Translating Stochastic CLS into Maude
(ONGOING WORK)

Thomas Anung Basuki[1] Antonio Cerone[1] Paolo Milazzo[2]

1. Int. Institute for Software Technology, United Nations Univeristy, Macau SAR, China
2. Dipartimento di Informatica, Università di Pisa, Italy

Iași – September, 2008

Introduction

The Calculus of Looping Sequences (CLS) is a formalism for the description of biological systems

- Stochastic CLS is the stochastic extension of CLS
- A simulator based on Stochastic CLS has been developed

Our original aim was to apply model checking to Stochastic CLS models

We wanted to use Probabilistic Maude (PMaude) as a model checker

- unfortunately, the model checking module of PMaude seems not to be available...

Consequently,

- we have chosen Real-Time Maude (Maude with a notion of time)
- we have adapted Gillespie's stochastic simulation algorithm in order to be used in Real-Time Maude
- we have used Real-Time Maude analysis tool to verify properties on the results of a number of simulations (statistical model checking)

Outline of the talk

1. Introduction

2. The Calculus of Looping Sequences (CLS)
   - Definition of CLS
   - The *lac* operon in CLS
   - Stochastic CLS

3. (Statistical) model checking of Stochastic CLS models
   - Choosing a model checker
   - Translation of Stochastic CLS into Real-Time Maude
   - Analysis examples

The Calculus of Looping Sequences (CLS)

We assume an alphabet $\mathcal{E}$. **Terms** T and **Sequences** S of CLS are given by the following grammar:

$$\begin{array}{lcl} T & ::= & S \mid (S)^L \rfloor T \mid T \mid T \\ S & ::= & \epsilon \mid a \mid S \cdot S \end{array}$$

where a is a generic element of $\mathcal{E}$, and ϵ is the empty sequence.

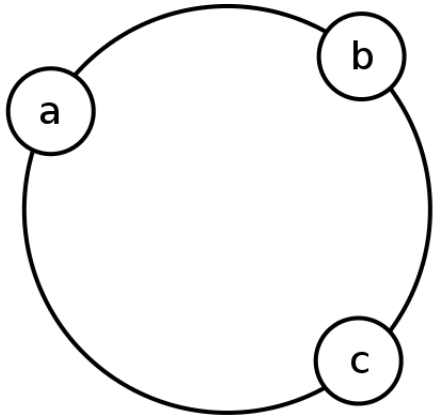
The operators are:

$S \cdot S$	:	Sequencing
$(S)^L$	:	Looping (S is closed and it can rotate)
$T_1 \rfloor T_2$	:	Containment (T_1 contains T_2)
$T \mid T$	:	Parallel composition (juxtaposition)

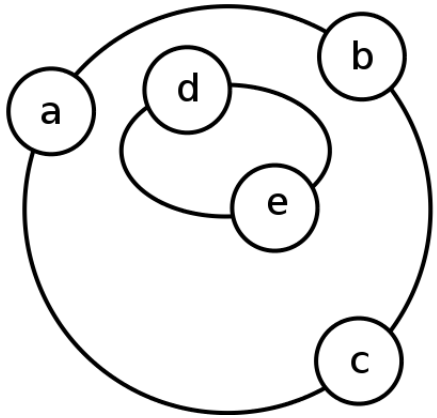
Actually, looping and containment form a single binary operator $(S)^L \rfloor T$.

Examples of Terms

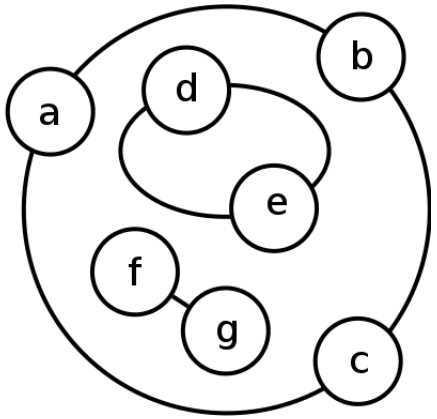
(i)



(ii)



(iii)



$$(i) \qquad \left(a \cdot b \cdot c\right)^{L} \rfloor \epsilon$$

$$(ii) \qquad \left(a \cdot b \cdot c\right)^{L} \rfloor \left(d \cdot e\right)^{L} \rfloor \epsilon$$

$$(iii) \qquad \left(a \cdot b \cdot c\right)^{L} \rfloor \left(f \cdot g \mid \left(d \cdot e\right)^{L} \rfloor \epsilon\right)$$

Structural Congruence

The **Structural Congruence** relations $\equiv_S$ and $\equiv_T$ are the least congruence relations on sequences and on terms, respectively, satisfying the following rules:

$$S_1 \cdot (S_2 \cdot S_3) \equiv_S (S_1 \cdot S_2) \cdot S_3 \qquad S \cdot \epsilon \equiv_S \epsilon \cdot S \equiv_S S$$

$$T_1 \mid T_2 \equiv_T T_2 \mid T_1 \qquad T_1 \mid (T_2 \mid T_3) \equiv_T (T_1 \mid T_2) \mid T_3$$

$$T \mid \epsilon \equiv_T T \quad \left(\epsilon\right)^L \rfloor \epsilon \equiv_T \epsilon \quad \left(S_1 \cdot S_2\right)^L \rfloor T \equiv_T \left(S_2 \cdot S_1\right)^L \rfloor T$$

We write $\equiv$ for $\equiv_T$.

CLS Patterns

Let us consider variables of three kinds:

- term variables $(X, Y, Z, \ldots)$
- sequence variables $(\widetilde{x}, \widetilde{y}, \widetilde{z}, \ldots)$
- element variables $(x, y, z, \ldots)$

Patterns P and **Sequence Patterns** SP of CLS extend CLS terms and sequences with variables:

$$\begin{array}{rcl} P & ::= & SP \mid (SP)^{L} \rfloor P \mid P \mid P \mid X \\ SP & ::= & \epsilon \mid a \mid SP \cdot SP \mid x \mid \widetilde{x} \end{array}$$

where a is a generic element of $\mathcal{E}$, ϵ is the empty sequence, and $x, \widetilde{x}$ and X are generic element, sequence and term variables

The structural congruence relation $\equiv$ extends trivially to patterns

Rewrite Rules

$P\sigma$ denotes the term obtained by replacing any variable in T with the corresponding term, sequence or element.

