assignments of s.* □

In general, F_x is a function whose first directive on x is the same as ψ if f **constrains** $\dot{x} \equiv \psi$ is a rule in the language and $s \models f$. Besides, F_x satisfies boundary conditions. Further more, no other triggers interrupt this function. In general, multiple states s_δ may satisfy Definition 3. *In this work, we assume that the domain $\mathcal{D}$ is such that a state s_δ, if it exists, is unique.* The existing state s_δ shall be denoted $\Psi(s, \delta)$. Also, let $\Psi(s, 0) = s$. The continuous transition function $\Psi(s, \delta)$ is analogous to the discrete transition function $\Phi(s, A)$ in that δ plays the role of A. We observe the following simple property of $\Psi(s, \delta)$.

Proposition 1. *Let $\mathcal{D}$ be a domain description and $\delta_1 > 0, \delta_2 > 0$. If $\Psi(s, \delta_1 + \delta_2)$ exists then so do $\Psi(s, \delta_1)$ and $\Psi(s, \delta_2)$, and $\Psi(s, \delta_1 + \delta_2) = \Psi(\Psi(s, \delta_1), \delta_2)$.*

Example 3. *Let us consider the domain description in Example 1. Let $s = \{\neg active, \neg stable, x := 18\}$. We know that s is a state. We have a discrete state transition: $\Phi(s, \{bind\}) = \{active, \neg stable, x := 18\}$.*

Given $\delta > 0$, let $F_x(t) = 18 * exp(-0.1 * t)$ and $s_t = \{\neg active, \neg stable, x := F_x(t)\}$, for all $t \in [0, \delta]$. Then s_δ and F_x satisfy the conditions in Definition 3. In particular, observe that $18 * exp(-0.1 * \epsilon) < 20$ for all $\epsilon > 0$, thus no action (i.e. action degrade) can be triggered by s_ϵ. Hence, $s_\delta = \Psi(s_0, \delta)$ for all $\delta > 0$. □

Let $A_{trig}(s)$ be the set of actions triggered by a state s. Given a state s, let $\delta_{trig}(s)$ be the smallest δ such that the state $s_\delta = \Psi(s, \delta)$ is defined and s_δ triggers at least one action (i.e. $A_{trig}(\Psi(s, \delta)) \neq \emptyset$). Note that $\delta_{trig}(s)$ can be 0. In this case, these sets of actions happen at the same time in an order.

Let $\mathcal{D}$ be a domain description. Assume that s_0 is an initial state and sets A_is of exogenous actions are performed at time t_i ($0 \leq t_1 < \ldots < t_n$). At time $t \geq t_n$, the system will reach a state $s_t = \Phi^*(s_0, \{A_1 : t_1, \ldots, A_n : t_n\}, t)$ defined as follows.

Definition 4 (State transition due to action sequence). *We define $s^* = \Phi^*(s_0, \{A_1 : t_1, \ldots, A_n : t_n\}, t)$ recursively.*

- *If $n = 0$: if $t \leq \delta_{trig}(s_0)$ then $s^* = \Psi(s_0, t)$, otherwise let $s_\delta = \Psi(s_0, \delta_{trig}(s_0))$ then*
  $$s^* = \Phi^*(\Phi(s_\delta, A_{trig}(s_\delta)), t - \delta_{trig}(s_0))$$
- *For $n > 0$: Let $s_{t_n} = \Phi^*(s_0, \{A_1 : t_1, \ldots, A_{n-1} : t_{n-1}\}, t_n)$. Then*
  $$s^* = \Phi^*(\Phi(s_{t_n}, A_n \cup A_{trig}(s_{t_n})), t - t_n)$$

Example 4. *Let us consider the domain in Example 1. Let the initial cellular state be $s_0 = \{\neg active, \neg stable, x := 18\}$. Assume that the binding is induced at time 0.5, let us find out the cellular state at time 3.1 and 4.1 (Figure 1).*

During time 0 and 0.5, the level x decreases from 18, thus $x < 20$ and no action is triggered. At time 0.5, the cell reaches the state $s_{0.5} = \Psi(s_0, 0.5) = \{\neg active, \neg stable, x = 18 * exp(-0.05)\}$. The execution of action bind at time 0.5 transforms the cell from

$s_{0.5}$ *to* $s'_{0.5} = \{active, \neg stable, x = 18 * exp(-0.05)\}$. *From* $s'_{0.5}$, *the flow constraint of* x *becomes* $\dot{x} \equiv 1$. *Since action* $degrade$ *is triggered at* $x \geq 20$, *we have that* $\delta_{trig}(s'_{0.5}) = 20 - 18 * exp(-0.05) \sim 2.8779$. *At time* $0.5 + \delta_{trig}(s'_{0.5}) \sim 3.3779$, *the action* $degrade$ *is triggered, which stabilizes the level* x *of Protein.*

Hence, the state at time 3.1 *is* $\{active, \neg stable, x = 18 * exp(-0.05) + 2.6\}$ *and the state at time* 4.1 *is* $\{active, stable, x = 20\}$.

