

Generating SDEs from PEPA models

Dr. Jeremy Bradley *jb@doc.ic.ac.uk*

Mr. Richard Hayden *rah03@doc.ic.ac.uk*

Imperial College London

PASTA 2007

Motivation

Massively parallel computer systems with very *complex interplay* between components are fast becoming ubiquitous.

From the massive heterogeneous server clusters behind the likes of Google ...

... to the revolution in peer-to-peer file transfer

It's only going to get *more complicated*, at least from a *performance evaluation* perspective ...

Why does performance matter?

...or why do I care how long my download takes?

Systems have to be able to make high-level *quantitative guarantees* such as:

- A search engine responds to search requests in < 500ms average, 2000ms worst case
- The positions of aircraft on a radar screen will be updated at least every second

Solution: abstraction (as ever!)

We model a simplified version using a formalism, such as a:

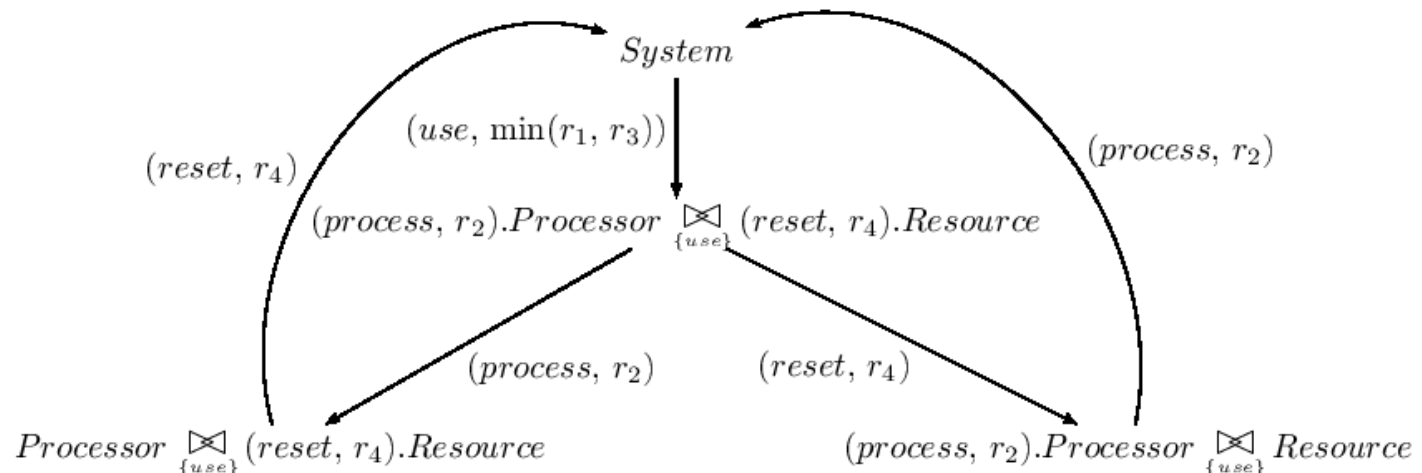
- Stochastic Petri net (SPN)
- Queueing network
- **Stochastic process algebra (SPA) ... e.g. PEPA**

We then 'solve' it and map what we learn on to the real-world problem.

Simple PEPA example

$$\text{Processor} \stackrel{\text{def}}{=} (\text{use}, r_1).(\text{process}, r_2).\text{Processor}$$

$$\text{Resource} \stackrel{\text{def}}{=} (\text{use}, r_3).(\text{reset}, r_4).\text{Resource}$$

$$\text{System} \stackrel{\text{def}}{=} \text{Processor} \bowtie_{\{\text{use}\}} \text{Resource}$$


Bigger, more realistic example

$$\text{System} \stackrel{\text{def}}{=} \underbrace{(\text{Processor} \parallel \dots \parallel \text{Processor})}_{N_P} \underset{\{use\}}{\bowtie} \underbrace{(\text{Resource} \parallel \dots \parallel \text{Resource})}_{N_R}$$

$$N_P = 50, N_R = 50 \Rightarrow 2^{100} \text{ states}$$

‘Solving’ the underlying CTMC means (generating and) solving a system of 2^{100} linear equations ...

Bigger, more realistic example

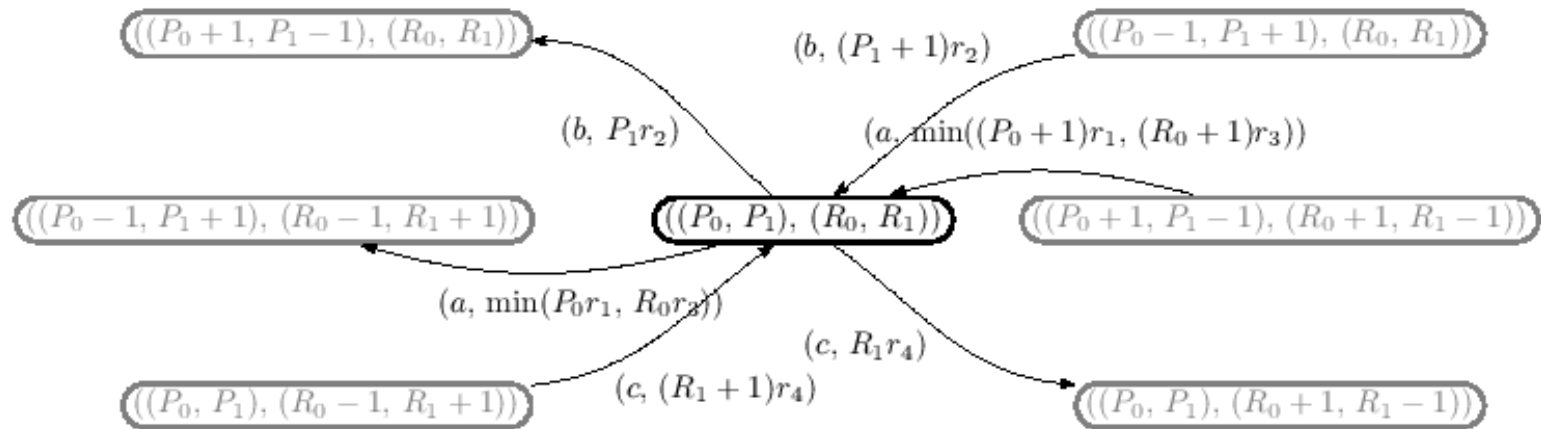
$$\text{System} \stackrel{\text{def}}{=} \underbrace{(\text{Processor} \parallel \dots \parallel \text{Processor})}_{N_P} \underset{\{use\}}{\bowtie} \underbrace{(\text{Resource} \parallel \dots \parallel \text{Resource})}_{N_R}$$

$$N_P = 50, N_R = 50 \Rightarrow 2^{100} \text{ states}$$

‘Solving’ the underlying CTMC means (generating and) solving a system of 2^{100} linear equations ...

This is the *state space explosion* problem

ODE continuous approximation



$$\frac{dP_0(t)}{dt} = -\min(P_0(t)r_1, R_0(t)r_2) + r_2P_1(t)$$

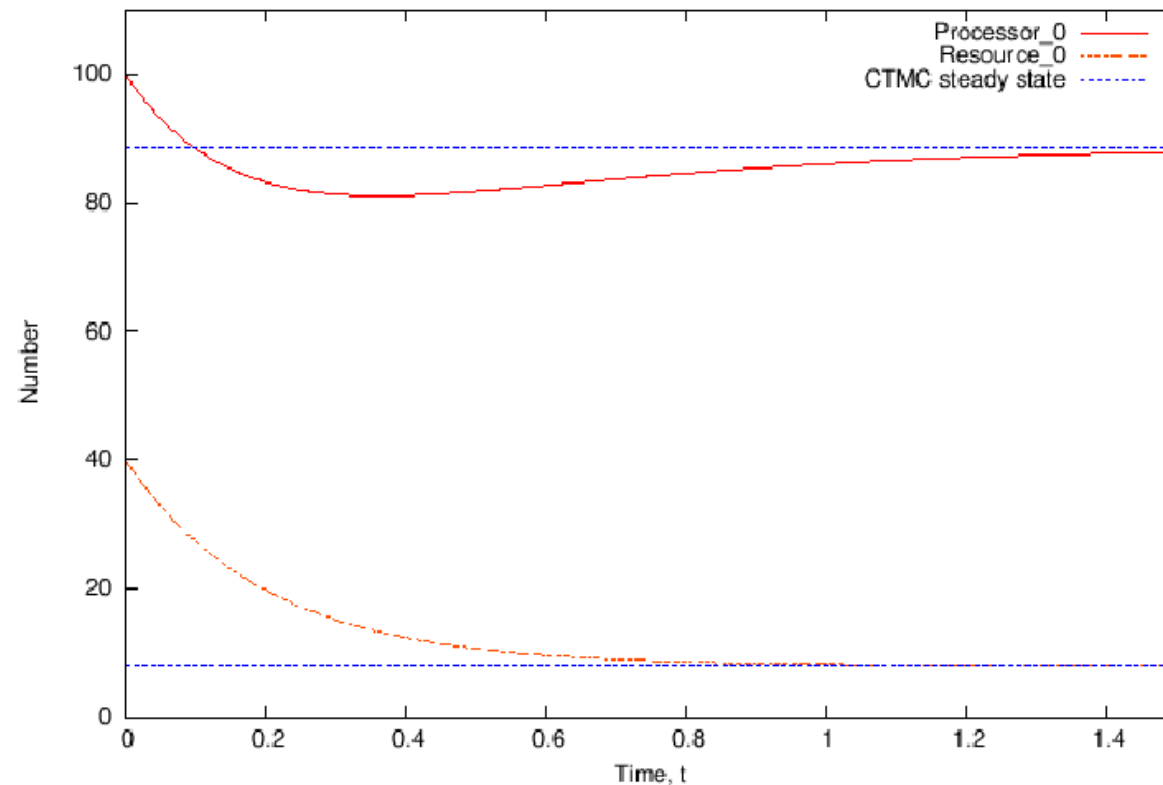
$$\frac{dP_1(t)}{dt} = \min(P_0(t)r_1, R_0(t)r_2) - r_2P_1(t)$$

$$\frac{dR_0(t)}{dt} = -\min(P_0(t)r_1, R_0(t)r_2) + r_4R_1(t)$$

$$\frac{dR_1(t)}{dt} = \min(P_0(t)r_1, R_0(t)r_2) - r_4R_1(t)$$

ODE continuous approximation

Intuitionist argument, but *seems* to work, at least in some cases ...



'Discrete' components

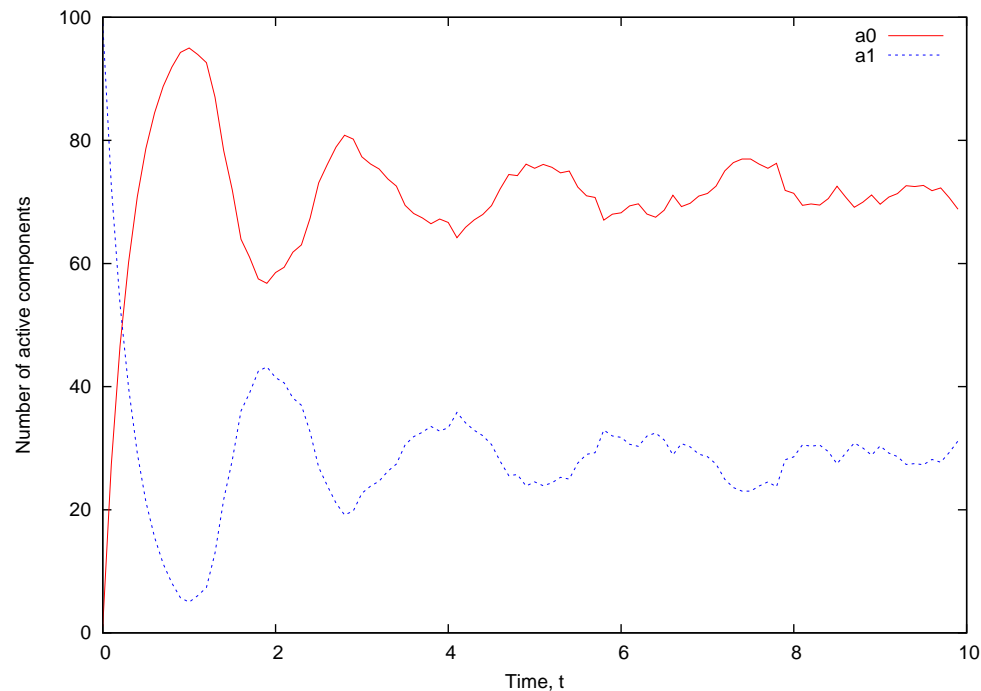
Albeit not in others ...

$$\begin{array}{l} A_0 \stackrel{def}{=} (a, \lambda).A_1 \\ A_1 \stackrel{def}{=} (b, \mu).A_0 \end{array} \quad Sys \stackrel{def}{=} \underbrace{\left(A_0 \begin{array}{c} \diagup \diagdown \\ \{a\} \end{array} A_0 \begin{array}{c} \diagup \diagdown \\ \{a\} \end{array} \dots \begin{array}{c} \diagup \diagdown \\ \{a\} \end{array} A_0 \right)}_{N_A}$$

'Discrete' components

Albeit not in others ...

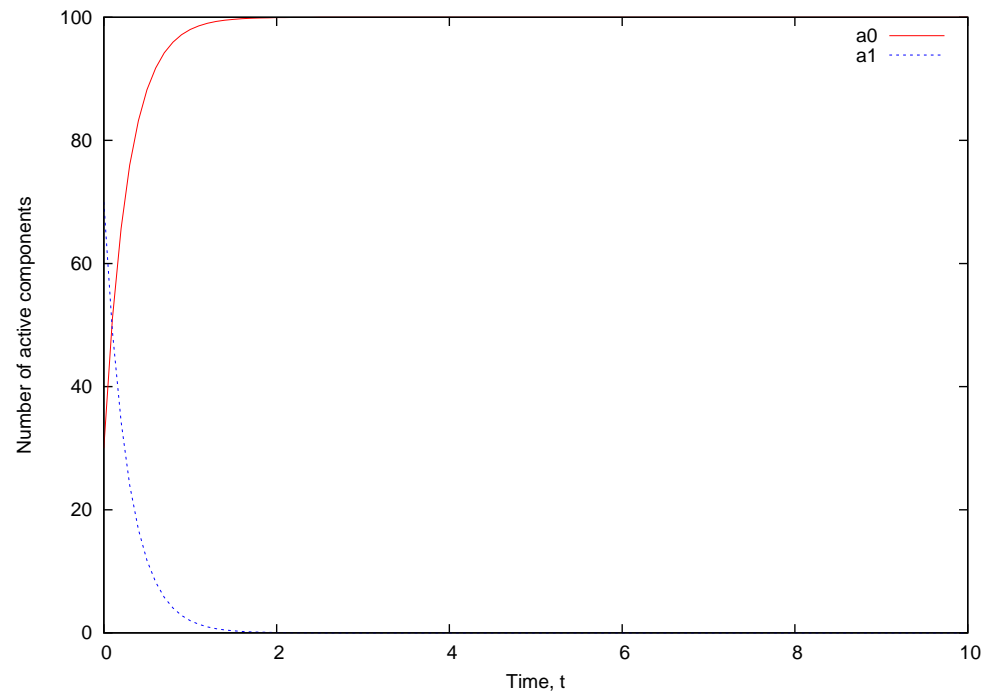
$$\begin{aligned} A_0 &\stackrel{\text{def}}{=} (a, \lambda).A_1 \\ A_1 &\stackrel{\text{def}}{=} (b, \mu).A_0 \end{aligned} \quad \text{Sys} \stackrel{\text{def}}{=} \underbrace{\left(A_0 \begin{array}{c} \diagup \diagdown \\ \{a\} \end{array} A_0 \begin{array}{c} \diagup \diagdown \\ \{a\} \end{array} \dots \begin{array}{c} \diagup \diagdown \\ \{a\} \end{array} A_0 \right)}_{N_A}$$



