SPAs for performance modelling: Lecture 2 — Stochastic Process Algebras

Jane Hillston

LFCS, School of Informatics
The University of Edinburgh
Scotland

9th April 2013

THE UNIVERSITY
of EDINBURGH

Outline

1 Process algebra and Markov processes

2 A semantics for PEPA — informally

3 A formal semantics for PEPA

4 The nature of synchronisation

Outline

1 Process algebra and Markov processes

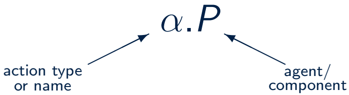
2 A semantics for PEPA — informally

3 A formal semantics for PEPA

4 The nature of synchronisation

Process Algebra

- Models consist of agents which engage in actions.



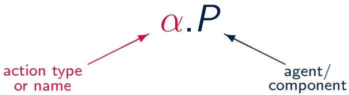
$\alpha.P$

action type or name

agent/ component

Process Algebra

- Models consist of agents which engage in actions.



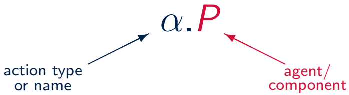
$$\alpha.P$$

action type
or name

agent/
component

Process Algebra

- Models consist of agents which engage in actions.



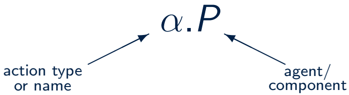
$\alpha.P$

action type or name

agent/ component

Process Algebra

- Models consist of agents which engage in actions.



$$\alpha.P$$

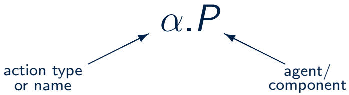
action type or name

agent/ component

- The structured operational (interleaving) semantics of the language is used to generate a labelled transition system.

Process Algebra

- Models consist of agents which engage in actions.



$\alpha.P$

action type or name

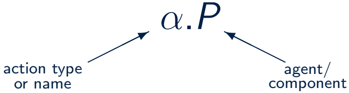
agent/ component

- The structured operational (interleaving) semantics of the language is used to generate a labelled transition system.

Process algebra model

Process Algebra

- Models consist of agents which engage in actions.



$\alpha.P$

action type or name

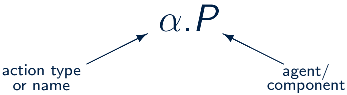
agent/ component

- The structured operational (interleaving) semantics of the language is used to generate a labelled transition system.

Process algebra model $\xrightarrow{\text{SOS rules}}$

Process Algebra

- Models consist of agents which engage in actions.



$\alpha.P$

action type or name

agent/ component

- The structured operational (interleaving) semantics of the language is used to generate a labelled transition system.

Process algebra model $\xrightarrow{\text{SOS rules}}$ Labelled transition system

Example

Consider a web server which offers html pages for download:

$$Server \overset{def}{=} get.download.rel.Server$$

Example

Consider a web server which offers html pages for download:

$$Server \overset{def}{=} get.download.rel.Server$$

Its clients might be web browsers, in a domain with a local cache of frequently requested pages. Thus any display request might result in an access to the server or in a page being loaded from the cache.

$$Browser \overset{def}{=} display.(cache.Browser \;+\; get.download.rel.Browser)$$

Example

Consider a web server which offers html pages for download:

$$Server \overset{def}{=} get.download.rel.Server$$

Its clients might be web browsers, in a domain with a local cache of frequently requested pages. Thus any display request might result in an access to the server or in a page being loaded from the cache.

$$Browser \overset{def}{=} display.(cache.Browser \ + \ get.download.rel.Browser)$$

A simple version of the Web can be considered to be the interaction of these components:

$$WEB \overset{def}{=} \big(Browser \parallel Browser\big) \mid Server$$

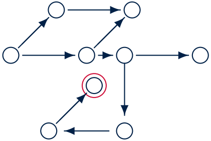
Qualitative Analysis

- The labelled transition system underlying a process algebra model can be used for functional verification e.g.: reachability analysis, specification matching and model checking.

Qualitative Analysis

- The labelled transition system underlying a process algebra model can be used for functional verification e.g.: reachability analysis, specification matching and model checking.

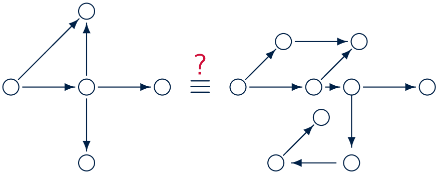
Will the system arrive
in a particular state?



Qualitative Analysis

- The labelled transition system underlying a process algebra model can be used for functional verification e.g.: reachability analysis, specification matching and model checking.

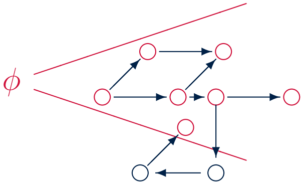
Does system behaviour match its specification?



Qualitative Analysis

- The labelled transition system underlying a process algebra model can be used for functional verification e.g.: reachability analysis, specification matching and model checking.

Does a given property ϕ hold within the system?



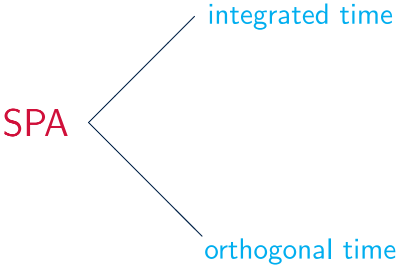
Stochastic process algebras

Process algebras where models are decorated with quantitative information used to generate a stochastic process are stochastic process algebras (SPA).