Σ is the set of all possible instantiations σ

A **Rewrite Rule** is a pair (P, P'), denoted $P \mapsto P'$, where:

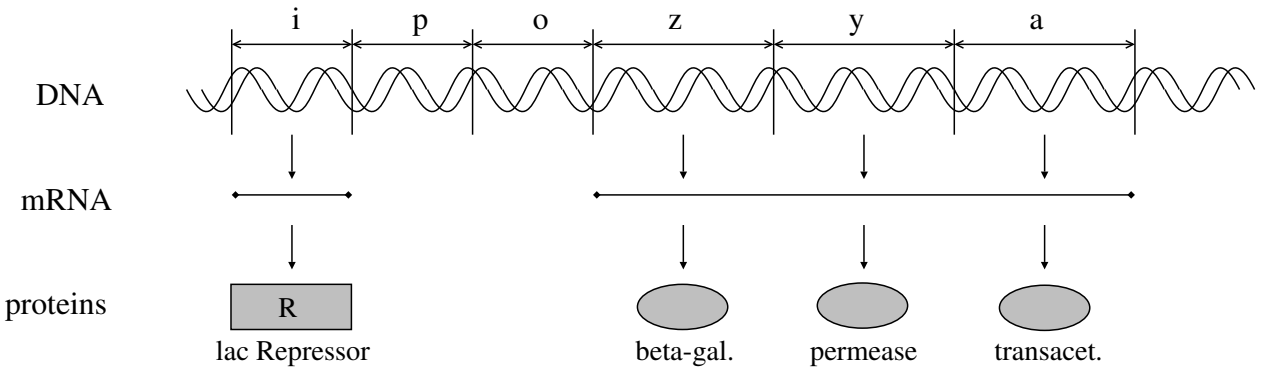
- P, P' are patterns
- variables in P' are a subset of those in P

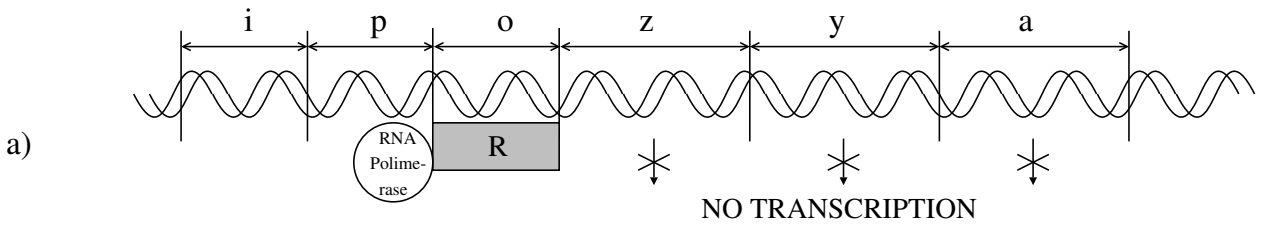
A rule $P \mapsto P'$ can be applied to all terms $P\sigma$.

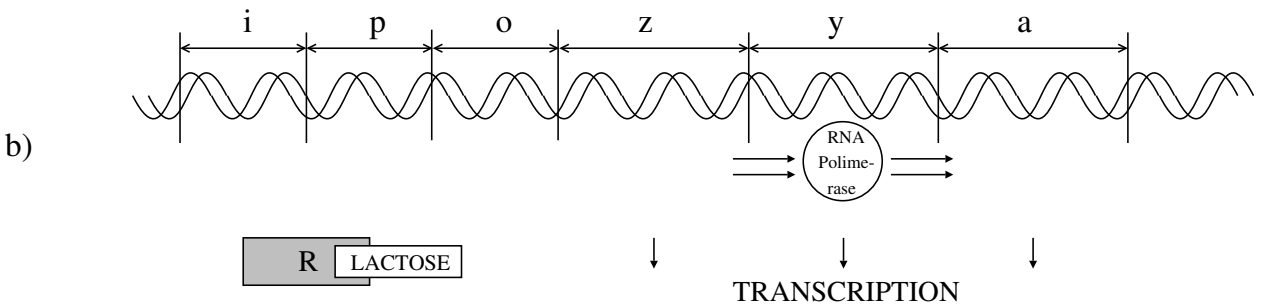
Example: $a \cdot x \cdot a \mapsto b \cdot x \cdot b$

- can be applied to $a \cdot c \cdot a$ (producing $b \cdot c \cdot b$)
- cannot be applied to $a \cdot c \cdot c \cdot a$

CLS modeling examples: the *lac* operon (1)







CLS modeling examples: the *lac* operon (2)

$$Ecoli \;::=\; \left(m\right)^{L} \rfloor \left(lacI \cdot lacP \cdot lacO \cdot lacZ \cdot lacY \cdot lacA \mid polym\right)$$

Rules for DNA transcription/translation:

$$lacI \cdot \widetilde{x} \;\mapsto\; lacI' \cdot \widetilde{x} \mid repr \qquad \text{(R1)}$$

$$polym \mid \widetilde{x} \cdot lacP \cdot \widetilde{y} \;\mapsto\; \widetilde{x} \cdot PP \cdot \widetilde{y} \qquad \text{(R2)}$$

$$\widetilde{x} \cdot PP \cdot lacO \cdot \widetilde{y} \;\mapsto\; \widetilde{x} \cdot lacP \cdot PO \cdot \widetilde{y} \qquad \text{(R3)}$$

$$\widetilde{x} \cdot PO \cdot lacZ \cdot \widetilde{y} \;\mapsto\; \widetilde{x} \cdot lacO \cdot PZ \cdot \widetilde{y} \qquad \text{(R4)}$$

$$\widetilde{x} \cdot PZ \cdot lacY \cdot \widetilde{y} \;\mapsto\; \widetilde{x} \cdot lacZ \cdot PY \cdot \widetilde{y} \mid betagal \qquad \text{(R5)}$$

$$\widetilde{x} \cdot PY \cdot lacA \;\mapsto\; \widetilde{x} \cdot lacY \cdot PA \mid perm \qquad \text{(R6)}$$

$$\widetilde{x} \cdot PA \;\mapsto\; \widetilde{x} \cdot lacA \mid transac \mid polym \qquad \text{(R7)}$$

CLS modeling examples: the *lac* operon (3)

$$Ecoli \;::=\; \left(m\right)^{L} \rfloor \left(lacI \cdot lacP \cdot lacO \cdot lacZ \cdot lacY \cdot lacA \mid polym\right)$$