'Discrete' components

Albeit not in others ...

$$\begin{aligned} A_0 &\stackrel{\text{def}}{=} (a, \lambda).A_1 \\ A_1 &\stackrel{\text{def}}{=} (b, \mu).A_0 \end{aligned} \quad \text{Sys} \stackrel{\text{def}}{=} \underbrace{\left(A_0 \begin{array}{c} \bowtie \\ \{a\} \end{array} A_0 \begin{array}{c} \bowtie \\ \{a\} \end{array} \dots \begin{array}{c} \bowtie \\ \{a\} \end{array} A_0 \right)}_{N_A}$$



'Discrete' components

Why not?

$$\begin{array}{l}
 A_0 \stackrel{def}{=} (a, \lambda).A_1 \\
 A_1 \stackrel{def}{=} (b, \mu).A_0
 \end{array}
 \quad
 Sys \stackrel{def}{=} \underbrace{\left(A_0 \begin{array}{c} \diagup \diagdown \\ \{a\} \end{array} A_0 \begin{array}{c} \diagup \diagdown \\ \{a\} \end{array} \dots \begin{array}{c} \diagup \diagdown \\ \{a\} \end{array} A_0 \right)}_{N_A}$$

$$(A_0 \parallel \dots \parallel A_0) \xrightarrow{(a, \lambda)} (A_1 \parallel \dots \parallel A_1)$$

$$(A_1 \parallel A_0 \parallel \dots \parallel A_0) \not\xrightarrow{(a, \lambda)}$$

'Discrete' components

Why not?

$$\begin{array}{l}
 A_0 \stackrel{\text{def}}{=} (a, \lambda).A_1 \\
 A_1 \stackrel{\text{def}}{=} (b, \mu).A_0
 \end{array}
 \quad
 Sys \stackrel{\text{def}}{=} \underbrace{\left(A_0 \begin{array}{c} \diagup \diagdown \\ \{a\} \end{array} A_0 \begin{array}{c} \diagup \diagdown \\ \{a\} \end{array} \dots \begin{array}{c} \diagup \diagdown \\ \{a\} \end{array} A_0 \right)}_{N_A}$$

$$(A_0 \parallel \dots \parallel A_0) \xrightarrow{(a, \lambda)} (A_1 \parallel \dots \parallel A_1)$$

$$(A_1 \parallel A_0 \parallel \dots \parallel A_0) \not\xrightarrow{(a, \lambda)}$$

$$\frac{dA_0(t)}{dt} = \mu A_1(t) - \lambda N_A \mathbf{I}'(A_1(t))$$

$$\frac{dA_1(t)}{dt} = -\mu A_1(t) + \lambda N_A \mathbf{I}'(A_1(t))$$

Motivation

- Is possible to formally justify the generated ODEs for a subclass of the PEPA models, in at least two senses:
 - ◆ In a limiting sense as shown by the last speaker
 - ◆ In a transient sense using generating functions
- At least the second sense can also be generalised to higher-order moments

Motivation

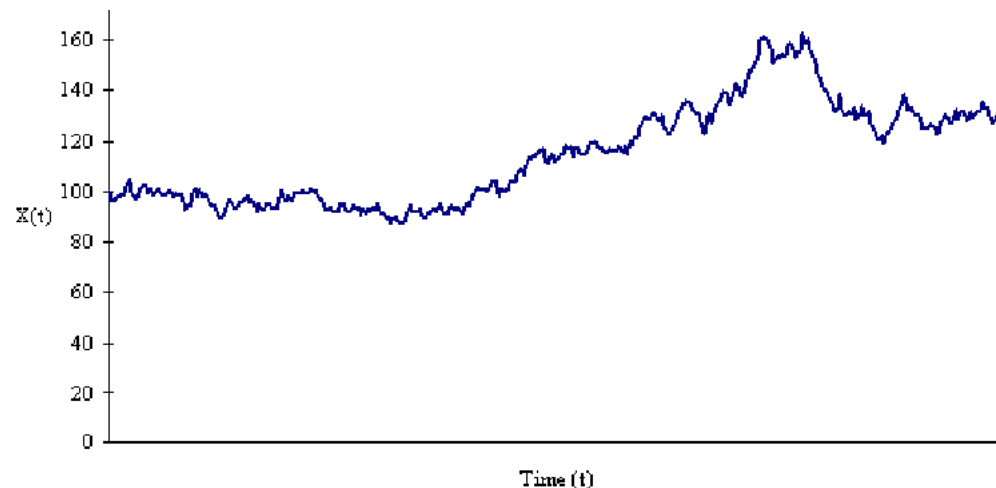
- Is possible to formally justify the generated ODEs for a subclass of the PEPA models, in at least two senses:
 - ◆ In a limiting sense as shown by the last speaker
 - ◆ In a transient sense using generating functions
- At least the second sense can also be generalised to higher-order moments
- Having identified the limitations of this kind of fluid-flow analysis (ODEs), can we address them with more powerful tools?

Stochastic differential equations

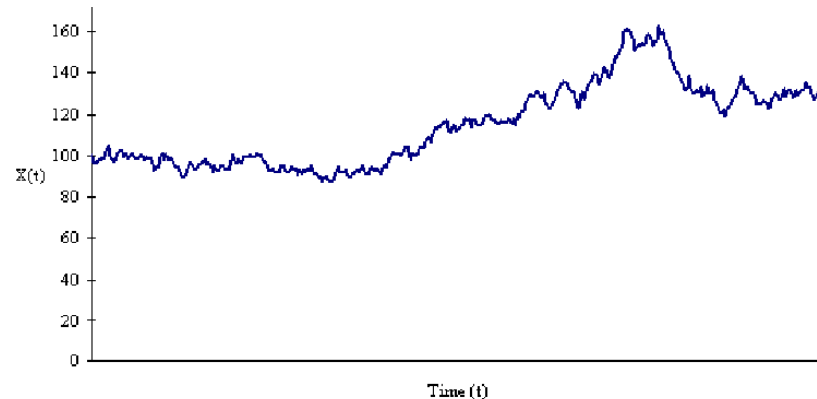
- Differential equations where *one or more terms is a stochastic process* and thus so are solutions
- *Random noise* often modelled using *Brownian motion*
- Analogous to *normal random variables*, can often be used to construct *functional central limit theorems*

Stochastic differential equations

- Differential equations where *one or more terms is a stochastic process* and thus so are solutions
- *Random noise* often modelled using *Brownian motion*
- Analogous to *normal random variables*, can often be used to construct *functional central limit theorems*

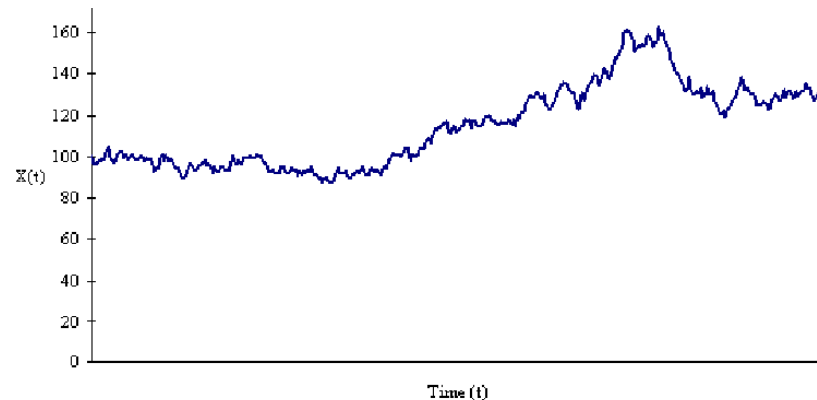


Brownian motion



- But *mathematically nasty* — almost always nowhere differentiable, infinite variation, not monotone on almost every interval, no matter how small(!)
- Necessitates different calculus — Itô calculus

Brownian motion

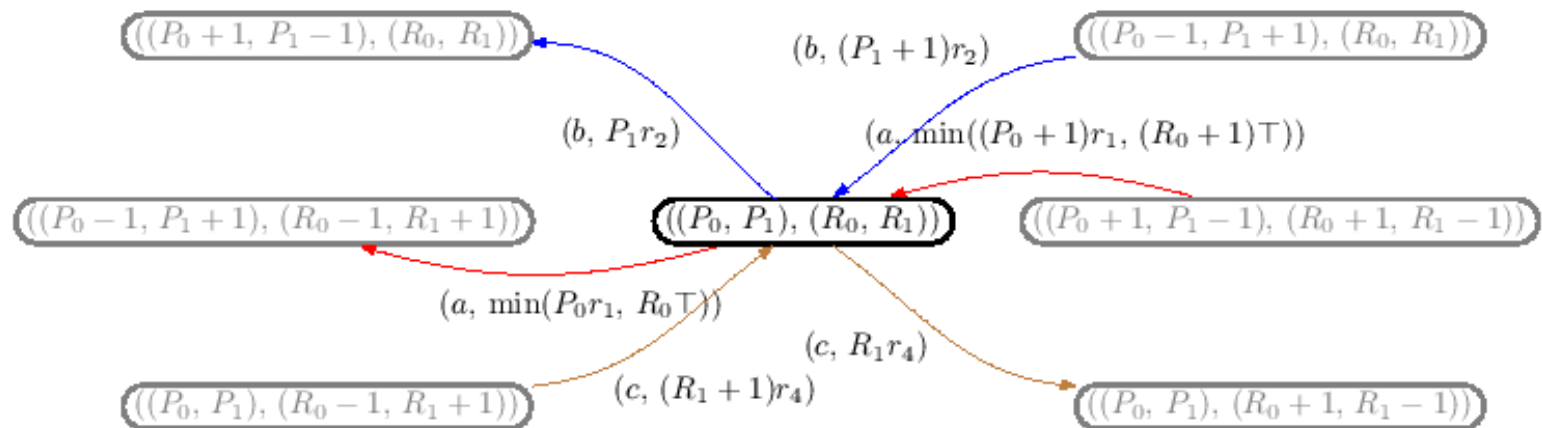


- But *mathematically nasty* — almost always nowhere differentiable, infinite variation, not monotone on almost every interval, no matter how small(!)
- Necessitates different calculus — Itô calculus

$$\int_0^t B(s) dB(s) = \frac{1}{2}B^2(t) - \frac{1}{2}t$$

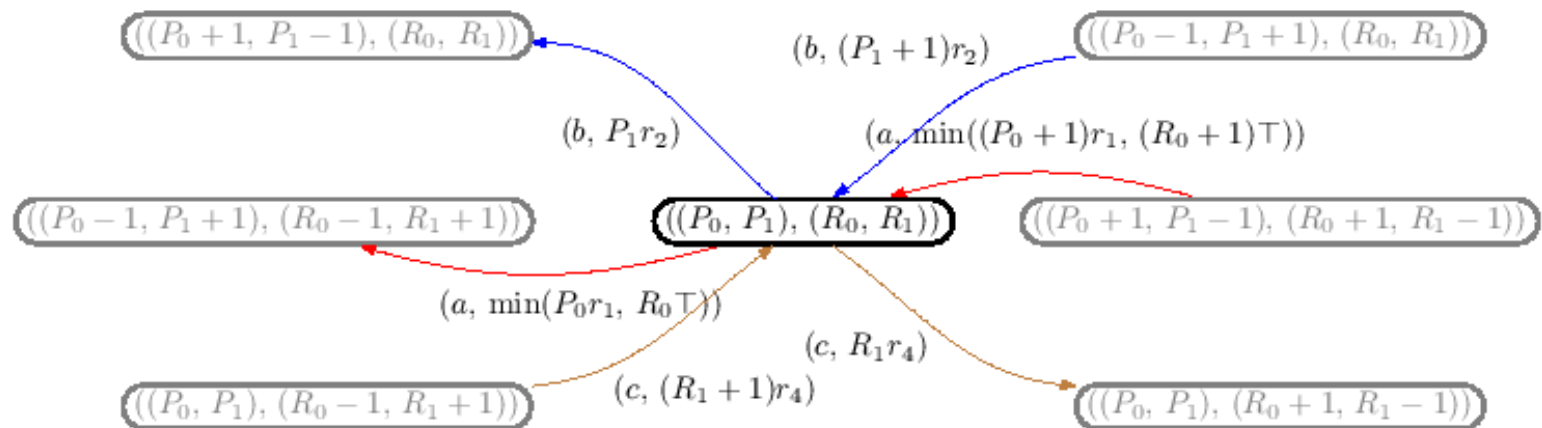
FCLT for PEPA?

Need to identify the stochastic processes to approximate. We define *transition counting processes* associated with PEPA models.



FCLT for PEPA?

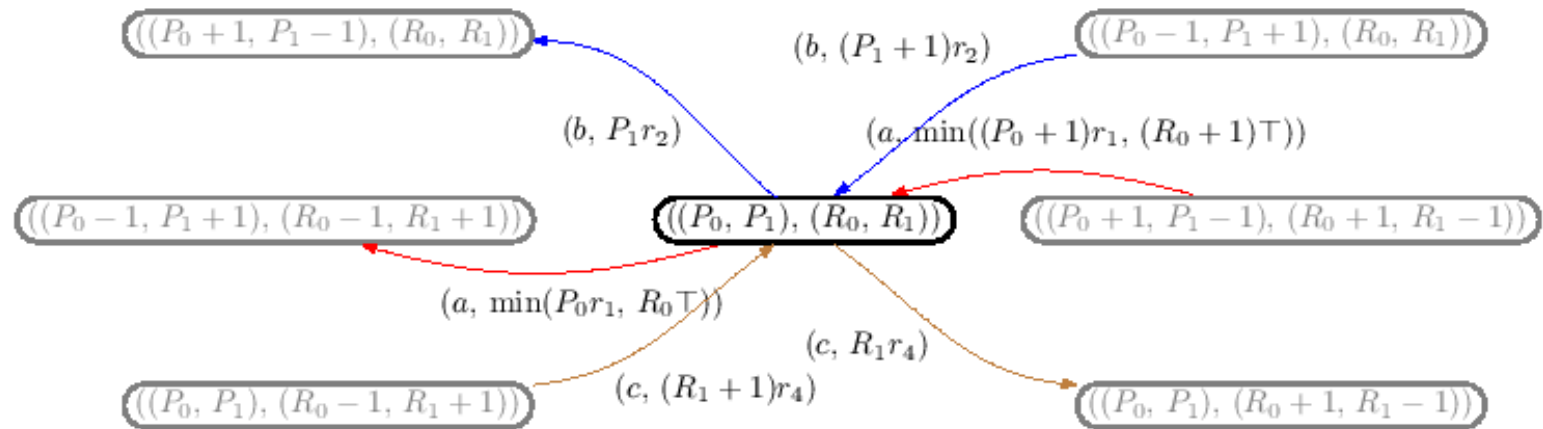
Need to identify the stochastic processes to approximate. We define *transition counting processes* associated with PEPA models.



$$p_0(t) = p_0(0) - N_1(t) + N_2(t) \quad r_0(t) = r_0(0) - N_1(t) + N_3(t)$$

$$p_1(t) = p_1(0) + N_1(t) - N_2(t) \quad r_1(t) = r_1(0) + N_1(t) - N_3(t)$$

FCLT for PEPA?



$$p_0(t) = p_0(0) - N_1(t) + N_2(t) \quad r_0(t) = r_0(0) - N_1(t) + N_3(t)$$

$$p_1(t) = p_1(0) + N_1(t) - N_2(t) \quad r_1(t) = r_1(0) + N_1(t) - N_3(t)$$

$$f_{-1,1,-1,1}(x_1, x_2, x_3, x_4) = \min(x_1 r_1, x_3 T) := \begin{cases} x_1 r_1 & \text{if } x_3 > 0 \\ 0 & \text{if } x_3 = 0 \end{cases}$$

$$f_{1,-1,0,0}(x_1, x_2, x_3, x_4) = x_2 r_2$$

$$f_{0,0,1,-1}(x_1, x_2, x_3, x_4) = x_4 r_4$$

FCLT for PEPA?