SPA Languages

SPA

SPA Languages

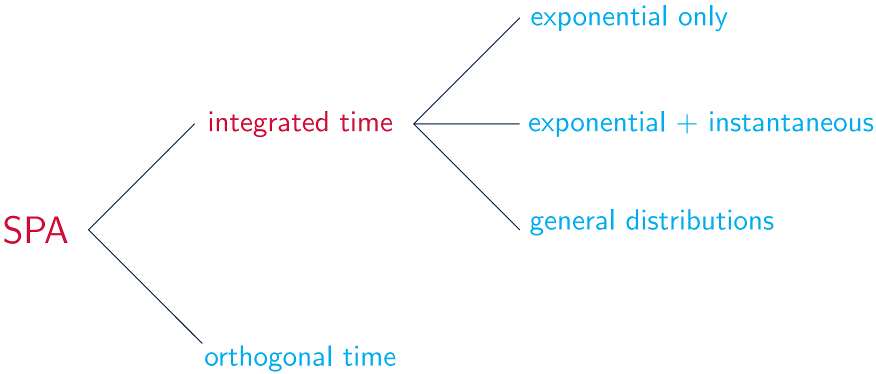


SPA

integrated time

orthogonal time

SPA Languages



SPA

integrated time

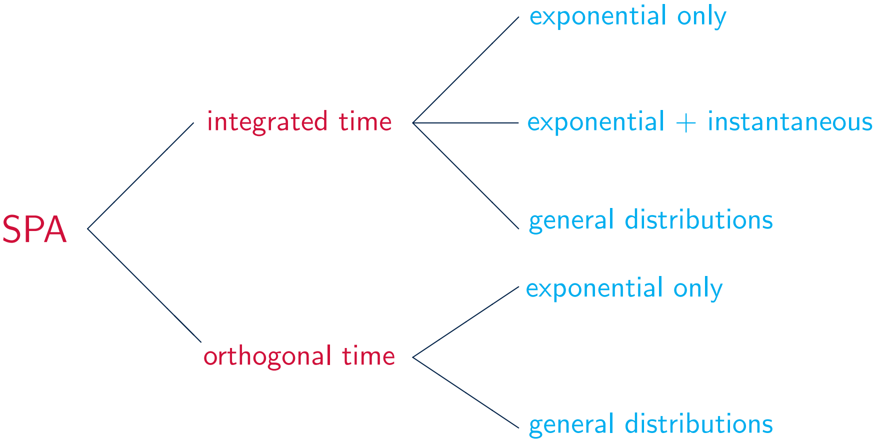
exponential only

exponential + instantaneous

general distributions

orthogonal time

SPA Languages



SPA

integrated time

exponential only

exponential + instantaneous

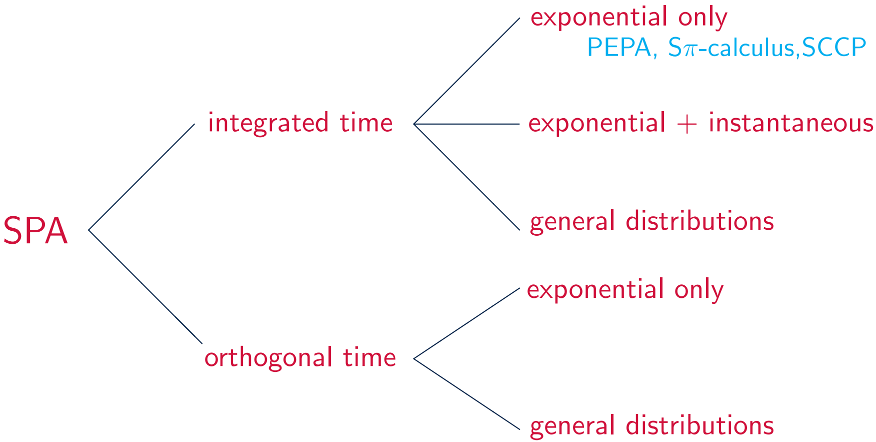
general distributions

orthogonal time

exponential only

general distributions

SPA Languages



SPA

integrated time

exponential only

PEPA, Sπ-calculus,SCCP

exponential + instantaneous

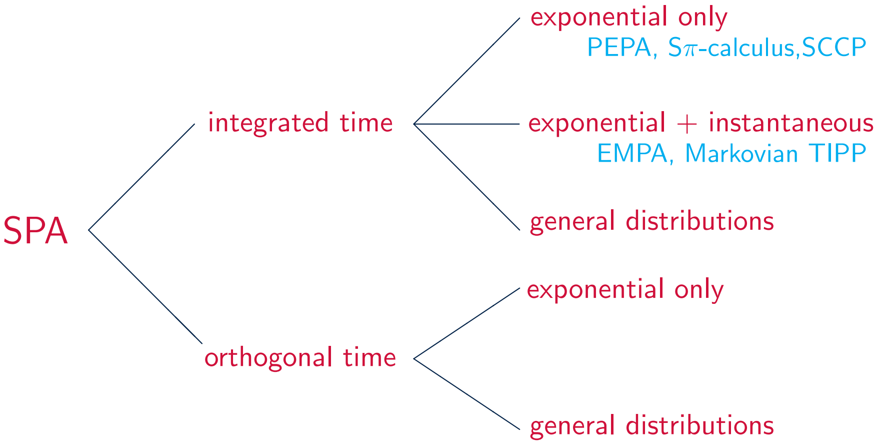
general distributions

orthogonal time

exponential only

general distributions

SPA Languages



SPA

integrated time

exponential only

PEPA, Sπ-calculus,SCCP

exponential + instantaneous

EMPA, Markovian TIPP

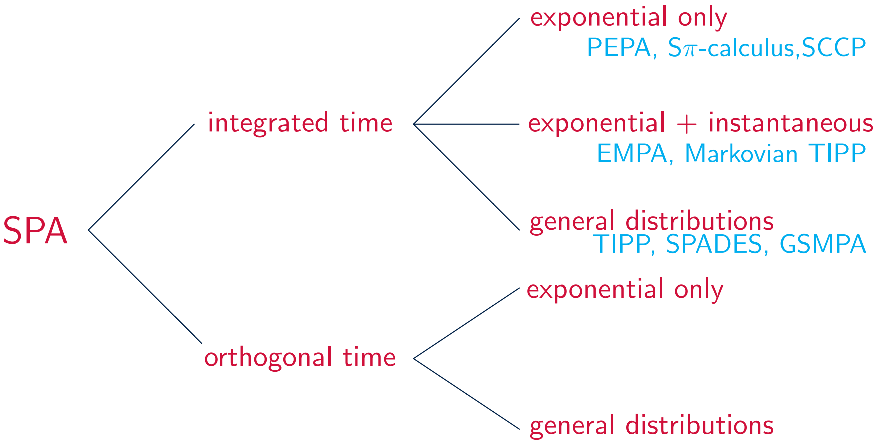
general distributions

orthogonal time

exponential only

general distributions

SPA Languages



SPA

integrated time

exponential only

PEPA, Sπ-calculus,SCCP

exponential + instantaneous

EMPA, Markovian TIPP

general distributions

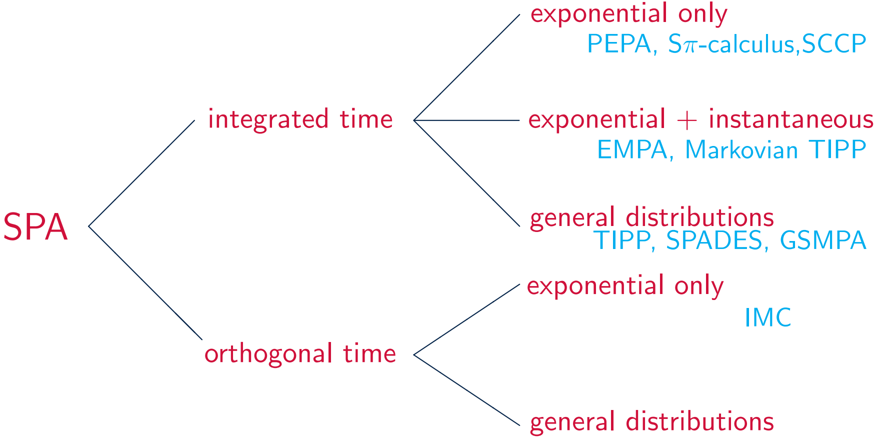
TIPP, SPADES, GSMPA

orthogonal time

exponential only

general distributions

SPA Languages



SPA

integrated time

exponential only

PEPA, Sπ-calculus,SCCP

exponential + instantaneous

EMPA, Markovian TIPP

general distributions

TIPP, SPADES, GSMPA

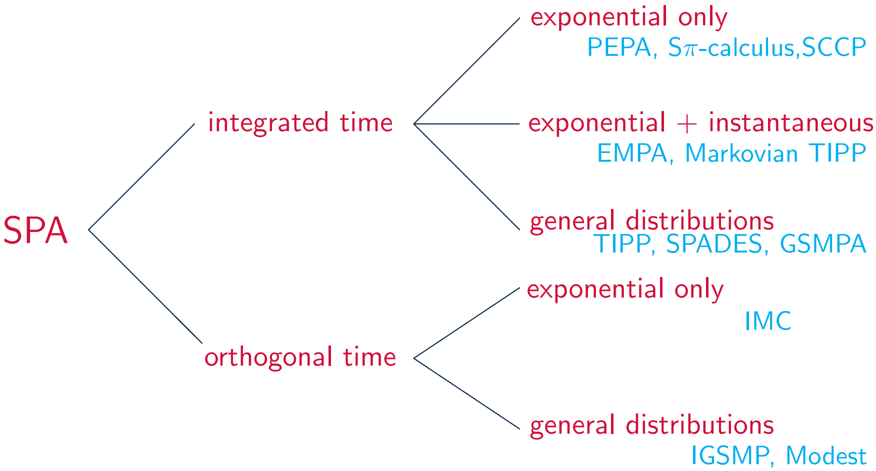
orthogonal time

exponential only

IMC

general distributions

SPA Languages



SPA

integrated time

exponential only

PEPA, Sπ-calculus,SCCP

exponential + instantaneous

EMPA, Markovian TIPP

general distributions

TIPP, SPADES, GSMPA

orthogonal time

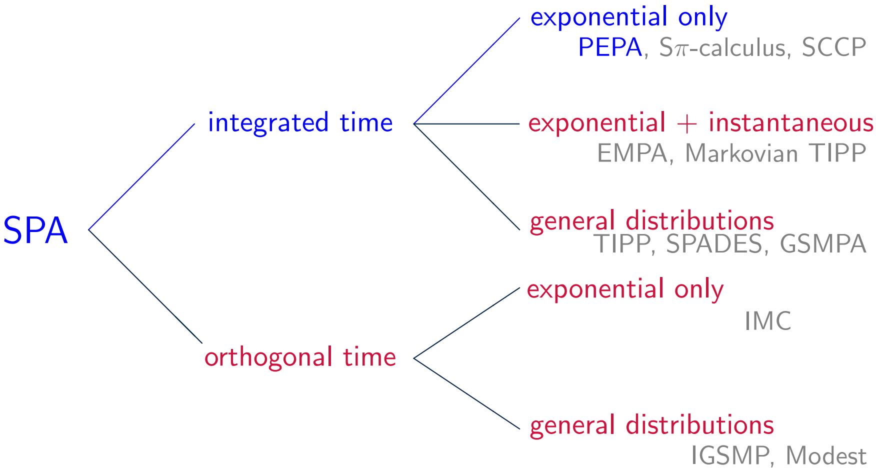
exponential only

IMC

general distributions

IGSMP, Modest

SPA Languages



SPA

integrated time

exponential only

PEPA, Sπ-calculus, SCCP

exponential + instantaneous

EMPA, Markovian TIPP

general distributions

TIPP, SPADES, GSMPA

orthogonal time

exponential only

IMC

general distributions

IGSMP, Modest

