Towards a logic of biochemical processes

Chris Banks

BCTCS, April 2012

Introduction

In this talk I will

- show how we are applying formal computational techniques to systems biology, using
  - formal high level language design
  - process calculi
  - modal/temporal logic
  - model checking
- present a language for modelling biochemical processes
- and a logic for querying and classifying model behaviour.

Motivation

- Current scientific practice:
  - Biochemical processes are modelled using ODEs
  - Tools like MATLAB used to write/solve/experiment with these models

Motivation

- Current scientific practice:
  - Biochemical processes are modelled using ODEs
  - Tools like MATLAB used to write/solve/experiment with these models
- Challenges?
  - ODEs do not clearly reflect the domain objects
  - Structural alteration is difficult – non-compositional
  - How to query/classify model behaviour – hand coded analyses?

Motivation

- Current scientific practice:
  - Biochemical processes are modelled using ODEs
  - Tools like MATLAB used to write/solve/experiment with these models
- Challenges?
  - ODEs do not clearly reflect the domain objects
  - Structural alteration is difficult – non-compositional
  - How to query/classify model behaviour – hand coded analyses?
- Solution?
  - Abstraction of the problem domain into a high level language
  - A declarative DSL reflecting the structure of biochemical processes
  - Compositional semantics
  - Compilation to ODEs for standard analyses
  - A logic for querying/classifying model behaviour
  - An efficient model checker

Motivation

- Current scientific practice:
  - Biochemical processes are modelled using ODEs
  - Tools like MATLAB used to write/solve/experiment with these models
- Challenges?
  - ODEs do not clearly reflect the domain objects
  - Structural alteration is difficult – non-compositional
  - How to query/classify model behaviour – hand coded analyses?
- Solution?
  - Abstraction of the problem domain into a high level language
  - A declarative DSL reflecting the structure of biochemical processes
  - Compositional semantics
  - Compilation to ODEs for standard analyses
  - A logic for querying/classifying model behaviour
  - An efficient model checker

The Language

- Cπ (continuous π-calculus)
  - A continuous time, continuous state-space, process calculus
  - Based on π-calculus with some alterations
  - A minimal language for biochemical processes.

$c\pi$ syntax

Species syntax

$$\begin{aligned} \text{(Species) } A, B &::= 0 \mid \sum_{i=0}^{n} \pi_i.S_i \mid A|B \mid (\nu M)A \\ \text{(Prefix) } \pi &::= a(\vec{x};\vec{y}) \mid \tau @ k \\ S &::= D(\vec{a}) \mid A \end{aligned}$$

Process syntax

$$\begin{aligned} \text{(Process) } P, Q &::= c \cdot A \mid P||Q \\ &\quad \text{where } c \in \mathbb{R} \text{ is the species concentration} \\ \text{(Affinity Network) } M &::= \text{a labelled undirected graph: } a \overset{k}{-} b \\ &\quad \text{where } a, b \text{ are sites and } k \in \mathbb{R} \text{ is a reaction rate} \end{aligned}$$

