Varieties of Stochastic Calculi

Luca Cardelli

Microsoft Research

The Microsoft Research - University of Trento Centre for Computational and Systems Biology

Trento, 2006-05-22..26

www.luca.demon.co.uk/ArtificialBiochemistry.htm

Research is what I'm doing when I don't know what I'm doing. Wernher Von Braun.
A continuous-time Markov chain (CTMC) is a pair \( M=(S,R) \) where \( S \) is a countable set of states, and \( R:S \times S \rightarrow \text{Real}_{\geq 0} \) is the rate matrix.

The exit rate \( E(s) = \sum_{s' \in S} R(s,s') \) denotes that the probability of taking a transition from \( s \) within \( t \) time units equals \( 1-e^{-E(s)t} \).

(The time between state changes is a random variable with a memoryless distribution.)

The embedded discrete-time Markov chain.

(N.B. when we do this, we loose information on holding time in each state)

The probability \( P(s,s') \) of moving from \( s \) to \( s' \) in a single step is:

- if \( E(s)>0 \) then \( P(s,s')=R(s,s')/E(s) \)
- if \( E(s)=0 \) then \( P(s,s)=1 \) and \( P(s,s')=0 \) for \( s \neq s' \).

\( P(s,s') \) equals the probability that the delay of going from \( s \) to \( s' \) “finishes before” the delay of any other outgoing transition from \( s \).

(1) The more standard generator matrix has \(-E(s)\) on the diagonal, which requires forbidding self-loops.

Model checking continuous-time Markov chains by transient analysis

Christel Baier\textsuperscript{a}, Boualem Haverkort\textsuperscript{b}, Holger Hermanns\textsuperscript{a} and Joost-Pieter Katoen\textsuperscript{c,*}
CTMC Modeling Formalism

- **CTMCs**
  - Finite State Machines with rates on transitions.
  - All durations are exponentially distributed (memoryless)
  - Well studied class of stochastic processes
  - Efficient analysis algorithms for stationary and transient analysis

- **We consider only:**
  - Modeling formalisms that map to CTMC's
  - But there are still quite a few of those:

- **High level formalisms mapping to CTMCs**
  - Stochastic Petri Nets [Molloy]
  - Markovian Queuing Networks [Muppala & Triverdi]
  - Stochastic Automata Networks [Plateau]
  - Probabilistic I/O Automata [Wu et al.]
  - Stochastic Process Algebras [Herzog et al.] [Hillston]
Stochastic Calculi
A Compositional Approach to Performance Modelling


Jane Hillston

**PEPA**

**Labeled Transition Semantics**

\[ P ::= (\alpha, r).P \mid P + Q \mid P \Downarrow A \mid P / L \mid A \]

**Definition 3.3.1** The apparent rate of action of type \( \alpha \) in a component \( P \), denoted \( r_\alpha(P) \), is the sum of the rates of all activities of type \( \alpha \) in \( \text{Act}(P) \).

1. \( r_\alpha(\beta, r).P) = \begin{cases} r & \text{if } \beta = \alpha \\ 0 & \text{if } \beta \neq \alpha \end{cases} \)

2. \( r_\alpha(P + Q) = r_\alpha(P) + r_\alpha(Q) \)

3. \( r_\alpha(P / L) = \begin{cases} r_\alpha(P) & \text{if } \alpha \notin L \\ 0 & \text{if } \alpha \in L \end{cases} \)

4. \( r_\alpha(P \Downarrow A) = \begin{cases} \min(r_\alpha(P), r_\alpha(Q)) & \text{if } \alpha \in L \\ r_\alpha(P) + r_\alpha(Q) & \text{if } \alpha \notin L \end{cases} \)

**Prefix**

\[
\frac{}{(\alpha, r).E \xrightarrow{(\alpha, r)} E}
\]

**Choice**

\[
\frac{E \xrightarrow{(\alpha, r)} E'}{E + F \xrightarrow{(\alpha, r)} E'} \quad \frac{F \xrightarrow{(\alpha, r)} F'}{E + F \xrightarrow{(\alpha, r)} F'}
\]