Rules to describe the binding of the lac Repressor to gene o, and what happens when lactose is present in the environment of the bacterium:

$$repr \mid \widetilde{x} \cdot lacO \cdot \widetilde{y} \;\mapsto\; \widetilde{x} \cdot RO \cdot \widetilde{y} \qquad \text{(R8)}$$

$$LACT \mid \left(m \cdot \widetilde{x}\right)^{L} \rfloor X \;\mapsto\; \left(m \cdot \widetilde{x}\right)^{L} \rfloor \left(X \mid LACT\right) \qquad \text{(R9)}$$

$$\widetilde{x} \cdot RO \cdot \widetilde{y} \mid LACT \;\mapsto\; \widetilde{x} \cdot lacO \cdot \widetilde{y} \mid RLACT \qquad \text{(R10)}$$

$$\left(\widetilde{x}\right)^{L} \rfloor \left(perm \mid X\right) \;\mapsto\; \left(perm \cdot \widetilde{x}\right)^{L} \rfloor X \qquad \text{(R11)}$$

$$LACT \mid \left(perm \cdot \widetilde{x}\right)^{L} \rfloor X \;\mapsto\; \left(perm \cdot \widetilde{x}\right)^{L} \rfloor \left(LACT \mid X\right) \qquad \text{(R12)}$$

$$betagal \mid LACT \;\mapsto\; betagal \mid GLU \mid GAL \qquad \text{(R13)}$$

CLS modeling examples: the *lac* operon (4)

$$Ecoli \;::=\; \left(m\right)^{L} \rfloor \left(lacI \cdot lacP \cdot lacO \cdot lacZ \cdot lacY \cdot lacA \mid polym\right)$$

Example:

$$\begin{aligned}
& Ecoli|LACT|LACT \\
\rightarrow^{*} & \left(m\right)^{L} \rfloor \left(lacI' \cdot lacP \cdot lacO \cdot lacZ \cdot lacY \cdot lacA \mid polym \mid repr\right)|LACT|LACT \\
\rightarrow^{*} & \left(m\right)^{L} \rfloor \left(lacI' \cdot lacP \cdot RO \cdot lacZ \cdot lacY \cdot lacA \mid polym\right)|LACT|LACT \\
\rightarrow^{*} & \left(m\right)^{L} \rfloor \left(lacI' \cdot lacP \cdot lacO \cdot lacZ \cdot lacY \cdot lacA|polym|RLACT\right)|LACT \\
\rightarrow^{*} & \left(perm \cdot m\right)^{L} \rfloor \left(lacI'{-}A|betagal|transac|polym|RLACT\right)|LACT \\
\rightarrow^{*} & \left(perm \cdot m\right)^{L} \rfloor \left(lacI'{-}A|betagal|transac|polym|RLACT|GLU|GAL\right)
\end{aligned}$$

Outline of the talk

1. Introduction
2. The Calculus of Looping Sequences (CLS)
   - Definition of CLS
   - The *lac* operon in CLS
   - Stochastic CLS
3. (Statistical) model checking of Stochastic CLS models
   - Choosing a model checker
   - Translation of Stochastic CLS into Real-Time Maude
   - Analysis examples

Background: the kinetics of chemical reactions

Usual notation for chemical reactions:

$$\ell_1 S_1 + \ldots + \ell_\rho S_\rho \underset{k_{-1}}{\overset{k}{\rightleftharpoons}} \ell'_1 P_1 + \ldots + \ell'_\gamma P_\gamma$$

where:

- S_i, P_i are molecules (reactants)
- ℓ_i, ℓ'_i are stoichiometric coefficients
- k, k_{-1} are the kinetic constants

The kinetics is described by the *law of mass action*:

$$\frac{d[P_i]}{dt} = \ell'_i \underbrace{k[S_1]^{\ell_1} \cdots [S_\rho]^{\ell_\rho}}_{\textit{reaction rate}} - \ell'_i \underbrace{k_{-1}[P_1]^{\ell'_1} \cdots [P_\gamma]^{\ell'_\gamma}}_{\textit{reaction rate}}$$

Background: Gillespie's simulation algorithm

- represents a chemical solution as a multiset of molecules
- computes the reaction rate a_μ by multiplying the kinetic constant by the number of possible combinations of reactants

Example: chemical solution with X_1 molecules S_1 and X_2 molecules S_2

reaction $R_1 : S_1 + S_2 \rightarrow 2S_1$ rate $a_1 = \binom{X_1}{1}\binom{X_2}{1}k_1 = X_1 X_2 k_1$

reaction $R_2 : 2S_1 \rightarrow S_1 + S_2$ rate $a_2 = \binom{X_1}{2}k_2 = \frac{X_1(X_1-1)}{2}k_2$

Given a set of reactions $\{R_1, \ldots R_M\}$ and a current time t

- The time $t + \tau$ at which the next reaction will occur is randomly chosen with τ exponentially distributed with parameter $\sum_{\nu=1}^{M} a_\nu$;
- The reaction R_μ that has to occur at time $t + \tau$ is randomly chosen with probability $\frac{a_\mu}{\sum_{\nu=1}^{M} a_\nu}$.

At each step t is incremented by τ and the chemical solution is updated.

Stochastic CLS

Stochastic CLS incorporates Gillespie's stochastic framework into the semantics of CLS

- Rewrite rules are enriched with a kinetic constant

What is a reactant combination in Stochastic CLS?

- A *reactant combination* is an occurence (up to $\equiv$) of a left hand side of a rewrite rule

Two definitions of the semantics of Stochastic CLS exist

- The first computes the number of reactant combinations by adding unique labels to the alphabet symbols in the term
- The second computes the number of reactant combinations compositionally

The *occ* function

For the sake of the translation into Maude we have defined a recursive function $occ(T, T')$ that gives the number of combinations of reactants T in the term T'.

$$occ(T, \epsilon) = 0$$

$$occ(S, E \cdot S_1 | T) = \begin{cases} 1 + occ(S, S_1) + occ(S, T) & \text{if } S \text{ is a prefix of } E \cdot S_1 \\ occ(S, S_1) + occ(S, T) & \text{otherwise} \end{cases}$$

$$occ(S, (S_1)^L \rfloor (T) \mid T_1) = occ'(S, S_1) + occ(S, T) + occ(S, T_1)$$

$$occ((S)^L \rfloor T_1, (S_1)^L \rfloor T | T_2) = \begin{cases} 1 + occ((S)^L \rfloor T_1, T_2) & \text{if } S \equiv S_1 \text{ and } T_1 \equiv T \\ occ((S)^L \rfloor T_1, T) + occ((S)^L \rfloor T_1, T_2) & \text{otherwise} \end{cases}$$

$$occ(S | T, S | T_1) = \begin{cases} \frac{(1+exactocc(S, T_1))exactocc(T, T_1)}{1+exactocc(S, T)} & \text{if } exactocc(T, T_1) > 0 \\ occ(S | T, T_1) & \text{otherwise} \end{cases}$$

$$occ((S)^L \rfloor T | T_1, (S_1)^L \rfloor T_2 | T_3) = \begin{cases} \frac{(1+exactocc((S)^L \rfloor T, T_3))exactocc(T_1, T_3)}{1+exactocc((S)^L \rfloor T, T_1)} & \text{if } S \equiv S_1,\ T \equiv T_2 \\ & \text{and } exactocc(T_1, T_3 \\ occ((S)^L \rfloor T | T_1, T_2) + occ((S)^L \rfloor T | T_1, T_3) & \text{otherwise} \end{cases}$$

