From Markov to Milner and back: Stochastic process algebras

Jane Hillston

School of Informatics
The University of Edinburgh
Scotland

15th August 2010

Outline

Outline

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.

$$\underbrace{\alpha}_{\text{action type or name}}.\underbrace{P}_{\text{agent/ component}}$$

Process Algebra

- Models consist of agents which engage in actions.

$$\underbrace{\alpha}_{\text{action type or name}}.\underbrace{P}_{\text{agent/ component}}$$

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.

$$\underbrace{\alpha}_{\text{action type or name}}.\underbrace{P}_{\text{agent/ component}}$$

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

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

Example

Consider a web server which offers html pages for download:

$$Server \stackrel{\mathit{def}}{=} get.download.rel.Server$$

Example

Consider a web server which offers html pages for download:

$$Server \stackrel{\mathit{def}}{=} get.download.rel.Server$$

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.

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

Example

Consider a web server which offers html pages for download:

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

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.

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

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

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

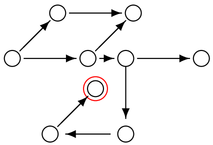
Qualitative Analysis

- 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.

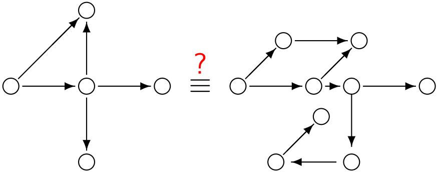
Will the system arrive in a particular state?



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.

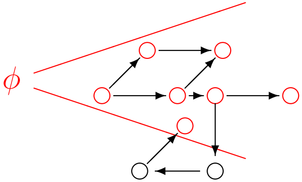
Does system behaviour match its specification?



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?



Modelling computer systems: the challenges

- Physical distance
  - Network latency

Modelling computer systems: the challenges

- Physical distance
  - Network latency
- Partial failures
  - Server may be down
  - Routers may be down

Modelling computer systems: the challenges

- Physical distance
    - Network latency
- Partial failures
    - Server may be down
    - Routers may be down
- Scale
    - Workload characterisation

Modelling computer systems: the challenges

- Physical distance
  - Network latency
- Partial failures
  - Server may be down
  - Routers may be down
- Scale
  - Workload characterisation
- Resource sharing
  - Network contention
  - CPU load

Modelling computer systems: the challenges

- Physical distance — need to represent time
  - Network latency
- Partial failures
  - Server may be down
  - Routers may be down
- Scale
  - Workload characterisation
- Resource sharing
  - Network contention
  - CPU load

Modelling computer systems: the challenges

- Physical distance — need to represent time
  - Network latency
- Partial failures — randomness and probability
  - Server may be down
  - Routers may be down
- Scale
  - Workload characterisation
- Resource sharing
  - Network contention
  - CPU load

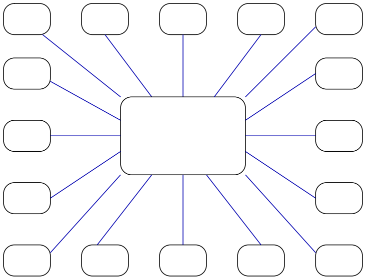
Modelling computer systems: the challenges

- Physical distance — need to represent time
  - Network latency
- Partial failures — randomness and probability
  - Server may be down
  - Routers may be down
- Scale — need to quantify population sizes
  - Workload characterisation
- Resource sharing
  - Network contention
  - CPU load

Modelling computer systems: the challenges

- Physical distance — need to represent time
  - Network latency
- Partial failures — randomness and probability
  - Server may be down
  - Routers may be down
- Scale — need to quantify population sizes
  - Workload characterisation
- Resource sharing — need to express percentages
  - Network contention
  - CPU load

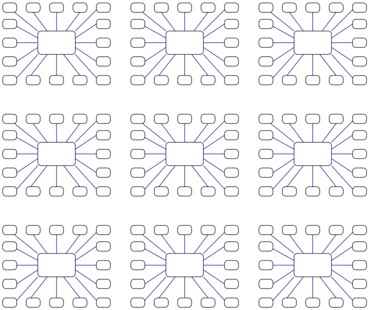
Quantitative Modelling: Motivation



Quality of Service issues

- Can the server maintain reasonable response times?

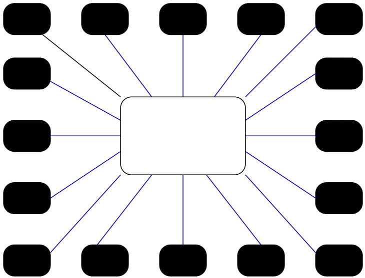
Quantitative Modelling: Motivation



Scalability issues

- How many times do we have to replicate this service to support all of the subscribers?

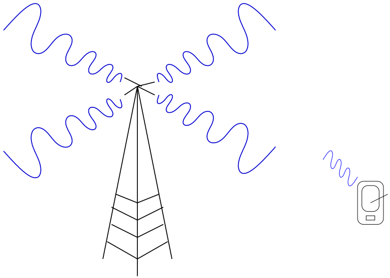
Quantitative Modelling: Motivation



Scalability issues

- Will the server withstand a distributed denial of service attack?

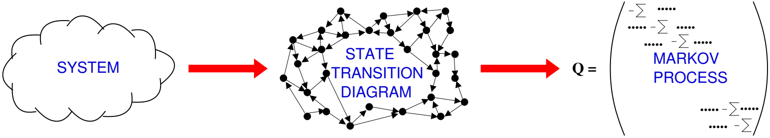
Quantitative Modelling: Motivation



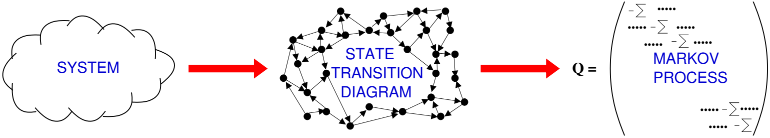
Service-level agreements

- What percentage of downloads do complete within the time we advertised?

