Model checking a continuous state-space process algebra.

Chris Banks

PEPA club

7th Oct, 2011

Introduction

This talk will be a summary of my thesis proposal and recent work:

- continuous π-calculus,
- the logic (or at least a stab in the dark),
- some examples of the logic,
- model checking: checking ODE numerical solutions,
- future work,
- software: tool support so far and planned.

Warning: contains more questions than answers!

Continuous π-calculus

- A process algebra for modelling the evolutionary variation of biochemical systems.
- Continuous-time, continuous-state, semnantics – equivalent to a set of ODEs.

Kwiatkowski, M. (2010). *A formal computational framework for the study of molecular evolution*. PhD Thesis, University of Edinburgh.

Continuous π-calculus – Syntax

- The set *Spec* of *species* is defined by the following grammar:

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

- A *prefix* is a *communication prefix* of the form $a(\vec{x};\vec{y})$ or a *silent prefix* of the form $\tau@k$ where $k \in \mathbb{R}$. Denoted: π, π', π_i, etc.
- An *affinity network* is a finite undirected graph where vertices are names and edges are labelled with $k \in \mathbb{R}$. Denoted: M, N, K, etc.

Continuous π-calculus – Syntax

- The set $\mathcal{P}$ of *processes* is defined by the following grammar:

$$\text{(Process) } P, Q \quad ::= \quad c \cdot A \;\mid\; P||Q$$

- Each species in a process has an initial concentration $c \in \mathbb{R}$.

Continuous π-calculus – Species Semantics

- The set of *concretions* *Conc* is defined by the following grammar:

$$\text{(Concretion) } F \quad ::= \quad (\vec{b};\vec{y})A \;\mid\; F|A \;\mid\; A|F \;\mid\; (\nu M)F$$

- A concretion is essentially a species which is committed to a specific potential interaction.
- If two concretions (F, G) are compatible $F \downarrow G$ then they can interact by pseudoapplication $F \circ G$ to form a new species.

$$(\vec{a};\vec{x})A \circ (\vec{b};\vec{y})B = A\{\vec{b}/\vec{x}\}|B\{\vec{a}/\vec{y}\}$$

$$\vdots$$

Continuous π-calculus – Species Semantics

A multiset of transitions *Tran* is generated by the SOS rules:

$$\frac{\pi_j = a_j(\vec{b}_j; \vec{y}_j)A_j}{\sum_{i=0}^{n} \pi_i.A_i \xrightarrow{a_j} (\vec{b}_j; \vec{y}_j)A_j} \quad Sum1(j, n)$$

$$\frac{\pi_j = \tau @ k}{\sum_{i=0}^{n} \pi_i.A_i \xrightarrow{\tau @ k} A_j} \quad Sum2(j, n)$$

$$\frac{A \xrightarrow{a} F \qquad B \xrightarrow{b} G \qquad F \downarrow G}{A|B \xrightarrow{\tau\langle a,b\rangle} F \circ G} \quad Com1$$

$$\frac{A \xrightarrow{\tau\langle a,b\rangle} B \qquad a, b \in M}{(\nu M)A \xrightarrow{\tau @ M(a,b)} (\nu M)B} \quad Com2$$

$$\frac{B \xrightarrow{\alpha} E \qquad D(\vec{y}) \triangleq B}{D(\vec{b}) \xrightarrow{\alpha\{\vec{b}/\vec{y}\}} E\{\vec{b}/\vec{y}\}} \quad Defn$$

$$\frac{A \xrightarrow{\alpha} E}{A|B \xrightarrow{\alpha} E|B} \quad Par - Left$$

$$\frac{B \xrightarrow{\alpha} E}{A|B \xrightarrow{\alpha} A|E} \quad Par - Right$$

$$\frac{A \xrightarrow{\alpha} E \qquad \alpha \notin M}{(\nu M)A \xrightarrow{\alpha} (\nu M)E} \quad Res1$$

$$\frac{A \xrightarrow{\tau\langle a,b\rangle} E \qquad a, b \notin M}{(\nu M)A \xrightarrow{\tau\langle a,b\rangle} (\nu M)E} \quad Res2$$

Continuous π-calculus – Vector Semantics

- The established approach to giving process algebra a continuous dynamics (e.g. in Bio-PEPA, BioCHAM) is a direct translation from the syntactic process to ODEs.
- However, as ODE descriptions are non-compositional, i.e. we cannot derive the ODEs for a system from the ODEs of its sub-systems, without extra information.
- cπ is therefore given a vector semantics which encodes this extra information and gives us a compositional model.
- ODEs can then be derived from the model in case we want to use standard solvers.