$$\begin{aligned}p_0(t) &= p_0(0) - N_1(t) + N_2(t) & r_0(t) &= r_0(0) - N_1(t) + N_3(t) \\p_1(t) &= p_1(0) + N_1(t) - N_2(t) & r_1(t) &= r_1(0) + N_1(t) - N_3(t)\end{aligned}$$

$$f_{-1,1,-1,1}(x_1, x_2, x_3, x_4) = \min(x_1 r_1, x_3 \top) := \begin{cases} x_1 r_1 & \text{if } x_3 > 0 \\ 0 & \text{if } x_3 = 0 \end{cases}$$

$$f_{1,-1,0,0}(x_1, x_2, x_3, x_4) = x_2 r_2$$

$$f_{0,0,1,-1}(x_1, x_2, x_3, x_4) = x_4 r_4$$

FCLT for PEPA?

$$\begin{aligned} p_0(t) &= p_0(0) - N_1(t) + N_2(t) & r_0(t) &= r_0(0) - N_1(t) + N_3(t) \\ p_1(t) &= p_1(0) + N_1(t) - N_2(t) & r_1(t) &= r_1(0) + N_1(t) - N_3(t) \end{aligned}$$

$$f_{-1,1,-1,1}(x_1, x_2, x_3, x_4) = \min(x_1 r_1, x_3 \top) := \begin{cases} x_1 r_1 & \text{if } x_3 > 0 \\ 0 & \text{if } x_3 = 0 \end{cases}$$

$$f_{1,-1,0,0}(x_1, x_2, x_3, x_4) = x_2 r_2$$

$$f_{0,0,1,-1}(x_1, x_2, x_3, x_4) = x_4 r_4$$

$$N_1(t) \Rightarrow \int_0^t \min(p_1(s)r_1, r_0(s)\top) ds + \int_0^t \sqrt{\min(p_1(s)r_1, r_0(s)\top)} dB_1(s)$$

FCLT for PEPA?

$$\begin{aligned}p_0(t) &= p_0(0) - N_1(t) + N_2(t) & r_0(t) &= r_0(0) - N_1(t) + N_3(t) \\p_1(t) &= p_1(0) + N_1(t) - N_2(t) & r_1(t) &= r_1(0) + N_1(t) - N_3(t)\end{aligned}$$

$$f_{-1,1,-1,1}(x_1, x_2, x_3, x_4) = \min(x_1 r_1, x_3 \top) := \begin{cases} x_1 r_1 & \text{if } x_3 > 0 \\ 0 & \text{if } x_3 = 0 \end{cases}$$

$$f_{1,-1,0,0}(x_1, x_2, x_3, x_4) = x_2 r_2$$

$$f_{0,0,1,-1}(x_1, x_2, x_3, x_4) = x_4 r_4$$

$$N_1(t) \Rightarrow \int_0^t \min(p_1(s)r_1, r_0(s)\top) ds + \int_0^t \sqrt{\min(p_1(s)r_1, r_0(s)\top)} dB_1(s)$$

$$N_2(t) \Rightarrow \int_0^t p_1(s)r_2 ds + \int_0^t \sqrt{p_1(s)r_2} dB_2(s)$$

FCLT for PEPA?

$$\begin{aligned}p_0(t) &= p_0(0) - N_1(t) + N_2(t) & r_0(t) &= r_0(0) - N_1(t) + N_3(t) \\p_1(t) &= p_1(0) + N_1(t) - N_2(t) & r_1(t) &= r_1(0) + N_1(t) - N_3(t)\end{aligned}$$

$$f_{-1,1,-1,1}(x_1, x_2, x_3, x_4) = \min(x_1 r_1, x_3 \top) := \begin{cases} x_1 r_1 & \text{if } x_3 > 0 \\ 0 & \text{if } x_3 = 0 \end{cases}$$

$$f_{1,-1,0,0}(x_1, x_2, x_3, x_4) = x_2 r_2$$

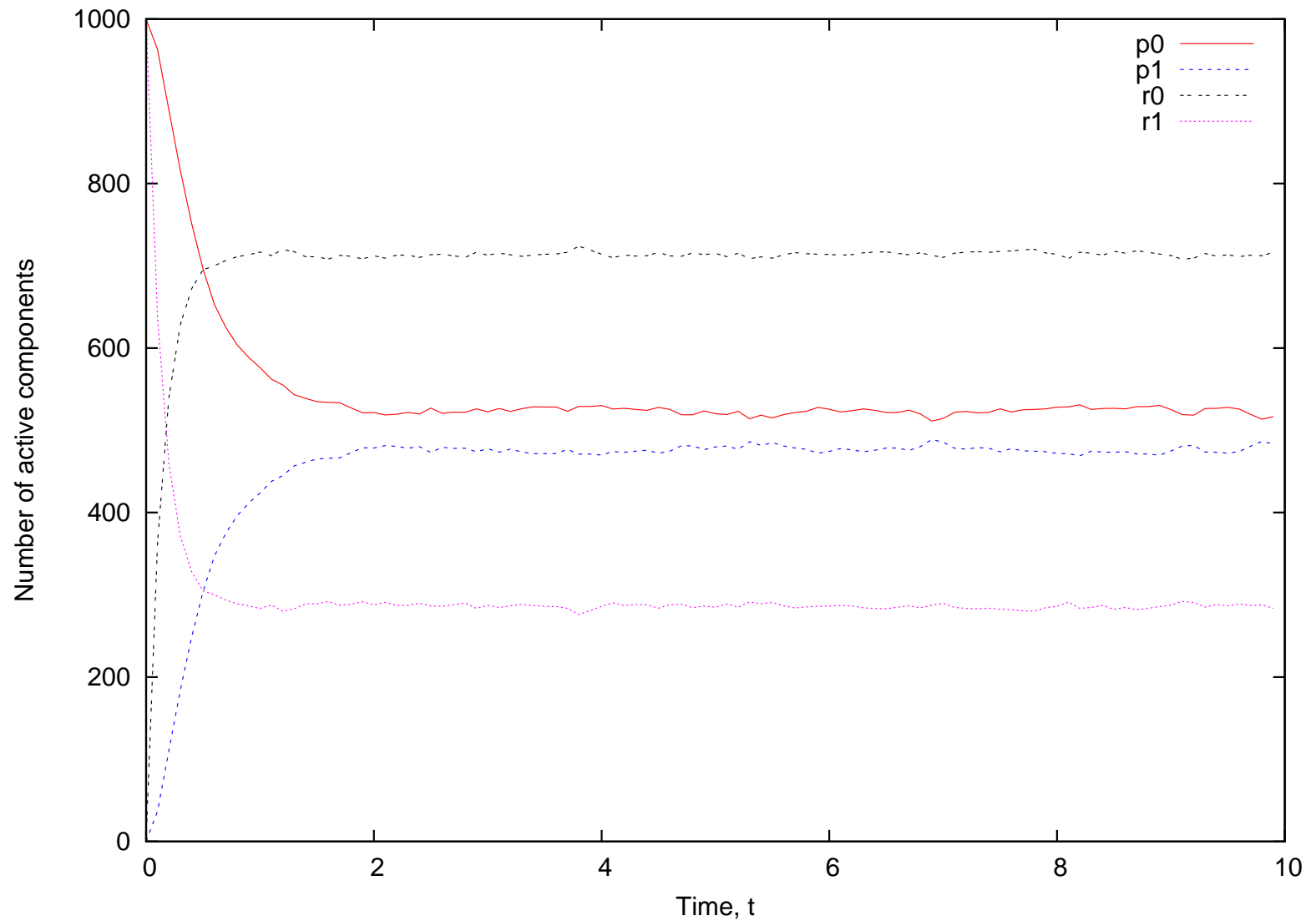
$$f_{0,0,1,-1}(x_1, x_2, x_3, x_4) = x_4 r_4$$

$$N_1(t) \Rightarrow \int_0^t \min(p_1(s)r_1, r_0(s)\top) ds + \int_0^t \sqrt{\min(p_1(s)r_1, r_0(s)\top)} dB_1(s)$$

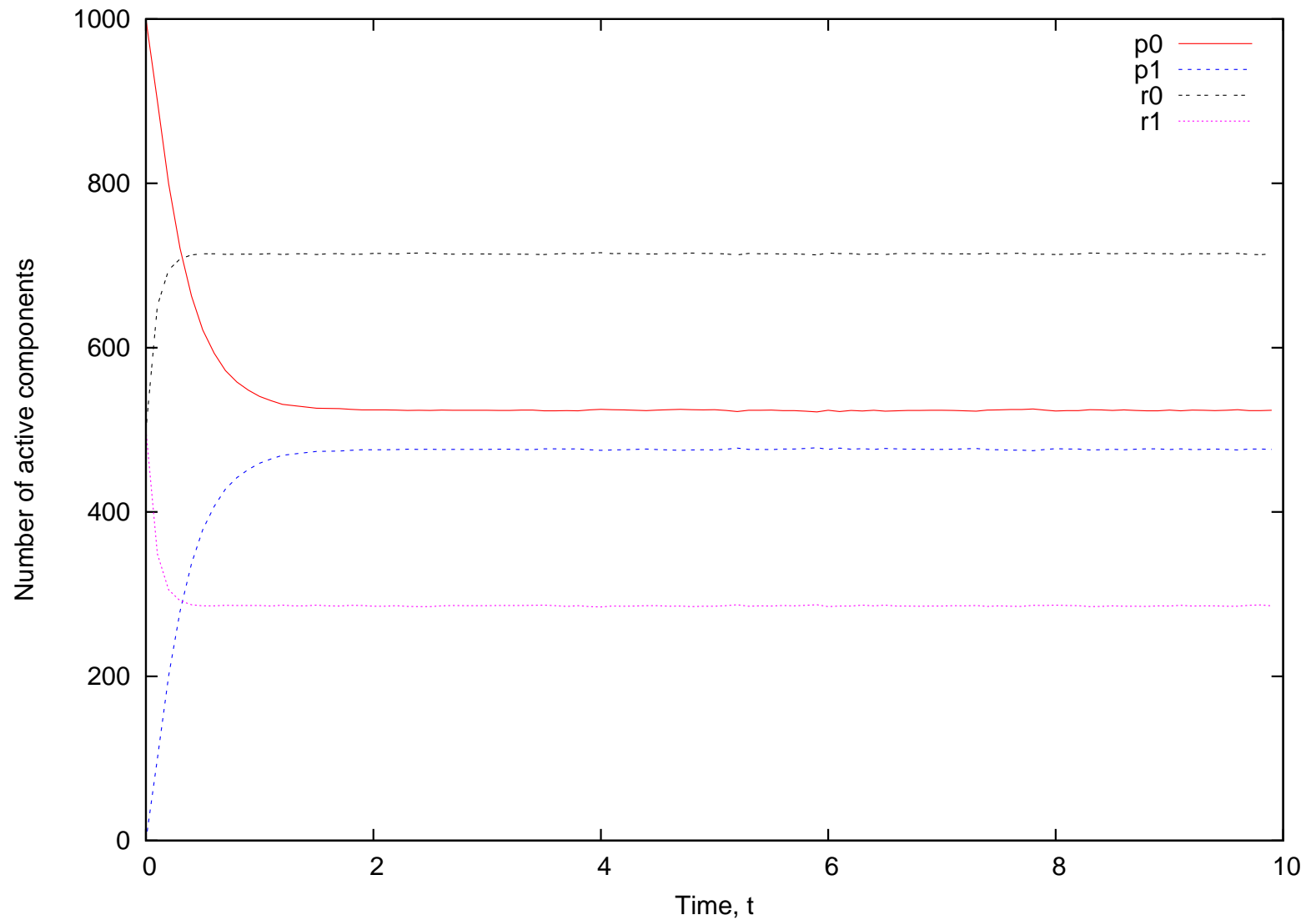
$$N_2(t) \Rightarrow \int_0^t p_1(s)r_2 ds + \int_0^t \sqrt{p_1(s)r_2} dB_2(s)$$

