Getting Performance out of Process Algebra

Jane Hillston. LFCS, University of Edinburgh

16th August 2005

Outline

Introduction

Interplay: Process Algebra and Markov Process

Applications

Conclusions

Outline

Introduction

Interplay: Process Algebra and Markov Process

Applications

Conclusions

The PEPA project

- The PEPA project started in Edinburgh in 1991.

The PEPA project

- The PEPA project started in Edinburgh in 1991.
- It was motivated by problems encountered when carrying out performance analysis of large computer and communication systems, based on numerical analysis of Markov processes.

The PEPA project

- The PEPA project started in Edinburgh in 1991.
- It was motivated by problems encountered when carrying out performance analysis of large computer and communication systems, based on numerical analysis of Markov processes.
- Process algebras offered a compositional description technique supported by apparatus for formal reasoning.

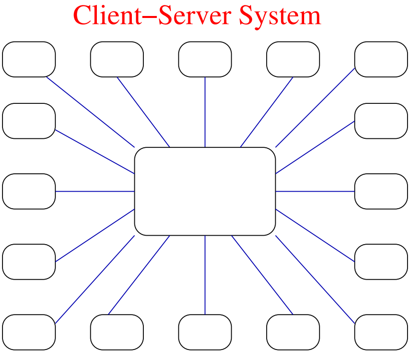
The PEPA project

- The PEPA project started in Edinburgh in 1991.
- It was motivated by problems encountered when carrying out performance analysis of large computer and communication systems, based on numerical analysis of Markov processes.
- Process algebras offered a compositional description technique supported by apparatus for formal reasoning.
- Performance Evaluation Process Algebra (PEPA) sought to address these problems by the introduction of a suitable process algebra.

The PEPA project

- The PEPA project started in Edinburgh in 1991.
- It was motivated by problems encountered when carrying out performance analysis of large computer and communication systems, based on numerical analysis of Markov processes.
- Process algebras offered a compositional description technique supported by apparatus for formal reasoning.
- Performance Evaluation Process Algebra (PEPA) sought to address these problems by the introduction of a suitable process algebra.
- We have sought to investigate and exploit the interplay between the process algebra and the continuous time Markov chain (CTMC).

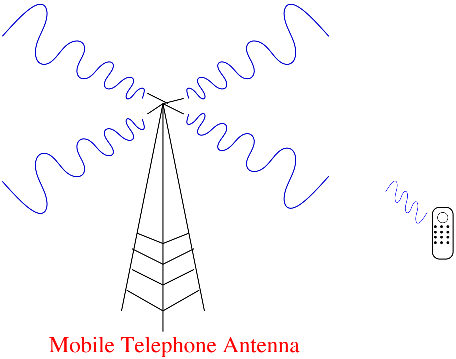
Performance Modelling: Motivation



Scalability issues

- How many clients can the existing server support and maintain reasonable response times?

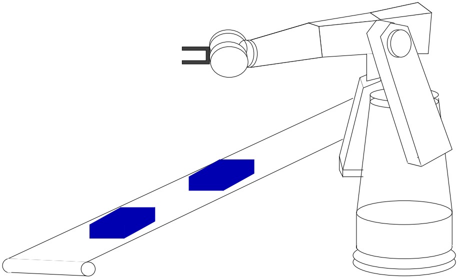
Performance Modelling: Motivation



Frequency Division Multiplexing

- How many frequencies do you need to keep blocking probabilities low?

Performance Modelling: Motivation

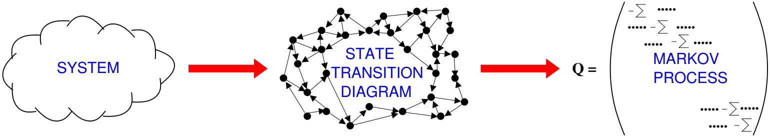


Flexible Manufacturing System

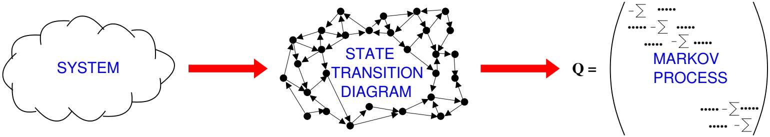
System Tuning

- What speed of conveyor belt will minimize robot idle time and maximize throughput whilst avoiding lost widgets?