**Cooperation**

\[
\frac{E \xrightarrow{(\alpha, r)} E'}{E [A] F \xrightarrow{(\alpha, r)} E'[A] F'} \quad \frac{F \xrightarrow{(\alpha, r)} F'}{E [A] F \xrightarrow{(\alpha, r)} E'[A] F'}
\]

\[
E \xrightarrow{(\alpha, r)} E' \quad F \xrightarrow{(\alpha, r)} F' \quad (\alpha \notin L) \quad \text{where } R = \frac{r_1}{r_\alpha(E)} \frac{r_2}{r_\alpha(F)} \min(r_\alpha(E), r_\alpha(F))
\]

**Hiding**

\[
\frac{E \xrightarrow{(\alpha, r)} E'}{E / L \xrightarrow{(\alpha, r)} E'/L} \quad \frac{E \xrightarrow{(\alpha, r)} E'}{E / L \xrightarrow{(\tau_f)} E'/L} \quad (\alpha \in L)
\]

**Constant**

\[
\frac{E \xrightarrow{(\alpha, r)} E'}{A \xrightarrow{(\alpha, r)} E'} \quad (A \equiv E)
\]
**Syntax**

\[
\pi \ ::= \ x(y) \quad \text{receive } y \text{ along } x \\
\bar{x}(y) \quad \text{send } y \text{ along } x
\]

\[
P \ ::= \ 0 \mid \sum_{i \in I} \pi_i \cdot P_i \mid [x = y] P \mid P_1 \parallel P_2 \mid (\text{new } x)P \mid 1P
\]

**Structural congruence laws**

Renaming of bound variables

\[
x(y).P = x(z).\{z/y\}P \quad \text{if } z \notin FN(P)
\]

\[
(\text{new } y).P = (\text{new } z).\{z/y\}P \quad \text{if } z \notin FN(P)
\]

**Chemical Mixing**

Reduction Semantics (with structural congruence)

**Reactions**

Communication (COMM):

\[
(\cdots + \bar{x}(z).Q)(\cdots + x(y).P) \to Q\parallel P\{z/y\}
\]

Reaction under parallel composition (PAR):

\[
P \to P' \\
Q \to Q'
\]
Stochastic π-calculus

- **Stochastic extension of π-calculus.**
  - Associate a single parameter \( r \) (rate) in \((0, \infty)\) to each activity \( a \).
  - The rate and the associated exponential distribution describes the stochastic behavior of the activity.

  \( a.P \) is replaced by \( a@r.P \)

- **Exponential distribution**
  - Guarantees the memoryless property: the time at which a change of state occurs is independent of the time at which the last change of state occurred.
  - I.e. we do not have to remember past states transitions (e.g. in the implementation).
  - Conversely, if we *do* want to remember past transitions, we can do it by programming.

- **Race resolution**
  - Races are resolved in a probabilistic competitive way: all the activities that are enabled in a state compete and the fastest one (stochastically) succeeds. [Gillespie]

---

Stochastic π-Calculus

**Corrado Priami**

Dipartimento di Informatica, Università di Pisa, Corso Italia, 40, I-56125 Pisa, Italy
Email: priami@di.unipi.it


---

\[
\begin{align*}
\text{Act:} & \quad (\mu, r), P \frac{\mu r}{P} P, \mu \text{ not input} \\
\text{Ein:} & \quad (x(y), r). P \frac{\mu r}{P} P[y/x] \\
\text{Idc:} & \quad P(x/y) \frac{\mu r}{P} P, Q(y) = P \\
\text{Par0:} & \quad P \frac{\mu r}{P} P, Q \frac{\mu r}{P} P \\
\text{Sum0:} & \quad P \frac{\mu r}{P} P + Q \frac{\mu r}{P} P \\
\text{Par1:} & \quad P \frac{\mu r}{P} P, Q \frac{\mu r}{P} P \\
\text{Sum1:} & \quad P \frac{\mu r}{P} P + Q \frac{\mu r}{P} P \\
\text{Open:} & \quad P \frac{\mu r}{P} P \\
\text{Race resolution:} & \quad \text{All the activities that are enabled in a state compete and the fastest one (stochastically) succeeds. [Gillespie]}
\end{align*}
\]
This paper was motivated by the need to model “chemical” interaction laws and mixed choice. But it supports mixed choice, only over a distinguished set of channels with exactly 1 mixed choice per summation (homodimerization).