$$N_3(t) \Rightarrow \int_0^t r_1(s)r_4 ds + \int_0^t \sqrt{r_1(s)r_4} dB_3(s)$$

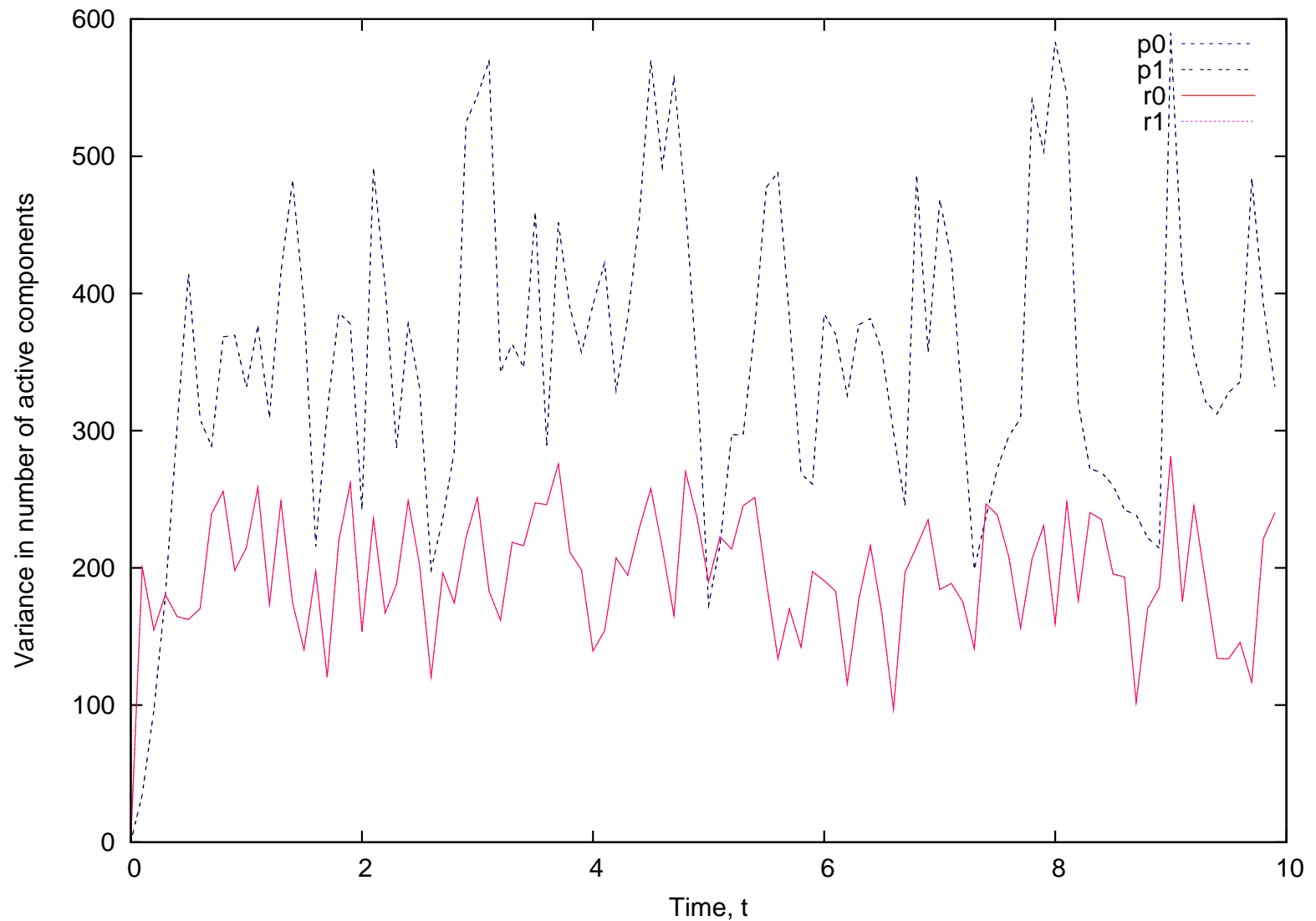
Simulation - expectations



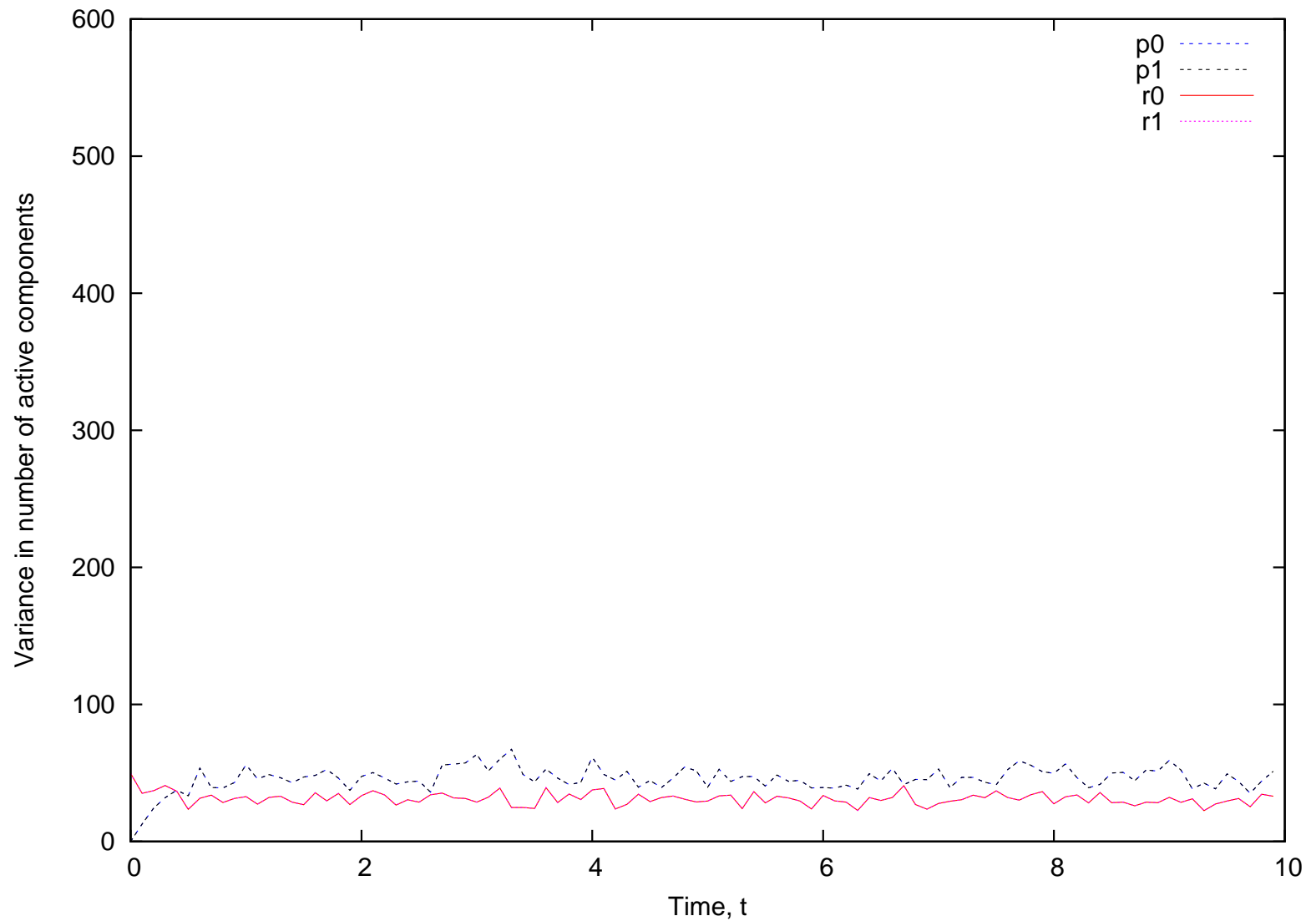
SDEs - expectations



Simulation - variances



SDEs - variances



FCLT for PEPA?

- Rates *need to be fast* for the approximation to work

FCLT for PEPA?

- Rates *need to be fast* for the approximation to work
- However, even when not, *trends seem to be reflected*

FCLT for PEPA?

- Rates *need to be fast* for the approximation to work
- However, even when not, *trends seem to be reflected*

But how do we deal with (pathologically) slow rates such as those caused by internal cooperation?

$$\begin{array}{l}
 A_0 \stackrel{\text{def}}{=} (a, \lambda).A_1 \\
 A_1 \stackrel{\text{def}}{=} (b, \mu).A_0
 \end{array}
 \quad
 Sys \stackrel{\text{def}}{=} \underbrace{\left(A_0 \begin{array}{c} \diagup \diagdown \\ \{a\} \end{array} A_0 \begin{array}{c} \diagup \diagdown \\ \{a\} \end{array} \dots \begin{array}{c} \diagup \diagdown \\ \{a\} \end{array} A_0 \right)}_{N_A}$$

$$f_{-N_A, N_A}(a_0, a_1) = \mathbf{I}'(x_2)\lambda$$

This is *discrete behaviour* — fluid-flow approximation here is not a good idea ...

Hybrid models

$$A_0 \stackrel{def}{=} (z, r_1).A_1 + (z, r_2).A_2$$

$$A_1 \stackrel{def}{=} (b, r_3).A_0$$

$$A_2 \stackrel{def}{=} (c, r_4).A_0$$

$$B_0 \stackrel{def}{=} (b, r_5).B_1$$

$$B_1 \stackrel{def}{=} (d, r_6).B_0 + (e, r_7).B_0$$

$$C_0 \stackrel{def}{=} (e, r_8).C_1$$

$$C_1 \stackrel{def}{=} (f, r_9).C_0$$

$$Sys \stackrel{def}{=} \left[\underbrace{\left(A_0 \begin{array}{c} \diagdown \diagup \\ \{z\} \end{array} \dots \begin{array}{c} \diagdown \diagup \\ \{z\} \end{array} A_0 \right)}_{N_A} \begin{array}{c} \diagdown \diagup \\ \{b\} \end{array} \underbrace{(B_0 \parallel \dots \parallel B_0)}_{N_B} \begin{array}{c} \diagdown \diagup \\ \{e\} \end{array} \underbrace{(C_0 \parallel \dots \parallel C_0)}_{N_C} \right]$$

Hybrid models

$$A_0 \stackrel{def}{=} (z, r_1).A_1 + (z, r_2).A_2$$

$$A_1 \stackrel{def}{=} (b, r_3).A_0$$

$$A_2 \stackrel{def}{=} (c, r_4).A_0$$

$$B_0 \stackrel{def}{=} (b, r_5).B_1$$

$$B_1 \stackrel{def}{=} (d, r_6).B_0 + (e, r_7).B_0$$

$$C_0 \stackrel{def}{=} (e, r_8).C_1$$

$$C_1 \stackrel{def}{=} (f, r_9).C_0$$

$$Sys \stackrel{def}{=} \left[\underbrace{\left(A_0 \begin{array}{c} \diagdown \diagup \\ \{z\} \end{array} \dots \begin{array}{c} \diagdown \diagup \\ \{z\} \end{array} A_0 \right)}_{N_A} \begin{array}{c} \diagdown \diagup \\ \{b\} \end{array} \underbrace{(B_0 \parallel \dots \parallel B_0)}_{N_B} \begin{array}{c} \diagdown \diagup \\ \{e\} \end{array} \underbrace{(C_0 \parallel \dots \parallel C_0)}_{N_C} \right]$$

$$f_{-N_A, (N_A-k), k, 0, 0, 0, 0}(x_1, x_2, x_3, x_4, x_5, x_6, x_7) = \mathbf{I}'(x_2)\mathbf{I}'(x_3) \left(\frac{1}{r_1 + r_2} \right)^{N_A-1} r_1^{N_A-k} r_2^k \binom{N_A}{k}$$

$$f_{0, 0, 0, 1, -1, 0, 0}(x_1, x_2, x_3, x_4, x_5, x_6, x_7) = x_5 r_6$$

Hybrid scheme

$$Sys \stackrel{def}{=} \left[\underbrace{\left(A_0 \bowtie_{\{z\}} \dots \bowtie_{\{z\}} A_0 \right)}_{N_A} \bowtie_{\{b\}} \underbrace{(B_0 \parallel \dots \parallel B_0)}_{N_B} \right] \bowtie_{\{e\}} \underbrace{(C_0 \parallel \dots \parallel C_0)}_{N_C}$$

$$x_1(t) = x_1(0) - N_A \sum_{i=0}^{N_A} M_i(t) + N_1(t) + N_2(t)$$

$$x_2(t) = x_2(0) + \sum_{i=0}^{N_A} (N_A - i) M_i(t) - N_1(t)$$

$$x_3(t) = x_3(0) + \sum_{i=0}^{N_A} i M_i(t) - N_2(t)$$

$$x_4(t) = x_4(0) - N_1(t) + N_3(t) + N_4(t)$$

$$x_5(t) = x_5(0) + N_1(t) - N_3(t) - N_4(t)$$

$$x_6(t) = x_6(0) - N_4(t) + N_5(t)$$

$$x_7(t) = x_7(0) + N_4(t) - N_5(t)$$

Hybrid scheme

$$\text{Sys} \stackrel{\text{def}}{=} \left[\underbrace{\left(A_0 \bowtie_{\{z\}} \dots \bowtie_{\{z\}} A_0 \right)}_{N_A} \bowtie_{\{b\}} \underbrace{(B_0 \parallel \dots \parallel B_0)}_{N_B} \right] \bowtie_{\{e\}} \underbrace{(C_0 \parallel \dots \parallel C_0)}_{N_C}$$

$$x_1(t) = x_1(0) - N_A \sum_{i=0}^{N_A} M_i(t) + N_1(t) + N_2(t)$$

$$x_2(t) = x_2(0) + \sum_{i=0}^{N_A} (N_A - i) M_i(t) - N_1(t)$$

$$x_3(t) = x_3(0) + \sum_{i=0}^{N_A} i M_i(t) - N_2(t)$$

$$x_4(t) = x_4(0) - N_1(t) + N_3(t) + N_4(t)$$

$$x_5(t) = x_5(0) + N_1(t) - N_3(t) - N_4(t)$$

$$x_6(t) = x_6(0) - N_4(t) + N_5(t)$$

$$x_7(t) = x_7(0) + N_4(t) - N_5(t)$$

Can we fluid-flow approximate the $N_i(t)$ using our FCLT and treat the $M_i(t)$ discretely?

Hybrid scheme

- Can we fluid-flow approximate the $N_i(t)$ using our FCLT and treat the $M_i(t)$ discretely?

Hybrid scheme

- Can we fluid-flow approximate the $N_i(t)$ using our FCLT and treat the $M_i(t)$ discretely?
- Not obviously — can't simulate jump times for $M_i(t)$ because their *rates change continuously* ...

Hybrid scheme

- Can we fluid-flow approximate the $N_i(t)$ using our FCLT and treat the $M_i(t)$ discretely?
- Not obviously — can't simulate jump times for $M_i(t)$ because their *rates change continuously* ...
- However, *similar problems arise in the area of financial modelling* (specifically LIBOR markets) — we adapt these methods to PEPA models ...

Thinning

- Crucial point is being able to bound the rate of the $M_i(t)$ - possible because *PEPA models are finite*

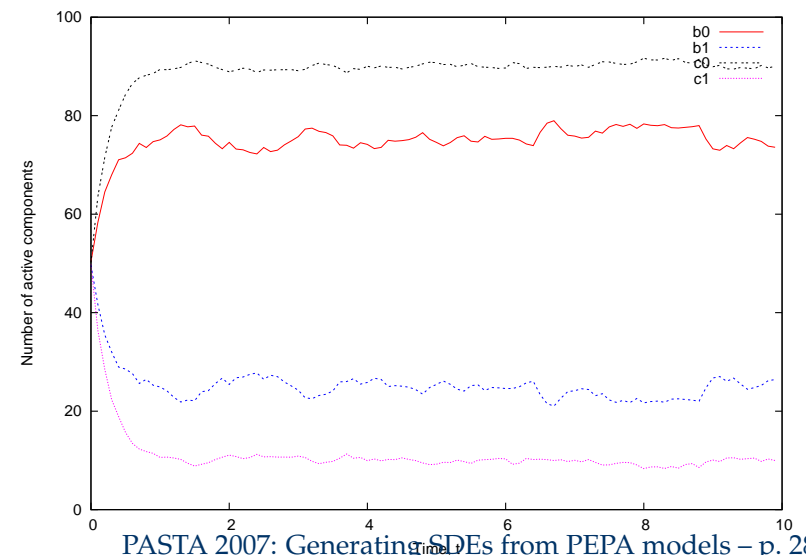
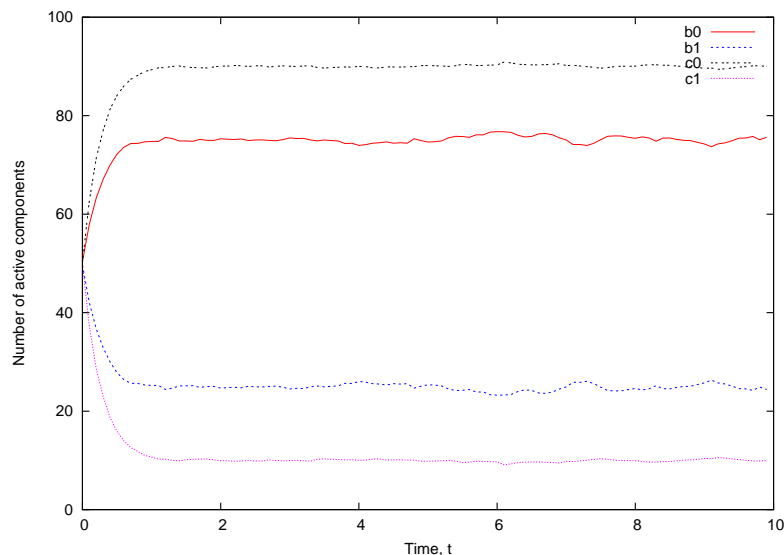
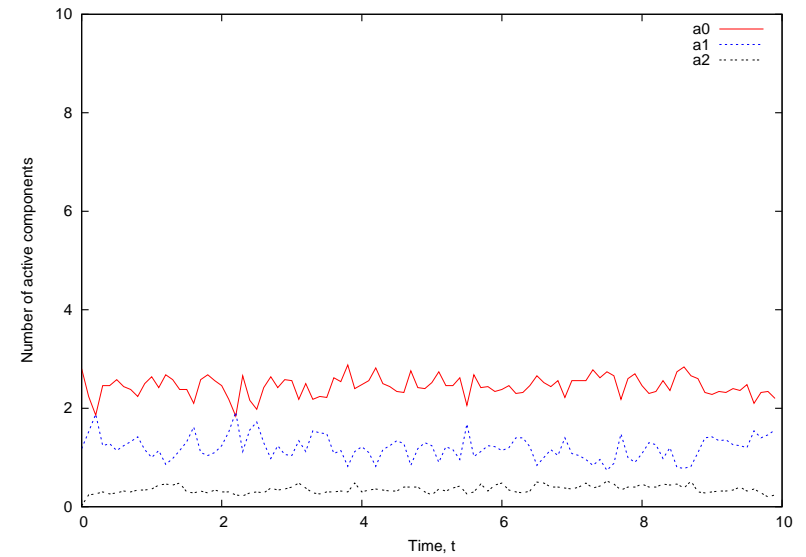
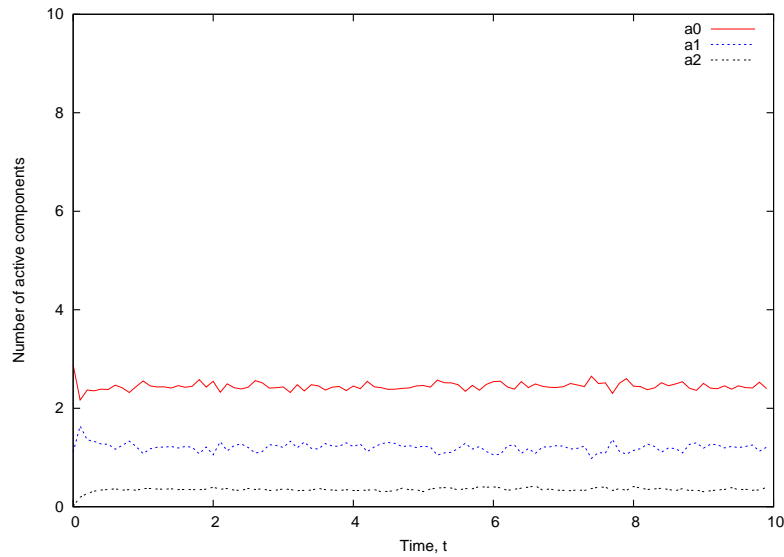
Thinning

- Crucial point is being able to bound the rate of the $M_i(t)$ - possible because *PEPA models are finite*
- We then simulate jumps at the maximum (**deterministic**) rate in advance ...