Interplay between process algebra and Markov process

- The theoretical development underpinning PEPA has focused on the interplay between the process algebra and the underlying mathematical structure, the Markov process.

Interplay between process algebra and Markov process

- The theoretical development underpinning PEPA has focused on the interplay between the process algebra and the underlying mathematical structure, the Markov process.
- From the process algebra side the Markov chain had a profound influence on the design of the language and in particular on the interactions between components.

Interplay between process algebra and Markov process

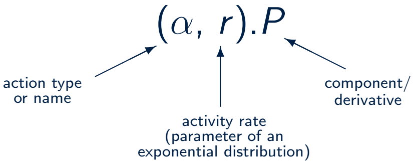
- The theoretical development underpinning PEPA has focused on the interplay between the process algebra and the underlying mathematical structure, the Markov process.

- From the process algebra side the Markov chain had a profound influence on the design of the language and in particular on the interactions between components.

- From the Markov chain perspective the process algebra structure has been exploited to find aspects of independence even between interacting components.

Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.



$$(\alpha, r).P$$

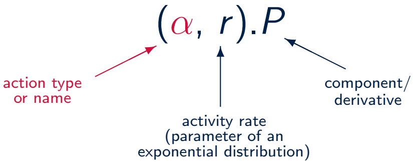
action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.



$$(\alpha, r).P$$

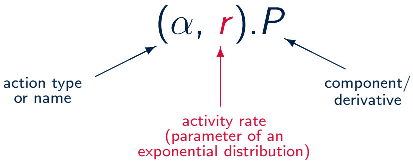
action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.



$$(\alpha, r).P$$

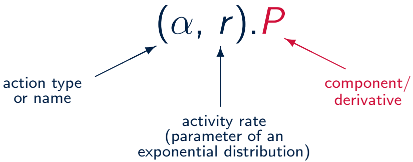
action type
or name

activity rate
(parameter of an
exponential distribution)

component/
derivative

Performance Evaluation Process Algebra

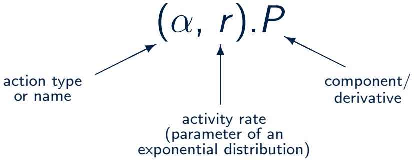
- Models are constructed from components which engage in activities.



$$(\alpha, r).P$$

Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.



$$(\alpha, r).P$$

action type or name

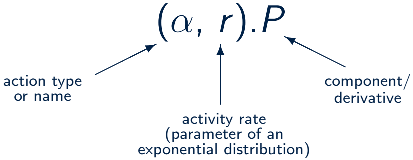
activity rate (parameter of an exponential distribution)

component/ derivative

- The language is used to generate a CTMC for performance modelling.

Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.



$$(\alpha, r).P$$

action type
or name

activity rate
(parameter of an
exponential distribution)

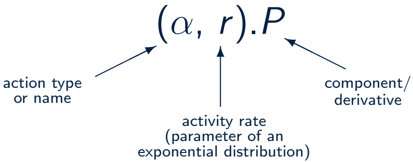
component/
derivative

- The language is used to generate a CTMC for performance modelling.

PEPA
MODEL

Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.



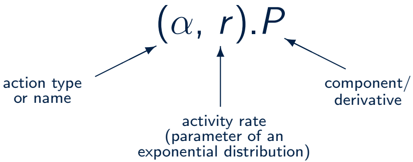
$$(\alpha, r).P$$

- The language is used to generate a CTMC for performance modelling.

PEPA
MODEL
SOS rules

Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.



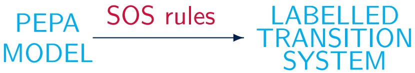
$(\alpha, r).P$

action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

- The language is used to generate a CTMC for performance modelling.



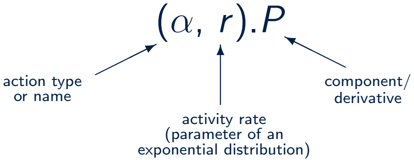
PEPA MODEL

SOS rules

LABELLED TRANSITION SYSTEM

Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.



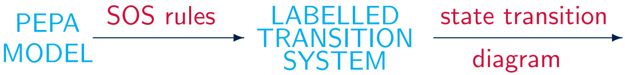
$(\alpha, r).P$

action type
or name

activity rate
(parameter of an
exponential distribution)

component/
derivative

- The language is used to generate a CTMC for performance modelling.



PEPA MODEL

SOS rules

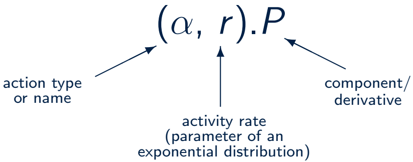
LABELLED TRANSITION SYSTEM

state transition

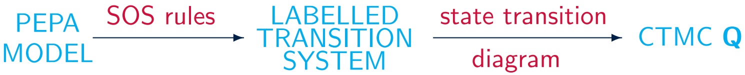
diagram

Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.



- The language is used to generate a CTMC for performance modelling.



PEPA

$$\begin{aligned} S &::= (\alpha, r).S \mid S + S \mid A \\ P &::= S \mid P \underset{L}{\bowtie} P \mid P/L \end{aligned}$$

PEPA

$$\begin{array}{rcl} S & ::= & (\alpha, r).S \mid S + S \mid A \\ P & ::= & S \mid P \underset{L}{\bowtie} P \mid P/L \end{array}$$

PREFIX: $(\alpha, r).S$ designated first action

PEPA

$$
\begin{array}{lcl}
S & ::= & (\alpha, r).S \mid S + S \mid A \\
P & ::= & S \mid P \underset{L}{\bowtie} P \mid P/L
\end{array}
$$

PREFIX:	$(\alpha, r).S$	designated first action
CHOICE:	$S + S$	competing components

PEPA

$$\begin{array}{lcl} S & ::= & (\alpha, r).S \mid S + S \mid A \\ P & ::= & S \mid P \underset{L}{\bowtie} P \mid P/L \end{array}$$

PREFIX:	$(\alpha, r).S$	designated first action
CHOICE:	$S + S$	competing components
CONSTANT:	$A \overset{\mathit{def}}{=} S$	assigning names

PEPA

$$
\begin{aligned}
S &::= (\alpha, r).S \mid S + S \mid A \\
P &::= S \mid P \underset{L}{\bowtie} P \mid P/L
\end{aligned}
$$

PREFIX:	$(\alpha, r).S$	designated first action
CHOICE:	$S + S$	competing components
CONSTANT:	$A \overset{def}{=} S$	assigning names
COOPERATION:	$P \underset{L}{\bowtie} P$	$\alpha \notin L$ individual actions
		$\alpha \in L$ shared actions

PEPA

$$
\begin{array}{rcl}
S & ::= & (\alpha, r).S \mid S + S \mid A \\
P & ::= & S \mid P \underset{L}{\bowtie} P \mid P/L
\end{array}
$$

PREFIX:	$(\alpha, r).S$	designated first action
CHOICE:	$S + S$	competing components
CONSTANT:	$A \overset{def}{=} S$	assigning names
COOPERATION:	$P \underset{L}{\bowtie} P$	$\alpha \notin L$ individual actions $\alpha \in L$ shared actions
HIDING:	P/L	abstraction $\alpha \in L \Rightarrow \alpha \rightarrow \tau$

Example: Browsers, server and download

$$
\begin{aligned}
Server &\stackrel{\mathit{def}}{=} (get, \top).(download, \mu).(rel, \top).Server \\
Browser &\stackrel{\mathit{def}}{=} (display, p\lambda).(get, g).(download, \top).(rel, r).Browser \\
&\quad + (display, (1-p)\lambda).(cache, m).Browser \\
WEB &\stackrel{\mathit{def}}{=} \big(Browser \parallel Browser\big) \underset{L}{\bowtie} Server
\end{aligned}
$$

where $L = \{get, download, rel\}$

Outline

1 Process algebra and Markov processes

2 A semantics for PEPA — informally

3 A formal semantics for PEPA

4 The nature of synchronisation

PEPA activities and rates

When enabled an activity, $a = (\alpha, \lambda)$, will delay for a period determined by its associated distribution function, i.e. the probability that the activity a happens within a period of time of length t is $F_a(t) = 1 - e^{-\lambda t}$.

PEPA activities and rates

- We can think of this as the activity setting a timer whenever it becomes enabled.

PEPA activities and rates

- We can think of this as the activity setting a timer whenever it becomes enabled.
- The time allocated to the timer is determined by the rate of the activity.

PEPA activities and rates

- We can think of this as the activity setting a timer whenever it becomes enabled.
- The time allocated to the timer is determined by the rate of the activity.
- If several activities are enabled at the same time each will have its own associated timer.

PEPA activities and rates

- We can think of this as the activity setting a timer whenever it becomes enabled.
- The time allocated to the timer is determined by the rate of the activity.
- If several activities are enabled at the same time each will have its own associated timer.
- When the first timer finishes that activity takes place—the activity is said to complete or succeed.

PEPA activities and rates

- We can think of this as the activity setting a timer whenever it becomes enabled.
- The time allocated to the timer is determined by the rate of the activity.
- If several activities are enabled at the same time each will have its own associated timer.
- When the first timer finishes that activity takes place—the activity is said to complete or succeed.
- This means that the activity is considered to "happen": an external observer will witness the event of activity of type α.