$\vdots \qquad \vdots \qquad \vdots \qquad \vdots$

The *occ* function: examples

$$occ(\quad a \mid b \quad , \quad a \mid a \mid b \mid b \mid b \quad) = 6$$

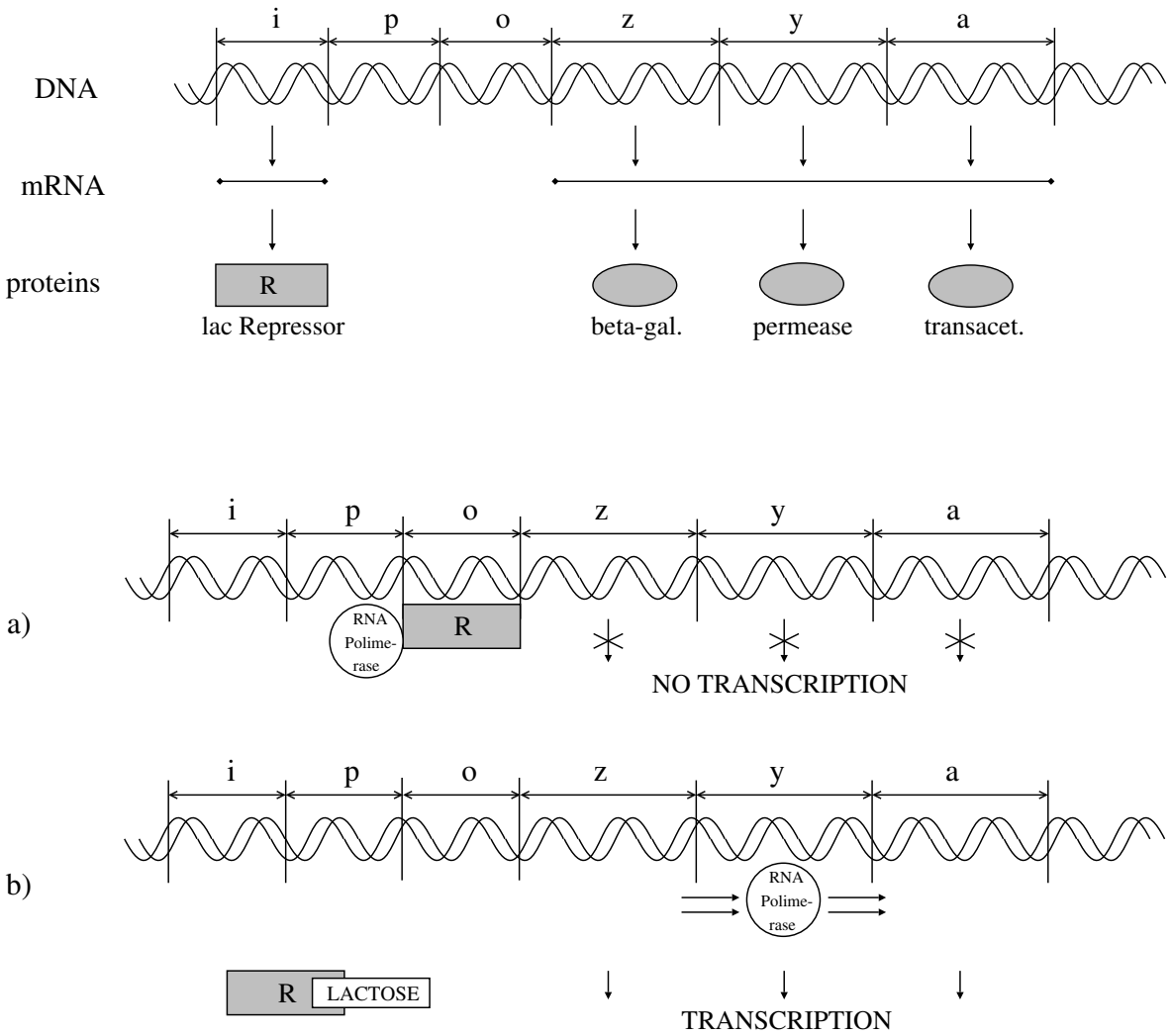
$$occ\left(\quad (m)^L \rfloor a \quad , \quad (m)^L \rfloor a \quad \mid \quad (m)^L \rfloor (m)^L \rfloor a \quad\right) = 2$$

$$occ\left(\quad a \mid b \quad , \quad (m)^L \rfloor (a \mid b) \quad \mid \quad (m)^L \rfloor (a \mid b \mid a) \quad\right) = 3$$

$$occ(\quad a \cdot b \quad , \quad a \cdot b \cdot c \cdot a \cdot b \quad) = 2$$

$$occ\left(\quad a \cdot b \quad , \quad (b \cdot a \cdot b \cdot a)^L \rfloor a \cdot b \cdot c \quad\right) = 3$$

A Stochastic CLS model of the *lac* operon (1)



A Stochastic CLS model of the *lac* operon (2)