Continuous π-calculus – Vector Semantics

- The *process space* $\mathbb{P}$ is the vector space $\mathbb{R}^{(\mathcal{S}^{\#})}$.
- Every process is identified with a vector in $\mathbb{P}$.
- $\mathbb{P}$ is the *phase space* of cπ systems, where:
  - every dimension refers to a prime species,
  - every point determines the state of the system.
- The dynamic evolution of a cπ system is a continuous trajectory in $\mathbb{P}$.
- We specify the trajectory by giving the gradient vector $\frac{dP}{dt}$ for every point.
- This is equivalent to a set of ODEs with an initial state.
- To give a compositional model we need information on potential interactions. . .

Continuous π-calculus – Vector Semantics

- The *space of potentials* $\mathbb{D}$ is the vector space $\mathbb{R}^{(\mathcal{S}^{\#} \times \mathcal{C} \times \mathcal{N})}$.
- The extra information we need about potential interactions is encoded in the *interaction potential* $\partial P \in \mathbb{D}$ which is associated with every process.
- ∂P is an encoding of the potential interactions with information about species concentrations.

Continuous π-calculus – Vector Semantics – composition

- If we have two systems P and Q:
  - $\frac{dP}{dt} \in \mathbb{P}$ and $\partial P \in \mathbb{D}$.
  - $\frac{dQ}{dt} \in \mathbb{P}$ and $\partial Q \in \mathbb{D}$.
- The *immediate behaviour* of P composed with Q:

$$\frac{d(P||Q)}{dt} \triangleq \frac{dP}{dt} + \frac{dQ}{dt} + \partial P \oplus \partial Q$$

The logic – Syntax

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

The logic – Semantics

For P, a process in $\mathcal{P}$, the following satisfaction relation defines the proposed logic:

$$
\begin{aligned}
P \vDash \phi \quad &\iff \phi \text{ is an atomic proposition which is true of } P \\
P \vDash \phi \wedge \psi \quad &\iff P \vDash \phi \text{ and } P \vDash \psi \\
P \vDash \neg\phi \quad &\iff P \nvDash \phi \\
P \vDash \phi \mathbf{U} \psi \quad &\iff \exists t'.\big((P^{t'} \vDash \psi) \wedge (\forall (t < t').P^{t} \vDash \phi)\big) \\
P \vDash \Pi \triangleright \phi \quad &\iff (\Pi \parallel P) \vDash \phi
\end{aligned}
$$

where

- the function $T : \mathcal{P} \times \mathbb{R} \mapsto \mathcal{P}$ where $T(P, t)$, denoted P^t, is the process reached from process P after time t;
- $\Pi \in \mathcal{P}$ equipped with the new global affinity network.

The logic - Derived symbols

The remaining propositional connectives can be derived in the normal way and the remaining temporal operators can be defined as follows:

$$\mathbf{F}\phi \equiv \top \mathbf{U} \phi$$
$$\mathbf{G}\phi \equiv \neg \mathbf{F} \neg \phi$$
$$\phi \mathbf{W} \psi \equiv (\phi \mathbf{U} \psi) \vee \mathbf{G}\phi$$
$$\phi \mathbf{R} \psi \equiv \neg(\neg \phi \mathbf{U} \neg \psi)$$

The logic - Examples

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

Model checking

An immediate algorithm for model checking:

- extract ODEs from the cπ model,
- solve numerically and produce a trace,
- compute satisfaction using the constraint LTL algorithm of:

Calzone, L., Chabrier-Rivier, N., Fages, F., & Soliman, S. (2006). *Machine learning biochemical networks from temporal logic properties.* Transactions on Computational Systems Biology VI, 6894. Springer.