PEPA activities and rates

- We can think of this as the activity setting a timer whenever it becomes enabled.
- The time allocated to the timer is determined by the rate of the activity.
- If several activities are enabled at the same time each will have its own associated timer.
- When the first timer finishes that activity takes place—the activity is said to complete or succeed.
- This means that the activity is considered to “happen”: an external observer will witness the event of activity of type α.
- An activity may be preempted, or aborted, if another one completes first.

PEPA and Markov processes

In a PEPA model if we define the stochastic process $X(t)$, such that $X(t) = C_i$ indicates that the system behaves as component C_i at time t, then $X(t)$ is a Markov process which can be characterised by a matrix, $\boldsymbol{Q}$.

PEPA and Markov processes

In a PEPA model if we define the stochastic process $X(t)$, such that $X(t) = C_i$ indicates that the system behaves as component C_i at time t, then $X(t)$ is a Markov process which can be characterised by a matrix, $\boldsymbol{Q}$.

A stationary or equilibrium probability distribution, $\boldsymbol{\pi}(\cdot)$, exists for every time-homogeneous irreducible Markov process whose states are all positive-recurrent.

PEPA and Markov processes

In a PEPA model if we define the stochastic process $X(t)$, such that $X(t) = C_i$ indicates that the system behaves as component C_i at time t, then $X(t)$ is a Markov process which can be characterised by a matrix, $\boldsymbol{Q}$.

A stationary or equilibrium probability distribution, $\boldsymbol{\pi}(\cdot)$, exists for every time-homogeneous irreducible Markov process whose states are all positive-recurrent.

This distribution is found by solving the global balance equation

$$\boldsymbol{\pi}\boldsymbol{Q} = \mathbf{0}$$

subject to the normalisation condition

$$\sum \boldsymbol{\pi}(C_i) = 1.$$

PEPA and time

All PEPA models are time-homogeneous since all activities are time-homogeneous: the rate and type of activities enabled by a component are independent of time.

PEPA and irreducibility and positive-recurrence

The other conditions, irreducibility and positive-recurrent states, are easily expressed in terms of the derivation graph of the PEPA model.

We only consider PEPA models with a finite number of states so if the model is irreducible then all states must be positive-recurrent i.e. the derivation graph is strongly connected.

PEPA and irreducibility and positive-recurrence

The other conditions, irreducibility and positive-recurrent states, are easily expressed in terms of the derivation graph of the PEPA model.

We only consider PEPA models with a finite number of states so if the model is irreducible then all states must be positive-recurrent i.e. the derivation graph is strongly connected.

In terms of the PEPA model this means that all behaviours of the system must be recurrent; in particular, for every choice, whichever path is chosen it must eventually return to the point where the choice can be made again, possibly with a different outcome.

Outline

1 Process algebra and Markov processes

2 A semantics for PEPA — informally

3 A formal semantics for PEPA

4 The nature of synchronisation

Structured Operational Semantics

PEPA is defined using a Plotkin-style structured operational semantics (a “small step” semantics).

Structured Operational Semantics

PEPA is defined using a Plotkin-style structured operational semantics (a “small step” semantics).

In this style of semantics we build a labelled transition system to capture the possible evolutions or derivations of a model.

Structured Operational Semantics

PEPA is defined using a Plotkin-style structured operational semantics (a “small step” semantics).

In this style of semantics we build a labelled transition system to capture the possible evolutions or derivations of a model.

A labelled transition system is a set of process terms $\mathcal{P}$, a set of action labels $\mathcal{A}$ and a relation $\mathcal{P} \times \mathcal{A} \times \mathcal{P}$ given by the operational rules.

Structured Operational Semantics

PEPA is defined using a Plotkin-style structured operational semantics (a “small step” semantics).

In this style of semantics we build a labelled transition system to capture the possible evolutions or derivations of a model.

A labelled transition system is a set of process terms $\mathcal{P}$, a set of action labels $\mathcal{A}$ and a relation $\mathcal{P} \times \mathcal{A} \times \mathcal{P}$ given by the operational rules.

In the rules, the derivation below the line can be inferred from the premise above the line.

Structured Operational Semantics

PEPA is defined using a Plotkin-style structured operational semantics (a “small step” semantics).

In this style of semantics we build a labelled transition system to capture the possible evolutions or derivations of a model.

A labelled transition system is a set of process terms $\mathcal{P}$, a set of action labels $\mathcal{A}$ and a relation $\mathcal{P} \times \mathcal{A} \times \mathcal{P}$ given by the operational rules.

In the rules, the derivation below the line can be inferred from the premise above the line.

Note that in this semantics the rate information is only treated as an additional label.

Structured Operational Semantics: Prefix and Choice

Prefix

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

Structured Operational Semantics: Prefix and Choice

Prefix

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

Choice

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

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

Structured Operational Semantics: Cooperation ($\alpha \notin L$)

Cooperation

$$\frac{E \xrightarrow{(\alpha,r)} E'}{E \underset{L}{\bowtie} F \xrightarrow{(\alpha,r)} E' \underset{L}{\bowtie} F} \; (\alpha \notin L)$$

Structured Operational Semantics: Cooperation ($\alpha \notin L$)

Cooperation

$$\frac{E \xrightarrow{(\alpha,r)} E'}{E \underset{L}{\bowtie} F \xrightarrow{(\alpha,r)} E' \underset{L}{\bowtie} F} \; (\alpha \notin L)$$

$$\frac{F \xrightarrow{(\alpha,r)} F'}{E \underset{L}{\bowtie} F \xrightarrow{(\alpha,r)} E \underset{L}{\bowtie} F'} \; (\alpha \notin L)$$

Structured Operational Semantics: Cooperation ($\alpha \in L$)

Cooperation $$\frac{E \xrightarrow{(\alpha,r_1)} E' \qquad F \xrightarrow{(\alpha,r_2)} F'}{E \underset{L}{\bowtie} F \xrightarrow{(\alpha,R)} E' \underset{L}{\bowtie} F'} \ (\alpha \in L)$$

Structured Operational Semantics: Cooperation ($\alpha \in L$)

Cooperation $$\frac{E \xrightarrow{(\alpha,r_1)} E' \qquad F \xrightarrow{(\alpha,r_2)} F'}{E \underset{L}{\bowtie} F \xrightarrow{(\alpha,R)} E' \underset{L}{\bowtie} F'} \; (\alpha \in L)$$

where $$R = \frac{r_1}{r_\alpha(E)} \frac{r_2}{r_\alpha(F)} min(r_\alpha(E), r_\alpha(F))$$