Performance Modelling using CTMC



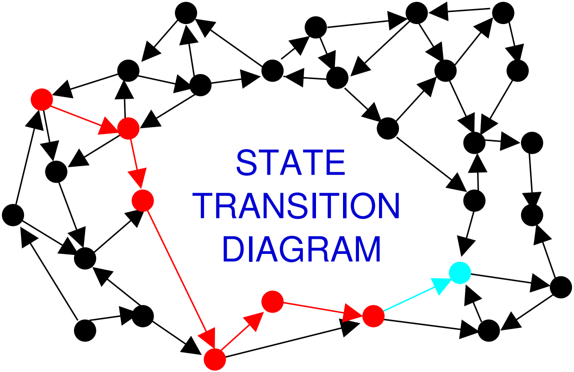
Performance Modelling using CTMC



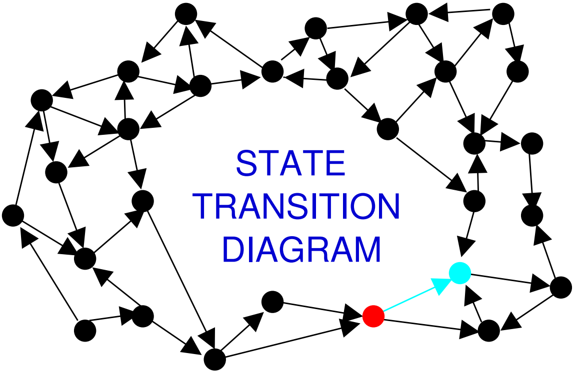
A stochastic process $X(t)$ is a Markov process iff for all $t_0 < t_1 < \ldots < t_n < t_{n+1}$, the joint probability distribution of $(X(t_0), X(t_1), \ldots, X(t_n), X(t_{n+1}))$ is such that

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

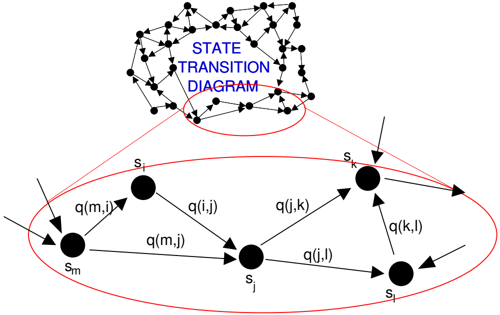
Performance Modelling using CTMC



Performance Modelling using CTMC

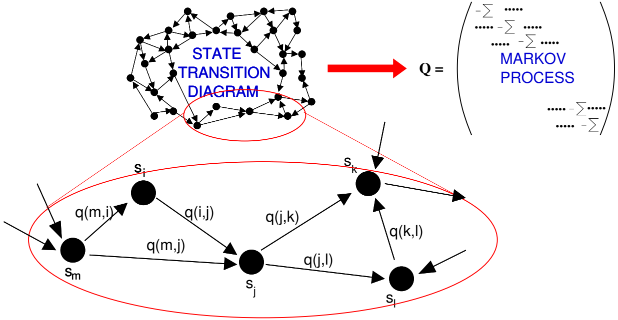


Performance Modelling using CTMC



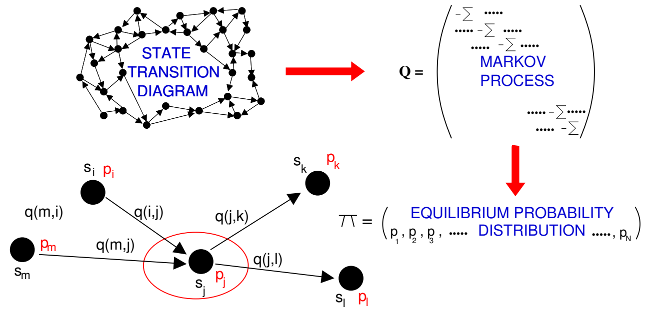
A negative exponentially distributed duration is associated with each transition.