Model checking – constraint LTL algorithm

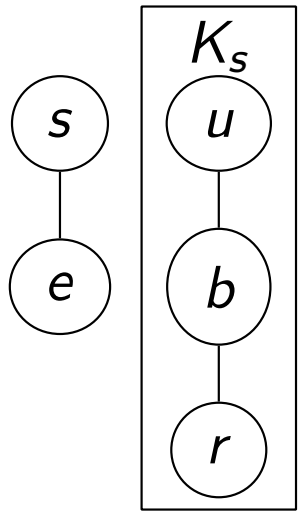
- A trace T is a finite linear Kripke structure with n real-valued states:
- $\langle t_0, \vec{s}_0, \frac{d\vec{s}_0}{dt} \rangle \rightarrow \langle t_1, \vec{s}_1, \frac{d\vec{s}_1}{dt} \rangle \rightarrow \ldots \rightarrow \langle t_{n-1}, \vec{s}_{n-1}, \frac{d\vec{s}_{n-1}}{dt} \rangle$
- To check that a formula ϕ in the logic holds each time-point of the trace is labelled with the sub-formulae of ϕ which hold.
- To adapt the algorithm to handle the guarantee $\Pi \triangleright \psi$ operator we need to re-compute the trace from each time-point with the new process Π in parallel.
- Note that when the guarantee is at the top level of ϕ then we only need to re-compute the whole trace once with Π in parallel, then check for ψ.
- This algorithm is fairly naive and there are plenty of optimisations that follow from it. E.g. when computing $\mathbf{F}\phi$ we can stop as soon as we find a witness.

Example

A simple cπ model:

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

$$S(s) \triangleq (\nu K_s)s\langle b\rangle.(u.S(s) + r.P())$$
$$E(e) \triangleq e(x).x.E(e)$$
$$P() \triangleq 0$$