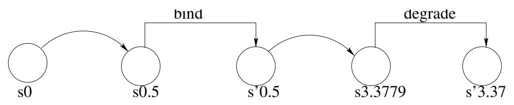

Figure 1: A trajectory due to action $bind : 0.5$

The intuition of Φ^* can be seen in light of trajectories representing how the system evolves due to an action sequence.

Definition 5 (Trajectory due to action sequence)**.** *Let* A_i*s be sets of exogenous actions, and* t_i*s be continuous time points* ($t_0 = 0$, $0 < t_i < t_{i+1}$ *for all* $i = 1, \dots, n-1$)*. Let* s_{δ_j} *and* $s'_{\delta_{j+1}}$ ($j = 1, \dots, m$) *be states. The sequence* $\tau = s_0 s_{\delta_1}(B_1, \delta_1)s'_{\delta_1} s_{\delta_2}(B_2, \delta_2) \dots (B_m, \delta_m)s'_{\delta_m}$ *is a trajectory due to the sequence of actions* $\{A_1 : t_1, \dots, A_n : t_n\}$ *if:*

- $0 \leq \delta_j \leq \delta_{j+1}$ *for* $j = 1, \dots, m$, *and* $\delta_m = t_n$.
- *There exists* $j_1, j_2, \dots, j_n = m$ *such that* $\delta_{j_i} = t_i$ *for* $i = 1, \dots, n$. *Furthermore:*
  - *if* $j = j_i$ *for some* i, *then* $B_j = A_i \cup A_{trig}(s_{\delta_{j_i}})$.
  - *else,* $B_j = A_{trig}(s_{\delta_{j_i}})$.
- *For all* $0 \leq j \leq m-1$: $s_{\delta_{j+1}} = \Psi(s'_{\delta_j}, \delta_{j+1} - \delta_j)$ *(where* $s'_0 = s_0$*) and* $s'_{\delta_{j+1}} = \Phi(s_{\delta_{j+1}}, B_{j+1})$.

The following proposition shows how a trajectory due to a sequence of actions can be computed by Φ^*.

Proposition 2. *Let* $\mathcal{D}$ *be a domain description and* s_0 *be an initial state. Let* A_i*s be sets of exogenous actions and* t_i*s be continuous time points* ($1 \leq i \leq n$)*. A trajectory due to the sequence of actions* $\{A_1 : t_1, \dots, A_n : t_n\}$*, if it exists, is unique. If* $\tau = s_0 s_{t_1}(B_1, t_1)s'_{t_1} s_{t_2}(B_2, t_2) \dots (B_n, t_n)s'_{t_n}$ *is the existing trajectory, then for all* $1 \leq i \leq n$*:*

- $s_{t_i} = \Phi^*(s_0, \{A_1 : t_1, \dots A_{i-1} : t_{i-1}\}, t_i)$.
- $s'_{t_i} = \Phi^*(s_0, \{A_1 : t_1, \dots A_i : t_i\}, t_i)$.

Entailment of queries Let $\mathcal{D}$ be a domain description, s be a state in $\mathcal{D}$ and f be a temporal goal. We define an entailment $\langle \mathcal{D}, s \rangle \models f$ constructively as follows.

- If f is a atom, then $\langle \mathcal{D}, s \rangle \models f$ iff $s \models f$.
- If $f = \neg g$, then $\langle \mathcal{D}, s \rangle \models \neg f$ iff $\langle \mathcal{D}, s \rangle \not\models g$.
- If $f = g_1 \vee g_2$, then $\langle \mathcal{D}, s \rangle \models f$ iff $\langle \mathcal{D}, s \rangle \models g_1$ or $\langle \mathcal{D}, s \rangle \models g_2$.
- If $f = g_1 \wedge g_2$, then $\langle \mathcal{D}, s \rangle \models f$ iff $\langle \mathcal{D}, s \rangle \models g_1$ and $\langle \mathcal{D}, s \rangle \models g_2$.
- If $f = \bigcirc[\delta]g$, then $\langle \mathcal{D}, s \rangle \models f$ iff $\langle \mathcal{D}, \Phi^*(s, \delta) \rangle \models g$.
- If $f = \Box g$, then $\langle \mathcal{D}, s \rangle \models f$ iff $\langle \mathcal{D}, \Phi^*(s, t) \rangle \models g$ for all $t \geq 0$.
- If $f = \Diamond g$, then $\langle \mathcal{D}, s \rangle \models f$ iff $\langle \mathcal{D}, \Phi^*(s, t) \rangle \models g$ for some $t \geq 0$.
- If $f = g_1 \cup g_2$, then $\langle \mathcal{D}, s \rangle \models f$ iff there exists $t_0 \geq 0$ such that $\langle \mathcal{D}, \Phi^*(s, t) \rangle \models g_1$ for all $t \in [0, t_0)$ and $\langle \mathcal{D}, \Phi^*(s, t_0) \rangle \models g_2$.