Performance Modelling using CTMC



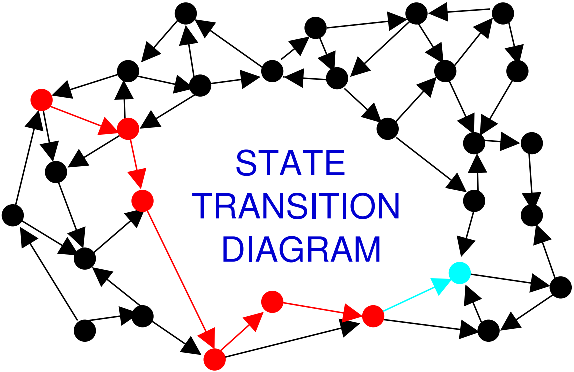
Performance Modelling using CTMC



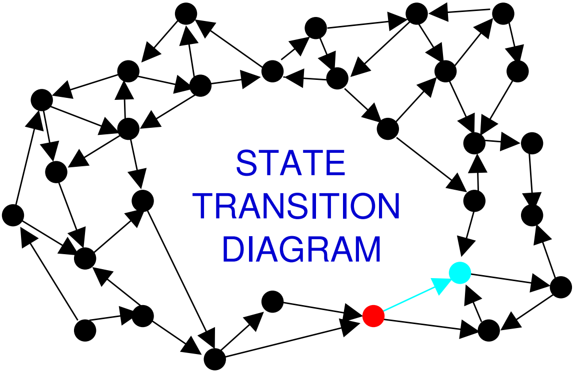
A stochastic process $X(t)$ is a Markov process iff for all $t_0 < t_1 < \ldots < t_n < t_{n+1}$, the joint probability distribution of $(X(t_0), X(t_1), \ldots, X(t_n), X(t_{n+1}))$ is such that

$$\Pr(X(t_{n+1}) = s_{i_{n+1}} \mid X(t_0) = s_{i_0}, \ldots, X(t_n) = s_{i_n}) = \Pr(X(t_{n+1}) = s_{i_{n+1}} \mid X(t_n) = s_{i_n})$$

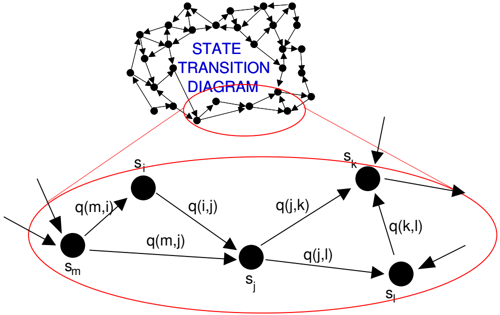
Performance Modelling using CTMC



Performance Modelling using CTMC

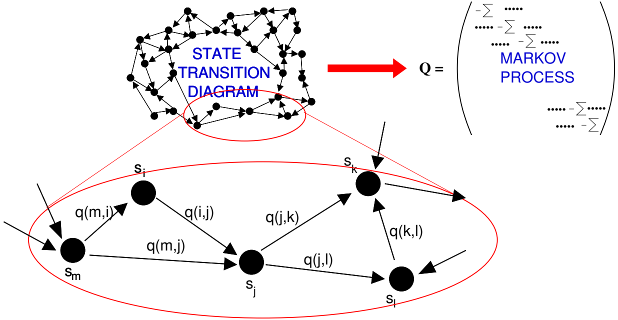


Performance Modelling using CTMC



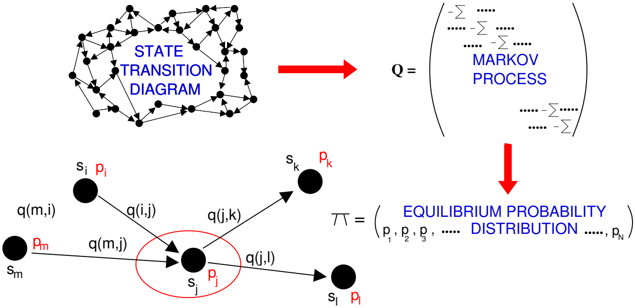
A negative exponentially distributed duration is associated with each transition.

Performance Modelling using CTMC



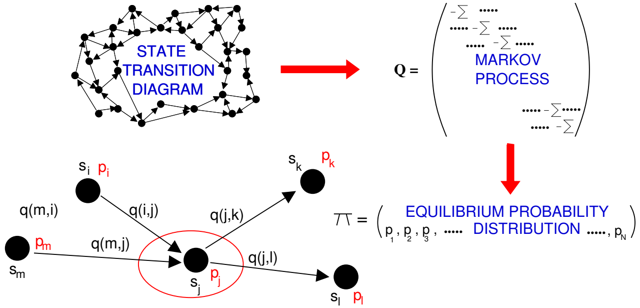
these parameters form the entries of the infinitesimal generator matrix Q

Performance Modelling using CTMC



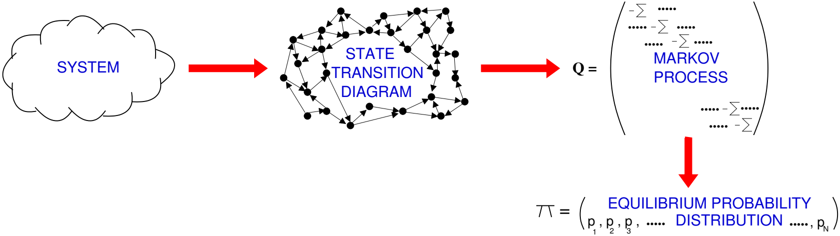
In steady state the probability flux out of a state is balanced by the flux in.