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

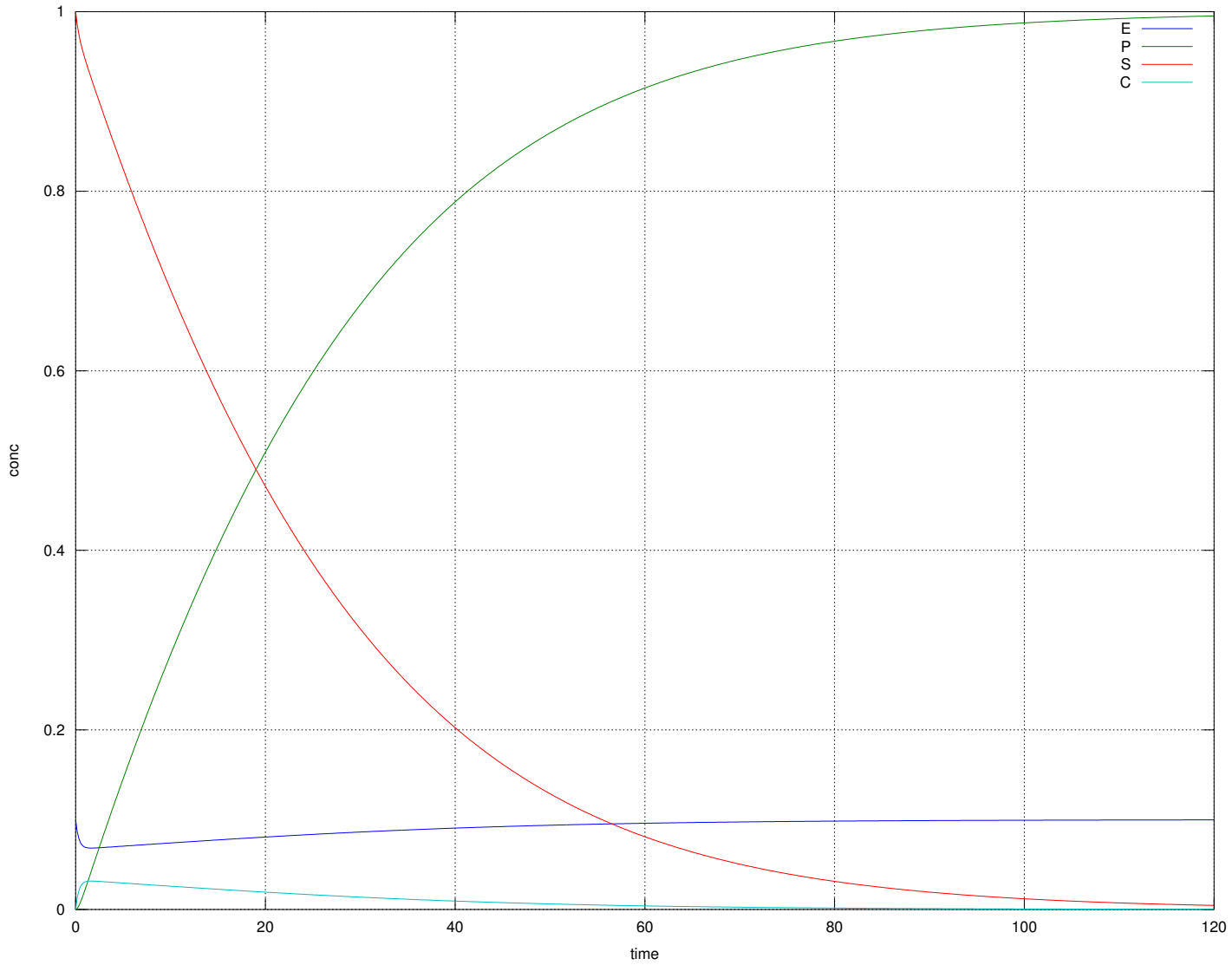
$$ES \equiv (\nu K_s)(b.E(e)|(u.S(s) + r.P()))$$

Example

ODEs:

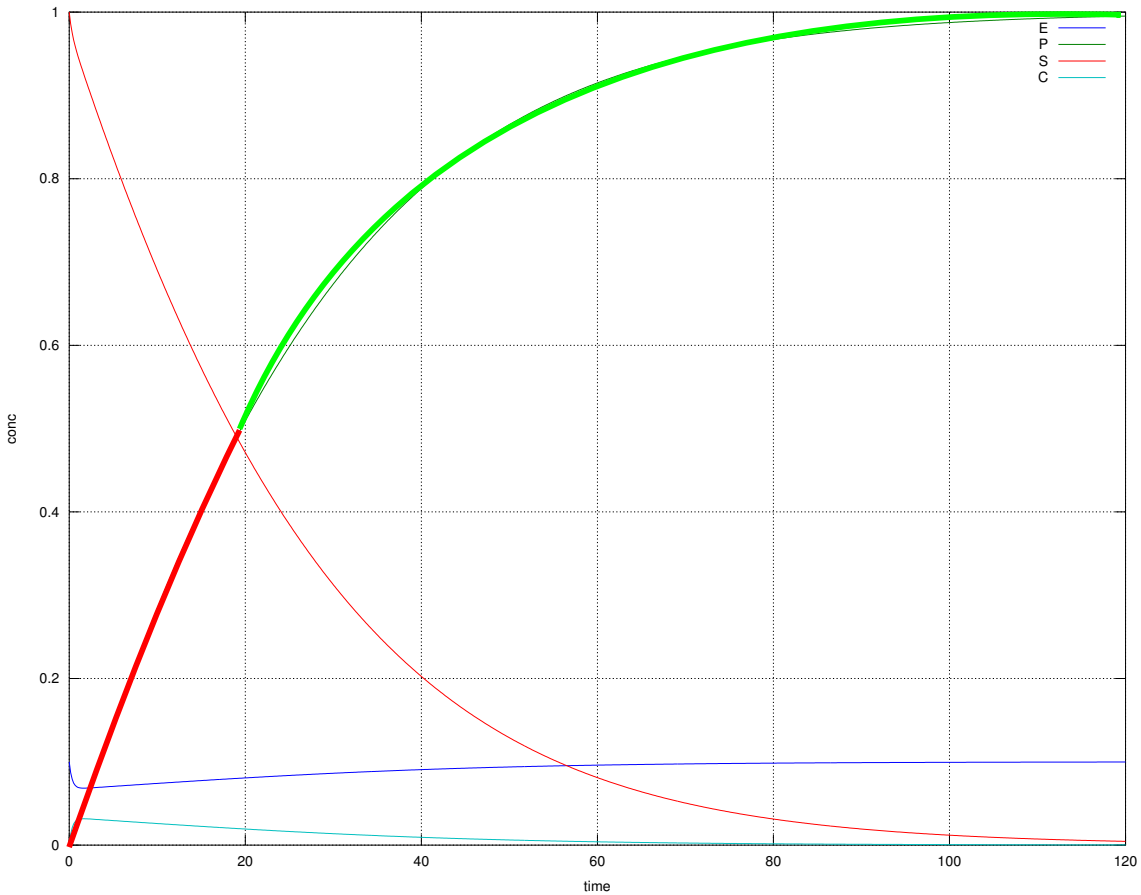
$$\begin{aligned}
\frac{d[S]}{dt} &= -r_b[E][S] + r_u[ES] \\
\frac{d[E]}{dt} &= -r_b[E][S] + r_u[ES] + r_r[ES] \\
\frac{d[P]}{dt} &= r_r[ES] \\
\frac{d[ES]}{dt} &= r_b[E][S] - r_u[ES] - r_r[ES]
\end{aligned}$$