A set $\mathcal{O}$ of observation is complete if for all variable $v \in \mathbf{B} \cup \mathbf{X}$, there exists an observation about v in $\mathcal{O}$. In the following, we shall consider only action theories $(\mathcal{D}, \mathcal{O})$ where $\mathcal{O}$ is complete.

Let $(\mathcal{D}, \mathcal{O})$ be a theory and let $Q = f$ **after** $A_1 : t_1, A_2 : t_2, \dots A_n : t_n$. Let s_0 be the initial state corresponding to $\mathcal{O}$ and $s_{t_n} = \Phi^*(s_0, \{A_1 : t_1, \dots, A_n : t_n\}, t_n)$. The theory $(\mathcal{D}, \mathcal{O})$ entails the query Q, written as $(\mathcal{D}, \mathcal{O}) \models Q$, iff $\langle \mathcal{D}, s_{t_n} \rangle \models f$.

Given a policy P, define

$$D(P) = \{f \ \textbf{triggers} \ a \mid (\ \textbf{do} \ \ a \ \textbf{if} \ f) \in P\}$$

Let $(\mathcal{D}, \mathcal{O})$ be a theory and let $Q = f$ **given** P. The theory $(\mathcal{D}, \mathcal{O})$ entails the query Q, if $(\mathcal{D} \cup D(P), \mathcal{O}) \models f$.

Example 5. *Let* $\mathcal{D}$ *be the domain in Example 1. Let* $\mathcal{O} = \{$ **initially** $\neg active$, **initially** $\neg stable$, **initially** $x = 18\}$. *Then we have the following entailments:*

$$(\mathcal{D}, \mathcal{O}) \models \Box(x < 20 \wedge \neg active)$$
$$(\mathcal{D}, \mathcal{O}) \models \Diamond(x \geq 20) \ \textbf{after} \ bind : 0.5$$
$$(\mathcal{D}, \mathcal{O}) \models \bigcirc[6]\Diamond stable \ \textbf{given} \ \{ \ \textbf{do} \ \ bind \ \textbf{if} \ x = 17\}$$

Approximation of ALCP

In the presence of continuous processes and differential equations, we may not be able to have exact values of continuous variables at a time. As a consequence, we need approximation approaches. However, the approximation approaches should have similar properties as the original system. For example, consider two balls rolling on the ground. It is possible that they hit each other at a time point, and then change their directions. In approximation, if we just take a few time points and then calculate the positions and velocities of these two balls, we may conclude that these two balls keep rolling without hitting each other. We should prevent this from happening. In our approach, we consider that hitting occurs whenever these two balls are close enough based on the parameter θ.

Approximating transition functions Let $\mathcal{D}$ be a domain description and $\theta \in \mathbb{R}^+$. An approximation Ψ_θ of the continuous transition Ψ is defined as follows.

Let s be a state in which no action is triggered. The state $s' = \Psi_\theta(s, 1)$ is such that

- For all Boolean variable $f \in \mathbf{B}$: $s'(f) = s(f)$.
- For all variable $x \in \mathbf{X}$, if a flow constraint $\dot{x} \equiv \psi$ is entailed by s, then $s'(x) = s(x) + \theta * s(\psi)$. Otherwise, $s'(x) = s(x)$.

And we define $\Psi_\theta(s, n) = \Psi_\theta(\Psi_\theta(s, n-1), 1)$, for all $n > 1$. Based on Φ and Ψ_θ, we can define the approximation Φ^*_θ of Φ^*.