Performance Modelling using CTMC

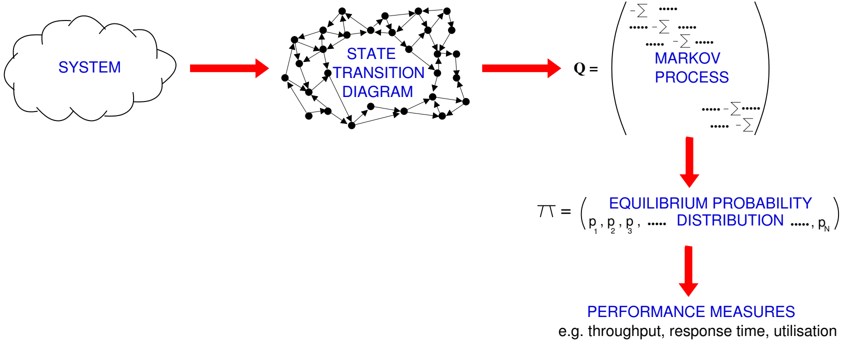


"Global balance equations" captured by $\pi Q = 0$ solved by linear algebra

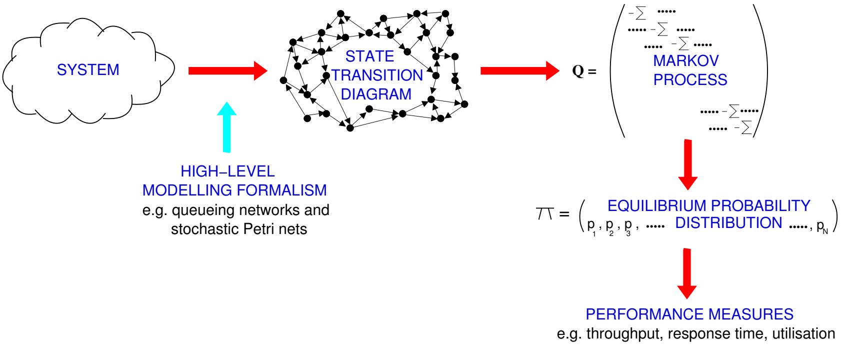
Performance Modelling using CTMC



Performance Modelling using CTMC



Performance Modelling using CTMC



Stochastic process algebras

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

Stochastic 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

Stochastic 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

Stochastic 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

Stochastic 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

Stochastic 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.

Stochastic 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.

SPA
MODEL

Stochastic 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.

SPA MODEL $\xrightarrow{\text{SOS rules}}$

Stochastic 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.

SPA MODEL $\xrightarrow{\text{SOS rules}}$ LABELLED TRANSITION SYSTEM

Stochastic 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.

SPA MODEL $\xrightarrow{\text{SOS rules}}$ LABELLED TRANSITION SYSTEM $\xrightarrow[\text{diagram}]{\text{state transition}}$

Stochastic 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.

$$\text{SPA MODEL} \xrightarrow{\text{SOS rules}} \text{LABELLED TRANSITION SYSTEM} \xrightarrow[\text{diagram}]{\text{state transition}} \text{CTMC } \mathbf{Q}$$

Stochastic 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.

SPA MODEL $\xrightarrow{\text{SOS rules}}$ LABELLED MULTI-TRANSITION SYSTEM $\xrightarrow[\text{diagram}]{\text{state transition}}$ CTMC $\mathbf{Q}$

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

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

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

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:	$A \stackrel{\mathit{def}}{=} S$	assigning names

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:	$A \stackrel{\mathit{def}}{=} S$	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:	$A \stackrel{\mathit{def}}{=} S$	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

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

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

SPA Languages

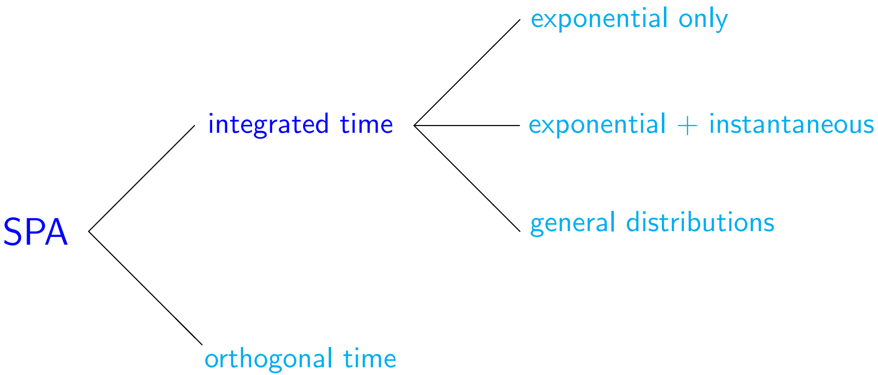
SPA

SPA Languages

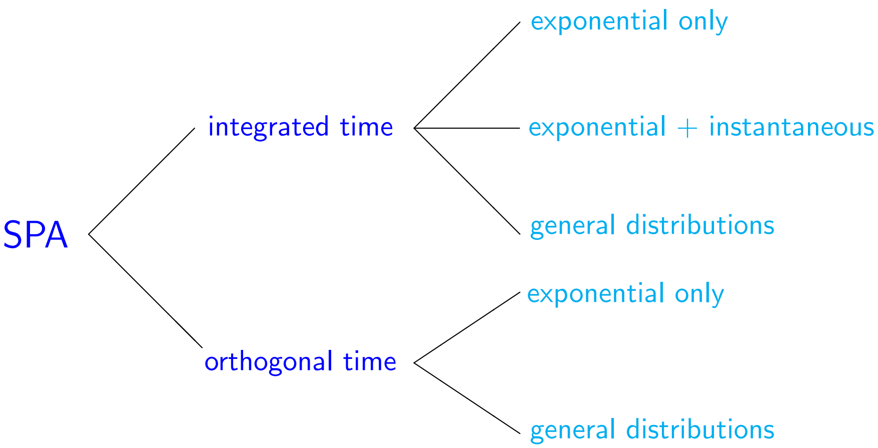
SPA

- integrated time
- orthogonal time

SPA Languages



SPA Languages



SPA Languages

- SPA
  - integrated time
    - exponential only
      - PEPA, Stochastic π-calculus
    - exponential + instantaneous
    - general distributions
  - orthogonal time
    - exponential only
    - general distributions

SPA Languages

- SPA
  - integrated time
    - exponential only
      - PEPA, Stochastic π-calculus
    - exponential + instantaneous
      - EMPA, Markovian TIPP
    - general distributions
  - orthogonal time
    - exponential only
    - general distributions

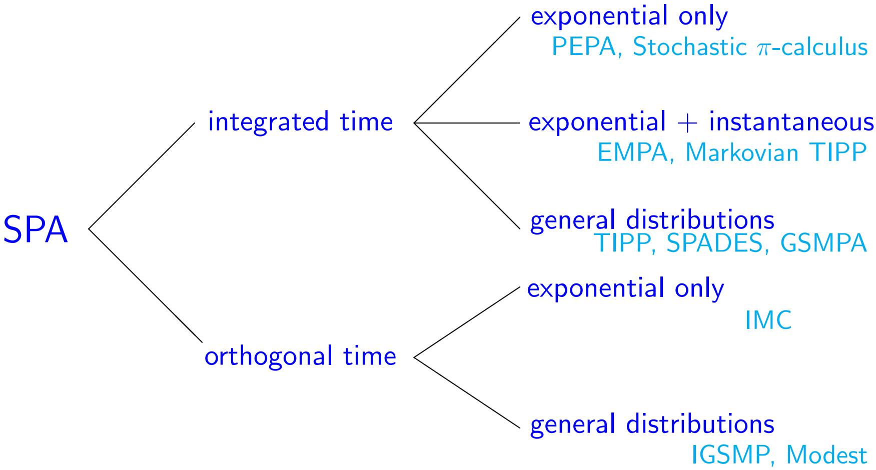
SPA Languages

- SPA
  - integrated time
    - exponential only
      - PEPA, Stochastic π-calculus
    - exponential + instantaneous
      - EMPA, Markovian TIPP
    - general distributions
      - TIPP, SPADES, GSMPA
  - orthogonal time
    - exponential only
    - general distributions