Performance Modelling using CTMC



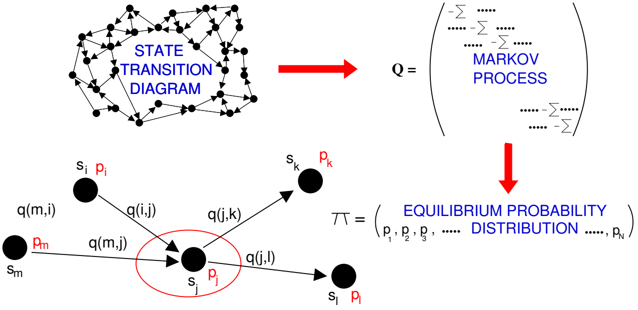
these parameters form the entries of the infinitesimal generator matrix Q

Performance Modelling using CTMC



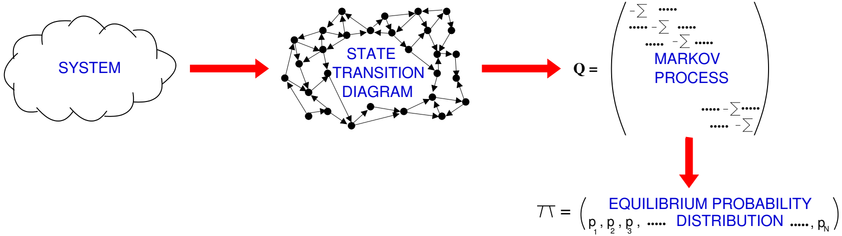
In steady state the probability flux out of a state is balanced by the flux in.

Performance Modelling using CTMC

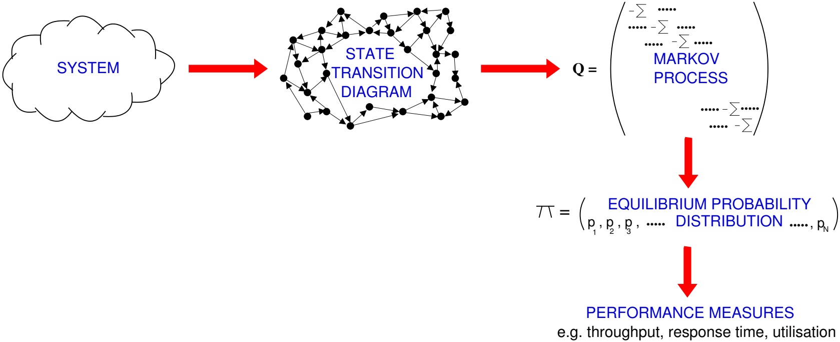


"Global balance equations" captured by $\pi Q = 0$ solved by linear algebra

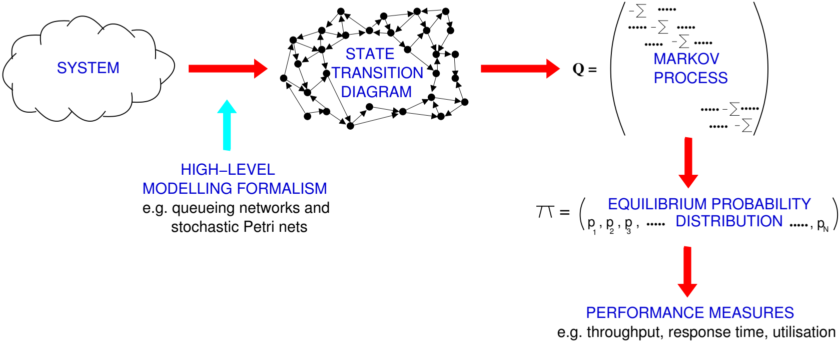
Performance Modelling using CTMC



Performance Modelling using CTMC



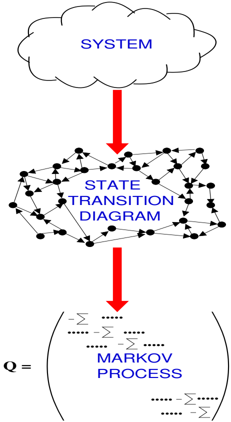
Performance Modelling using CTMC



Performance Modelling using CTMC

Model Construction

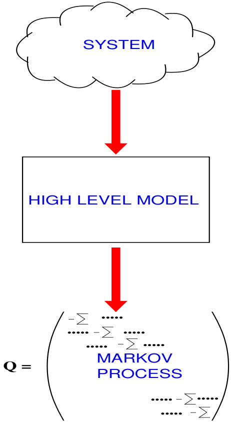
- describing the system using a high level modelling formalism
- generating the underlying CTMC



Performance Modelling using CTMC

Model Construction

- describing the system using a high level modelling formalism
- generating the underlying CTMC



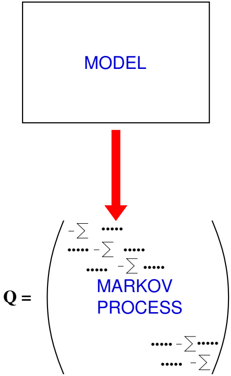
Performance Modelling using CTMC

Model Construction

- describing the system using a high level modelling formalism
- generating the underlying CTMC