Definition 6. *We define $s_k = \Phi^*_\theta(s_0, \{A_1 : k_1, \ldots, A_n : k_n\}, k)$ recursively, where A_is are sets of exogenous actions and k_i and k are integers, $0 \leq k_1 < \ldots < k_n \leq k$.*

- *If $n = 0$: if $k \leq \lfloor \delta_{trig}(s_0)/\theta \rfloor$ then $s_k = \Psi_\theta(s_0, k)$, otherwise let $s' = \Psi_\theta(s_0, \lfloor \delta_{trig}(s_0)/\theta \rfloor)$ then*
  $s_k = \Phi^*_\theta(\Phi(s'), A_{trig}(s')), k - \lfloor \delta_{trig}(s_0)/\theta \rfloor)$
- *For $n > 0$: Let $s_{k_n} = \Phi^*_\theta(s_0, \{A_1 : k_1, \ldots, A_{n-1} : k_{n-1}\}, k_n)$. Then*
  $s_t = \Phi^*_\theta(\Phi(s_{k_n}, A_n \cup A_{trig}(s_{k_n})), k - k_n)$

Approximating the entailment Let $\mathcal{D}$ be a domain description, s be a state in $\mathcal{D}$ and f be a temporal goal. We define an entailment $\langle \mathcal{D}, s \rangle \models_\theta f$ constructively, similarly to the entailment $\langle \mathcal{D}, s \rangle \models f$. We shall present only a part the definition.

- If f is an atom, then $\langle \mathcal{D}, s \rangle \models_\theta f$ iff $s \models_\theta f$.
- If $f = g_1 \vee g_2$, then $\langle \mathcal{D}, s \rangle \models_\theta f$ iff $\langle \mathcal{D}, s \rangle \models_\theta g_1$ or $\langle \mathcal{D}, s \rangle \models_\theta g_2$.
- If $f = \bigcirc[\delta]g$, then $\langle \mathcal{D}, s \rangle \models_\theta f$ iff $\langle \mathcal{D}, \Phi^*_\theta(s, \delta) \rangle \models_\theta g$.
- If $f = \Box g$, then $\langle \mathcal{D}, s \rangle \models_\theta f$ iff $\langle \mathcal{D}, \Phi^*_\theta(s, k) \rangle \models_\theta g$ for all integer $k \geq 0$.

Let $(\mathcal{D}, \mathcal{O})$ be a theory and let $Q = f$ **after** $A_1 : k_1, A_2 : k_2, \ldots A_n : k_n$. Let s_0 be the initial state corresponding to $\mathcal{O}$ and $s_k = \Phi^*(s_0, \{A_1 : k_1, \ldots, A_n : k_n\}, k)$. The theory $(\mathcal{D}, \mathcal{O})$ $\theta-$entails the query Q, written as $(\mathcal{D}, \mathcal{O}) \models_\theta Q$, if $\langle \mathcal{D}, s_k \rangle \models_\theta f$.

Approximating domain description Let $\mathcal{D}$ be a domain description and $\epsilon \in \mathbb{R}^+$. An approximation $\mathcal{D}[\epsilon]$ of $\mathcal{D}$ is obtained from $\mathcal{D}$ as follows. In the rules of $\mathcal{D}$:

- replace any atom of the form $E_1 \leq E_2$ with $E_1 < E_2 + \epsilon$;
- replace any atom of the form $E_1 = E_2$ with $(E_1 - \epsilon < E_2) \wedge (E_2 < E_1 + \epsilon)$;

Proposition 3. *Let $\mathcal{D}$ be a domain description and $\mathcal{A}_Q$ be a set of atoms. Assume that the functions in* **F** *all have bounded and continuous first derivative. Let A_is be sets of exogenous actions ($1 \leq i \leq n$) and $0 \leq t_1 < t_2 < \ldots < t_n$. Let s_0 be any initial state and $\mathcal{O}$ be the observations about s_0. Then for any fixed $\epsilon > 0$ there exists $\Theta > 0$ such that for all $\theta \in (0, \Theta)$*

$$(\mathcal{D}, \mathcal{O}) \models Q \Leftrightarrow (\mathcal{D}[\epsilon], \mathcal{O}) \models_\theta Q_\theta$$

where $Q = f$ **after** *$A_1 : t_1, \ldots, A_n : t_n$ with f being built of atoms from $\mathcal{A}_Q$, and $Q_\theta = f$* **after** *$A_1 : \lfloor t_1/\theta \rfloor, \ldots, A_n : \lfloor t_n/\theta \rfloor$.*

Reasoning about Cellular Processes

The repressilator network We shall study a synthetic cellular network termed the repressilator in *Escherichia coli* (Elowitz & Leibler 2000). It contains a negative feedback loop of three proteins: *lacI* from E. coli, *tetR* from the tetracycline-resistance transposon Tn10, and cI from λ phage. *LacI* represses *tetR*, *tetR* represses *cI* and *cI* represses *LacI* (Figure 2). Let us denote n_is ($i = 0, 1, 2$) the names *lacI*, *tetR* and *cI*, respectively. Let p_i and m_i be the protein concentration and mRNA concentration of n_is. We assume