\( \text{In}_x(0) = 0, \)
\( \text{In}_x \left( \sum_{i \in I} (\pi_i, r_i) \cdot P_i \right) = |\{(\pi_i, r_i) | i \in I \land \text{sbj}(\pi_i) = x\}|, \)
\( \text{In}_x(P_1 \cdot P_2) = \text{In}_x(P_1) + \text{In}_x(P_2), \)
\( \text{In}_x((v z) P) = \begin{cases} \text{In}_x(P) & \text{if } z \neq x, \\ 0 & \text{otherwise.} \end{cases} \)

\( \text{Out}_x \) is similarly defined, by replacing any occurrence of \( \text{In} \) with \( \text{Out} \) and the condition \( \text{sbj}(\pi_i) = x \) with \( \text{sbj}(\pi_i) = \overline{x} \).

A usual reaction is implemented by the three parameters \( r_b, r_0 \) and \( r_1 \), where \( r_b \) represents the basical rate, and \( r_0 \) and \( r_1 \) denote the quantities of interacting molecules, and are computed compositionally via \( \text{In}_x \) and \( \text{Out}_x \), while deducing transitions. The first axiom in Table 1 corresponds to usual reactions, with two different molecules. The second one corresponds to homodimerization reactions.
**Def. 2.1 (Process Expressions)**
The set of action prefixes $\pi$ are defined by

$$
\pi ::= x\langle\vec{y}\rangle \quad \text{receive } \vec{y} \text{ along } x \\
| \quad \vec{x}\langle\vec{y}\rangle \quad \text{send } \vec{y} \text{ along } x.
$$

where $\vec{y}$ denotes a tuple $(y_1, \ldots, y_n)$ of channel names called a channel vector. The set of process expressions in the StoPi-calculus is defined by the following syntax with the syntactic categories processes ($P$), guarded sums ($M$) and process declarations ($D$):

$$
D ::= D; D \mid A(x) \overset{df}{=} P
$$

$$
P ::= P|P \mid \text{new } (x_1 : r_1, \ldots, x_n : r_n) \ P
| M \mid A(x) \mid 0
$$

$$
M ::= \pi.P \mid M + M
$$

where 0 is the nil process (or empty sum) which cannot do any transitions and $A(x)$ is an invocation of the process with identifier $A$ with actual parameters $\vec{x}$.

**Def. 2.2 (Structural Congruence)**
Two processes $P$ and $Q$ in the StoPi-calculus are structurally congruent, written $P \equiv Q$, if we can transform one into the other by using the following equations (in either direction):

1. $P \equiv \{\vec{x}/\vec{y}\} P$ if $y_i \notin f(n)(P)$ for $1 \leq i \leq |\vec{y}|$. (alpha-conversion)
2. $M + N \equiv N + M, \quad (M + N) + L \equiv M + (N + L)$
3. $P|0 \equiv P, P|Q \equiv Q|P, P|(Q|R) \equiv (P|Q)|R$
4. new $\vec{x}\langle\vec{y}\rangle P \equiv \text{new } \vec{y}\langle\vec{x}\rangle P$
5. new $\vec{x}\langle\vec{y}\rangle P \equiv \text{new } \vec{y}\langle\vec{x}\rangle P$ if $x_i \notin f(n)(P)$ for $1 \leq i \leq |\vec{x}|$

$$
A(\vec{x}) \overset{df}{=} P \Rightarrow A(\vec{y}) \equiv \{\vec{y}/\vec{x}\} P
$$

**STOCH**

$$
P \overset{\pi}{\rightarrow} P' \quad \text{if } P \equiv Q \text{ and } P' \equiv Q'
$$