Model Manipulation

- model simplification
- model aggregation



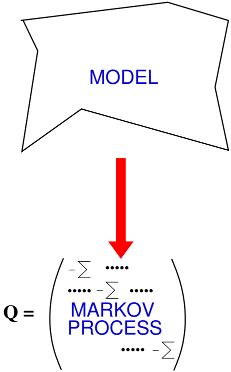
Performance Modelling using CTMC

Model Construction

- describing the system using a high level modelling formalism
- generating the underlying CTMC

Model Manipulation

- model simplification
- model aggregation



Performance Modelling using CTMC

Model Construction

- describing the system using a high level modelling formalism
- generating the underlying CTMC

Model Manipulation

- model simplification
- model aggregation

Model Solution

- solving the CTMC to find steady state probability distribution
- deriving performance measures

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.

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

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.

$$\alpha.P$$

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:

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

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.

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

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:

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

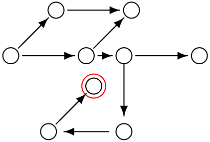
Qualitative Analysis

- 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.

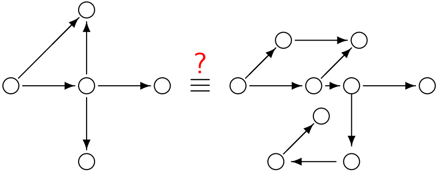
Will the system arrive in a particular state?



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.

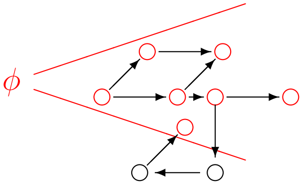
Does system behaviour match its specification?



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?



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.

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

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

- 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.

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

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

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.

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

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.

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

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.

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

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

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

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

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 (race policy)

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 (race policy)
CONSTANT:	$A \stackrel{\text{def}}{=} S$	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 (race policy)
CONSTANT:	$A \stackrel{def}{=} S$	assigning names
COOPERATION:	$P \underset{L}{\bowtie} P$	$\alpha \notin L$ concurrent activity (*individual actions*) $\alpha \in L$ cooperative activity (*shared actions*)

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 (race policy)
CONSTANT:	$A \stackrel{\text{def}}{=} S$	assigning names
COOPERATION:	$P \underset{L}{\bowtie} P$	$\alpha \notin L$ concurrent activity (*individual actions*) $\alpha \in L$ cooperative activity (*shared actions*)

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 (race policy)
CONSTANT:	$A \stackrel{def}{=} S$	assigning names
COOPERATION:	$P \underset{L}{\bowtie} P$	$\alpha \notin L$ concurrent activity (*individual actions*) $\alpha \in L$ cooperative activity (*shared actions*)
HIDING:	P/L	abstraction $\alpha \in L \Rightarrow \alpha \to \tau$

Example revisited

The behaviour of the server is the same but now quantitative information is recorded for each operation:

$$Server \stackrel{def}{=} (get, \top).(download, \mu).(rel, \top).Server$$

Example revisited

The behaviour of the server is the same but now quantitative information is recorded for each operation:

$$Server \stackrel{\mathit{def}}{=} (get, \top).(download, \mu).(rel, \top).Server$$

In addition to duration we also incorporate information about the relative frequencies of the different actions which take place after a display request:

$$\begin{aligned} Browser \stackrel{\mathit{def}}{=} \ & (display, p_1\lambda).(cache, m).Browser \ + \\ & (display, p_2\lambda).(get, g).(download, \top).(rel, r).Browser \end{aligned}$$

Example revisited

The behaviour of the server is the same but now quantitative information is recorded for each operation:

$$Server \stackrel{\mathit{def}}{=} (get, \top).(download, \mu).(rel, \top).Server$$

In addition to duration we also incorporate information about the relative frequencies of the different actions which take place after a display request:

$$\begin{aligned} Browser \stackrel{\mathit{def}}{=} & \ (display, p_1\lambda).(cache, m).Browser \ + \\ & \ (display, p_2\lambda).(get, g).(download, \top).(rel, r).Browser \end{aligned}$$

The configuration is recorded as before; using the PEPA cooperation the actions which must be shared are explicitly named:

$$WEB \stackrel{\mathit{def}}{=} \left(Browser \parallel Browser\right) \underset{L}{\bowtie} Server \qquad L = \{get, download, rel\}$$