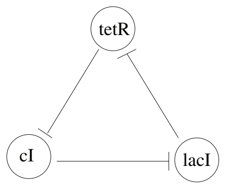

Figure 2: The repressilator network

the same set of parameters as in (Elowitz & Leibler 2000), where β denotes the ratio of protein decay rate to the mRNA decay rate, n is a Hill coefficient and the number of protein copies produced from a given promoter type during continuous growth is α_0 in the presence of saturating amounts of repressor, while $\alpha + \alpha_0$ in its absence. We assume $n = 2$, which is a special case of the model proposed in (Elowitz & Leibler 2000). Before the network reaches the steady state, the protein concentrations p_is change according to the differential equations ($i = 0, 1, 2$):

$$\dot{p_i} = -\beta(p_i - m_i)$$

When the binding between transcription factors and DNA is non-cooperative, the changes in mRNA concentrations are described by the differential equations

$$\dot{m_i} = -m_i + \frac{\alpha}{1 + p_{i-1}} + \alpha_0$$

where $i = 0, 1, 2$ and $i - 1$ is computed in module 3.

When the binding is cooperative due to some enzymes, we have the other set of differential equations ($i = 0, 1, 2$)

$$\dot{m_i} = -m_i + \frac{\alpha}{1 + p^2_{i-1}} + \alpha_0$$

Representing the repressilator Let us consider the alphabet containing Boolean variables *coop*, $steady$ and continuous variables p_i and m_i ($i = 0, 1, 2$). The variable *coop* represents that the binding is cooperative, while $steady$ represents the steady state. The continuous variables represent the concentrations described previously. Let **F** be the set of all polynomial functions of the continuous variables. Let $stabilize$ be a triggered action, which is triggered when the concentration of a protein is equal to its mRNA concentration. Let $add(enzymes)$ be a exogenous action, which introduce enzymes catalyzing the cooperative binding.

The repressilator is represented by the following rules, where $i = 0, 1, 2$:

$$\begin{aligned}
\neg coop \ \textbf{constrains} \ & \dot{m_i} \equiv -m_i + \frac{\alpha}{1 + p_{i-1}} + \alpha_0 \\
coop \ \textbf{constrains} \ & \dot{m_i} \equiv -m_i + \frac{\alpha}{1 + p^2_{i-1}} + \alpha_0 \\
\neg steady \ \textbf{constrains} \ & \dot{p_i} \equiv -\beta(p_i - m_i) \\
add(enzymes) \ \textbf{causes} \ & coop \\
stabilize \ \textbf{causes} \ & steady \\
p_i = m_i \ \textbf{triggers} \ & stabilize
\end{aligned}$$

Approximation results We implemented the approximation and evaluated an example query $Q = \Diamond\Box \ steady$. The

α	α_0	β	ϵ	θ	convergence steps	C	p_0	p_1	p_2
10	0	5	10^{-5}	0.01	≥ 20000	0.1	1.99990	2.00028	1.99983
10	0	5	10^{-5}	0.01	≥ 20000	100	1.99993	1.99998	2.00008
10	0	20	10^{-5}	0.01	4857	0.1	1.99989	2.00016	1.99992
10	0	20	10^{-5}	0.01	3921	100	2.00010	1.99984	2.00005
10^3	1	5	10^{-5}	0.01	≥ 20000	0.1	6.03828	17.68638	12.23005
10^3	1	5	10^{-5}	0.01	≥ 20000	100	15.0747	6.92404	12.90286
10^3	1	120	10^{-5}	0.01	9945	0.1	10.31323	10.31308	10.31443
10^3	1	120	10^{-5}	0.01	7743	100	10.31270	10.31398	10.31406

Table 1: Experiment with Different Parameters

initial state is $s_0 = \{coop, \neg steady, p_0 = p_1 = p_2 = 0, m_0 = m_1 = m_2 = C\}$, where C is a constant. Let $\mathcal{O}$ be the observations about s_0.