Apparent Rate

$$
\begin{aligned}
r_\alpha((\beta, r).P) &= \begin{cases} r & \beta = \alpha \\ 0 & \beta \neq \alpha \end{cases} \\
r_\alpha(P + Q) &= r_\alpha(P) + r_\alpha(Q) \\
r_\alpha(A) &= r_\alpha(P) \qquad \text{where } A \overset{def}{=} P \\
r_\alpha(P \underset{L}{\bowtie} Q) &= \begin{cases} r_\alpha(P) + r_\alpha(Q) & \alpha \notin L \\ min(r_\alpha(P), r_\alpha(Q)) & \alpha \in L \end{cases} \\
r_\alpha(P/L) &= \begin{cases} r_\alpha(P) & \alpha \notin L \\ 0 & \alpha \in L \end{cases}
\end{aligned}
$$

Structured Operational Semantics: Hiding

Hiding

$$\frac{E \xrightarrow{(\alpha,r)} E'}{E/L \xrightarrow{(\alpha,r)} E'/L} \; (\alpha \notin L)$$

Structured Operational Semantics: Hiding

Hiding

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

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

Structured Operational Semantics: Constants

Constant

$$\frac{E \xrightarrow{(\alpha,r)} E'}{A \xrightarrow{(\alpha,r)} E'} \; (A \overset{\mathit{def}}{=} E)$$

Properties of the definition (1)

PEPA has no “nil” (a deadlocked process).

This is because the PEPA language is intended for modelling non-stop processes (such as Web servers, operating systems, or manufacturing processes) rather than for modelling terminating processes (a compilation, a sorting routine, and so forth).

Creating a deadlocked process

When we are interested in transient behaviour we use the deadlocked process *Stop* to signal a component which performs no further actions.

$$Stop \overset{def}{=} \Big(((a, r).Stop) \underset{\{a,b\}}{\bowtie} ((b, r).Stop)\Big)/\{\, a, b \,\}$$

Properties of the definition (2)

Cooperation in PEPA is multi-way. Two, three, four or more partners may cooperate, and they all need to synchronise for the activity to happen.

Properties of the definition (2)

Cooperation in PEPA is multi-way. Two, three, four or more partners may cooperate, and they all need to synchronise for the activity to happen.

This comes from the fact that synchronisation has the form $a, a \rightarrow a$ (as in CSP) instead of $a, \bar{a} \rightarrow \tau$ (as in CCS and the π-calculus).

Properties of the definition (2)

Cooperation in PEPA is multi-way. Two, three, four or more partners may cooperate, and they all need to synchronise for the activity to happen.

This comes from the fact that synchronisation has the form $a, a \rightarrow a$ (as in CSP) instead of $a, \bar{a} \rightarrow \tau$ (as in CCS and the π-calculus).

This is used to have "witnesses" to events (known as stochastic probes).

Properties of the definition (3)

- Because of its mapping onto a CTMC, PEPA has an interleaving semantics.

Properties of the definition (3)

- Because of its mapping onto a CTMC, PEPA has an interleaving semantics.
- Other modelling formalisms based on CTMCs are also based on an interleaving semantics (e.g. Generalised Stochastic Petri nets).

Properties of the definition (3)

- Because of its mapping onto a CTMC, PEPA has an interleaving semantics.
- Other modelling formalisms based on CTMCs are also based on an interleaving semantics (e.g. Generalised Stochastic Petri nets).
- As we have seen a continuous time Markov chain (CTMC) is generated from a PEPA model via its structured operational semantics.

Properties of the definition (3)

- Because of its mapping onto a CTMC, PEPA has an interleaving semantics.
- Other modelling formalisms based on CTMCs are also based on an interleaving semantics (e.g. Generalised Stochastic Petri nets).
- As we have seen a continuous time Markov chain (CTMC) is generated from a PEPA model via its structured operational semantics.
- Linear algebra is used to solve the model in terms of equilibrium behaviour.

Properties of the definition (3)

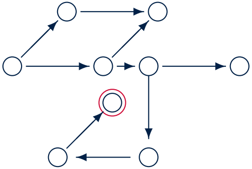
- Because of its mapping onto a CTMC, PEPA has an interleaving semantics.
- Other modelling formalisms based on CTMCs are also based on an interleaving semantics (e.g. Generalised Stochastic Petri nets).
- As we have seen a continuous time Markov chain (CTMC) is generated from a PEPA model via its structured operational semantics.
- Linear algebra is used to solve the model in terms of equilibrium behaviour.
- The resulting probability distribution is seldom the ultimate goal of performance analysis; a modeller derives performance measures from this distribution via a reward structure.

Integrated analysis

Qualitative verification can now be complemented by quantitative verification.

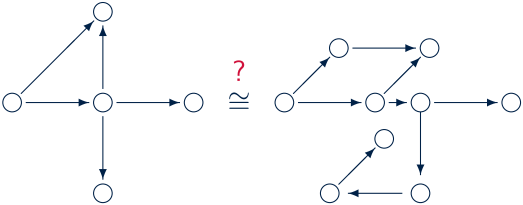
Integrated analysis: Reachability analysis

How long will it take for the system to arrive in a particular state?



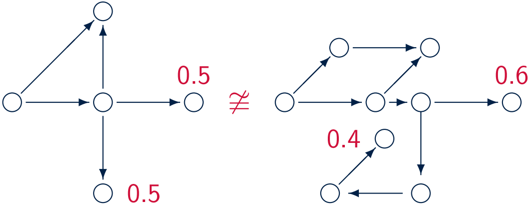
Integrated analysis: Specification matching

With what probability does system behaviour match its specification?



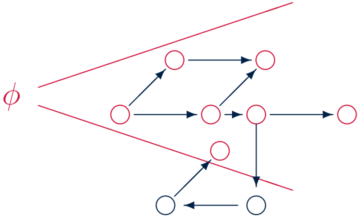
Integrated analysis: Specification matching

Does the "*frequency profile*" of the system match that of the specification?



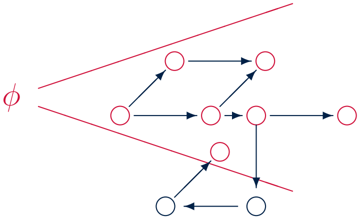
Integrated analysis: Model checking

Does a given property ϕ hold within the system with a given probability?



Integrated analysis: Model checking

For a given starting state
how long is it until
a given property ϕ holds?