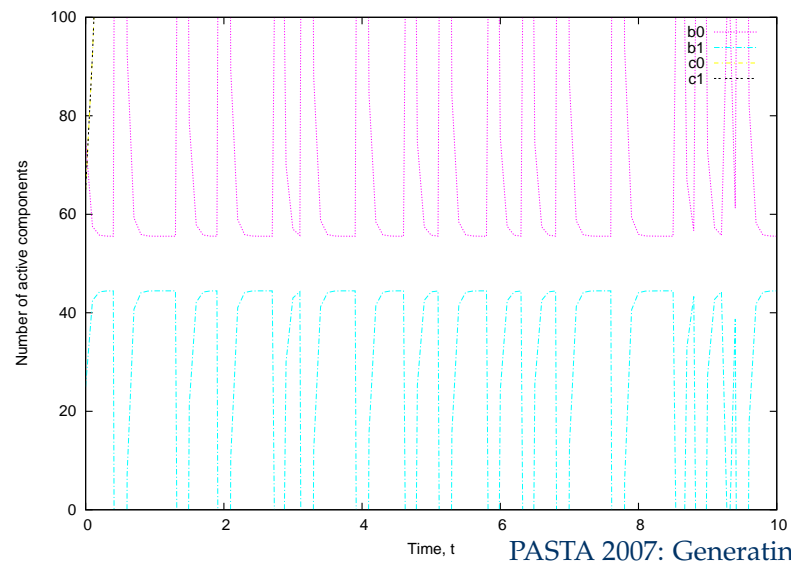
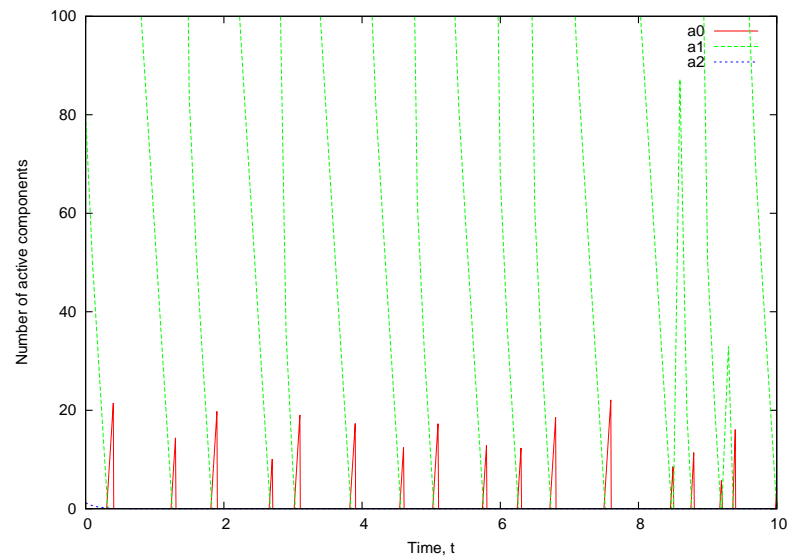
Thinning

- Crucial point is being able to bound the rate of the $M_i(t)$ - possible because *PEPA models are finite*
- We then simulate jumps at the maximum (**deterministic**) rate in advance ...
- By clever maths — ‘throw-away’ jumps that turn out to be ‘wrong predictions’ as we simulate the continuous parts

Hybrid comparison



ODEs



Contributions

- Novel decomposition of PEPA processes
- FCLT to approximate continuously-behaving components
- Hybrid scheme to handle models with discretely-behaving components

Further work

- Generalisation of these techniques to other Markovian modelling formalisms
- Similar FCLTs for non-Markovian modelling formalisms, semi-Markov models?
- Nature of convergence rate of FCLT needs investigation — highly mathematically technical — can we guarantee at least trend will always be correct?
- Formal classification of the class of PEPA models for which the SDEs have unique solution — technical to check for each model
- Correct handling of out-of-bounds situations?



PEPA syntax and semantics

Syntax: sequential components

$$S ::= (\alpha, r).S \mid S + S \mid A$$

- Prefix: $(a, r).S$ (*designated first action*)
- Choice: $S + S$ (*competing components — race*)
- Constant: $A \stackrel{\text{def}}{=} S$ (*assignment of names*)

$\text{Server} \stackrel{\text{def}}{=} (\text{task}, r_1).\text{Server} + (\text{break}, r_2).\text{BrokenServer}$

Syntax: model components

$$P ::= S \mid P \underset{L}{\bowtie} P \mid P / L$$

- Cooperation: $P \underset{L}{\bowtie} P$
 - ◆ $a \notin L \Rightarrow$ concurrent activity (*individual actions*)
 - ◆ $a \in L \Rightarrow$ cooperative activity (*shared actions*)
- Hiding: P / L (*abstraction $a \in L \Rightarrow a \rightarrow \tau$*)

$$\text{ServerFarm} \stackrel{\text{def}}{=} \text{Server} \underset{\{\text{reset}\}}{\bowtie} \text{Server} \underset{\{\text{reset}\}}{\bowtie} \text{Server}$$

Semantics: sequential

- Prefix:

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

- Choice:

$$\frac{E \xrightarrow{(\alpha, r)} E'}{E + F \xrightarrow{(\alpha, r)} E'}$$

$$\frac{F \xrightarrow{(\alpha, r)} F'}{E + F \xrightarrow{(\alpha, r)} F'}$$

- Constant:

$$\frac{E \xrightarrow{(\alpha, r)} E'}{A \xrightarrow{(\alpha, r)} E'} \quad (A \stackrel{def}{=} E)$$

Semantics: model

■ Hiding:

$$\frac{E \xrightarrow{(\alpha, r)} E'}{E/H \xrightarrow{(\alpha, r)} E'/H} \quad (\alpha \notin H)$$

$$\frac{E \xrightarrow{(\alpha, r)} E'}{E/H \xrightarrow{(\tau, r)} E'/H} \quad (\alpha \in H)$$

■ Cooperation:

$$\frac{E \xrightarrow{(\alpha, r)} E'}{E \begin{array}{c} \diagdown \diagup \\ \text{ } \\ \diagup \diagdown \\ S \end{array} F \xrightarrow{(\alpha, r)} E' \begin{array}{c} \diagdown \diagup \\ \text{ } \\ \diagup \diagdown \\ S \end{array} F} \quad (\alpha \notin S)$$

$$\frac{F \xrightarrow{(\alpha, r)} F'}{E \begin{array}{c} \diagdown \diagup \\ \text{ } \\ \diagup \diagdown \\ S \end{array} F \xrightarrow{(\alpha, r)} E \begin{array}{c} \diagdown \diagup \\ \text{ } \\ \diagup \diagdown \\ S \end{array} F'}$$

$$\frac{E \xrightarrow{(\alpha, r_1)} E' \quad F \xrightarrow{(\alpha, r_2)} F'}{E \begin{array}{c} \diagdown \diagup \\ \text{ } \\ \diagup \diagdown \\ S \end{array} F \xrightarrow{(\alpha, R)} E' \begin{array}{c} \diagdown \diagup \\ \text{ } \\ \diagup \diagdown \\ S \end{array} F'} \quad (\alpha \in S)$$

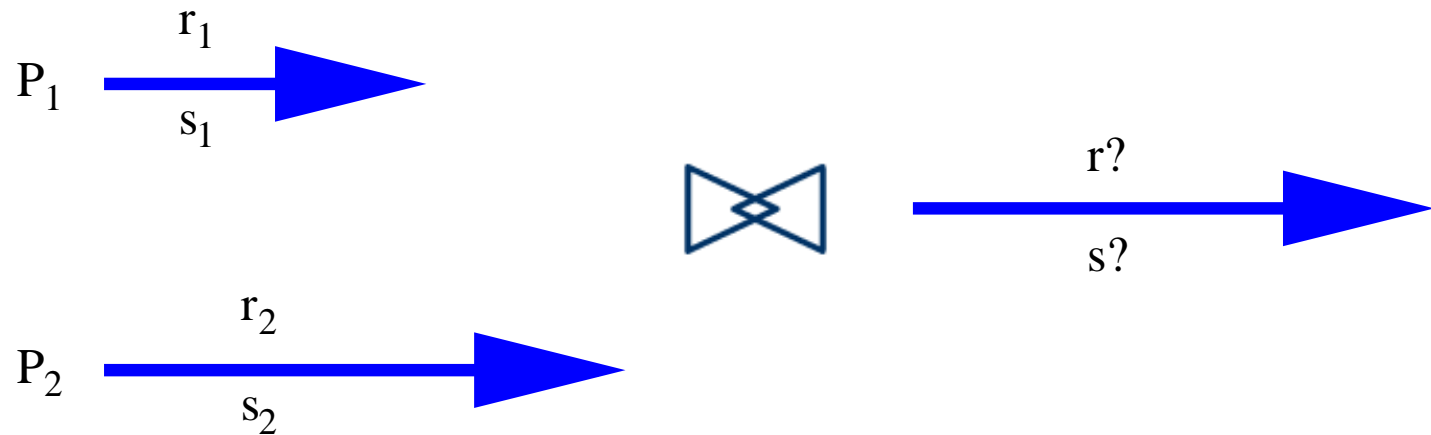
What should R be?

Timed synchronisation

How to specify the rate in a timed synchronisation is not obvious ...

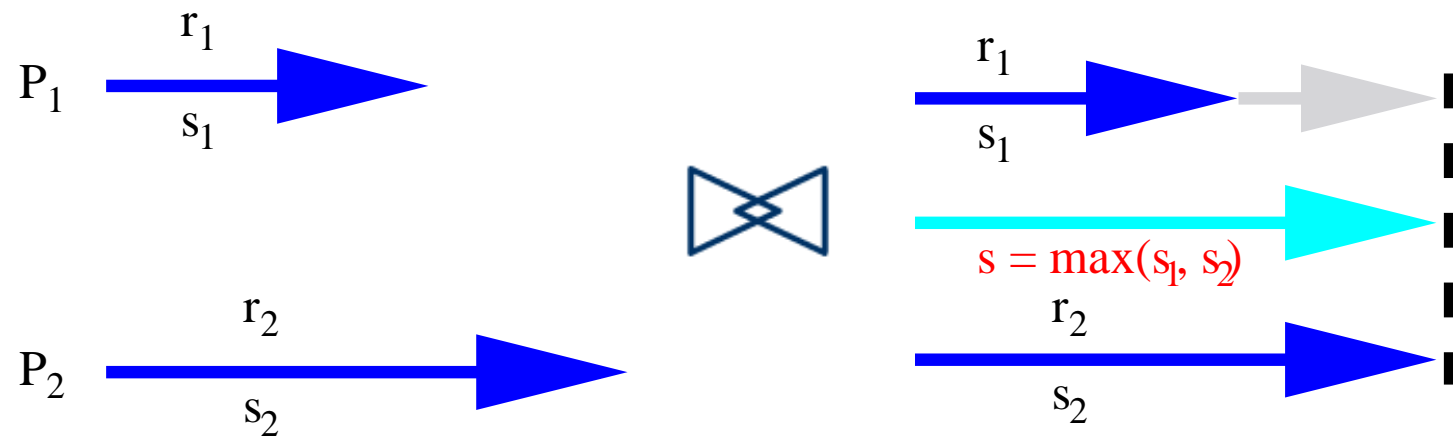
Timed synchronisation

How to specify the rate in a timed synchronisation is not obvious ...



Timed synchronisation

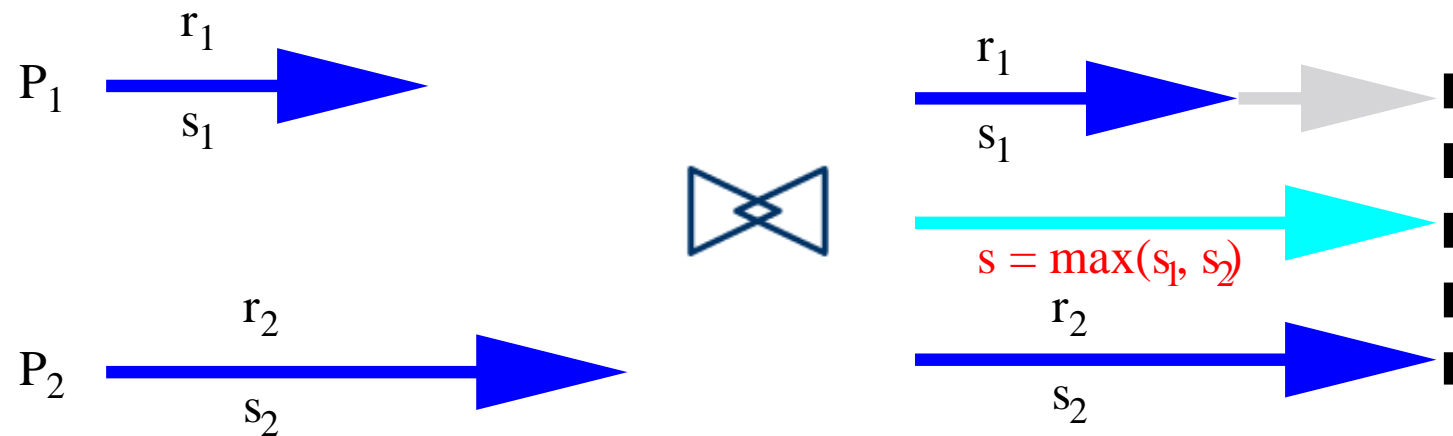
How to specify the rate in a timed synchronisation is not obvious ...



Barrier synchronisation ...

Timed synchronisation

How to specify the rate in a timed synchronisation is not obvious ...

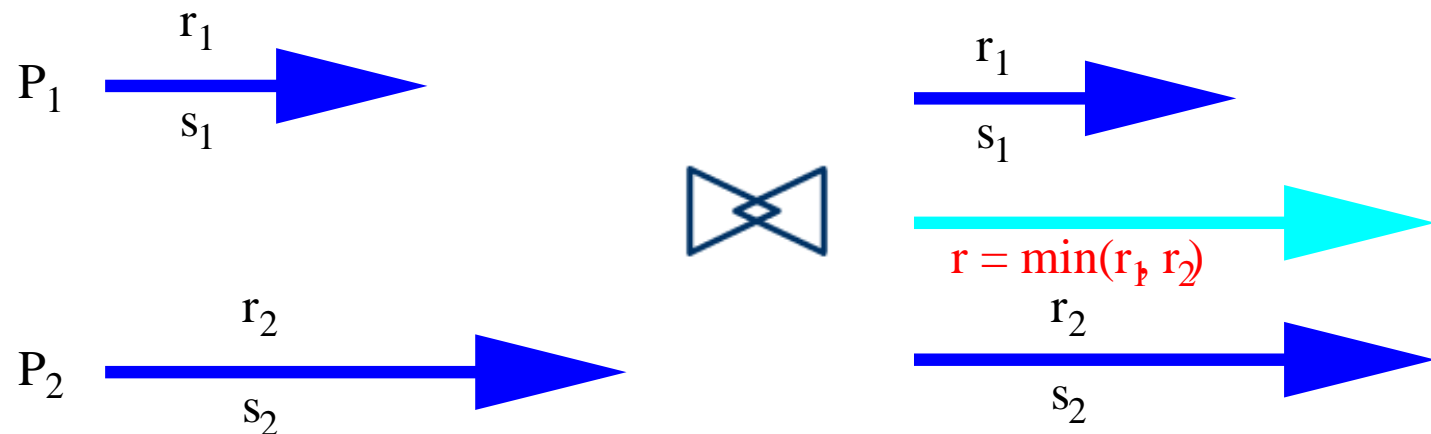


Barrier synchronisation ...