Example



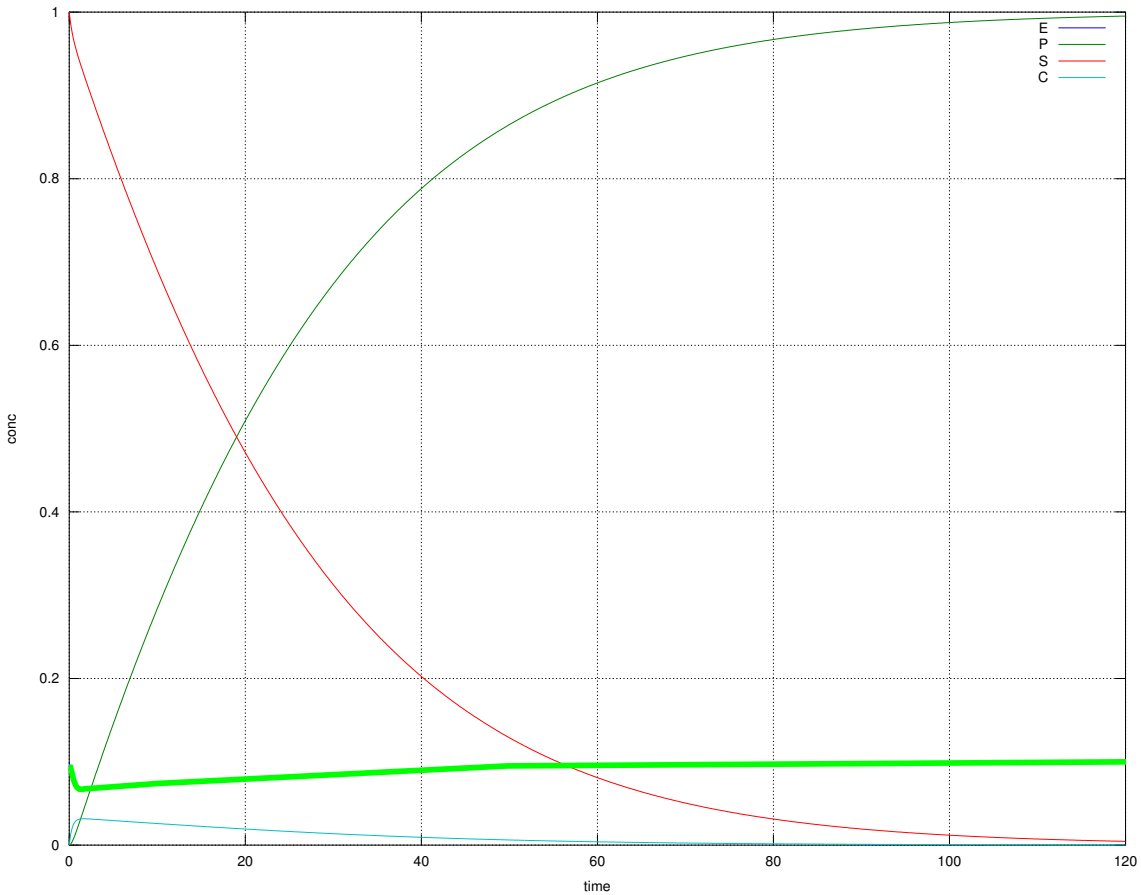
Example – Model checking

$$\mathbf{F}([P] > 0.5)$$



Example – Model checking

$$\mathbf{G}([E] > 0)$$



Example – Model checking – Guarantee

- We wish to introduce an inhibitor

$$I + E \leftrightarrows EI$$

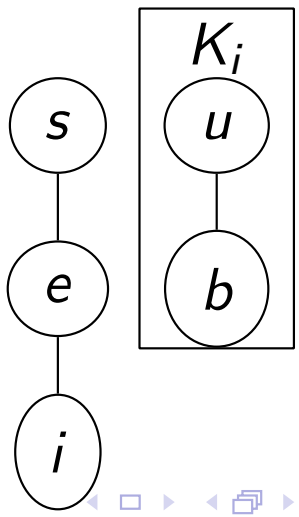