The approximation result is shown in Table 1. The parameters C corresponds to the initial concentration of mRNA, p_0, p_1, p_2 are the protein concentrations of their respective n_i, while θ and ϵ have the same meanings as in the previous section. That is, θ is the constant in the approximation Ψ_θ of transition function Ψ and ϵ is the constant used in the approximation $\mathcal{D}[\epsilon]$ of the domain $\mathcal{D}$. The third line of the table shows that the formula $\Box\ steady$ becomes true at the 4857^{th} approximate state: $\Phi^*_{0.01}(s_0, 4857) \models \Box\ steady$ (with respect to domain $\mathcal{D}[0.00001]$). Thus $(\mathcal{D}[0.00001], \mathcal{O}) \models_{0.01} \Diamond\Box\ steady$. Notice that the results correspond to the ones presented by (Elowitz & Leibler 2000) also. Experiments with various settings of θ suggests that $\theta < 1$ is good for $\epsilon < 0.001$.

Related Works

The language **ALCP** can be distinguished from existing works in action languages based on various important aspects: the intuitiveness of high-level action language $\mathcal{A}$, transition function based and hybrid automata based semantics (Henzinger 1996), reasoning about continuous changes, triggered actions, and available support for efficient implementation of reasoning. We now discuss a few representative related works.

Two extensions of the language $\mathcal{A}$ have been proposed for reasoning about continuous changes (Baral, Son, & Tuan 2002; Chintabathina, Gelfond, & Watson 2005). In (Baral, Son, & Tuan 2002), there is no easy way to represent triggers, since actions are associated with durations and delayed effects. In the process description language of (Chintabathina, Gelfond, & Watson 2005), states are associated with predefined durations. Due to this notion of states, it would be difficult to extend the process description language with triggers. The reason is that we usually do not know the execution of the trigger unless when we know that its conditions are satisfied, which we can only be known exactly by solving (usually complex) sets of functions and/or differential equations. Thus the duration of s that lead to a trigger is not known beforehand.

Some special type of triggers can be modeled in the planning language PDDL2.1 (Fox & Long 2003) by a combination of conditional effects and duration inequalities. Yet planning languages are not aimed to support issues of reasoning about actions such as counterfactual reasoning or explanation.

There exists a large body of research in hybrid (discrete-continuous) systems, for example (Sandewall 1989; Kolen & Zhao 1996; Mosterman, Zhao, & Biswas 1997; R. Alur, T.A. Henzinger, & G.J. Pappas 2000; McIlraith *et al.* 2000; F. Zhao *et al.* 2001; Narasimhan & Biswas 2002; 2003; K.Nakamura & Fusaoka 2004). Nevertheless, most of works in hybrid systems have not been aimed at reasoning about actions and triggers and their effects. The focus of this work is to provide an action language for reasoning about continuous processes in the cell.

Discussion and Future Work

In this paper, we proposed a new language for representing and reasoning about a cell as a hybrid system that also takes care of both exogenous actions and triggered actions. Our language is based on the hybrid automata semantics. It can be easily extended to model actions that have delayed effects, concurrent actions, actions with durations etc. Let us take the modeling of delayed effects as an example. Whenever an action with delayed effects is executed, we set up a trigger for each of its delayed effects. For example, if we know that one of the effects of an action is delayed for 20 minutes, we may have the following rules:

$$\begin{aligned} a\ &\textbf{causes}\ y = 20 \\ a\ &\textbf{causes}\ g \\ g\ &\textbf{constrains}\ \dot{y} \equiv -1 \\ y = 0\ &\textbf{triggers}\ delayed\_action \\ delayed\_action\ &\textbf{causes}\ delayed\_effects \end{aligned}$$

In the above program, a Boolean variable g, a real variable y, and a trigger $delayed\_action$ are all auxiliary variables that do not occur in other parts of the program.

Action Language approach of dealing with transition systems considers the translation of discrete states in a step manner but not a time manner. In this approach, a few steps may belong to the same time point. In a system with continuous properties, the continuous variables may change their values at different time, so it is not sufficient to consider only steps. We should also take into account real time. Keep in mind that our language can model hybrid systems that have both continuous properties and discrete properties. In our language, states are distinguished based on real time points. Thus a variable can changes its value for multiple times at one particular time point. For example, consider the following domain description:

$$\begin{aligned} f \wedge (v = 2) \ &\textbf{triggers}\ a \\ a\ &\textbf{causes}\ \neg f \\ \neg f\ &\textbf{triggers}\ b \\ b\ &\textbf{causes}\ g \end{aligned}$$

We have a state $s = \{f, \neg g, v = 2\}$. We know $s_1 = \{\neg f, \neg g, v = 2\} = \Phi(s, a)$ and $s_2 = \{f, g, v = 2\} = \Phi(s_1, b)$. It can be implied that two triggers a and b are executed one after another but they happened at the same time point. In this paper and in reality, we assume that there are at most finite number of transitions at one time point.