Transcription of DNA, binding of lac Repressor to gene o, and interaction between lactose and lac Repressor:

$$lacI \cdot \widetilde{x} \overset{0.02}{\mapsto} lacI \cdot \widetilde{x} \mid Irna \tag{S1}$$

$$Irna \overset{0.1}{\mapsto} Irna \mid repr \tag{S2}$$

$$polym \mid \widetilde{x} \cdot lacP \cdot \widetilde{y} \overset{0.1}{\mapsto} \widetilde{x} \cdot PP \cdot \widetilde{y} \tag{S3}$$

$$\widetilde{x} \cdot PP \cdot \widetilde{y} \overset{0.01}{\mapsto} polym \mid \widetilde{x} \cdot lacP \cdot \widetilde{y} \tag{S4}$$

$$\widetilde{x} \cdot PP \cdot lacO \cdot \widetilde{y} \overset{20.0}{\mapsto} polym \mid Rna \mid \widetilde{x} \cdot lacP \cdot lacO \cdot \widetilde{y} \tag{S5}$$

$$Rna \overset{0.1}{\mapsto} Rna \mid betagal \mid perm \mid transac \tag{S6}$$

$$repr \mid \widetilde{x} \cdot lacO \cdot \widetilde{y} \overset{1.0}{\mapsto} \widetilde{x} \cdot RO \cdot \widetilde{y} \tag{S7}$$

$$\widetilde{x} \cdot RO \cdot \widetilde{y} \overset{0.01}{\mapsto} repr \mid \widetilde{x} \cdot lacO \cdot \widetilde{y} \tag{S8}$$

$$repr \mid LACT \overset{0.005}{\mapsto} RLACT \tag{S9}$$

$$RLACT \overset{0.1}{\mapsto} repr \mid LACT \tag{S10}$$

A Stochastic CLS model of the *lac* operon (3)

The behaviour of the three enzymes for lactose degradation:

$$\left(\widetilde{x}\right)^{L} \rfloor (perm \mid X) \overset{0.1}{\mapsto} \left(perm \cdot \widetilde{x}\right)^{L} \rfloor X \tag{S11}$$

$$LACT \mid \left(perm \cdot \widetilde{x}\right)^{L} \rfloor X \overset{0.001}{\mapsto} \left(perm \cdot \widetilde{x}\right)^{L} \rfloor (LACT \mid X) \tag{S12}$$

$$betagal \mid LACT \overset{0.001}{\mapsto} betagal \mid GLU \mid GAL \tag{S13}$$

Degradation of all the proteins and mRNA involved in the process:

$$perm \overset{0.001}{\mapsto} \epsilon \tag{S14}$$

$$betagal \overset{0.001}{\mapsto} \epsilon \tag{S15}$$

$$transac \overset{0.001}{\mapsto} \epsilon \tag{S16}$$

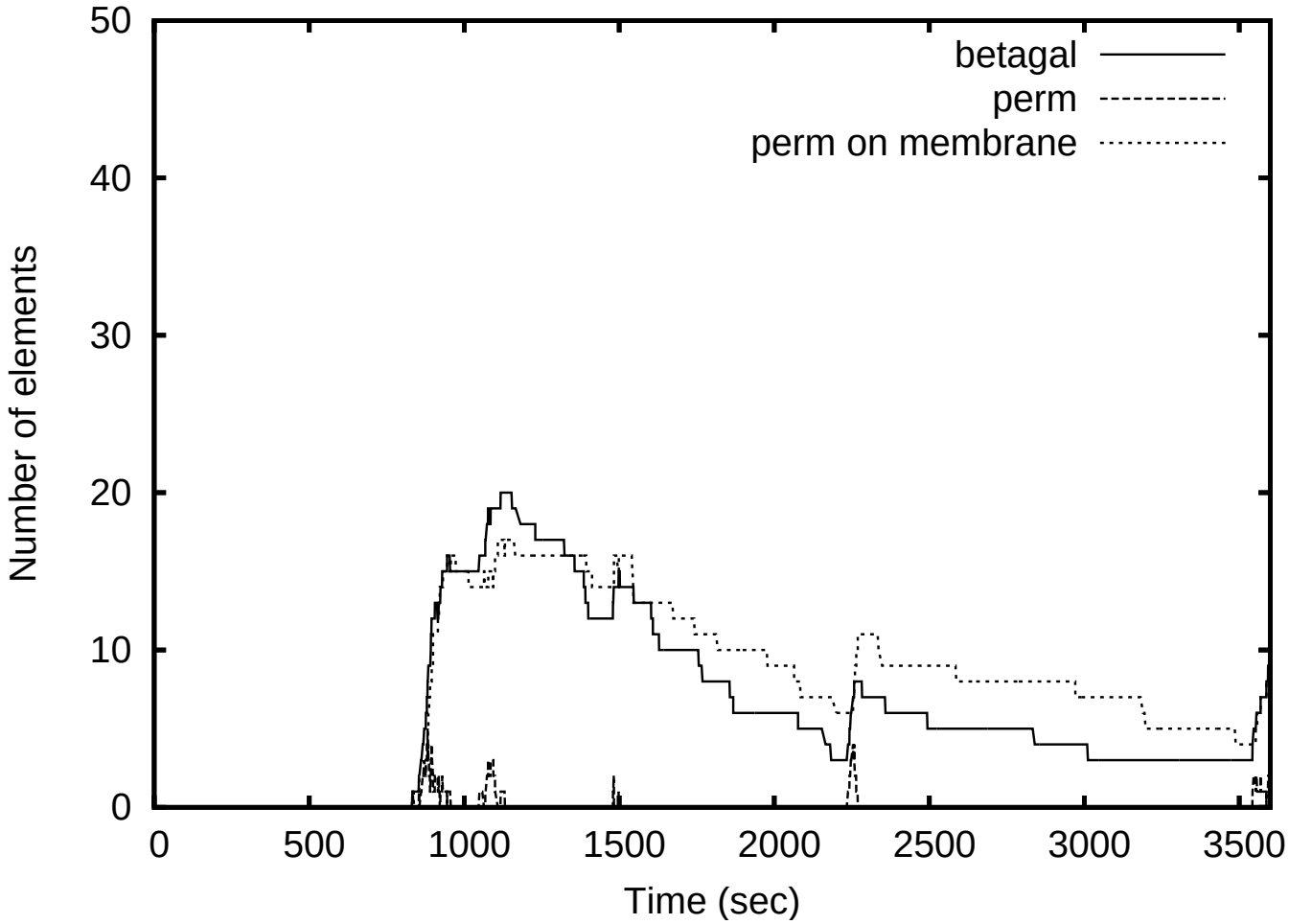
$$repr \overset{0.002}{\mapsto} \epsilon \tag{S17}$$