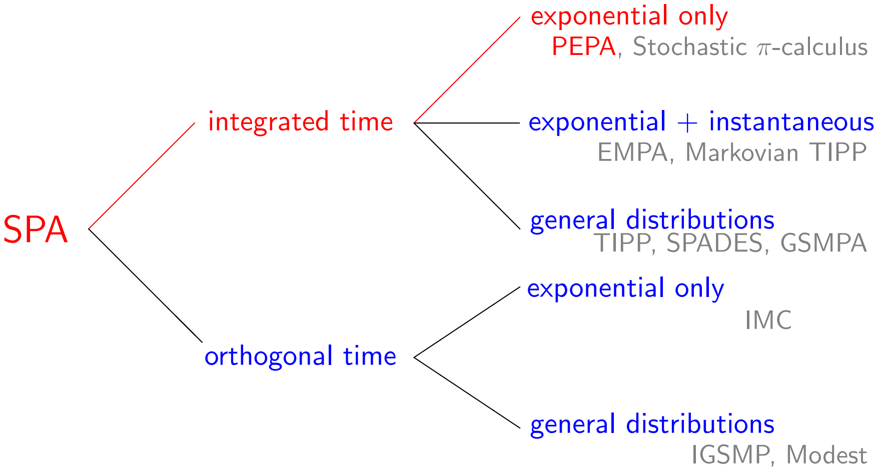
SPA Languages

- SPA
  - integrated time
    - exponential only
      - PEPA, Stochastic π-calculus
    - exponential + instantaneous
      - EMPA, Markovian TIPP
    - general distributions
      - TIPP, SPADES, GSMPA
  - orthogonal time
    - exponential only
      - IMC
    - general distributions

SPA Languages



SPA Languages

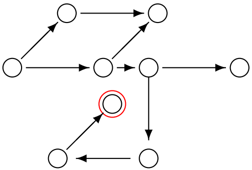


Integrated analysis

Qualitative verification can now be complemented by quantitative verification.

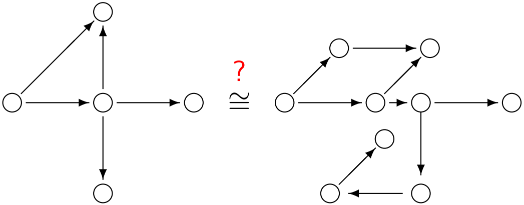
Integrated analysis: Reachability analysis

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



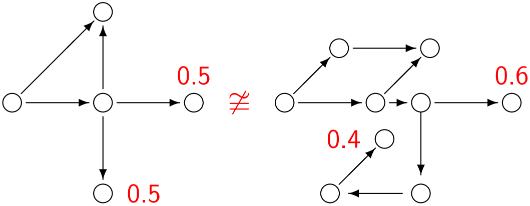
Integrated analysis: Specification matching

With what probability does system behaviour match its specification?



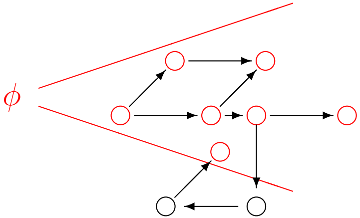
Integrated analysis: Specification matching

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



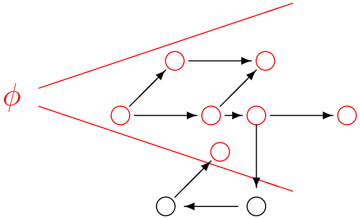
Integrated analysis: Model checking

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?



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).

Prefix

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

Structured Operational Semantics

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

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

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

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

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

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

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

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

Structured Operational Semantics: Constants

Constant

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

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).

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 $a, a \to a$ (as in CSP) instead of $a, \bar{a} \to \tau$ (as in CCS and the π-calculus).

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

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

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

The Importance of Being Exponential

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

The Importance of Being Exponential

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

The Importance of Being Exponential

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

The Importance of Being Exponential

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

The Importance of Being Exponential

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

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 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:

$$(\alpha, r).Stop \parallel (\beta, s).Stop = \\ (\alpha, r).(\beta, s).(Stop \parallel Stop) + (\beta, s).(\alpha, r).(Stop \parallel Stop)$$

only if the negative exponential distribution is used.

Outline

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

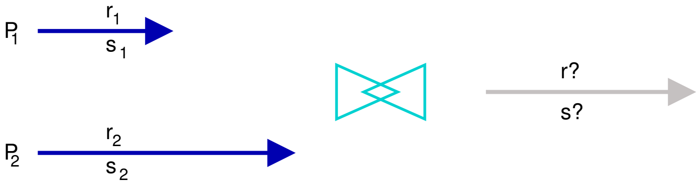
- Actions are partitioned into input and output pairs.
- Communication or synchronisation takes places between conjugate pairs.
- The resulting action has silent type τ.

CSP-style

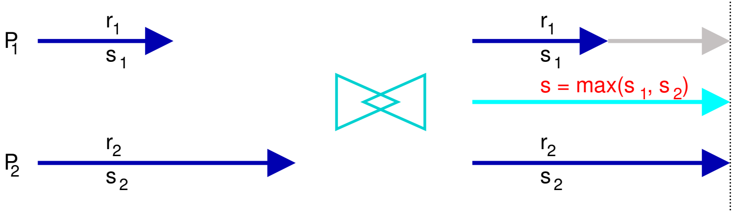
- No distinction between input and output actions.
- Communication or synchronisation takes place on the basis of shared names.
- The resulting action has the same name.

Most stochastic process algebras adopt CSP-style synchronisation.