Example revisited

The behaviour of the server is the same but now quantitative information is recorded for each operation:

$$Server \stackrel{def}{=} (get, \top).(download, \mu).(rel, \top).Server$$

In addition to duration we also incorporate information about the relative frequencies of the different actions which take place after a display request:

$$\begin{aligned} Browser \stackrel{def}{=} \; & (display, p_1\lambda).(cache, m).Browser \; + \\ & (display, p_2\lambda).(get, g).(download, \top).(rel, r).Browser \end{aligned}$$

The configuration is recorded as before; using the PEPA cooperation the actions which must be shared are explicitly named:

$$WEB \stackrel{def}{=} \left(Browser \parallel Browser\right) \underset{L}{\bowtie} Server \qquad L = \{get, download, rel\}$$

Integrated analysis

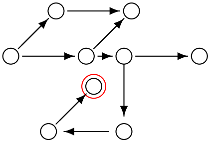
- Qualitative verification can now be complemented by quantitative verification:

Integrated analysis

- Qualitative verification can now be complemented by quantitative verification:

Reachability analysis

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

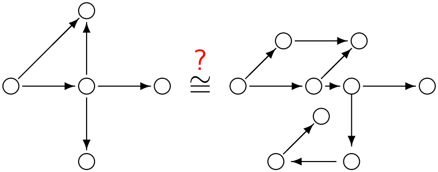


Integrated analysis

- Qualitative verification can now be complemented by quantitative verification:

Specification matching

With what probability does system behaviour match its specification?

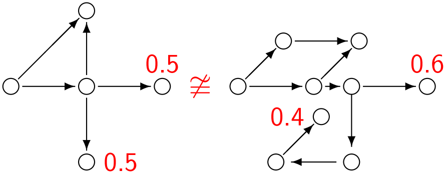


Integrated analysis

- Qualitative verification can now be complemented by quantitative verification:

Specification matching

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

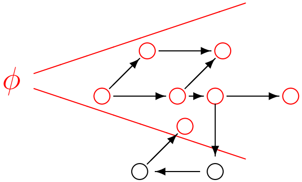


Integrated analysis

- Qualitative verification can now be complemented by quantitative verification:

Model checking

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

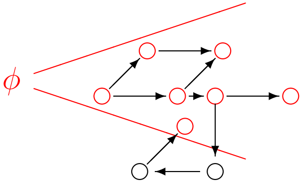


Integrated analysis

- Qualitative verification can now be complemented by quantitative verification:

Model checking

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



Outline

Introduction

Interplay: Process Algebra and Markov Process

Applications

Conclusions

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.

SPA Languages

SPA

SPA Languages

integrated time

SPA

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, 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
  - 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
      - IGSMP, Modest

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
      - IGSMP, Modest

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 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

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

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

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

The Importance of Being Exponential

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

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

The Importance of Being Exponential

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

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 assumed.

Parallel Composition

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

Parallel Composition

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

Parallel Composition

- Parellel 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

- Parellel 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.

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.

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.

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.

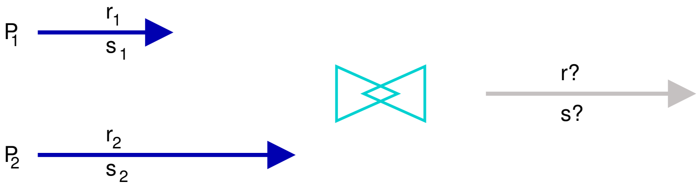
Most stochastic process algebras adopt CSP-style synchronisation.