If there are multiple different states at a time point, a variable may have a few different values at a time. This is a common property of systems that combines continuous and discrete variables. In terms of future work, we need to distinguish steps from time points. A better definition of trajectories, and consequently richer query languages are needed for such systems. We are also interested in having an efficient implementation of the reasoning mechanisms, in which we plan to take advantage of state-of-the-art methods from hybrid automata and symbolic model checking.

References

Alur, R.; Belta, C.; Ivanicic, F.; Kumar, V.; Mintz, M.; Pappas, G. J.; Rubin, H.; and Schug, J. 2001. Hybrid modeling and simulation of biomolecular networks. *Hybrid Systems: Computation and Control. LNCS* 2034:19–32.

Antoniotti, M.; Park, F.; Policriti, A.; Ugel, N.; and Mishra, B. 2003a. Foundations of a query and simulation system for the modeling of biochemical and biological processes. In *Pacific Symposium on Biocomputing 2003 (PSB 2003)*, 116127.

Antoniotti, M.; Mishra, B.; Piazza, C.; Policriti, A.; and Simeoni, M. 2003b. Modeling cellular behavior with hybrid automata: Bisimulation and collapsing. In *Computational Methods in Systems Biology (CMSB 2003)*, 57–74.

Baral, C.; Floriano, L.; Hardesty, A.; Morales, D.; Nogueira, M.; and Son, T. C. 1998. From theory to practice: The utep robot in the aaai 96 aaai 97 robot contests. In *Agents 1998*, number 32-38.

Baral, C.; Son, T.; and Tuan, L. 2002. A transition function based characterization of actions with delayed and continuous effects. In *Proc. of KR'02*, 291–302.

Batt, G.; Ropers, D.; de Jong, H.; Geiselmann, J.; Mateescu, R.; Page, M.; and Schneider, D. 2005. Analysis and verification of qualitative models of genetic regulatory networks: A model-checking approach. In *IJCAI*, 370–375.

Calder, M.; Hillston, J.; and Gilmore., S. 2004. Modelling the influence of rkip on the erk signalling pathway using the stochastic process algebra pepa. In *Proceedings of BioCONCUR 2004, Electronic Notes in Theoretical Computer Science*.

Chabrier, N.; Chiaverini, M.; Danos, V.; Fages, F.; and Schachter, V. 2004. Modeling and querying biomolecular interaction networks. *Theoretical Computer Science* 325(1):25–44.

Chintabathina, S.; Gelfond, M.; and Watson, R. 2005. Modeling hybrid domains using process description language. In *ASP-05*.

Cimatti, A.; Clarke, E.; Giunchiglia, E.; Giunchiglia, F.; Pistore, M.; Roveri, M.; Sebastiani, R.; and Tacchella, A. 2002. NuSMV version 2: An opensource tool for symbolic model checking. In *CAV 2002*, volume 2404 of *LNCS*. Springer.

Eker, S.; Knapp, M.; Laderoute, K.; Lincoln, P.; Meseguer, J.; and Sonmez, K. 2002. Pathway logic: symbolic analysis of biological signaling. In *Pacific Symposium on Biocomputing 2002 (PSB 2002)*, 400412.

Elowitz, M., and Leibler, S. 2000. A synthetic oscillatory network of transcriptional regulators. *Nature* 403:335–338.

Emerson, E. A., and Clarke, E. 1982. Using branching time temporal logic to synthesize synchronization skeletons. In *Science of Computer programming, vol 2*. 241–266.

F. Zhao; X. Koutsoukos; H. Haussecker; J. Reich; P. Cheung; and C. Picardi. 2001. Distributed monitoring of hybrid systems: a model-directed approach. In *Proc. IJCAI*, 557–564.

Fages, F.; Soliman, S.; and Chabrier-Rivier, N. Modelling and querying interaction networks in the biochemical abstract machine BIOCHAM. *Jour. Biol. Phys. Chem.* 4(2):64–73.

Fox, M., and Long, D. 2003. Pddl2.1: An extension to pddl for expressing temporal planning domains. *JAIR* 20:61–124.

Funahashi, A.; Tanimura, N.; Morohashi, M.; and Kitano, H. 2003. Celldesigner: a process diagram editor for gene-regulatory and biochemical networks. *BIOSILICO* 1(5):159–162.

Fusaoka, A. 1996. Situation calculus on a dense flow of time. In *AAAI/IAAI, Vol. 1*, 633–638.

Gerard J. Holzmann. 2003. *The SPIN Model Checker : Primer and Reference Manual*. Addison-Wesley.

Grosskreutz, H., and Lakemeyer, G. 2000. cc-Golog: Towards more realistic logic-based robot controllers. In *AAAI/IAAI*, 476–482.