$$Irna \overset{0.01}{\mapsto} \epsilon \tag{S18}$$

$$Rna \overset{0.01}{\mapsto} \epsilon \tag{S19}$$

$$RLACT \overset{0.002}{\mapsto} LACT \tag{S20}$$

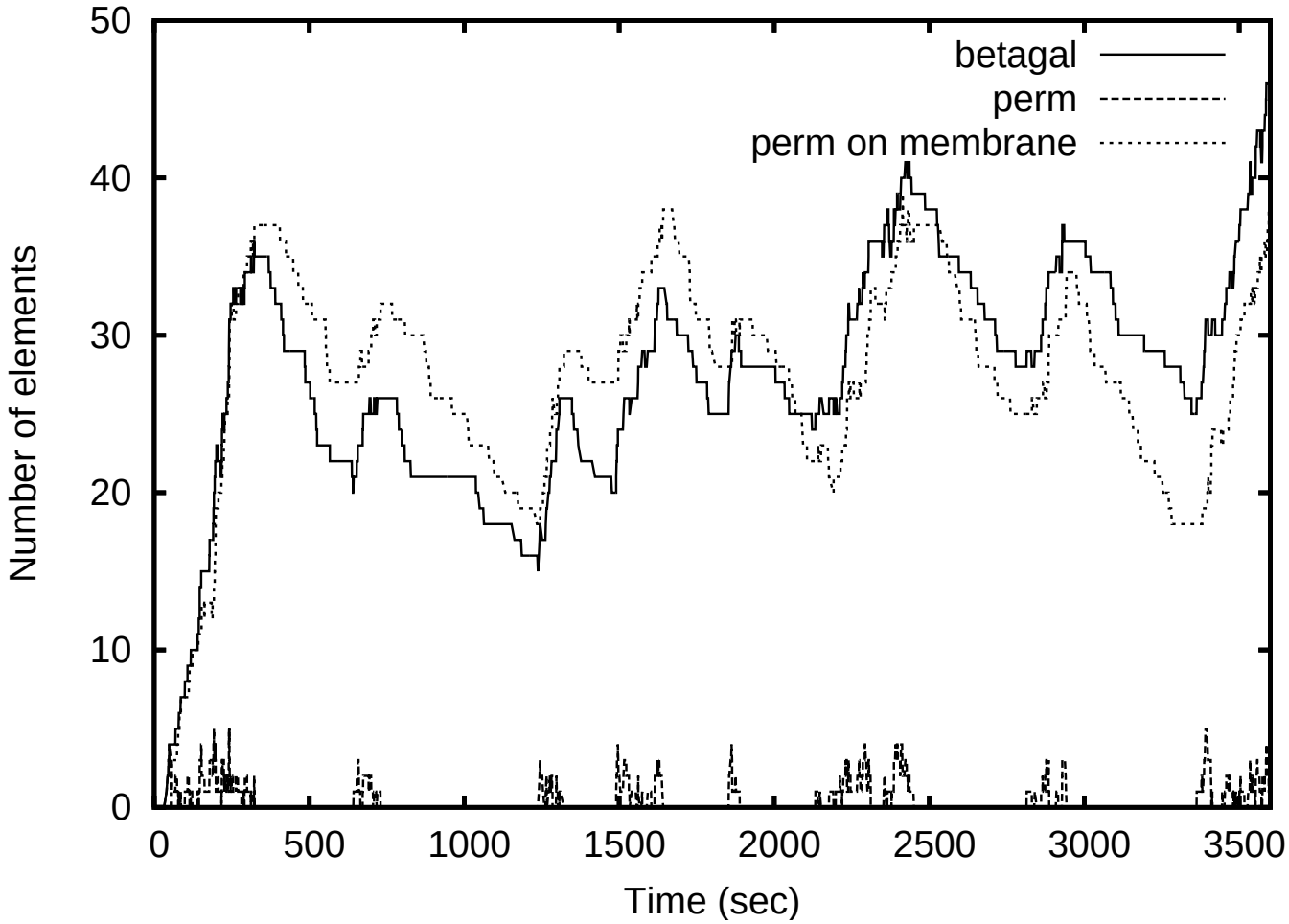
Simulation results (1)



Production of enzymes in the absence of lactose

$$(m)^{L} \rfloor (lacI - A \mid 30 \times polym \mid 100 \times repr)$$

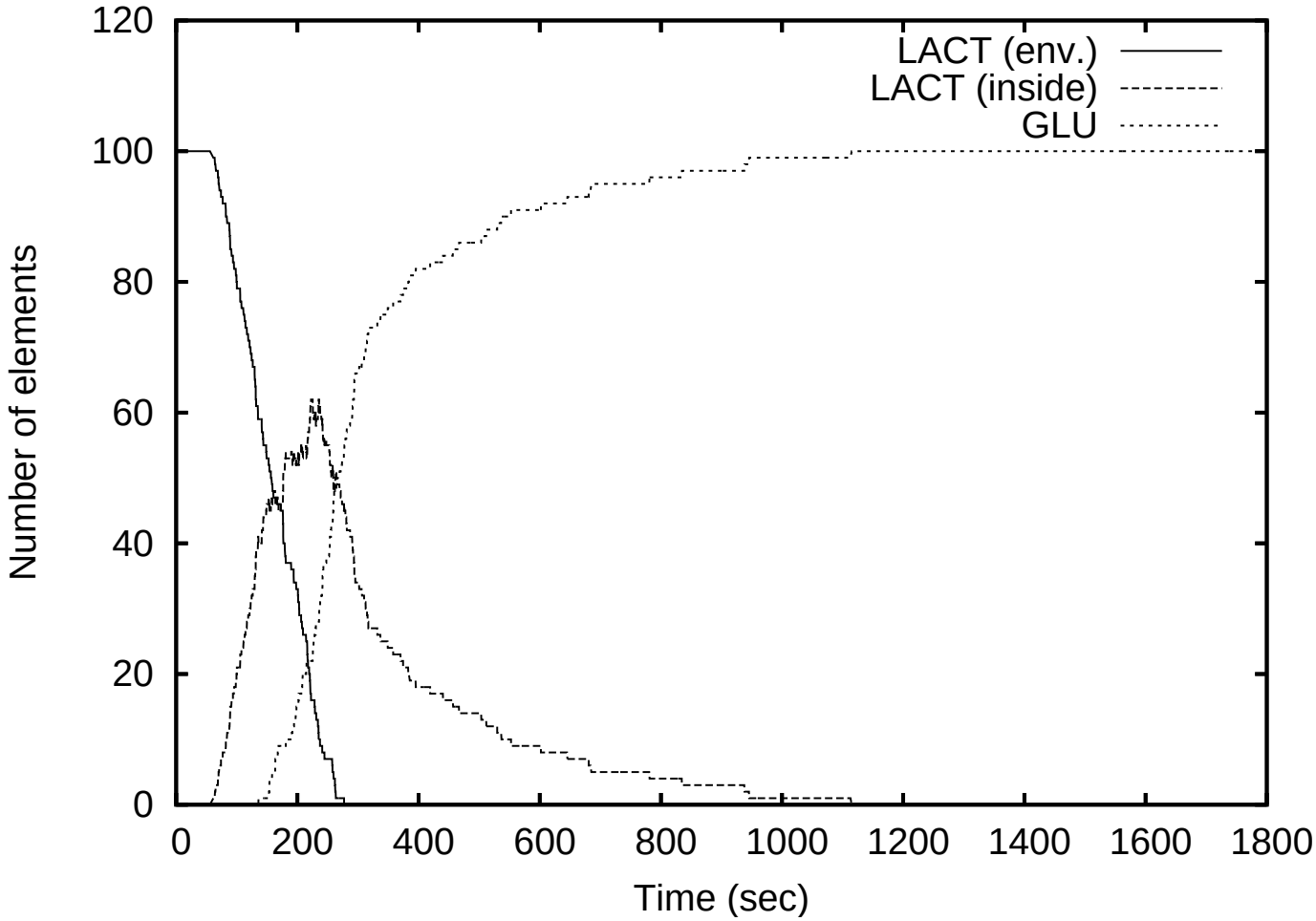
Simulation results (2)