The Importance of Being Exponential

$$\begin{tikzcd}
 & (\alpha, r).\mathit{Stop} \parallel (\beta, s).\mathit{Stop} \arrow[dl, "{(\alpha, r)}"'] \arrow[dr, "{(\beta, s)}"] & \\
\mathit{Stop} \parallel (\beta, s).\mathit{Stop} \arrow[dr, "{(\beta, s)}"'] & & (\alpha, r).\mathit{Stop} \parallel \mathit{Stop} \arrow[dl, "{(\alpha, r)}"] \\
 & \mathit{Stop} \parallel \mathit{Stop} &
\end{tikzcd}$$

The Importance of Being Exponential

$$\begin{tikzcd}
 & (\alpha, r).\mathit{Stop} \parallel (\beta, s).\mathit{Stop} \arrow[dl, swap, "(\alpha, r)"] \arrow[dr, "(\beta, s)"] & \\
\mathit{Stop} \parallel (\beta, s).\mathit{Stop} \arrow[dr, swap, "(\beta, s)"] & & (\alpha, r).\mathit{Stop} \parallel \mathit{Stop} \arrow[dl, "(\alpha, r)"] \\
 & \mathit{Stop} \parallel \mathit{Stop} &
\end{tikzcd}$$

The Importance of Being Exponential

$$\begin{tikzcd}
 & (\alpha, r).\mathit{Stop} \parallel (\beta, s).\mathit{Stop} \arrow[dl, swap, "(\alpha, r)"] \arrow[dr, "(\beta, s)"] & \\
\mathit{Stop} \parallel (\beta, s).\mathit{Stop} \arrow[dr, swap, "(\beta, s)"] & & (\alpha, r).\mathit{Stop} \parallel \mathit{Stop} \arrow[dl, "(\alpha, r)"] \\
 & \mathit{Stop} \parallel \mathit{Stop} &
\end{tikzcd}$$

The Importance of Being Exponential

$$\begin{tikzcd}
 & (\alpha, r).Stop \parallel (\beta, s).Stop \arrow[dl, "(\alpha, r)"'] \arrow[dr, "(\beta, s)"] \arrow[dd, "(\alpha, r)"', "(\beta, s)"] & \\
Stop \parallel (\beta, s).Stop \arrow[dr, "(\beta, s)"'] & & (\alpha, r).Stop \parallel Stop \arrow[dl, "(\alpha, r)"] \\
 & Stop \parallel Stop &
\end{tikzcd}$$

The Importance of Being Exponential

$$\begin{tikzcd}
& (\alpha, r).\mathit{Stop} \parallel (\beta, s).\mathit{Stop} \arrow[dl, "(\alpha, r)"'] \arrow[dr, "(\beta, s)"] & \\
\mathit{Stop} \parallel (\beta, s).\mathit{Stop} \arrow[dr, "(\beta, s)"'] & & (\alpha, r).\mathit{Stop} \parallel \mathit{Stop} \arrow[dl, "(\alpha, r)"] \\
& \mathit{Stop} \parallel \mathit{Stop} &
\end{tikzcd}$$

The Importance of Being Exponential

$$\begin{tikzcd}
 & (\alpha, r).\mathit{Stop} \parallel (\beta, s).\mathit{Stop} \arrow[dl, "{(\alpha, r)}"'] \arrow[dr, "{(\beta, s)}"] & \\
\mathit{Stop} \parallel (\beta, s).\mathit{Stop} \arrow[dr, "{(\beta, s)}"'] & & (\alpha, r).\mathit{Stop} \parallel \mathit{Stop} \arrow[dl, "{(\alpha, r)}"] \\
 & \mathit{Stop} \parallel \mathit{Stop} &
\end{tikzcd}$$

The Importance of Being Exponential

$$\begin{tikzcd}
 & (\alpha, r).Stop \parallel (\beta, s).Stop \arrow[dl, "{(\alpha, r)}"'] \arrow[dr, "{(\beta, s)}"] & \\
Stop \parallel (\beta, s).Stop \arrow[dr, "{(\beta, s)}"'] & & (\alpha, r).Stop \parallel Stop \arrow[dl, "{(\alpha, r)}"] \\
 & Stop \parallel Stop &
\end{tikzcd}$$

The Importance of Being Exponential

$$\begin{tikzcd}
 & (\alpha, r).Stop \parallel (\beta, s).Stop \arrow[dl, "(\alpha, r)"'] \arrow[dr, "(\beta, s)"] & \\
Stop \parallel (\beta, s).Stop \arrow[dr, "(\beta, s)"'] & & (\alpha, r).Stop \parallel Stop \arrow[dl, "(\alpha, r)"] \\
 & Stop \parallel Stop &
\end{tikzcd}$$

The memoryless property of the negative exponential distribution means that residual times do not need to be recorded.

The exponential distribution and the expansion law

We retain the expansion law of classical process algebra:

$$\begin{aligned}&(\alpha, r).\mathit{Stop} \parallel (\beta, s).\mathit{Stop} = \\ &\qquad (\alpha, r).(\beta, s).(\mathit{Stop} \parallel \mathit{Stop}) + (\beta, s).(\alpha, r).(\mathit{Stop} \parallel \mathit{Stop})\end{aligned}$$

only if the negative exponential distribution is used.

Outline

1. Process algebra and Markov processes
2. A semantics for PEPA — informally
3. A formal semantics for PEPA
4. The nature of synchronisation

Parallel Composition

- Parallel composition is the basis of the compositionality in a process algebra

Parallel Composition

- Parallel composition is the basis of the compositionality in a process algebra — it defines which components interact and how.

Parallel Composition

- Parallel composition is the basis of the compositionality in a process algebra — it defines which components interact and how.
- In classical process algebra is it often associated with communication.

Parallel Composition

- Parallel composition is the basis of the compositionality in a process algebra — it defines which components interact and how.
- In classical process algebra is it often associated with communication.
- When the activities of the process algebra have a duration the definition of parallel composition becomes more complex.

Parallel Composition

- Parallel composition is the basis of the compositionality in a process algebra — it defines which components interact and how.

- In classical process algebra is it often associated with communication.

- When the activities of the process algebra have a duration the definition of parallel composition becomes more complex.

- The issue of what it means for two timed activities to synchronise is a vexed one....

Who Synchronises...?

Even within classical process algebras there is variation in the interpretation of parallel composition:

Who Synchronises...?

Even within classical process algebras there is variation in the interpretation of parallel composition:

CCS-style

- Actions are partitioned into input and output pairs.
- Communication or synchronisation takes places between conjugate pairs.
- The resulting action has silent type τ.

Who Synchronises...?

Even within classical process algebras there is variation in the interpretation of parallel composition:

CCS-style

- Actions are partitioned into input and output pairs.
- Communication or synchronisation takes places between conjugate pairs.
- The resulting action has silent type τ.

CSP-style

- No distinction between input and output actions.
- Communication or synchronisation takes place on the basis of shared names.
- The resulting action has the same name.

Who Synchronises...?