- resulting in the new system:

$$S + EI \leftrightarrows S + E + I \leftrightarrows ES + I \rightarrow P + E + I$$

- We can achieve this by using the guarantee operator with the original system:

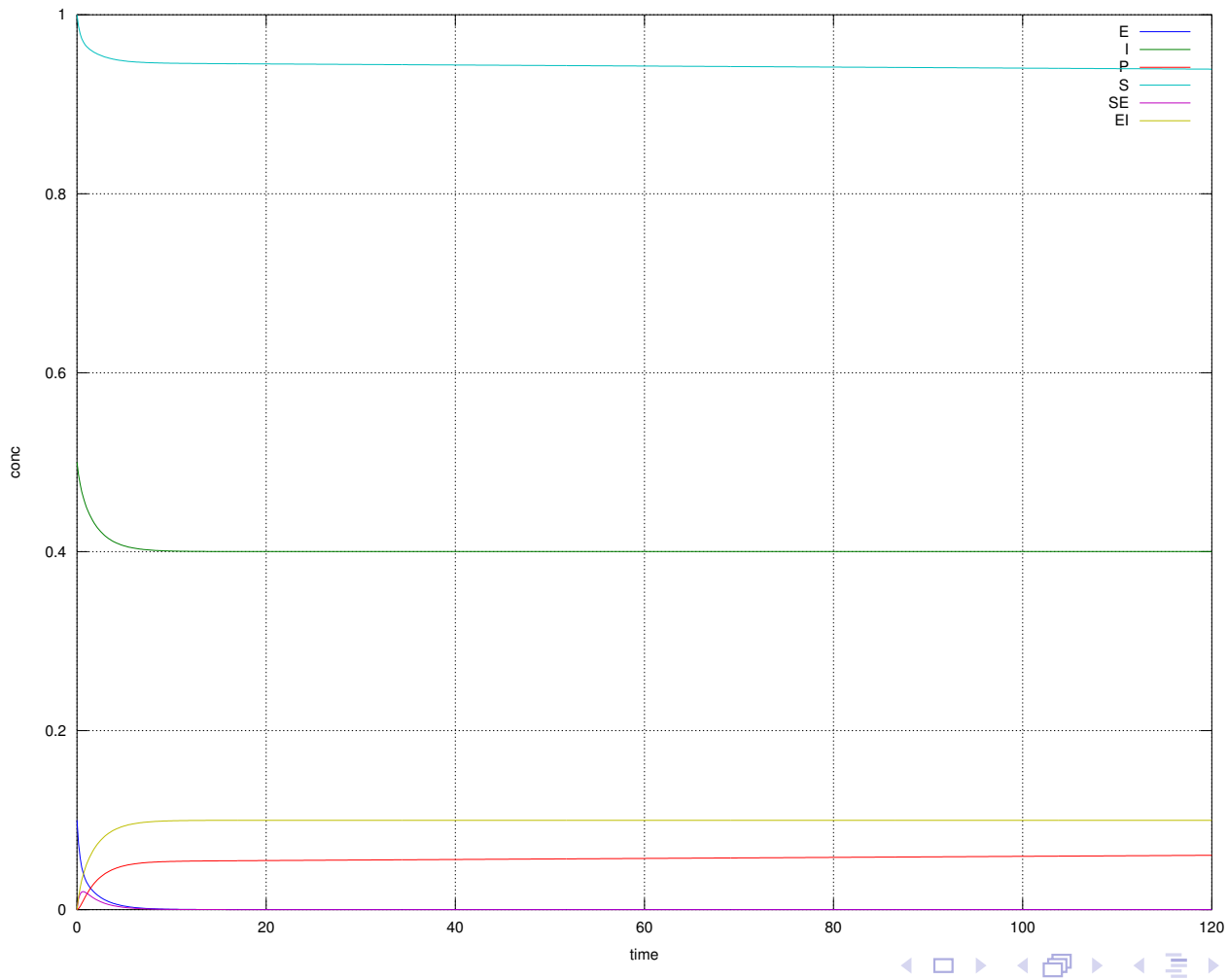
$$c \cdot I(i) \triangleright (\mathbf{G}([P] < 0.1))$$