...but s is no longer *exponentially distributed*

Timed synchronisation

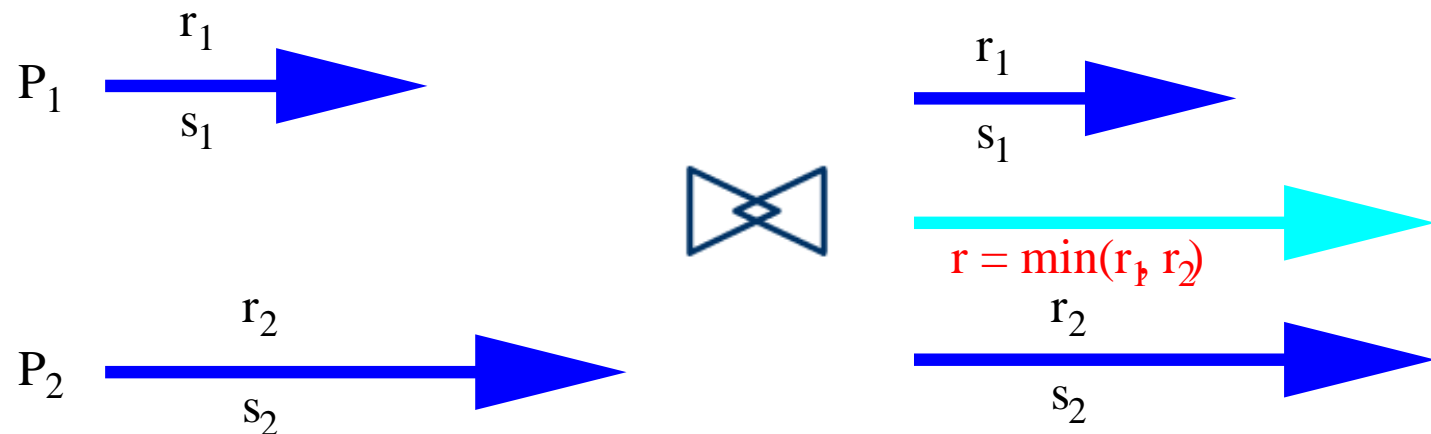
How to specify the rate in a timed synchronisation is not obvious ...



So we make an approximation — distribute exponentially with rate $\min(r_1, r_2)$

Timed synchronisation

How to specify the rate in a timed synchronisation is not obvious ...



Restriction to exponential distribution \Rightarrow natural interpretation of PEPA models as *continuous-time Markov chains* \Rightarrow (relative) tractability

(back)

A decorative graphic in the top-left corner consisting of a light green rounded rectangle and a dark blue horizontal bar with rounded ends.

Intuitionist ODE introduction

Idea: ignore individuals

- Stop trying to track the state of *every component*
- Instead just *how many* are in each state

Idea: ignore individuals

- Stop trying to track the state of *every component*
- Instead just *how many* are in each state

$$Processor_0 \stackrel{def}{=} (task_1, r_1).Processor_1$$

$$Processor_1 \stackrel{def}{=} (task_2, r_2).Processor_0$$

$$(P_0 \parallel P_0 \parallel P_0 \parallel P_0) \xrightarrow{(task_1, r_1/2)} (P_1 \parallel P_0 \parallel P_0 \parallel P_0)$$

$$(P_0 \parallel P_0 \parallel P_0 \parallel P_0) \xrightarrow{(task_1, r_1/2)} (P_0 \parallel P_1 \parallel P_0 \parallel P_0)$$

$$(P_0 \parallel P_0 \parallel P_0 \parallel P_0) \xrightarrow{(task_1, r_1/2)} (P_0 \parallel P_0 \parallel P_1 \parallel P_0)$$

$$(P_0 \parallel P_0 \parallel P_0 \parallel P_0) \xrightarrow{(task_1, r_1/2)} (P_0 \parallel P_0 \parallel P_0 \parallel P_1)$$

Aggregated state space

$$\begin{array}{l}
 \text{Processor}_0 \stackrel{\text{def}}{=} (a, r_1).\text{Processor}_1 \quad \text{Processor}_1 \stackrel{\text{def}}{=} (b, r_2).\text{Processor}_0 \\
 \text{Resource}_0 \stackrel{\text{def}}{=} (a, r_3).\text{Resource}_1 \quad \text{Resource}_1 \stackrel{\text{def}}{=} (c, r_4).\text{Resource}_0 \\
 \text{System} \stackrel{\text{def}}{=} \underbrace{(\text{Processor}_0 \parallel \dots \parallel \text{Processor}_0)}_{N_P} \underset{\{a\}}{\bowtie} \underbrace{(\text{Resource}_0 \parallel \dots \parallel \text{Resource}_0)}_{N_R}
 \end{array}$$

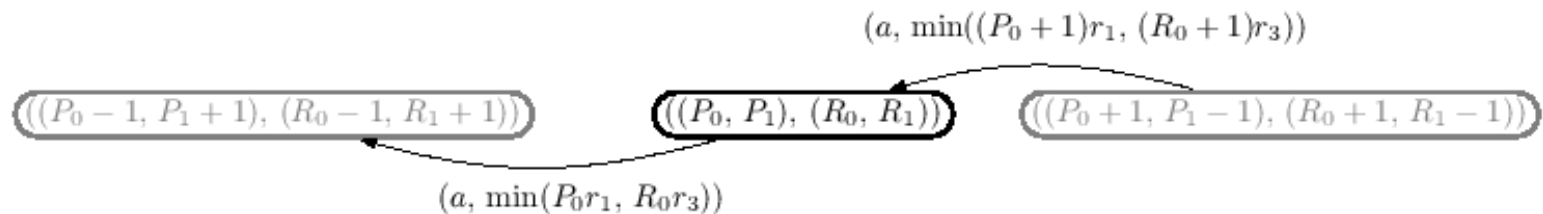
Aggregated state space

$$\begin{array}{l}
 \text{Processor}_0 \stackrel{\text{def}}{=} (a, r_1).\text{Processor}_1 \quad \text{Processor}_1 \stackrel{\text{def}}{=} (b, r_2).\text{Processor}_0 \\
 \text{Resource}_0 \stackrel{\text{def}}{=} (a, r_3).\text{Resource}_1 \quad \text{Resource}_1 \stackrel{\text{def}}{=} (c, r_4).\text{Resource}_0 \\
 \text{System} \stackrel{\text{def}}{=} \underbrace{(\text{Processor}_0 \parallel \dots \parallel \text{Processor}_0)}_{N_P} \underset{\{a\}}{\bowtie} \underbrace{(\text{Resource}_0 \parallel \dots \parallel \text{Resource}_0)}_{N_R}
 \end{array}$$

$$((P_0, P_1), (R_0, R_1))$$

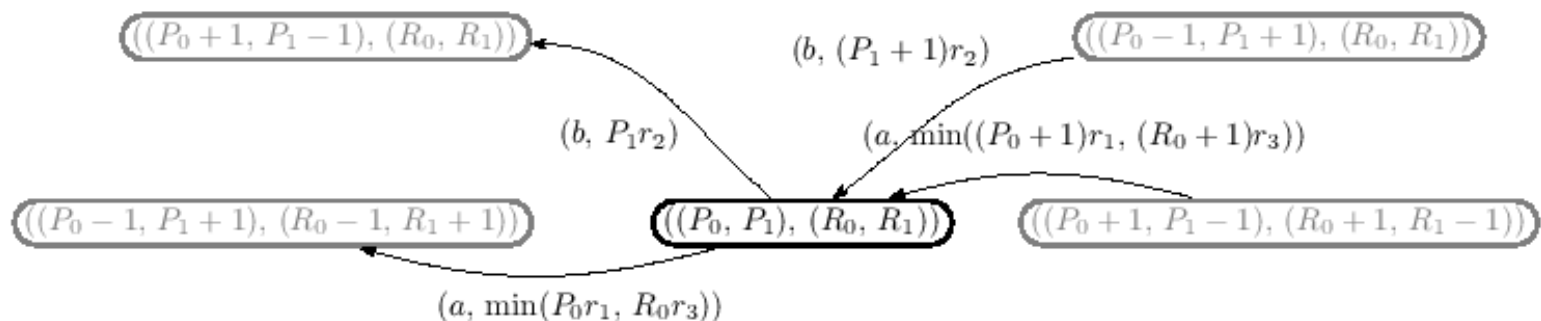
Aggregated state space

$$\begin{aligned}
 \text{Processor}_0 &\stackrel{\text{def}}{=} (a, r_1). \text{Processor}_1 & \text{Processor}_1 &\stackrel{\text{def}}{=} (b, r_2). \text{Processor}_0 \\
 \text{Resource}_0 &\stackrel{\text{def}}{=} (a, r_3). \text{Resource}_1 & \text{Resource}_1 &\stackrel{\text{def}}{=} (c, r_4). \text{Resource}_0 \\
 \text{System} &\stackrel{\text{def}}{=} \underbrace{(\text{Processor}_0 \parallel \dots \parallel \text{Processor}_0)}_{N_P} \underset{\{a\}}{\bowtie} \underbrace{(\text{Resource}_0 \parallel \dots \parallel \text{Resource}_0)}_{N_R}
 \end{aligned}$$



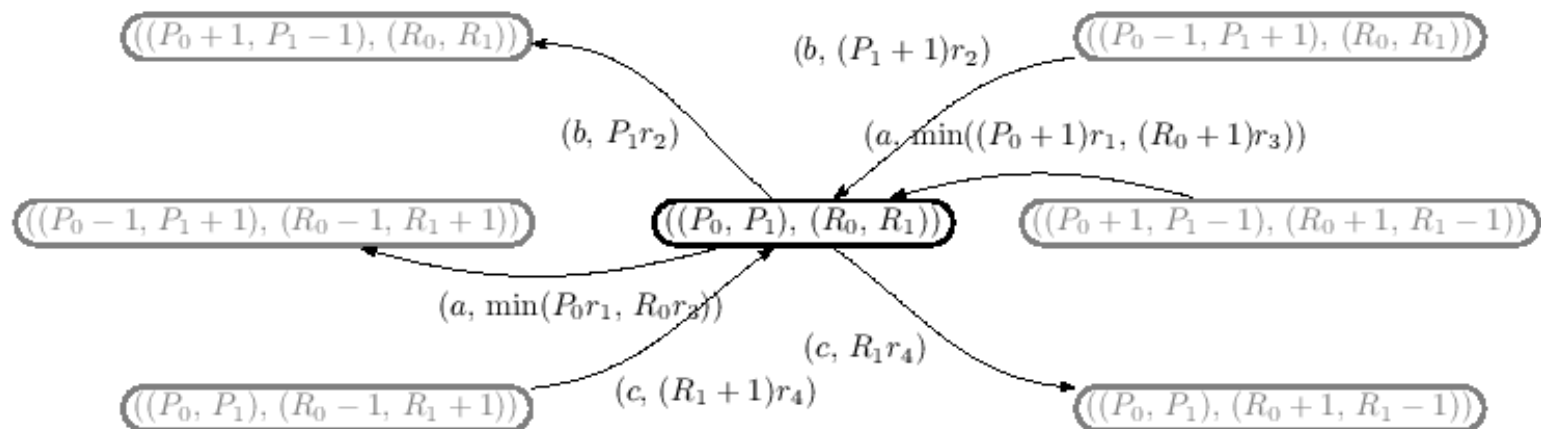
Aggregated state space

$$\begin{aligned}
 \text{Processor}_0 &\stackrel{\text{def}}{=} (a, r_1). \text{Processor}_1 & \text{Processor}_1 &\stackrel{\text{def}}{=} (b, r_2). \text{Processor}_0 \\
 \text{Resource}_0 &\stackrel{\text{def}}{=} (a, r_3). \text{Resource}_1 & \text{Resource}_1 &\stackrel{\text{def}}{=} (c, r_4). \text{Resource}_0 \\
 \text{System} &\stackrel{\text{def}}{=} \underbrace{(\text{Processor}_0 \parallel \dots \parallel \text{Processor}_0)}_{N_P} \underset{\{a\}}{\boxtimes} \underbrace{(\text{Resource}_0 \parallel \dots \parallel \text{Resource}_0)}_{N_R}
 \end{aligned}$$

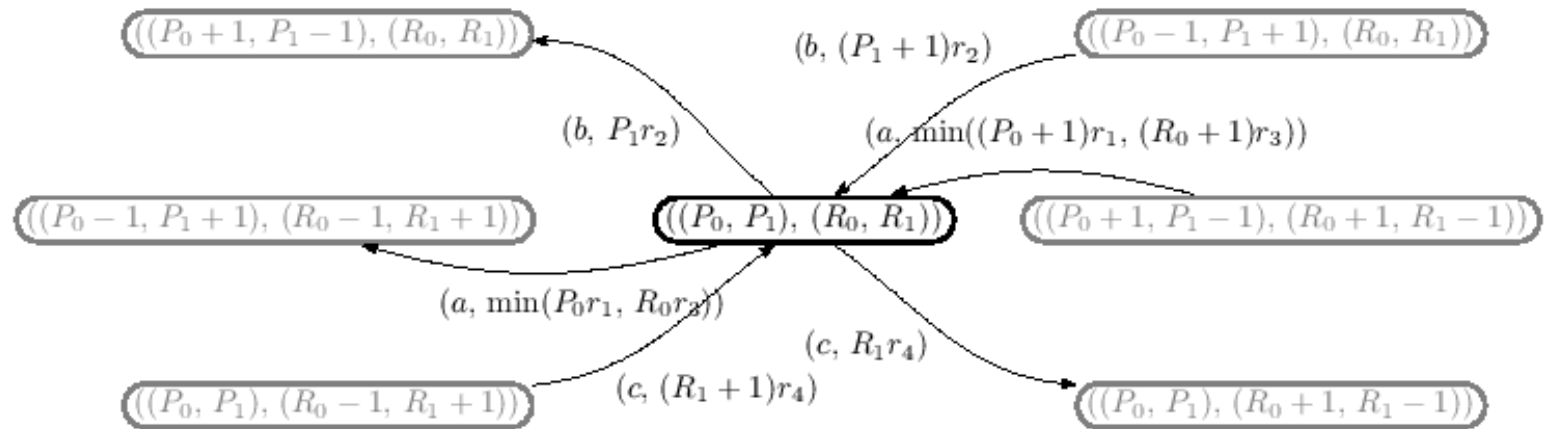


Aggregated state space

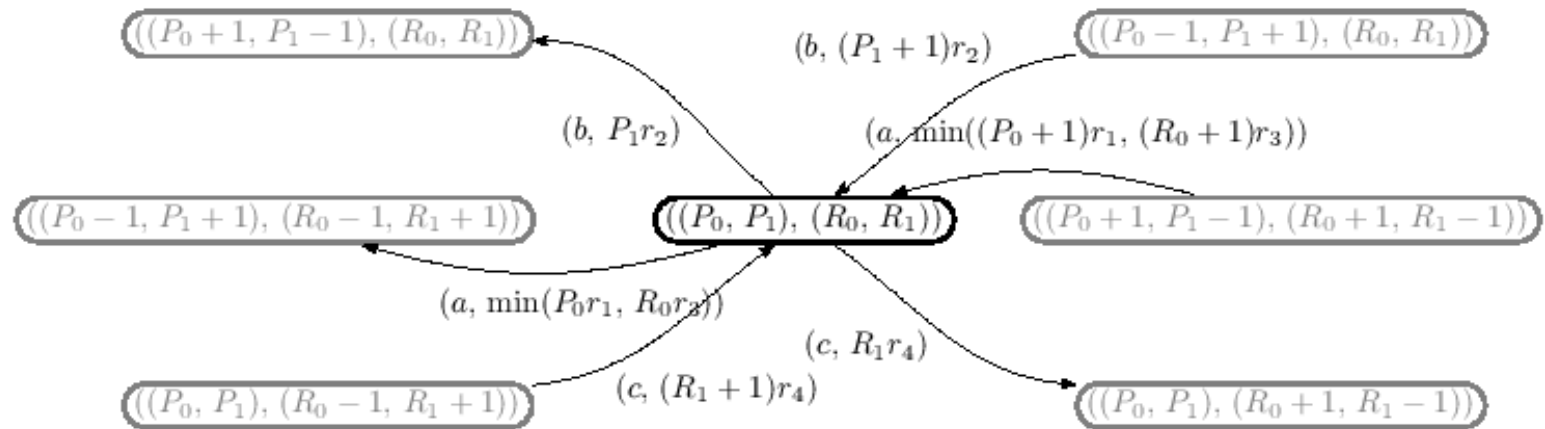
$$\begin{aligned}
 \text{Processor}_0 &\stackrel{\text{def}}{=} (a, r_1). \text{Processor}_1 & \text{Processor}_1 &\stackrel{\text{def}}{=} (b, r_2). \text{Processor}_0 \\
 \text{Resource}_0 &\stackrel{\text{def}}{=} (a, r_3). \text{Resource}_1 & \text{Resource}_1 &\stackrel{\text{def}}{=} (c, r_4). \text{Resource}_0 \\
 \text{System} &\stackrel{\text{def}}{=} \underbrace{(\text{Processor}_0 \parallel \dots \parallel \text{Processor}_0)}_{N_P} \underset{\{a\}}{\boxtimes} \underbrace{(\text{Resource}_0 \parallel \dots \parallel \text{Resource}_0)}_{N_R}
 \end{aligned}$$