Timed Synchronisation

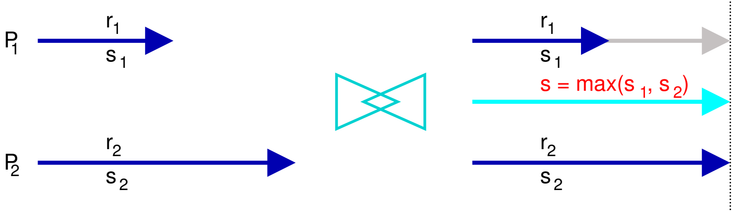


Timed Synchronisation



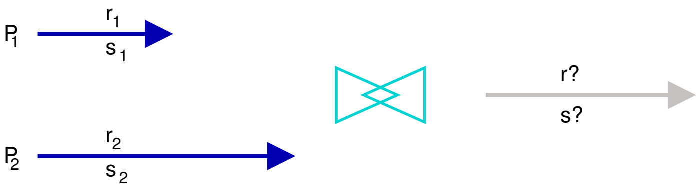
Barrier Synchronisation

Timed Synchronisation



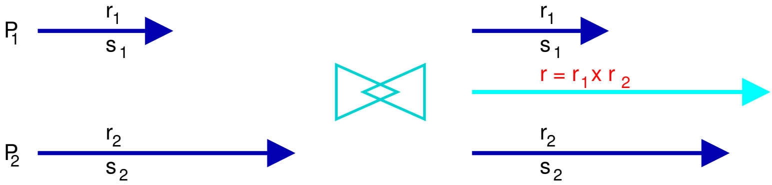
s is no longer exponentially distributed

Timed Synchronisation



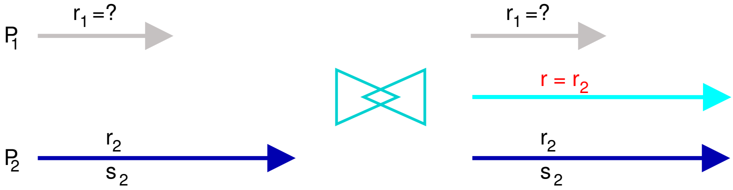
algebraic considerations limit choices

Timed Synchronisation



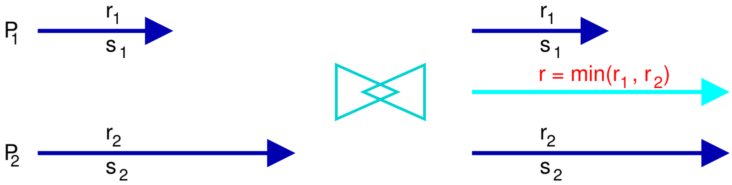
TIPP: new rate is product of individual rates

Timed Synchronisation



EMPA: one participant is passive

Timed Synchronisation



bounded capacity: new rate is the minimum of the rates

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.

Outline

Equivalence relations in Performance Modelling

Equivalence relations are used, often informally, in performance modelling to manipulate models into an alternative form which is somehow easier to solve:

Model simplification: use a model-model equivalence to substitute one model by another which is more attractive from a solution point of view, e.g. smaller state space, special class of model, etc.

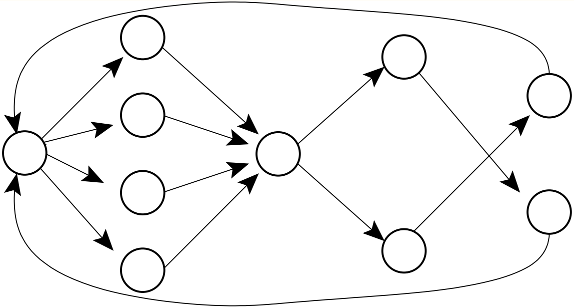
Equivalence relations in Performance Modelling

Equivalence relations are used, often informally, in performance modelling to manipulate models into an alternative form which is somehow easier to solve:

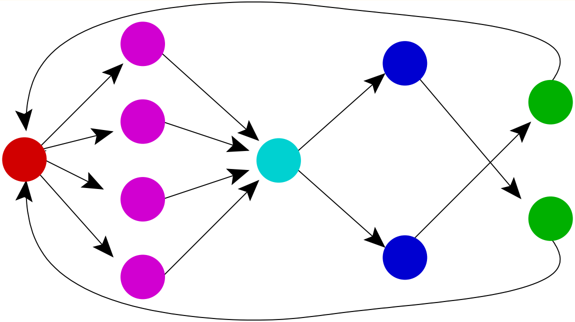
Model simplification: use a model-model equivalence to substitute one model by another which is more attractive from a solution point of view, e.g. smaller state space, special class of model, etc.

Model aggregation: use a state-state equivalence to establish a partition of the state space of a model, and replace each set of states by one macro-state, i.e. take a different stochastic representation of the same model.

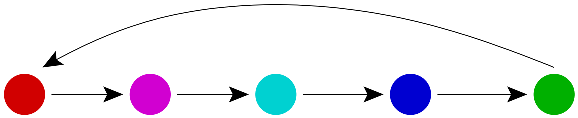
Reducing by lumping



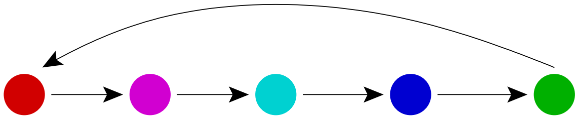
Reducing by lumping



Reducing by lumping



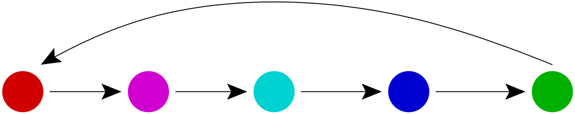
Reducing by lumping



As appealling as this is, it is not the case that it is always mathematically legitimate.

In particular, arbitarily lumping the states of a Markov chain, will typically give rise to a stochastic process which no longer satisfies the Markov condition.

Reducing by lumping



As appealling as this is, it is not the case that it is always mathematically legitimate.

In particular, arbitarily lumping the states of a Markov chain, will typically give rise to a stochastic process which no longer satisfies the Markov condition.

A lumpable partition is the only partition of a Markov process which preserves the Markov property.

Lumpability

- In the early 1960's Kemeny and Snell established the conditions under which it was possible to lump a Markov chain and still have a Markov chain afterwards.

Lumpability

- In the early 1960's Kemeny and Snell established the conditions under which it was possible to lump a Markov chain and still have a Markov chain afterwards.
- In particular these conditions were characterised by conditions on the rates which are straightforward to check.

Lumpability

- In the early 1960's Kemeny and Snell established the conditions under which it was possible to lump a Markov chain and still have a Markov chain afterwards.
- In particular these conditions were characterised by conditions on the rates which are straightforward to check.
- However checking the conditions did involve constructing the complete Markov chain first.