where $I(i) \triangleq (\nu K_i) i\langle b \rangle.u.I(i)$



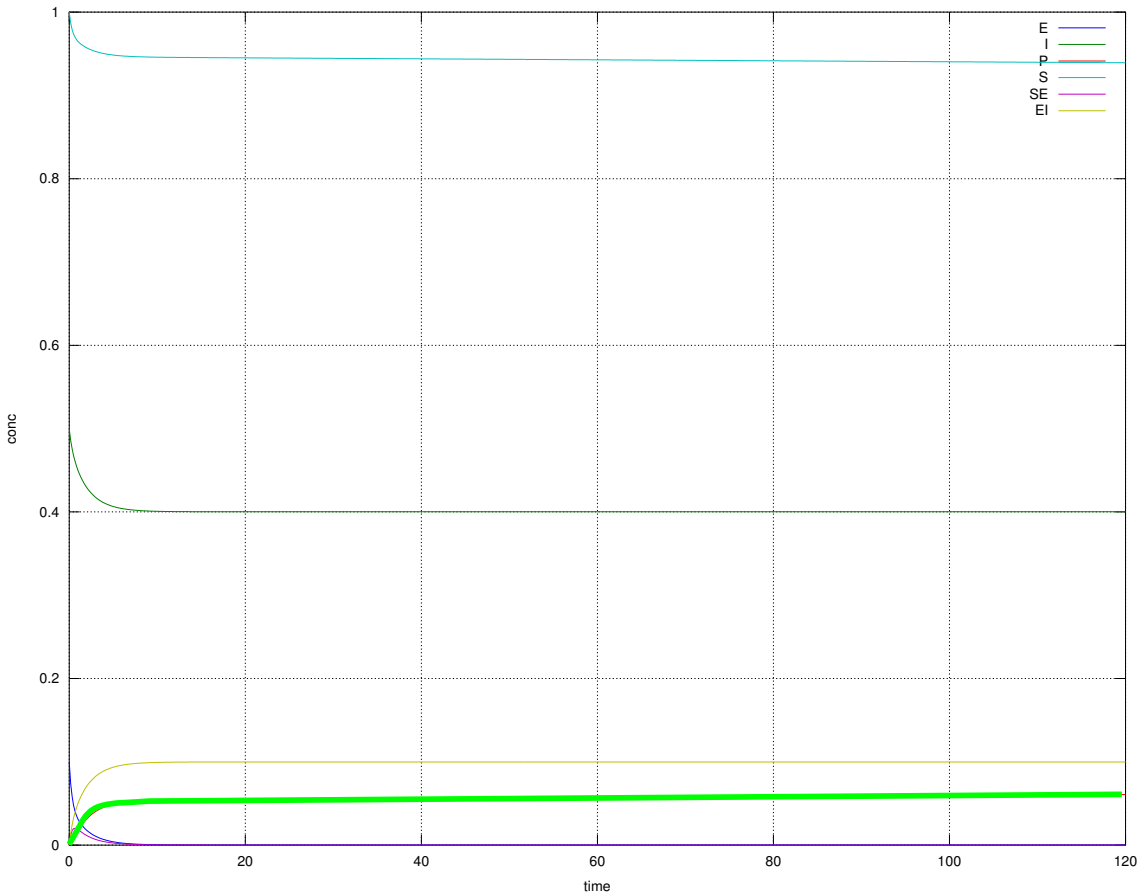
Example – Model checking – Guarantee

$$\Pi||I$$



Example – Model checking – Guarantee

$$I(i) \triangleright (\mathbf{G}([P] < 0.1))$$



Future work

Future work:

- bigger models, more complex properties,
- direct model checking of $c\pi$ vectors – or how to avoid solving ODEs,
- compositionality, parallelism,
- tool support.

A note on direct model checking

- At the moment it is possible solve ODEs numerically and do model checking on time series.
- This is inefficient because traces contain much more information than we might actually need.
- It is especially inefficient for guarantee, because we need to go back and re-solve the ODE after introducing the new species.
- Consider $\mathbf{G}(\Pi \rhd \phi)$ . . . ??
- I want to look at ways of solving the model checking problem over the cπ vector model directly, with a view to improving efficiency.
- Also, numerical solutions are a sample over a finite time period, thus making $\mathbf{G}\phi$ unprovable (although falsifiable) and $\mathbf{F}\phi$ unfalsifiable (although provable) in general. Maybe an analytical approach could help with this?

A note on direct model checking – Examples

Some easy things we could prove algebraically, without solving ODEs:

- $\mathbf{G}([A] \leq \mathit{init}([A]))$
- Invariants: sum of derivatives.

What else? Can we find a class of formulae we prove things about without solving?

A note on compositionality

NB: this is very speculative, not very well thought out, and likely to be quite hard, but would be very nice!

- $S \in \mathbb{P}$ and $\frac{dS}{dt} \in \mathbb{P}$ have corresponding basic formulae $[S]$ and $\frac{d[S]}{dt}$,
- but $\partial P \in \mathbb{D}$ does not.
- Consider a formula $[S \xrightarrow{a} C]$ species S, concretion C and name a, where $Q \vDash [S \xrightarrow{a} C] > k \iff \partial Q(S, a, C) > k$.
- This could be used to give structurally inductive evaluations for $[S]$ and $\frac{d[S]}{dt}$ and make use of the compositionality of cπ models.
- Using the knowledge of potential interactions in $\mathbb{D}$, we know which could contribute to $[S]$.

Tool support

Re-implemented Marek's cπ tool with a "workbench" style REPL interface, in Haskell:

- for my own understanding of the system,
- to do more than just take a cπ system and output ODEs,
- and to make the tool more amenable to extension and experimentation.

To do:

- finish implementing ODE solutions for cπ models,
  - numerical solutions using the `hmatrix` Haskell module,
  - graphical output with GNUplot Haskell bindings,
- implement Fages/Rizk's algorithm for checking numerical traces,
- automated experimentation – parameter perturbation etc.,
- implement a GUI
  - Gtk2Hs + Glade?
  - Java + Eclipse plugin?

- Thank you for staying awake.
- Questions?