Heidtke, K., and Schulze-Kremer, S. 1998. Design and implementation of a qualitative simulation model of lambda phage infection. *Bioinformatics* 14:81–91.

Henzinger, T. A.; Ho, P.-H.; and Wong-Toi, H. 1997. Hytech: A model checker for hybrid systems. *STTT* 1(1-2):110–122.

Henzinger, T. A. 1996. The theory of hybrid automata. In *LICS-96*, 278–292.

Herrmann, C. S., and Thielscher, M. 1996. Reasoning about continuous processes. In *AAAI-96*.

K.Nakamura, and Fusaoka, A. 2004. On a description and reasoning about hybrid system. In *Proc. 17th IEA/AIE*, 274–283.

Kolen, J. F., and Zhao, F. 1996. A computational analysis of the reachability problem for a class of hybrid dynamical systems. In *Hybrid Systems*, 215–227.

McIlraith, S. A.; Biswas, G.; Clancy, D.; and Gupta, V. 2000. Hybrid systems diagnosis. In *HSCC*, 282–295.

Miller, R., and Shanahan, M. 1996. Reasoning about discontinuities in the event calculus. In *Proc. KR*, 63–74.

Mosterman, P. J.; Zhao, F.; and Biswas, G. 1997. Sliding mode model semantics and simulation for hybrid systems. In *Hybrid Systems*, 218–237.

Narasimhan, S., and Biswas, G. 2002. *An Approach to Model-Based Diagnosis of Hybrid Systems*. chapter Lecture Notes in Computer Science.

Narasimhan, S., and Biswas, G. 2003. Model-based diagnosis of hybrid systems. In *IJCAI*, 376–381.

Peleg, M.; Yeh, I.; and Altman, R. B. 2002. Modelling biological processes using workflow and petri net models. *Bioinformatics* 18(6):825–837.

Peres, S., and Comet, J.-P. 2003. Contribution of computational tree logic to biological regulatory networks: Example from pseudomonas aeruginosa. In *CMSB*, 47–56.

Pinto, J. 1998. Integrating discrete and continuous change in a logical framework. *Computational Intelligence* 39–88.

R. Alur; T.A. Henzinger; and G.J. Pappas. 2000. Discrete abstractions of hybrid systems. In *Proc. IEEE*, 971-84.

R. Mateescu, and H. Garavel. 1998. XTL: A meta-language and tool for temporal logic model- checking. In *Proc. Workshop on Software Tools for Technology Transfer*, 33-42.

Reddy, V. N.; Liebman, M. N.; and Mavrovouniotis, M. L. 1996. Qualitative analysis of biochemical reaction systems. *Computers in Biology and Medicine* 26:9–24.

Regev, A.; Silverman, W.; and Shapiro, E. 2001. Representation and simulation of biochemical processes using π-calculus process algebra. In *Proc. PSB*, 459–470.

Reiter, R. 1996. Natural actions, concurrency and continuous time in the situation calculus. In *Proc. KR*, 2–13.

Reiter, R. 2001a. *Knowledge in action : logical foundations for specifying and implementing dynamical systems*. MIT Press.

Reiter, R. 2001b. *Time, concurrency and processes*. MIT. chapter Knowledge in action: Logical Fundations for specifying and implementing dynamical systems, 149–183.

Sandewall, E. 1989. Combining logic and differential equations for describing real-world systems. In *Proceedings of KR*, 412–420.

Shanahan, M. 1990. Representing continuous change in the event calculus. In *Proc. ECAI*, 598–603.

Shanahan, M. P. 1998. A logical account of the common sense informatic situation for a mobile robot. *Electronic Transactions on Artificial Intelligence* 2:69–104.

Talcott, C.; Eker, S.; Knapp, M.; Lincoln, P.; and Laderoute, K. 2004. Pathway logic modeling of protein functional domains in signal transduction. In *Proc. PSB*, 568–580.

Thielscher, M. 2000a. Representing the knowledge of a robot. In *Proc. KR*, 109–120.

Thielscher, M. 2000b. Modeling actions with ramifications in nondeterministic, concurrent, and continuous domains - and a case study. In *Proceedings of AAAI/IAAI 2000*, 497–502.

Thielscher, M. 2001. The concurrent, continuous fluent calculus. *Studia Logica* 67(3):315–331.

Tran, N., and Baral, C. 2004. Reasoning about triggered actions in AnsProlog and its application to molecular interactions in cells. In *KR*, 554–563.

Tran, N., and Baral, C. 2005. Issues in reasoning about interaction networks in cells: Necessity of event ordering knowledge. In *Proc. AAAI*.