Lumpability

- In the early 1960's Kemeny and Snell established the conditions under which it was possible to lump a Markov chain and still have a Markov chain afterwards.
- In particular these conditions were characterised by conditions on the rates which are straightforward to check.
- However checking the conditions did involve constructing the complete Markov chain first.
- This is something of a catch-22 situation when the problem is that the state space of the Markov chain is too large to handle.

Lumpability

If the original state space is $\{X_1, X_2, \ldots, X_n\}$ then the aggregated state space is some $\{X_{[1]}, X_{[2]}, \ldots, X_{[N]}\}$ where $N < n$ and ideally $N << n$.

Lumpability

If the original state space is $\{X_1, X_2, \ldots, X_n\}$ then the aggregated state space is some $\{X_{[1]}, X_{[2]}, \ldots, X_{[N]}\}$ where $N < n$ and ideally $N << n$.

If the transition rates of the original process are $q(X_i, X_k)$ then the transition rates into any partition from a state is

$$q(X_i, X_{[j]}) = \sum_{k \in [j]} q(X_i, X_j)$$

Lumpability

If the original state space is $\{X_1, X_2, \ldots, X_n\}$ then the aggregated state space is some $\{X_{[1]}, X_{[2]}, \ldots, X_{[N]}\}$ where $N < n$ and ideally $N << n$.

If the transition rates of the original process are $q(X_i, X_k)$ then the transition rates into any partition from a state is

$$q(X_i, X_{[j]}) = \sum_{k \in [j]} q(X_i, X_j)$$

Transition rates between partitions are the weighted sum of the transition rates of each state in the first partition to the second partition, weighted by the conditional steady state probability of that state in the partition, $\overline{\pi}_j(\cdot)$

$$q(X_{[j]}, X_{[i]}) = \sum_{k \in [j]} \overline{\pi}_j(X_k) q(X_k, X_{[i]})$$

Ordinary, Exact and Strict Lumpability

- A Markov process is ordinarily lumpable with respect to a partition $\chi = \{X_{[i]}\}$ iff, for any $X_{[k]}, X_{[l]} \in \chi, X_i, X_j \in X_{[k]}$

$$q(X_i, X_{[l]}) = q(X_j, X_{[l]})$$

Ordinary, Exact and Strict Lumpability

- A Markov process is ordinarily lumpable with respect to a partition $\chi = \{X_{[i]}\}$ iff, for any $X_{[k]}, X_{[l]} \in \chi, X_i, X_j \in X_{[k]}$

$$q(X_i, X_{[l]}) = q(X_j, X_{[l]})$$

- χ is an exactly lumpable partition iff, for any $X_{[k]}, X_{[l]} \in \chi, X_i, X_j \in X_{[l]}$

$$q(X_{[k]}, X_i) = q(X_{[k]}, X_j)$$

Ordinary, Exact and Strict Lumpability

- A Markov process is ordinarily lumpable with respect to a partition $\chi = \{X_{[i]}\}$ iff, for any $X_{[k]}, X_{[l]} \in \chi, X_i, X_j \in X_{[k]}$

$$q(X_i, X_{[l]}) = q(X_j, X_{[l]})$$

- χ is an exactly lumpable partition iff, for any $X_{[k]}, X_{[l]} \in \chi, X_i, X_j \in X_{[l]}$

$$q(X_{[k]}, X_i) = q(X_{[k]}, X_j)$$

- χ is a strictly lumpable partition iff it is ordinarily lumpable and exactly lumpable.

Equivalence within process algebras

Many different notions of equivalence have been developed for process algebra, but in stochastic process algebras some form of bisimulation is generally defined.

Equivalence within process algebras

Many different notions of equivalence have been developed for process algebra, but in stochastic process algebras some form of bisimulation is generally defined.

The notion of bisimulation, introduced by Milner and Park, is based on observability: two systems are bisimilar if they match each other's moves.

Equivalence within process algebras

Many different notions of equivalence have been developed for process algebra, but in stochastic process algebras some form of bisimulation is generally defined.

The notion of bisimulation, introduced by Milner and Park, is based on observability: two systems are bisimilar if they match each other's moves.

More formally a bisimulation is generally regarded as a binary relation between transition systems which associates systems which are able to simulate each other.

Equivalence within process algebras

Many different notions of equivalence have been developed for process algebra, but in stochastic process algebras some form of bisimulation is generally defined.

The notion of bisimulation, introduced by Milner and Park, is based on observability: two systems are bisimilar if they match each other's moves.

More formally a bisimulation is generally regarded as a binary relation between transition systems which associates systems which are able to simulate each other.

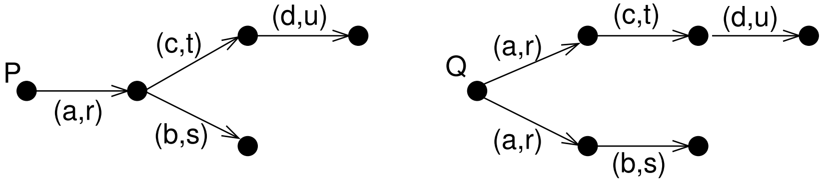
Different flavours of bisimulation may be defined depending on the power of the observer.

Strong Equivalence in PEPA

Strong equivalence in PEPA (somtimes termed Markovian bisimulation) is a bisimulation in the style of Larsen and Skou.

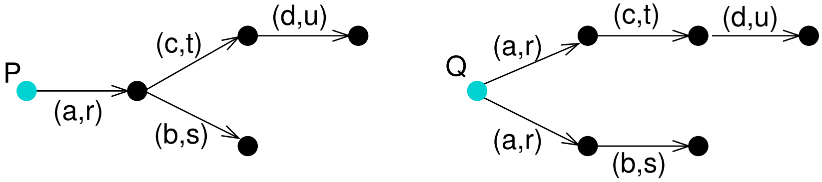
Strong Equivalence in PEPA

Strong equivalence in PEPA (somtimes termed Markovian bisimulation) is a bisimulation in the style of Larsen and Skou.



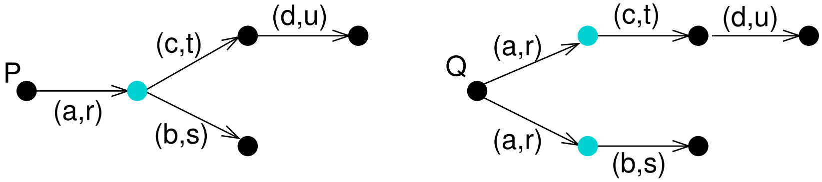
Strong Equivalence in PEPA

Strong equivalence in PEPA (somtimes termed Markovian bisimulation) is a bisimulation in the style of Larsen and Skou.