**PAR**

$$
P \overset{\pi}{\rightarrow} P' \quad P|Q \overset{\pi}{\rightarrow} P'|Q
$$

**RES**

$$
P \overset{\pi}{\rightarrow} P' \quad \text{(new } y : r) P \overset{\pi}{\rightarrow} \text{(new } y : r) P'
$$

**REACT**

$$
\left(\langle x(\vec{y}) P + M), (\vec{x}(\vec{z}) Q + N)\right) \overset{\pi}{\rightarrow} \left(\langle x(\vec{y}) P\right)
$$

**Figure 1: Semantics of the StoPi-calculus.**

**Def. 3.1 (Process Normal Form)**
For a process in the StoPi-calculus we define the normal form as

$$
\text{new } (y_1 : \tau_1, \ldots, y_n : \tau_n)(P_1 \cdots P_j)
$$

where each $P_i$ is either $\sum_{k=0}^{n} \pi_k P_k$ or $A(\vec{x})$.

In the following let $c = S$ be a configuration, $Ch_c$ the channels contained in the environments in the configuration $c$ and assume that $s$ and $s'$ are summations in $S$. Furthermore let

$$
|s|_c = |\{(x, y, P) \in A \mid s = (A, e) \land e(x) = ch\}|
$$

and

$$
|s|_c = |\{(x, y, P) \in A \mid s = (A, e) \land e(x) = ch\}|
$$

Then the apparent rate of a reaction on a channel in a configuration is defined as

**Def. 3.4 (Apparent Rates)**
The apparent rate of a reaction between guarded sums $s$ and $s'$ in a configuration $c$ on channel $ch \in Ch_c$, where $s$ receives input on $ch$ and $s'$ outputs on $ch$, is defined as:

$$
r(s, s', ch) = \begin{cases} 
\text{if } s \neq s' \\
0 
\end{cases}
$$

The apparent rate of channel $ch$ in a configuration $c$ is the sum of the apparent rates of a reactions on $ch$ on all possible guarded sums:

$$
r(ch) = \sum_{s, s' \in S} r(s, s', ch).
$$

Note that the expression (7) can explicitly be calculated as

$$
r(ch) = ch \left( \sum_{s \in S} |s|_c \sum_{s' \in S | s \neq s'} |s'|_c \right).
$$

**The StoPi-calculus and Simulator**
- a stochastic r-calculus and the implementation of a simulator

Anders Bloch
Bjørn Haagensen
Mithun Kurukshetrirayer
Steffen Ude Knoeser

Aalborg University, Department of Computer Science
Fredrik Hjelmsvej Vej 7E, DK-9220 Aalborg Øst
30th May 2003 (unpublished)
http://www.cs.auc.dk/~bh/files/dat4project.ps.gz

This is the most detailed analysis of a stochastic pi calculus (with mixed choice) yet. However, the apparent rate is defined on the underlying abstract machine after translation, not on the pi calculus itself.
Again, the apparent rate is defined on the underlying abstract machine.
\[ R(x, P) = \tau \text{rate}(x) \times (\text{In}_x(P) \times \text{Out}_x(P) - \text{Mix}_x(P)) \]
Luca Cardelli

Reduction Semantics
(with structural congruence)

\[ \begin{align*}
\tau_r . P + M & \xrightarrow{r} P \\
\text{!}x(n).P + M \mid ?x(m).Q + N & \xrightarrow{x} P \mid Q_{\{n/m\}} \\
P \xrightarrow{x} P' & \Rightarrow \nu x P \xrightarrow{R(x,P)} \nu x P'
\end{align*} \]

\[ \begin{align*}
\theta \neq x & \quad P \xrightarrow{\theta} P' \Rightarrow \nu x P \xrightarrow{\theta} \nu x P' \\
& \quad P \xrightarrow{\theta} P' \Rightarrow P \mid Q \xrightarrow{\theta} P' \mid Q \\
Q \equiv P & \xrightarrow{\theta} P' \equiv Q' \Rightarrow Q \xrightarrow{\theta} Q'
\end{align*} \]

Reduction in S\(\pi\), where \(\theta := x | r\) and \(R(x,P)\) is the apparent rate of \(x\) in \(P\).