Even within classical process algebras there is variation in the interpretation of parallel composition:

CCS-style

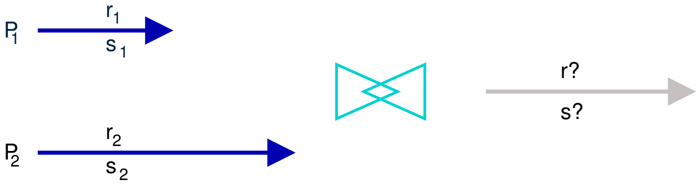
- Actions are partitioned into input and output pairs.
- Communication or synchronisation takes places between conjugate pairs.
- The resulting action has silent type τ.

CSP-style

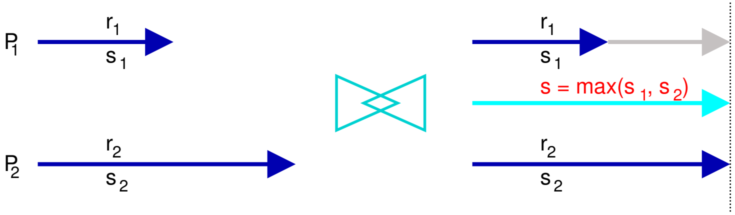
- No distinction between input and output actions.
- Communication or synchronisation takes place on the basis of shared names.
- The resulting action has the same name.

Most stochastic process algebras adopt CSP-style synchronisation.

Timed Synchronisation

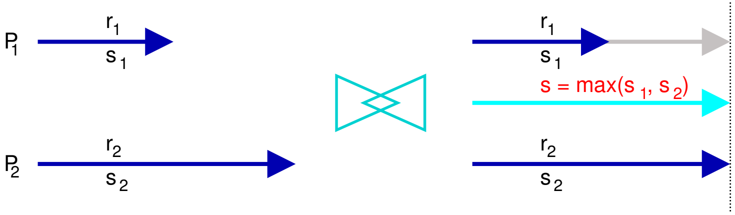


Timed Synchronisation



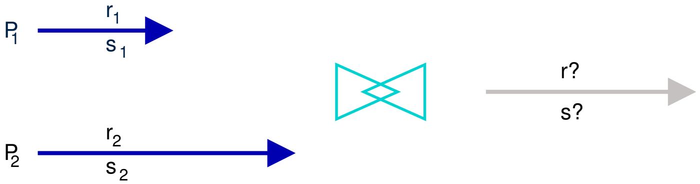
Barrier Synchronisation

Timed Synchronisation



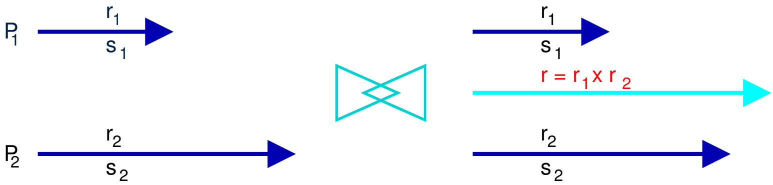
s is no longer exponentially distributed

Timed Synchronisation



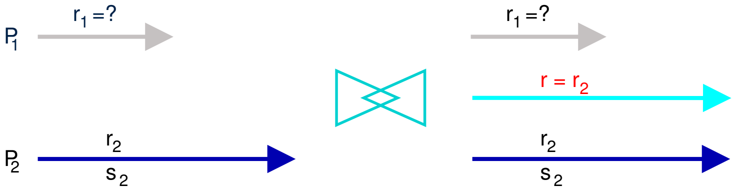
algebraic considerations limit choices

Timed Synchronisation



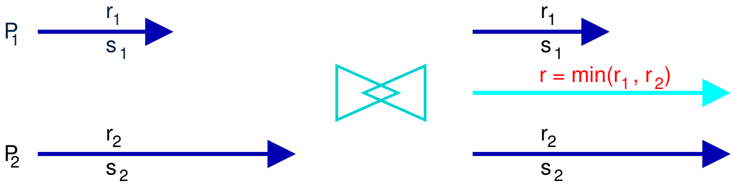
TIPP: new rate is product of individual rates

Timed Synchronisation



EMPA: one participant is passive

Timed Synchronisation



bounded capacity: new rate is the minimum of the rates

Bounded capacity

Within the cooperation framework, PEPA assumes bounded capacity: that is, a component cannot be made to perform an activity faster by cooperation, so the rate of a shared activity is the minimum of the apparent rates of the activity in the cooperating components.

Apparent Rate

$$
\begin{array}{rcll}
r_\alpha((\beta, r).P) & = & \begin{cases} r & \beta = \alpha \\ 0 & \beta \neq \alpha \end{cases} \\[2ex]
r_\alpha(P + Q) & = & r_\alpha(P) + r_\alpha(Q) \\[2ex]
r_\alpha(A) & = & r_\alpha(P) & \text{where } A \overset{def}{=} P \\[2ex]
r_\alpha(P \underset{L}{\bowtie} Q) & = & \begin{cases} r_\alpha(P) + r_\alpha(Q) & \alpha \notin L \\ min(r_\alpha(P), r_\alpha(Q)) & \alpha \in L \end{cases} \\[2ex]
r_\alpha(P/L) & = & \begin{cases} r_\alpha(P) & \alpha \notin L \\ 0 & \alpha \in L \end{cases}
\end{array}
$$

Apparent Rate

$$\begin{aligned}
r_\alpha((\beta, r).P) &= \begin{cases} r & \beta = \alpha \\ 0 & \beta \neq \alpha \end{cases} \\
r_\alpha(P + Q) &= r_\alpha(P) + r_\alpha(Q) \\
r_\alpha(A) &= r_\alpha(P) \qquad \text{where } A \overset{def}{=} P \\
r_\alpha(P \underset{L}{\bowtie} Q) &= \begin{cases} r_\alpha(P) + r_\alpha(Q) & \alpha \notin L \\ min(r_\alpha(P), r_\alpha(Q)) & \alpha \in L \end{cases} \\
r_\alpha(P/L) &= \begin{cases} r_\alpha(P) & \alpha \notin L \\ 0 & \alpha \in L \end{cases}
\end{aligned}$$

This is used to calculate pairwise cooperation rates: the overall rate of cooperation must not exceed either of the constituent apparent rates.

Cooperation in PEPA

- In PEPA each component has a bounded capacity to carry out activities of any particular type, determined by the apparent rate for that type.

Cooperation in PEPA

- In PEPA each component has a bounded capacity to carry out activities of any particular type, determined by the apparent rate for that type.
- Synchronisation, or cooperation cannot make a component exceed its bounded capacity.

Cooperation in PEPA

- In PEPA each component has a bounded capacity to carry out activities of any particular type, determined by the apparent rate for that type.
- Synchronisation, or cooperation cannot make a component exceed its bounded capacity.
- Thus the apparent rate of a cooperation is the minimum of the apparent rates of the co-operands.