cπ example

Enzyme catalysed reaction

$$S + E \rightleftarrows ES \rightarrow E + P$$

$c\pi$ example

Enzyme catalysed reaction

$$S + E \rightleftarrows ES \rightarrow E + P$$

$$S \triangleq s(u, r).(u.S + r.P)$$

$$E \triangleq (\nu\{{}_x\angle^u_r\})e\langle u, r\rangle.x.E$$

cπ example

Enzyme catalysed reaction

$$S + E \rightleftarrows ES \rightarrow E + P$$

$$S \triangleq s(u, r).(u.S + r.P)$$

$$E \triangleq (\nu\{{}_x\angle^u_r\})e\langle u, r\rangle.x.E$$

$$ES \equiv (\nu\{{}_x\angle^u_r\})(x.E|(u.S + r.P))$$

$c\pi$ example

Enzyme catalysed reaction

$$S + E \rightleftarrows ES \rightarrow E + P$$

$$S \triangleq s(u, r).(u.S + r.P)$$

$$E \triangleq (\nu\{{}_x\angle^u_r\})e\langle u, r\rangle.x.E$$

$$P \triangleq 0$$

$$\Pi \triangleq c_s \cdot S(s) || c_e \cdot E(e) || c_p \cdot P()$$

$$Aff = \{s - e\}$$

cπ example

Syntax

Compiler

ODEs

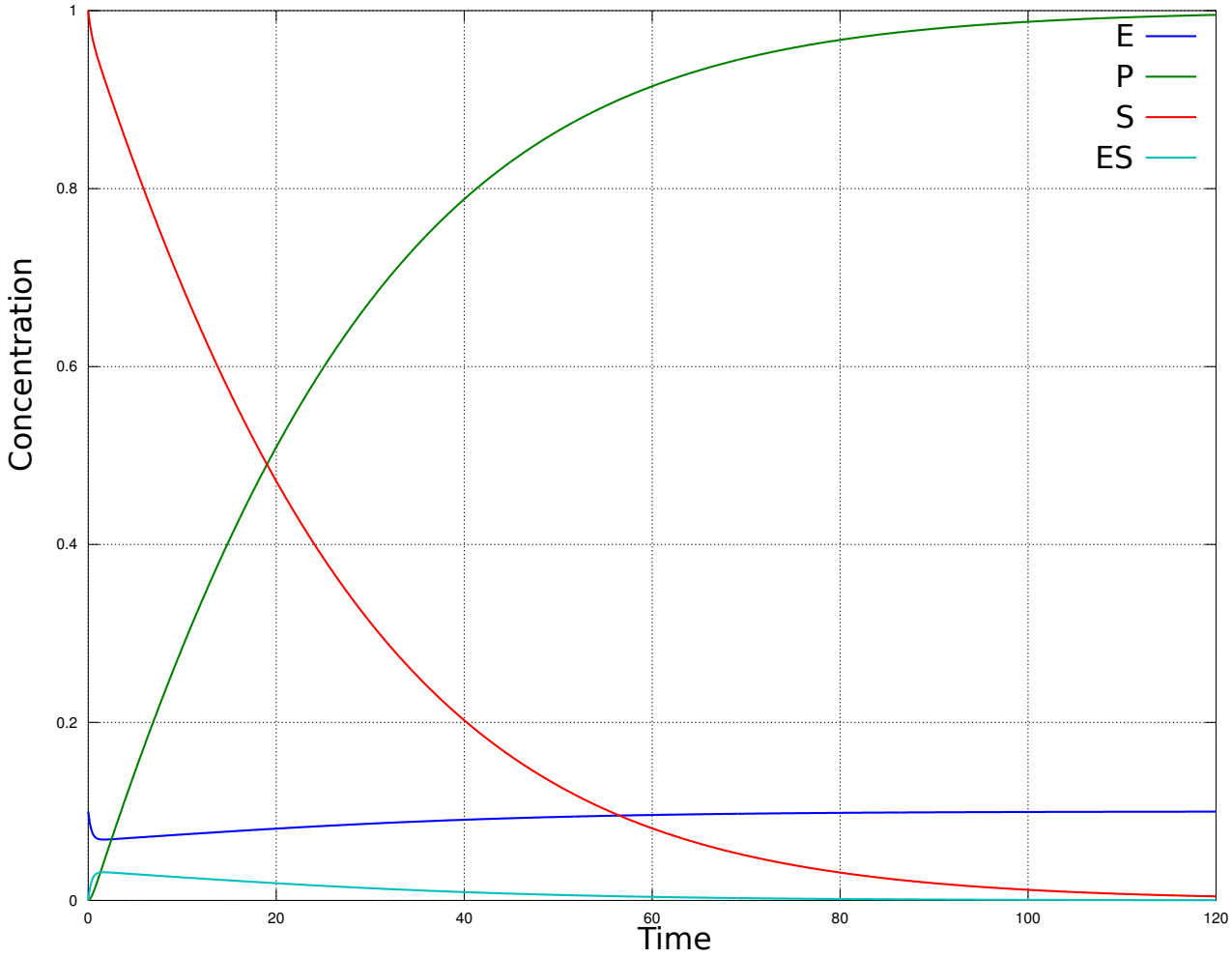
Solver

Time Series

Plotter

Plots

cπ example



$c\pi$ features

- Language reflects the structure of biochemical processes:
  - Prefix actions – reaction sites
  - Affinity network – complex reaction capabilities
  - Restriction – internal capabilities of complexes
- Compositional semantics – easier to construct large models from components
- Compilation to ODEs for standard analysis
- Computationally less expensive for large models than other approaches
  - than e.g. approaches using stochastic simulation or exhaustive state-space exploration

A logic for $c\pi$

- Formal system allows computational techniques – model checking
- Requirements?
  - Temporal logic
  - Real valued constraints
  - Contextual properties
- $LTL(\mathbb{R})$
  - LTL is sufficient for deterministic processes
  - $LTL(\mathbb{R})$ for real valued constraints on species concentrations
- Guarantee operator from spatial logic
  - $\phi \triangleright \psi$
  - $Q \triangleright \psi$