Continuous approximation



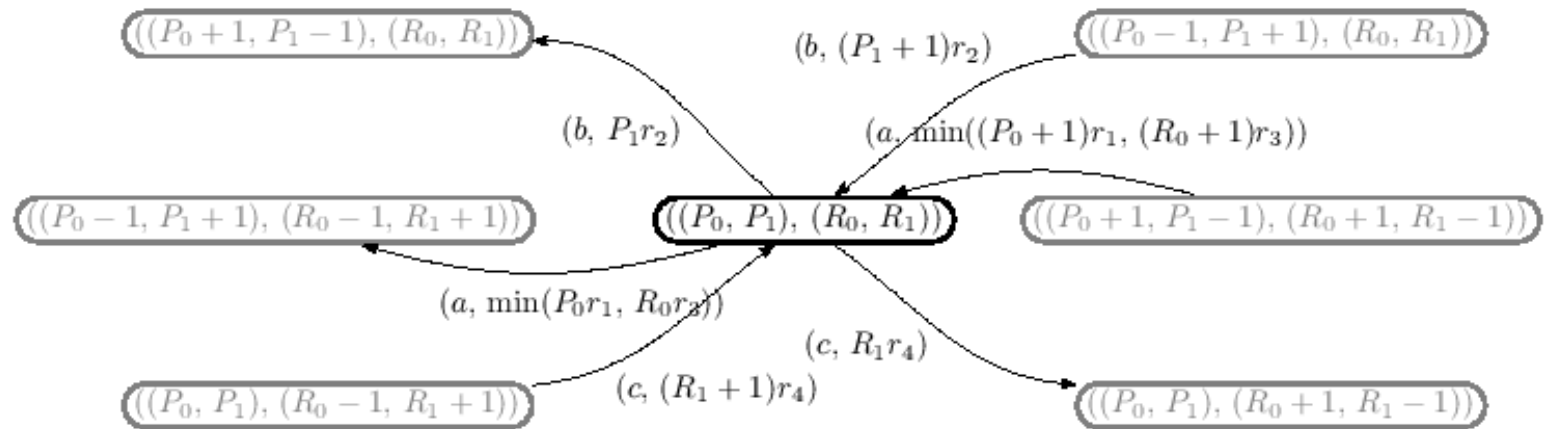
Continuous approximation



For large numbers of components approximate the counters by *real variables* ...

$$P_0(t + dt) - P_0(t) = -\min(P_0(t)r_1, R_0(t)r_2)dt + r_2P_1(t)dt$$

Continuous approximation

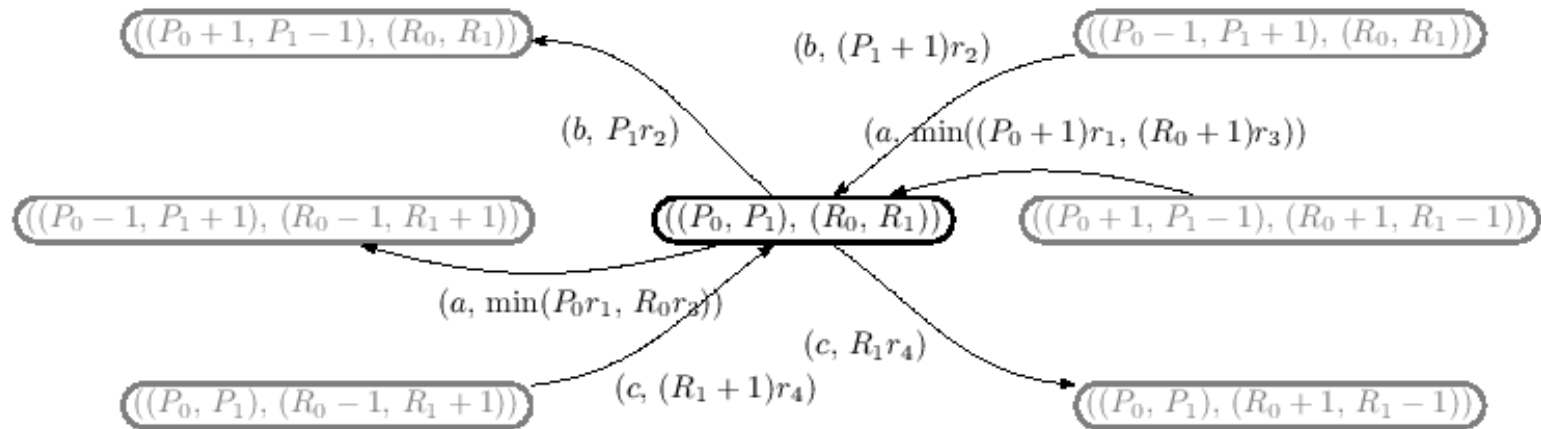


For large numbers of components approximate the counters by *real variables* ...

$$P_0(t + dt) - P_0(t) = -\min(P_0(t)r_1, R_0(t)r_2)dt + r_2P_1(t)dt$$

Can you see where this is going ...?

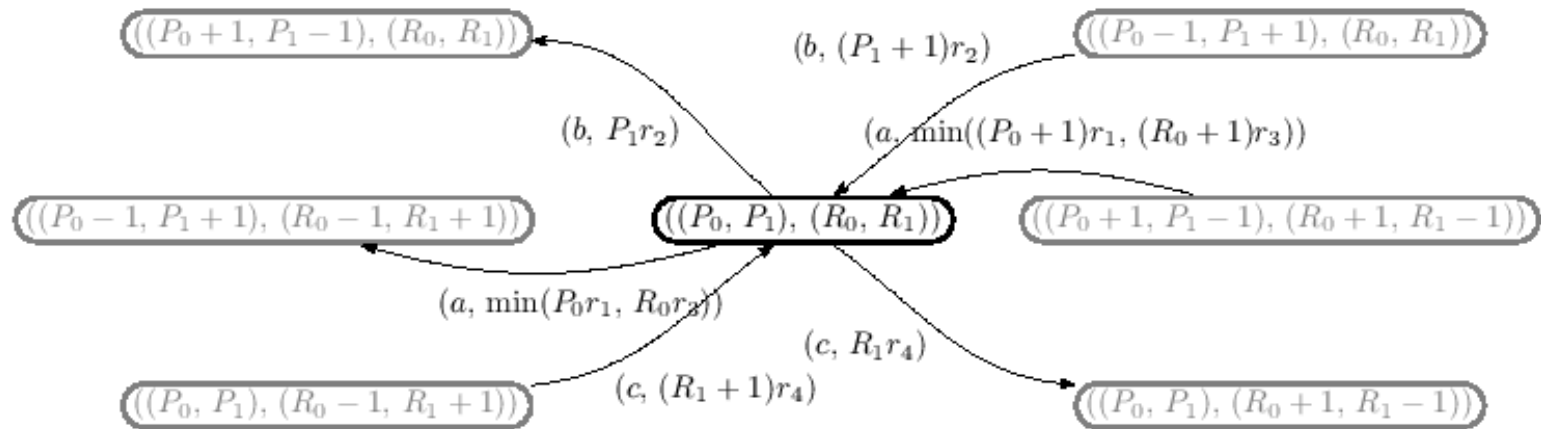
Continuous approximation



For large numbers of components approximate the counters by *real variables* ...

$$\frac{P_0(t + dt) - P_0(t)}{dt} = -\min(P_0(t)r_1, R_0(t)r_2) + r_2P_1(t)$$

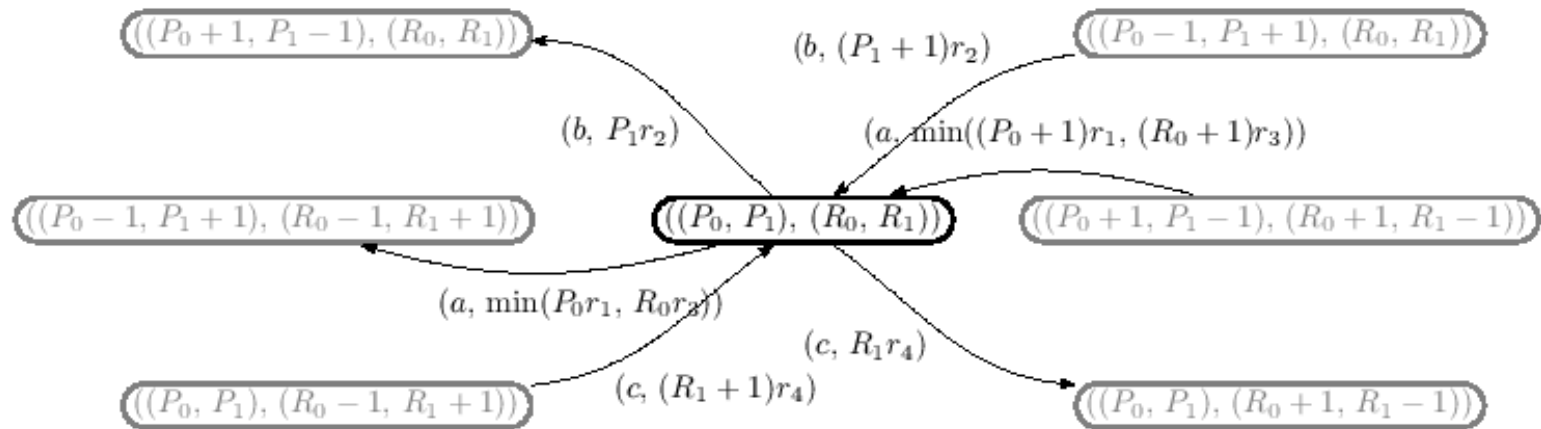
Continuous approximation



For large numbers of components approximate the counters by *real variables* ...

$$\lim_{dt \rightarrow 0} \frac{P_0(t + dt) - P_0(t)}{dt} = -\min(P_0(t)r_1, R_0(t)r_2) + r_2P_1(t)$$

Continuous approximation



For large numbers of components approximate the counters by *real variables* ...

$$\frac{dP_0(t)}{dt} = -\min(P_0(t)r_1, R_0(t)r_2) + r_2P_1(t)$$

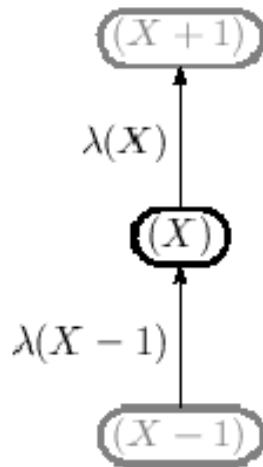
(back)

A decorative graphic consisting of a light green L-shaped corner element in the top-left and a dark blue horizontal bar with rounded ends extending from the left edge of the slide.

ODE derivation using generating functions

Idea

Consider a simple CTMC ...



Naturally (formally) associated with any CTMC are its *Chapman-Kolmogorov* equations (in this case infinitely-many):

$$\left. \begin{aligned} \frac{dp_x(t)}{dt} &= \lambda(x-1)p_{x-1}(t) - \lambda x p_x(t) & x > 1 \\ \frac{dp_1(t)}{dt} &= -\lambda p_1(t) \end{aligned} \right\}$$

Idea

$$\left. \begin{aligned} \frac{dp_x(t)}{dt} &= \lambda(x-1)p_{x-1}(t) - \lambda x p_x(t) & x > 1 \\ \frac{dp_1(t)}{dt} &= -\lambda p_1(t) \end{aligned} \right\}$$

Idea

$$\left. \begin{aligned} \frac{dp_x(t)}{dt} &= \lambda(x-1)p_{x-1}(t) - \lambda x p_x(t) & x > 1 \\ \frac{dp_1(t)}{dt} &= -\lambda p_1(t) \end{aligned} \right\}$$

Define the moment-generating function for the variable we are interested in:

$$M(\theta, t) := \mathbf{E} \left[e^{\theta X(t)} \right]$$

Idea

$$\left. \begin{aligned} \frac{dp_x(t)}{dt} &= \lambda(x-1)p_{x-1}(t) - \lambda x p_x(t) & x > 1 \\ \frac{dp_1(t)}{dt} &= -\lambda p_1(t) \end{aligned} \right\}$$

Define the moment-generating function for the variable we are interested in:

$$\begin{aligned} M(\theta, t) &:= \mathbf{E} \left[e^{\theta X(t)} \right] \\ &=: \sum_{y=1}^{\infty} [p_y(t) e^{\theta y}] \end{aligned}$$

Idea

$$\left. \begin{aligned} \frac{dp_x(t)}{dt} &= \lambda(x-1)p_{x-1}(t) - \lambda x p_x(t) & x > 1 \\ \frac{dp_1(t)}{dt} &= -\lambda p_1(t) \end{aligned} \right\}$$

Differentiate with respect to t :

$$\frac{\partial M}{\partial t} = \frac{\partial}{\partial t} \sum_{y=1}^{\infty} [p_y(t)e^{\theta y}] = \sum_{y=1}^{\infty} \left[\frac{dp_y(t)}{dt} e^{\theta y} \right]$$

Idea

$$\left. \begin{aligned} \frac{dp_x(t)}{dt} &= \lambda(x-1)p_{x-1}(t) - \lambda x p_x(t) & x > 1 \\ \frac{dp_1(t)}{dt} &= -\lambda p_1(t) \end{aligned} \right\}$$

Differentiate with respect to t :

$$\frac{\partial M}{\partial t} = \frac{\partial}{\partial t} \sum_{y=1}^{\infty} [p_y(t)e^{\theta y}] = \sum_{y=1}^{\infty} \left[\frac{dp_y(t)}{dt} e^{\theta y} \right]$$

Some (exciting) algebra later ...

$$\frac{\partial M}{\partial t} = \lambda(e^{\theta} - 1) \frac{\partial M}{\partial \theta}$$

Idea

$$\frac{\partial M}{\partial t} = \lambda(e^\theta - 1) \frac{\partial M}{\partial \theta}$$