**Definition 1.1. Syntax of S\(\pi\)**

\[ \begin{align*}
P, Q & := \nu x P & \text{Restriction} \\
| & P \mid Q & \text{Parallel} \\
| & M & \text{Choice} \\
| & X \langle n \rangle & \text{Instance} \\
M & := \pi . P + M & \text{Action} \\
| & 0 & \text{Null} \\
| & \tau_r & \text{Delay}
\end{align*} \]

**Definition 1.2. Structural Congruence in S\(\pi\)**

\[ \begin{align*}
P \mid 0 & = P \\
P \mid Q & = Q \mid P \\
(P \mid Q) \mid R & = (P \mid Q) \mid R \\
\pi_1 . P_1 + \pi_2 . P_2 + M & = \pi_2 . P_2 + \pi_1 . P_1 + M \\
\nu x 0 & = 0 \\
\nu x y P & = \nu y \nu x P \\
x \notin \text{fn}(P) & \Rightarrow \nu x (P \mid Q) = P \mid \nu x Q \\
X \langle n \rangle \equiv P & \Rightarrow X \langle n \rangle = P \langle x/n \rangle
\end{align*} \]

**Definition 1.3. Apparent Rate in S\(\pi\).** The definition of \(\text{Out}_x(P)\) is similar to that of \(\text{In}_x(P)\).

**Proposition 1.4.** \(\forall P, P' \in S\pi \land P \equiv Q \Rightarrow R(x, P) = R(x, Q)\)
BioAmbients

- **An extension of Sto-π-calculus**
  - Dynamic membranes: operations for merging, splitting, interacting through membrane channels.
  - Implemented by Aviv Regev.

- **An adaptation of Ambient Calculus**
  - A process language for dynamic containers (mobile agents, distributed locations, etc.)

**BioAmbients: An abstraction for biological compartments**

Aviv Regev a,† Ekaterina M. Panina b William Silverman c Luca Cardelli d Ehud Shapiro e
• **Theorem [Hillston, Priami]**
  - Let $P$ be a *finite control* $S\pi$ process.
    Then the set of derivative processes from $P$ defines a finite CTMC.
1. For all \( x \in fn(A) \) calculate \( a_x = Act_x(A) \times rate(x) \)
2. Store non-zero values of \( a_x \) in a list \( (x_\mu, a_\mu) \), where \( \mu \in 1...M \).
3. Calculate \( a_0 = \sum_{\nu=0}^{M} a_\nu \)
4. Generate two random numbers \( n_1, n_2 \in [0, 1] \) and calculate \( \tau, \mu \) such that:
   \[
   \tau = \frac{1}{a_0} \ln \left( \frac{1}{n_1} \right)
   \]
   \[
   \sum_{\nu=1}^{\mu-1} a_\nu < n_2 a_0 \leq \sum_{\nu=1}^{\mu} a_\nu
   \]
5. \( Next(A) = x_\mu \) and \( Delay(A) = \tau \).

**Definition 12.** Calculating \( Next(A) \) and \( Delay(A) \) according to Gillespie [6].

---

2340

**Exact Stochastic Simulation of Coupled Chemical Reactions**

Daniel T. Gillespie

Research Department, Naval Weapons Center, China Lake, California 93555 (Received May 12, 1977)

Publication costs assisted by the Naval Weapons Center


---

that’s Area 51!
One State of the Simulation

(François and Hakim example)

6 possible reactions on AB

2 possible reactions on bA

1 possible reactions on each τ

Use Gillespie to pick a reaction and move on
Summary

- Stochastic process calculi:
  - The modular representation of concurrent stochastic processes.
  - Cut down to CTMCs (Continuous Time Markov Chains). Then, standard analysis techniques are applicable.
  - Can be given friendly automata-like scalable graphical syntax [A.Phillips].
  - Is directly executable (via Gillespie).
  - Are analyzable [Hillston] [Hermanns].

Figure 2. Regulating Gene Expression by Positive Feedback

Figure 3. Protein A molecules v.s. time in presence (left) and absence (right) of TF

Q?