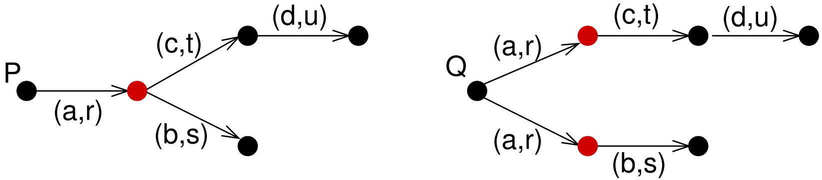
Strong Equivalence in PEPA

Strong equivalence in PEPA (somtimes termed Markovian bisimulation) is a bisimulation in the style of Larsen and Skou.



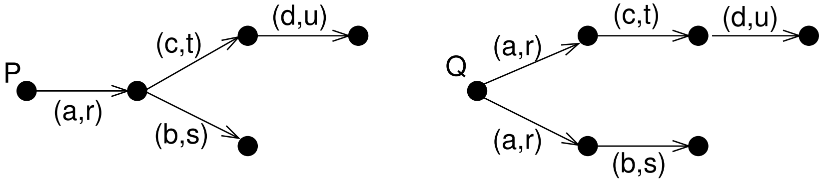
Strong Equivalence in PEPA

Strong equivalence in PEPA (somtimes termed Markovian bisimulation) is a bisimulation in the style of Larsen and Skou.



Strong Equivalence in PEPA

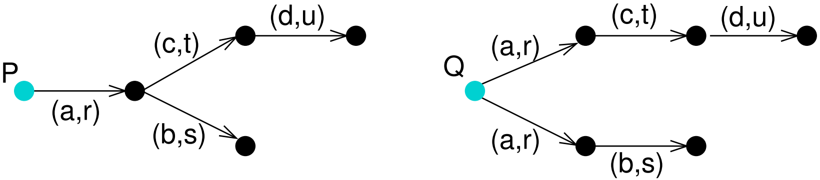
Strong equivalence in PEPA (somtimes termed Markovian bisimulation) is a bisimulation in the style of Larsen and Skou.



Observability is assumed to include the ability to record timing information over a number of runs.

Strong Equivalence in PEPA

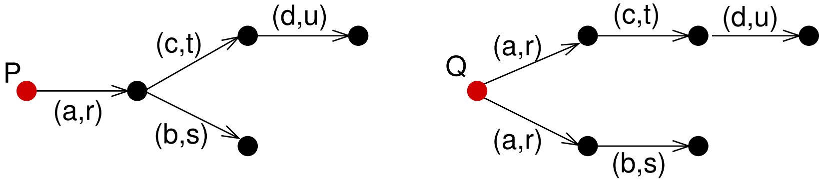
Strong equivalence in PEPA (somtimes termed Markovian bisimulation) is a bisimulation in the style of Larsen and Skou.



Observability is assumed to include the ability to record timing information over a number of runs.

Strong Equivalence in PEPA

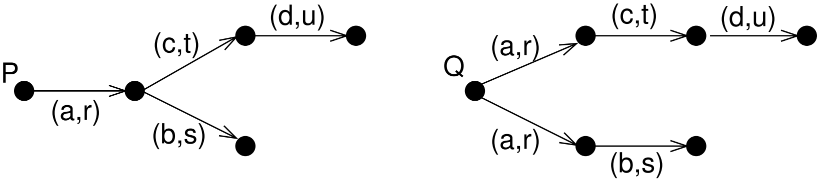
Strong equivalence in PEPA (somtimes termed Markovian bisimulation) is a bisimulation in the style of Larsen and Skou.



Observability is assumed to include the ability to record timing information over a number of runs.

Strong Equivalence in PEPA

Strong equivalence in PEPA (somtimes termed Markovian bisimulation) is a bisimulation in the style of Larsen and Skou.

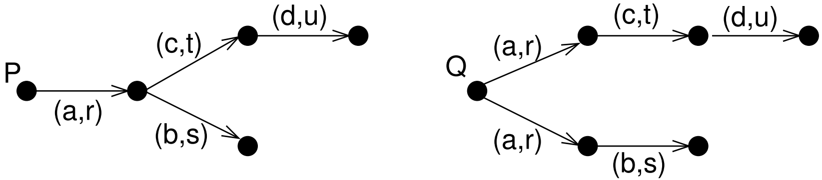


Observability is assumed to include the ability to record timing information over a number of runs.

Two processes are equivalent if they can undertake the same actions, at the same rate, and arrive at processes that are equivalent.

Strong Equivalence in PEPA

Strong equivalence in PEPA (somtimes termed Markovian bisimulation) is a bisimulation in the style of Larsen and Skou.



Observability is assumed to include the ability to record timing information over a number of runs.

Two processes are equivalent if they can undertake the same actions, at the same rate, and arrive at processes that are equivalent.

Expressed as rates to equivalence classes of processes

Strong Equivalence in PEPA

Definition

An equivalence relation $\mathcal{R} \subseteq \mathcal{C} \times \mathcal{C}$ is a *strong equivalence* if whenever $(P, Q) \in \mathcal{R}$ then for all $\alpha \in \mathcal{A}$ and for all $S \in \mathcal{C}/\mathcal{R}$

$$q[P, S, \alpha] = q[Q, S, \alpha].$$

where

$$q[C_i, S, \alpha] = \sum_{C_j \in S} q(C_i, C_j, \alpha)$$

Strong Equivalence and Lumpability

- Given this definition we can show that if we consider strong equivalence of states within a single model, it induces an ordinarily lumpable partition on the state space of the underlying Markov chain.

Strong Equivalence and Lumpability

- Given this definition we can show that if we consider strong equivalence of states within a single model, it induces an ordinarily lumpable partition on the state space of the underlying Markov chain.
- Moreover it can be shown that strong equivalence is a congruence.

Strong Equivalence and Lumpability

- Given this definition we can show that if we consider strong equivalence of states within a single model, it induces an ordinarily lumpable partition on the state space of the underlying Markov chain.
- Moreover it can be shown that strong equivalence is a congruence.
- This means that aggregation based on lumpability can be applied component by component, avoiding the previous problem of having to construct the complete state space in order to find the lumpable partitions.

Outline

Case study: active badges

We have used the PEPA modelling language to analyse the configuration of a location tracking system based on active badges.

Active badges transmit unique infra-red signals which are detected by networked sensors. These report locations back to a central database.