Timed Synchronisation

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

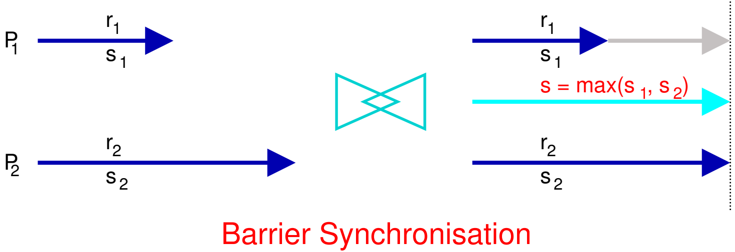
Timed Synchronisation

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

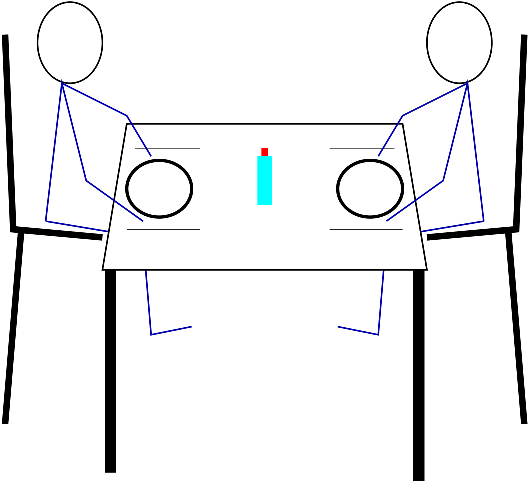


Timed Synchronisation

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

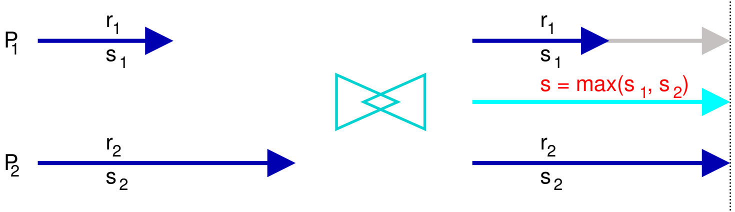


Timed Synchronisation



Timed Synchronisation

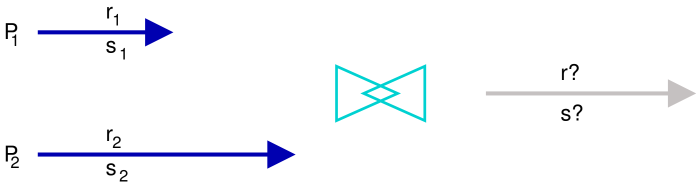
- The issue of what it means for two timed activities to synchronise is a vexed one....



s is no longer exponentially distributed

Timed Synchronisation

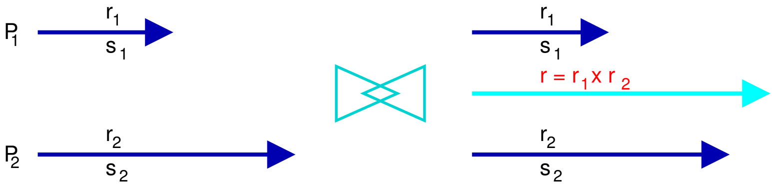
- The issue of what it means for two timed activities to synchronise is a vexed one....



algebraic considerations limit choices

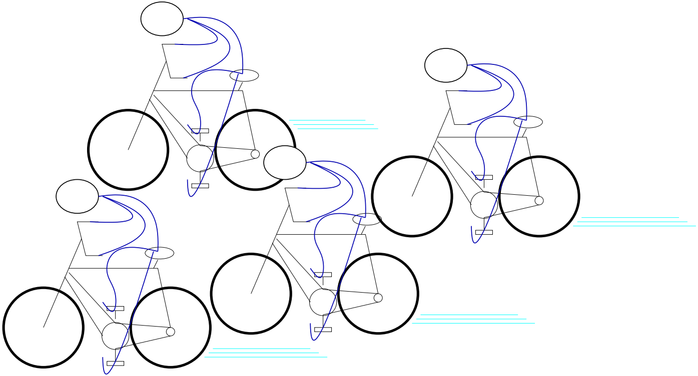
Timed Synchronisation

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



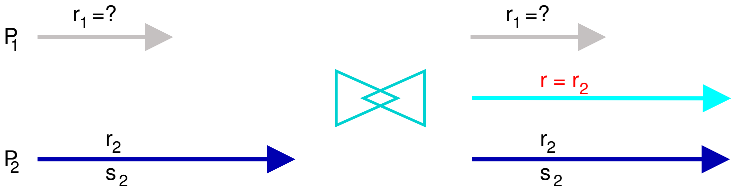
TIPP: new rate is product of individual rates