Production of enzymes in the presence of lactose

$$100 \times LACT \mid \left(m\right)^{L} \rfloor \left(lacI - A \mid 30 \times polym \mid 100 \times repr\right)$$

Simulation results (3)



Degradation of lactose into glucose

$$100 \times LACT \mid \left(m\right)^{L} \rfloor \left(lacI - A \mid 30 \times polym \mid 100 \times repr\right)$$

Outline of the talk

1. Introduction

2. The Calculus of Looping Sequences (CLS)
   - Definition of CLS
   - The *lac* operon in CLS
   - Stochastic CLS

3. (Statistical) model checking of Stochastic CLS models
   - Choosing a model checker
   - Translation of Stochastic CLS into Real-Time Maude
   - Analysis examples

A model checker for Stochastic CLS

As candidate model checkers we have considered:

- PRISM
- Murphi
- PMaude

All of them are probabilistic/stochastic model checkers

PMaude is the most suitable

- It uses a language based on rewrite rules (rewrite logic) that eases the translation of Stochastic CLS rules

Unfortunately, the model checking module of PMaude seems not to be available

- a possible alternative: Real-Time Maude

Real-Time Maude

Maude is a specification language equipped with efficient analysis tools, which supports three modelling paradigms:

- algebraic style (via equations)
- rewrite logic (via rewrite rules)
- object oriented (via classes and messages)

Real-Time Maude extends Maude with a notion of time

- rewrite rule applications might consume (a fixed amount of) time

Real-Time Maude has two kinds of rules

- istantaneous rules:
  `crl [l] : t => t' if cond`
- tick rules:
  `crl [l] : t => t' in time τ if cond`

Translation of Stochastic CLS into Real-Time Maude

Real-Time Maude is not stochastic

- we will include Gillespie's simulation algorithm (slightly changed) in the translation of Stochastic CLS models
- it will be used to generate single executions of the model
- Real-Time Maude analysis tools will be applied to the simulation results

This is *statistical model checking*

- we loose exhaustivity (properties are checked on a number of runs)
- huge systems could be handled
- may allow *property driven* simulations

A slight modification to Gillespie's algorithm

Given reactions $\{R_1, \ldots, R_M\}$ and initial molecular population $X_1, \ldots, X_M$:

Step 0 Initialize current time $t = 0$ and stepped time $t\_step = 0$

Step 1 Compute reaction propensities $a_1, \ldots, a_n$ and $\Sigma_{\nu=1}^{M} a_\nu$

Step 2 Choose reaction μ and reaction time τ

Step 3 If $t + \tau \geq t\_step$ then $t\_step = t\_step + \Delta t$

Step 4 Execute reaction μ and set $t = t + \tau$

Step 5 If $t \geq total\_time$ then conclude, else return to **Step 1**

Simulations are still exact

- our modification doesn't change states and choices of the simulation
- simulation results are sampled every Δt time units

(Modified) Gillespie's algorthm in Real-Time Maude

Every step of the simulation algorithm is translated into an instantaneous rule, apart from **Step 3** (the increase of the stepped time)

```
(set tick def 1/100 .)

crl [increase] :
  < step:3 , t:R1 , tau:R2 , t_step:R3 , delta_t:R4 >
=>
  < step:4 , t_step:R3+R4 >
in time R3
if (R1+R2) >= R3

crl [no_increase] :
  < step:3 , t:R1 , tau:R2 , t_step:R3 , delta_t:R4 >
=>
  < step:4 >
in time R3
if (R1+R2) < R3
```

Translation of Stochastic CLS into Real-Time Maude

$$
\begin{aligned}
T &::= S \mid (S)^L \rfloor T \mid T \mid T \\
S &::= \epsilon \mid a \mid S \cdot S
\end{aligned}
$$

```
(omod CLS is
  pr NAT
  sorts Elem Seq Term Loop
  subsorts Elem < Seq < Term

  op empty :  -> Seq [ctor]
  op _._ :  Seq Seq -> Seq
        [assoc gather (E e) id:  empty ctor]
  op `{_`}_ :  Elem Nat -> Term
  op `[_`]LContains`[_`] :  Seq Term -> Term
        [prec 41 gather (& &) ctor]
  op _|_ :  Term Term -> Term
        [assoc comm prec 45 gather (E e) id:  empty ctor]
endom)
```