Logic examples

- $\mathbf{F}([A] \geq c)$
- $\mathbf{G}([A] > 0)$
- $\mathbf{G}\left(\frac{d[A]}{dt} + \frac{d[B]}{dt} = 0\right)$
- $Q \triangleright \mathbf{F}([A] \geq c)$
- $\mathbf{G}([A] < c) \wedge Q \triangleright \mathbf{F}([A] \geq c)$
- $\mathbf{G}(Q \triangleright \mathbf{F}([A] < c))$

Logic syntax

$$
\begin{aligned}
\text{(Formula) } \phi, \psi \quad &::= \quad \mathit{Atom} \mid \phi \wedge \psi \mid \phi \vee \psi \mid \phi \Rightarrow \psi \mid \neg\phi \mid \\
&\qquad \phi \mathbf{U} \psi \mid \mathbf{F}\phi \mid \mathbf{G}\phi \mid \\
&\qquad Q \triangleright \phi \\
\mathit{Atom} \quad &::= \quad \top \mid \bot \mid \mathit{Val}\ \mathit{BOp}\ \mathit{Val} \\
\mathit{Val} \quad &::= \quad v \in \mathbb{R} \mid [S] \mid \frac{d[S]}{dt} \mid \mathit{Val}\ \mathit{AOp}\ \mathit{Val} \\
\mathit{BOp} \quad &::= \quad > \mid < \mid \geqslant \mid \leqslant \\
\mathit{AOp} \quad &::= \quad + \mid - \mid \times \mid \div
\end{aligned}
$$

Logic semantics

For P, a cπ process:

$$
\begin{array}{lcl}
P \models \phi & \iff & \phi \text{ is an atomic proposition which is true of } P \\
P \models \phi \wedge \psi & \iff & P \models \phi \text{ and } P \models \psi \\
P \models \neg\phi & \iff & P \not\models \phi \\
P \models \phi \mathbf{U} \psi & \iff & \exists t'.\big((P^{t'} \models \psi) \wedge (\forall (t < t').P^{t} \models \phi)\big) \\
P \models Q \triangleright \phi & \iff & (Q \parallel P) \models \phi
\end{array}
$$

where

- P^t, is the process reached from process P after time t;
- Q is a cπ process with any new global affinity network

Model checking

- A numerical ODE solver gives us a *Trace* of the form:

$$(t_0, \vec{c_0}, \vec{c_0'}), (t_1, \vec{c_1}, \vec{c_1'}), \ldots$$

- A model checking algorithm is then easily expressed as a functional program:

```
check ::  Trace → Formula → Bool

check (t:ts) (atom) =    valid atom t
check (t:ts) (φ ∧ ψ) =   (check (t:ts) φ) AND (check (t:ts) ψ)
check (t:ts) (φ ∨ ψ) =   (check (t:ts) φ) OR (check (t:ts) ψ)
check (t:ts) (¬φ) =      NOT(check (t:ts) φ)
check (t:ts) (φ U ψ) =   (check (t:ts) ψ) OR
                         ((check (t:ts) φ) AND (check (ts) (φ U ψ)))
check (t:ts) (Q ▷ φ) =   check (solve (compose Q (proc (t:ts)))) φ

solve ::  Process → Trace
compose ::  Process → Process → Process
proc ::  Trace → Process
```

Tools

- cπ workbench – implementation in Haskell
  - cπ interactive interpreter
  - transition system exploration
  - ODE output
  - ODE solvers
  - graph plotting
  - logic and model checker

Future work

- Time bounds on temporal modalities:
  - $\mathbf{F}_{t<30}\phi$
  - better semantics – $\mathbf{G}\phi$ unprovable, $\mathbf{F}\phi$ unfalsifiable in general.
  - more automated control of solver
- Reduce calls to the numerical solver:
  - solver is the bottleneck
  - can some class of formulae be solved symbolically?
- Convergence properties:
  - finding process convergence (within some tolerance)
  - equilibrium / steady state concentrations of species
  - default natural time bound for temporal modalities.
- Continued work on tool support / UI.
- Friendly syntax?

Fin

- Acknowledgements:
  - Ian Stark (supervisor)
  - Marek Kwiatkowski (cπ)

- Bibliography:

Kwiatkowski, M., & Stark, I.
The continuous π-calculus: A process algebra for biochemical modelling.
In *Proc. Computational Methods in Systems Biology '08*, LNCS 5307, pp.103122. Springer-Verlag, 2008.

Kwiatkowski, M.
A formal computational framework for the study of molecular evolution
Ph.D. Thesis, University of Edinburgh, 2010.

Kwiatkowski, M., & Stark, I.
On Executable Models of Molecular Evolution.
In *Proc. WCSB'11*, pp.105-108, 2011