Extract the first moment of $X(t)$ (differentiate wrt. θ , set $\theta = 0$):

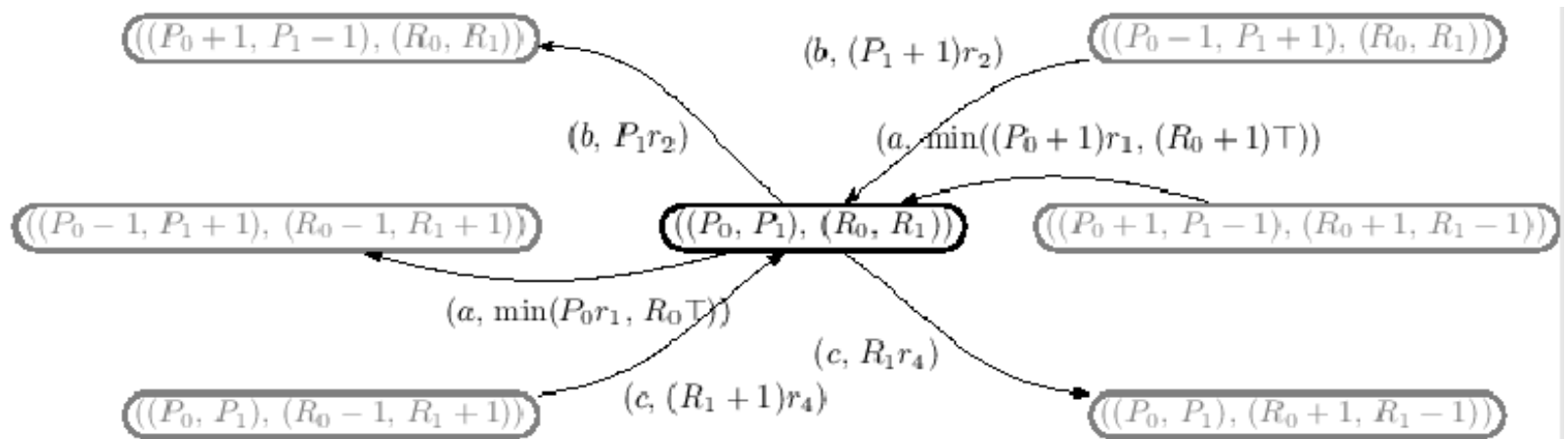
$$\frac{d\mathbf{E}[X(t)]}{dt} = \lambda \mathbf{E}[X(t)] \Rightarrow \mathbf{E}[X(t)] = Ce^{\lambda t}$$

...an ODE for the expected value of the counter

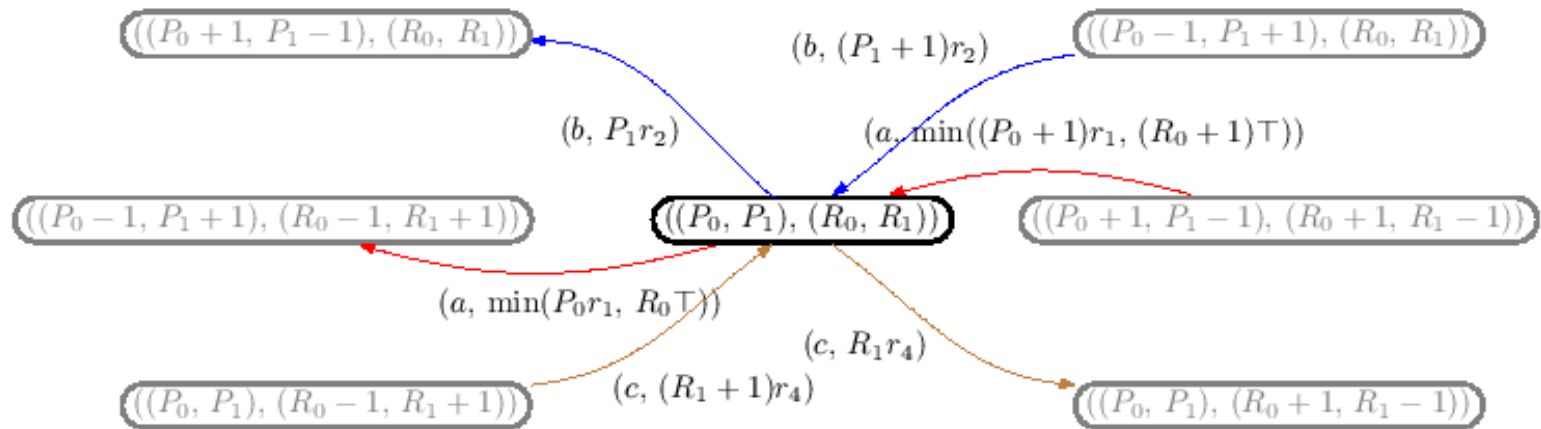
Formally deriving ODEs from PEPA models

$$\begin{aligned}
 \text{Processor}_0 &\stackrel{\text{def}}{=} (a, r_1).\text{Processor}_1 & \text{Processor}_1 &\stackrel{\text{def}}{=} (b, r_2).\text{Processor}_0 \\
 \text{Resource}_0 &\stackrel{\text{def}}{=} (a, \top).\text{Resource}_1 & \text{Resource}_1 &\stackrel{\text{def}}{=} (c, r_4).\text{Resource}_0
 \end{aligned}$$

$$\text{System} \stackrel{\text{def}}{=} \underbrace{(\text{Processor}_0 \parallel \dots \parallel \text{Processor}_0)}_{N_P} \bowtie_{\{a\}} \underbrace{(\text{Resource}_0 \parallel \dots \parallel \text{Resource}_0)}_{N_R}$$



Formally deriving ODEs from PEPA models



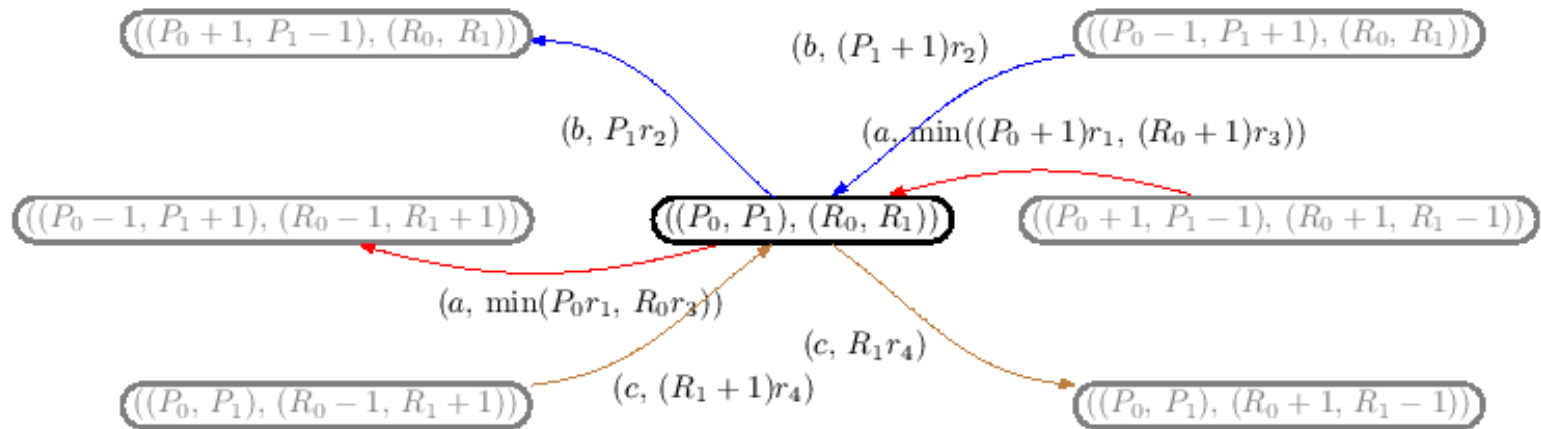
Define *transition rate functions* to classify *transition classes*:

$$f_{-1,1,-1,1}(x_1, x_2, x_3, x_4) = \min(x_1 r_1, x_3 \top) := \begin{cases} x_1 r_1 & \text{if } x_3 > 0 \\ 0 & \text{if } x_3 = 0 \end{cases}$$

$$f_{1,-1,0,0}(x_1, x_2, x_3, x_4) = x_2 r_2$$

$$f_{0,0,1,-1}(x_1, x_2, x_3, x_4) = x_4 r_4$$

Formally deriving ODEs from PEPA models



Define *joint moment generating function*:

$$M(\theta_1, \theta_2, \theta_3, \theta_4, t) := \mathbf{E} \left[e^{\theta_1 P_0(t) + \theta_2 P_1(t) + \theta_3 R_0(t) + \theta_4 R_1(t)} \right]$$

Formally deriving ODEs from PEPA models

We show in general that:

$$\frac{\partial M}{\partial t} = \sum_{i_1, \dots, i_N \in \mathbb{Z}} \left(e^{\theta_1 i_1 + \dots + \theta_N i_N} - 1 \right) f_{i_1, \dots, i_N} \left(\frac{\partial}{\partial \theta_1}, \dots, \frac{\partial}{\partial \theta_N} \right) M(\theta_1, \dots, \theta_N, t)$$

But $f_{i_1, \dots, i_N} \left(\frac{\partial}{\partial \theta_1}, \dots, \frac{\partial}{\partial \theta_N} \right)$ must make sense, i.e. the transition rate functions *can only be polynomials*

Formally deriving ODEs from PEPA models

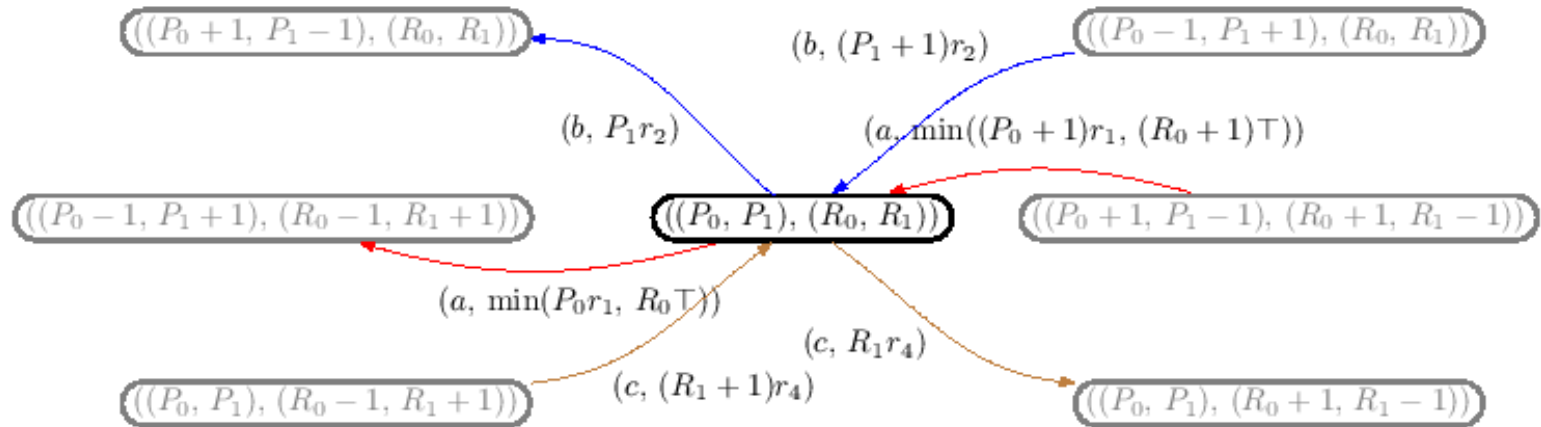
We show in general that:

$$\frac{\partial M}{\partial t} = \sum_{i_1, \dots, i_N \in \mathbb{Z}} \left(e^{\theta_1 i_1 + \dots + \theta_N i_N} - 1 \right) f_{i_1, \dots, i_N} \left(\frac{\partial}{\partial \theta_1}, \dots, \frac{\partial}{\partial \theta_N} \right) M(\theta_1, \dots, \theta_N, t)$$

But $f_{i_1, \dots, i_N} \left(\frac{\partial}{\partial \theta_1}, \dots, \frac{\partial}{\partial \theta_N} \right)$ must make sense, i.e. the transition rate functions *can only be polynomials*
For passive cooperation, might approximate:

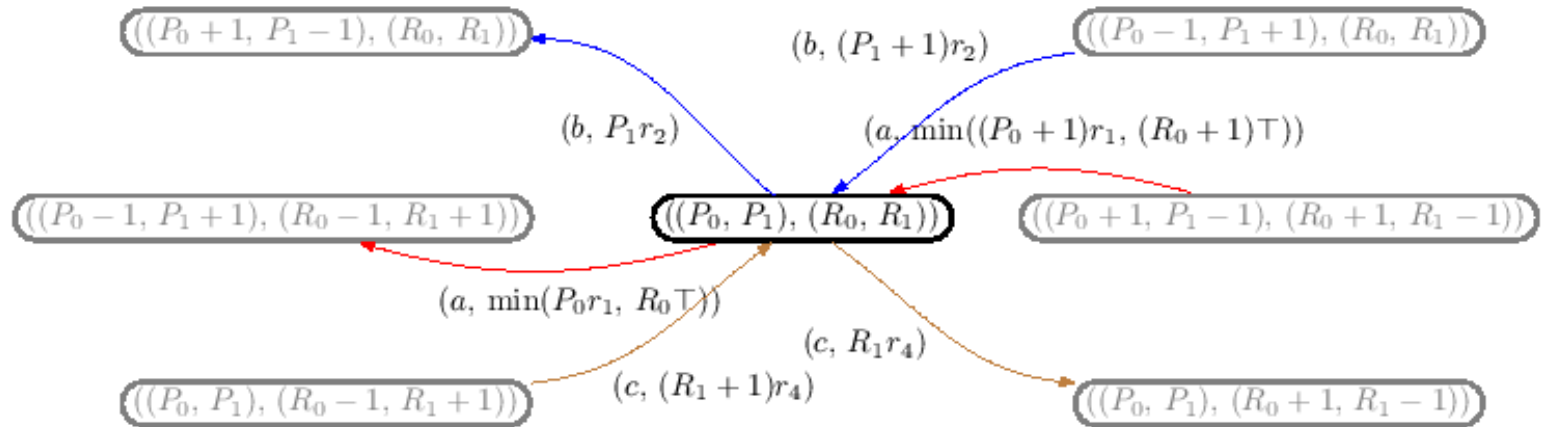
$$\min(x_1 r_1, x_3 \top) := \begin{cases} x_1 r_1 & \text{if } x_3 > 0 \\ 0 & \text{if } x_3 = 0 \end{cases} \approx x_1 r_1$$

Formally deriving ODEs from PEPA models



$$\begin{aligned} \frac{\partial M}{\partial t} = & r_1 \left(e^{-\theta_1 + \theta_2 - \theta_3 + \theta_4} - 1 \right) \frac{\partial M}{\partial \theta_1} \\ & + r_2 \left(e^{\theta_1 - \theta_2} - 1 \right) \frac{\partial M}{\partial \theta_2} + r_4 \left(e^{\theta_3 - \theta_4} - 1 \right) \frac{\partial M}{\partial \theta_4} \end{aligned}$$

Formally deriving ODEs from PEPA models



$$\frac{d\mathbf{E}[P_0(t)]}{dt} = -r_1 \mathbf{E}[P_0(t)] + r_2 \mathbf{E}[P_1(t)]$$

$$\frac{d\mathbf{E}[P_1(t)]}{dt} = r_1 \mathbf{E}[P_0(t)] - r_2 \mathbf{E}[P_1(t)]$$

$$\frac{d\mathbf{E}[R_0(t)]}{dt} = -r_1 \mathbf{E}[P_0(t)] + r_4 \mathbf{E}[R_1(t)]$$

$$\frac{d\mathbf{E}[R_1(t)]}{dt} = r_1 \mathbf{E}[P_0(t)] - r_4 \mathbf{E}[R_1(t)]$$