Translation of Stochastic CLS into Real-Time Maude

Lotka reactions as Stochastic CLS rules

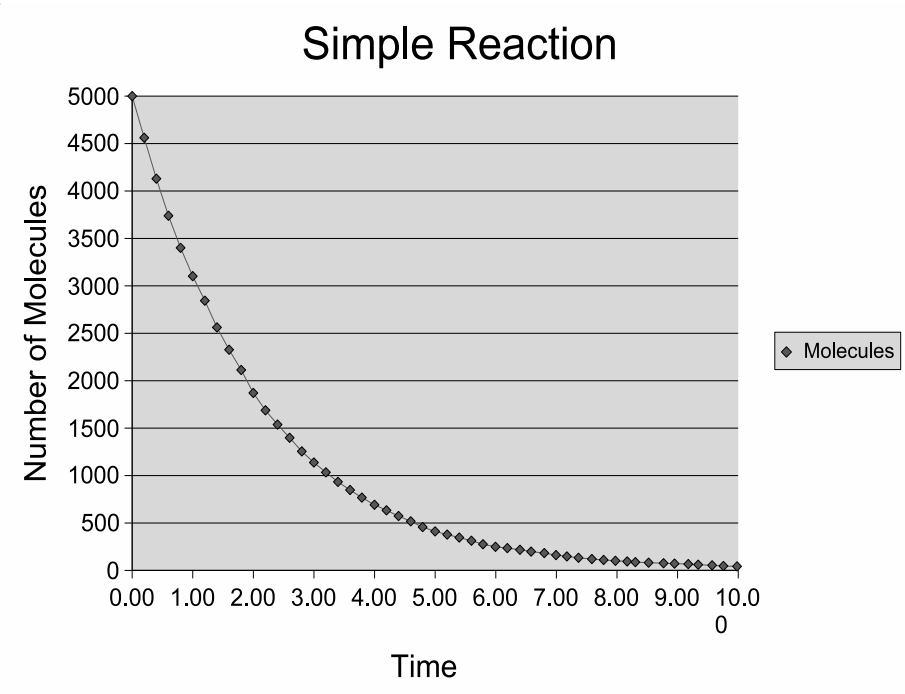
$$S_1 \xrightarrow{10} S_1|S_1 \qquad S_1|S_2 \xrightarrow{0.01} S_2|S_2 \qquad S_2 \xrightarrow{10} \epsilon$$

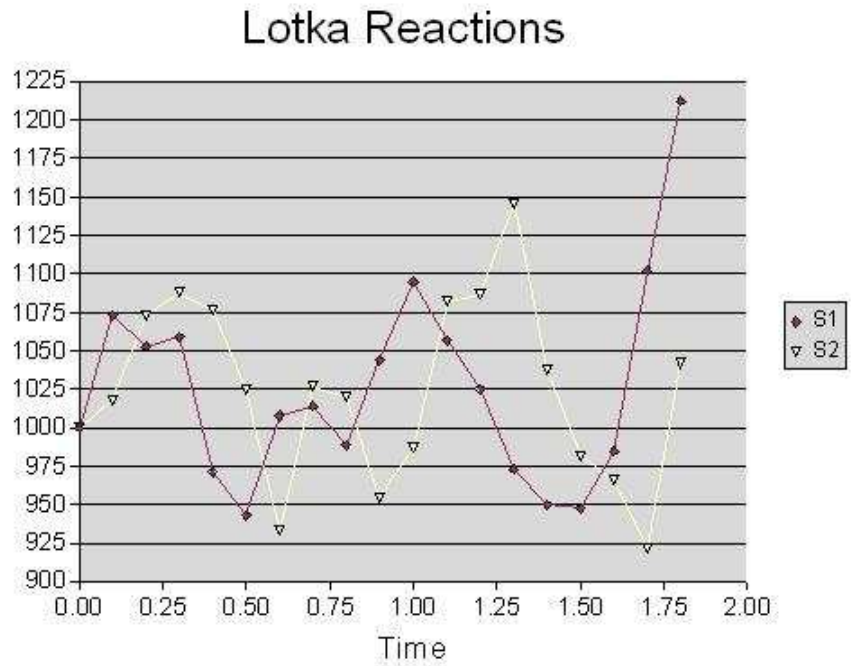
```
rl [ S1 ] :
    < O : CLSTerm | term :  (T | S1), mu :  1, step :  4 >
=>
    < O : CLSTerm | term :  (T | S1 | S1), step :  5 >

rl [ S2 ] :
    < O : CLSTerm | term :  (T | S1 | S2), mu :  2, step :  4 >
=>
    < O : CLSTerm | term :  (T | S2 | S2), step :  5 >

rl [ S3 ] :
    < O : CLSTerm | term :  (T | S2), mu :  3, step :  4 >
=>
    < O : CLSTerm | term :  T, step :  5 >
```

Analysis example: stochastic simulation





- Simple reaction: $S_1 \xrightarrow{0.5} S_2$
- Lotka reactions: $S_1 \xrightarrow{10} S_1|S_1$ $\quad S_1|S_2 \xrightarrow{0.01} S_2|S_2$ $\quad S_2 \xrightarrow{10} \epsilon$

Analysis example: statistical model checking

Initialisation of 100 stochastic simulations

```
rl [ initialise1 ] :
  < step :  0 >
=>
  < seed :  random(1), step :  1 >
⋮


⋮
rl [ initialise100 ] :
  < step :  0 >
=>
  < seed :  random(100), step :  1 >
```

Analysis example: statistical model checking

By using the `tsearch` command we can check all possible behaviours

Starting with 4 x S_1 and 4 x S_2 we search 10 states where S_2 is absent

```
(tsearch [10] INIT({S1}4 | {S2}4) =>* {< O:Oid :  CLSTerm | term
:  T:Term > C:Configuration} such that occ(S2,T:Term) = 0 in time
<= 1/10 .)

Solution 1
C:Configuration -->
  < term:  {S1}5, finaltime:  7.8293318117206676e-2 >
⋮
Solution 10
C:Configuration -->
  < term:  {S1}8, finaltime:  5.6307762323583766e-2 >
```

Analysis example: statistical model checking

The `find earliest` searches for the earliest time when a given state is reached

Starting with 4 x S_1 and 4 x S_2 we search the earliest time when S_2 disappear

```
(find earliest INIT({S1}4 | {S2}4) =>* {< CLSTerm | term:T:Term >
C:Configuration} such that occ(S2,T:Term) == 0 .)
```

```
Result:
{< term:  {S1}8, finaltime:  5.6307762323583766e-2 > in time 3/50}
```

Analysis example: statistical model checking

Verification of properties espressed as LTL formulas. Some state formulas:

vanished(T) indicates that term T has vanished from the system,

IsLessThan(T,T') indicates that the occurences of term T are less than the occurences of T'.

Starting with 4 x S_1 and 4 x S_2 we prove

- that S_2 will eventually disappear (i.e. $\Diamond$vanished(S_2))
- that the amount of S_2 will eventually become less than the amount of S_1 (i.e. $\Diamond$IsLessThan(S_2,S_1))

```
(mc INIT({S1}4 | {S2}4) |=t <> vanished(S2) in time<=1 .)

Result Bool :  true

(mc INIT({S1} 4 | {S2} 4) |=t <> IsLessThan(S2,S1) in time<=1 .)

Result Bool :  true
```

Conclusions

We have presented preliminary ideas for the application of (statistical) model checking to Stochastic CLS models

Many things to do:

- Prove the correctness of the translation into Real-Time Maude w.r.t. Stochastic CLS semantics
- Improve the efficiency of the computation of the *occ* function
- Model and analyse some more significant case study (e.g. the Lactose operon)