Case study: active badges

The badges are battery-powered and the tradeoff in the system is between the conservation of battery power and the accuracy of the information harvested from the sensors.

Case study: active badges

The badges are battery-powered and the tradeoff in the system is between the conservation of battery power and the accuracy of the information harvested from the sensors.

When transmissions from badges collide, the badges sleep for a randomly determined time before retrying.

Active badges: the PEPA model

The PEPA model of this system tracks the progress of one badge-wearer around three connected corridors (numbered 14, 15 and 16).

Active badges: the PEPA model

The PEPA model of this system tracks the progress of one badge-wearer around three connected corridors (numbered 14, 15 and 16).

The activities which are performed in the system include the badge registering with a sensor (at rate r), the person moving to another corridor (at rate m) and a sensor reporting back to the central database (at rate s).

Active badges: the PEPA model

Person

$$
\begin{aligned}
P_{14} &\stackrel{\mathit{def}}{=} (reg_{14}, r).P_{14} + (move_{15}, m).P_{15} \\
P_{15} &\stackrel{\mathit{def}}{=} (reg_{15}, r).P_{15} + (move_{14}, m).P_{14} + (move_{16}, m).P_{16} \\
P_{16} &\stackrel{\mathit{def}}{=} (reg_{16}, r).P_{16} + (move_{15}, m).P_{15}
\end{aligned}
$$

Active badges: the PEPA model

Person

$$
\begin{aligned}
P_{14} &\stackrel{def}{=} (reg_{14}, r).P_{14} + (move_{15}, m).P_{15} \\
P_{15} &\stackrel{def}{=} (reg_{15}, r).P_{15} + (move_{14}, m).P_{14} + (move_{16}, m).P_{16} \\
P_{16} &\stackrel{def}{=} (reg_{16}, r).P_{16} + (move_{15}, m).P_{15}
\end{aligned}
$$

Sensor

$$
\begin{aligned}
S_{14} &\stackrel{def}{=} (reg_{14}, \top).(rep_{14}, s).S_{14} \\
S_{15} &\stackrel{def}{=} (reg_{15}, \top).(rep_{15}, s).S_{15} \\
S_{16} &\stackrel{def}{=} (reg_{16}, \top).(rep_{16}, s).S_{16}
\end{aligned}
$$

Active badges: the PEPA model

Database

$$
\begin{aligned}
DB_{14} &\stackrel{\text{def}}{=} (rep_{14}, \top).DB_{14} + (rep_{15}, \top).DB_{15} + (rep_{16}, \top).DB_{16} \\
DB_{15} &\stackrel{\text{def}}{=} (rep_{14}, \top).DB_{14} + (rep_{15}, \top).DB_{15} + (rep_{16}, \top).DB_{16} \\
DB_{16} &\stackrel{\text{def}}{=} (rep_{14}, \top).DB_{14} + (rep_{15}, \top).DB_{15} + (rep_{16}, \top).DB_{16}
\end{aligned}
$$

Active badges: the PEPA model

Database

$$
\begin{aligned}
DB_{14} &\stackrel{\mathit{def}}{=} (rep_{14}, \top).DB_{14} + (rep_{15}, \top).DB_{15} + (rep_{16}, \top).DB_{16} \\
DB_{15} &\stackrel{\mathit{def}}{=} (rep_{14}, \top).DB_{14} + (rep_{15}, \top).DB_{15} + (rep_{16}, \top).DB_{16} \\
DB_{16} &\stackrel{\mathit{def}}{=} (rep_{14}, \top).DB_{14} + (rep_{15}, \top).DB_{15} + (rep_{16}, \top).DB_{16}
\end{aligned}
$$

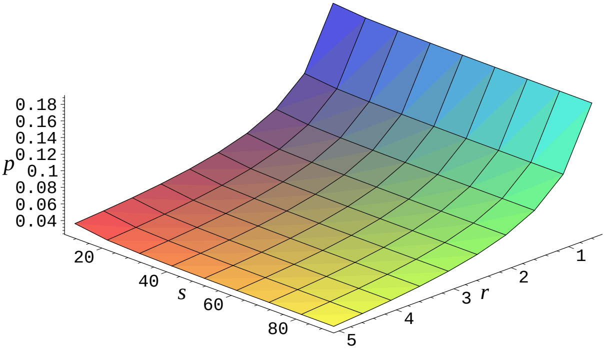
System

$$P_{14} \underset{L}{\bowtie} (S_{14} \parallel S_{15} \parallel S_{16}) \underset{M}{\bowtie} DB_{14}$$

where $L = \{\, reg_{14}, reg_{15}, reg_{16} \,\}$

$M = \{\, rep_{14}, rep_{15}, rep_{16} \,\}$

Probability that the database holds inaccurate information



Outline

Summary

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

Summary

- The theoretical development underpinning PEPA 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.

Summary

- The theoretical development underpinning PEPA 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, leading to efficient solution techniques.

The PEPA website

`http://www.dcs.ed.ac.uk/pepa`

From the website the PEPA Eclipse Plug-in and some other tools are available for download.

There is also information about people involved in the PEPA project, projects undertaken and a collection of published papers.

Thanks!

Thanks!

Acknowledgements: collaborators

Many co-authors and collaborators have contributed to the success of PEPA: Jeremy Bradley, Allan Clark, Graham Clark, Jie Ding, Adam Duguid, Stephen Gilmore, Leila Kloul, Marina Ribaudo, Mirco Tribastone, Nigel Thomas, and others.

Thanks!

Acknowledgements: collaborators
Many co-authors and collaborators have contributed to the success of PEPA: Jeremy Bradley, Allan Clark, Graham Clark, Jie Ding, Adam Duguid, Stephen Gilmore, Leila Kloul, Marina Ribaudo, Mirco Tribastone, Nigel Thomas, and others.

Acknowledgements: funding
Work on PEPA has been supported by EPRSC under a number of grants, most recently the Process Algebra for Collective Dynamics grant.

Thanks!

Acknowledgements: collaborators
Many co-authors and collaborators have contributed to the success of PEPA: Jeremy Bradley, Allan Clark, Graham Clark, Jie Ding, Adam Duguid, Stephen Gilmore, Leila Kloul, Marina Ribaudo, Mirco Tribastone, Nigel Thomas, and others.

Acknowledgements: funding
Work on PEPA has been supported by EPRSC under a number of grants, most recently the Process Algebra for Collective Dynamics grant.

More information:
http://www.dcs.ed.ac.uk/pepa