Timed Synchronisation



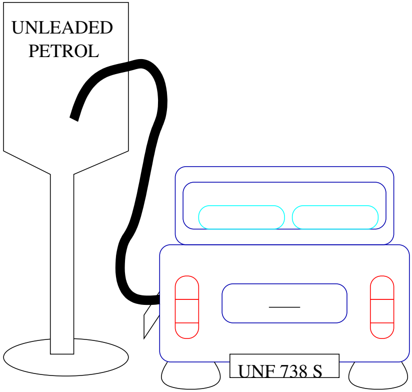
Timed Synchronisation

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



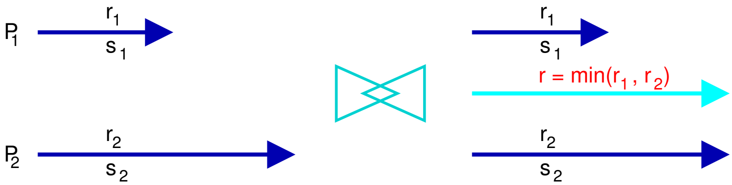
EMPA: one participant is passive

Timed Synchronisation



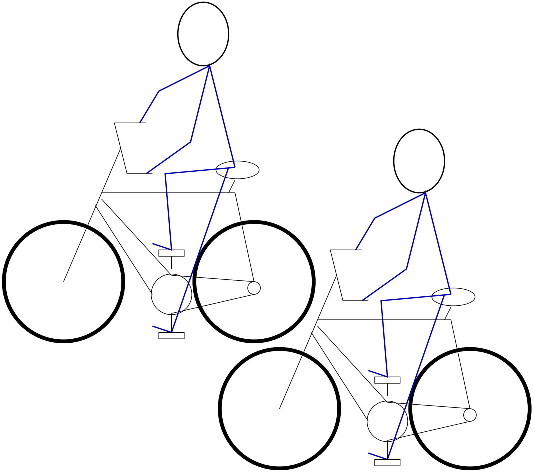
Timed Synchronisation

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



bounded capacity: new rate is the minimum of the rates

Timed Synchronisation



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

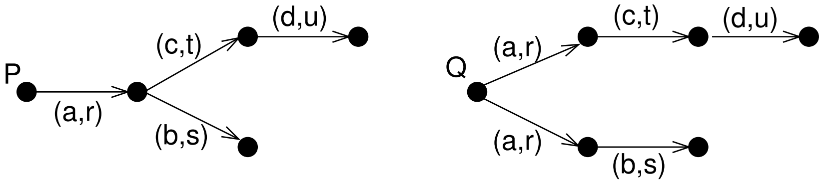
- In PEPA each component has a bounded capacity to carry out activities of any particular type, determined by the apparent rate for that type.
- Synchronisation, or cooperation cannot make a component exceed its bounded capacity.

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.

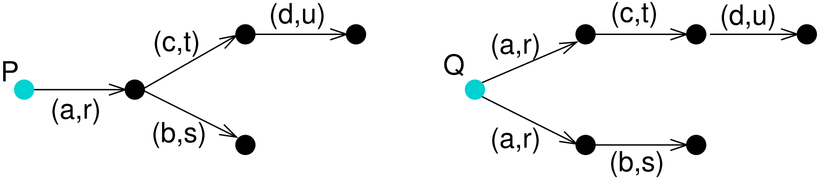
Equivalence Relations

In process algebra equivalence relations are defined based on the notion of observability:



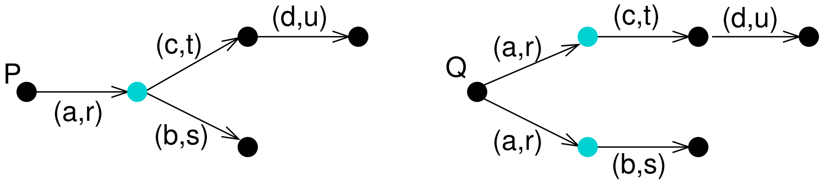
Equivalence Relations

In process algebra equivalence relations are defined based on the notion of observability:



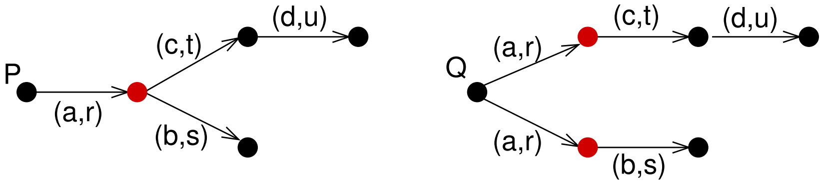
Equivalence Relations

In process algebra equivalence relations are defined based on the notion of observability:



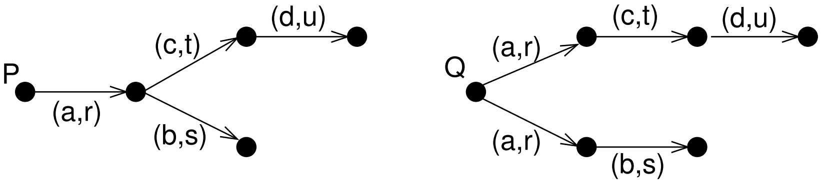
Equivalence Relations

In process algebra equivalence relations are defined based on the notion of observability:



Equivalence Relations

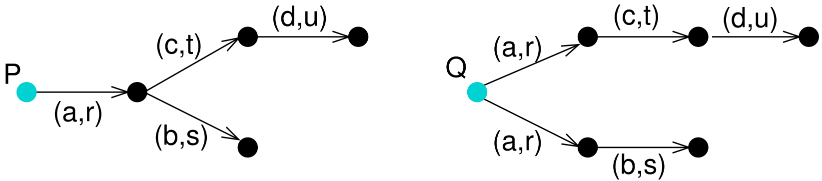
In process algebra equivalence relations are defined based on the notion of observability:



In PEPA observation is assumed to include the ability to record timing information over a number of runs.

Equivalence Relations

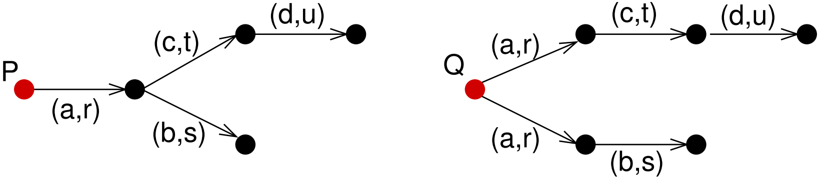
In process algebra equivalence relations are defined based on the notion of observability:



In PEPA observation is assumed to include the ability to record timing information over a number of runs.

Equivalence Relations

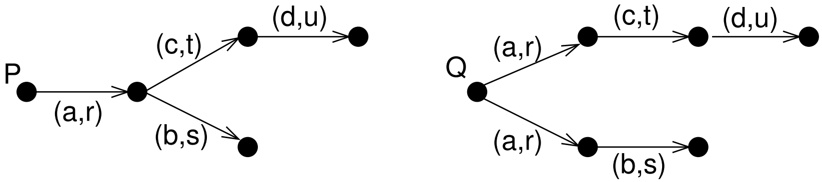
In process algebra equivalence relations are defined based on the notion of observability:



In PEPA observation is assumed to include the ability to record timing information over a number of runs.

Equivalence Relations

In process algebra equivalence relations are defined based on the notion of observability:



In PEPA observation is assumed to include the ability to record timing information over a number of runs.

The resulting equivalence relation is a bisimulation in the style of Larsen and Skou, and coincides with the Markov process notion of lumpability.

Aggregation and lumpability

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.

Aggregation and lumpability

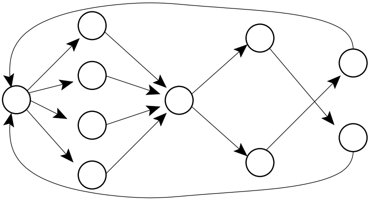
Model aggregation: use a state-state equivalence to establish a partition of the state space of a model, and replace each set of states by one macro-state, i.e. take a different stochastic representation of the same model.

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

Aggregation and lumpability

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.

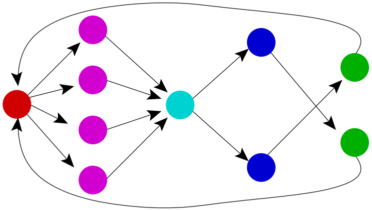
A lumpable partition is the only partition of a Markov process which preserves the Markov property.



Aggregation and lumpability

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.

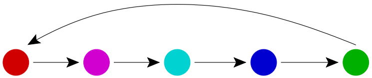
A lumpable partition is the only partition of a Markov process which preserves the Markov property.



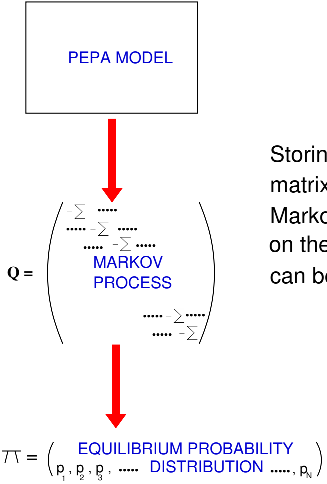
Aggregation and lumpability

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.

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



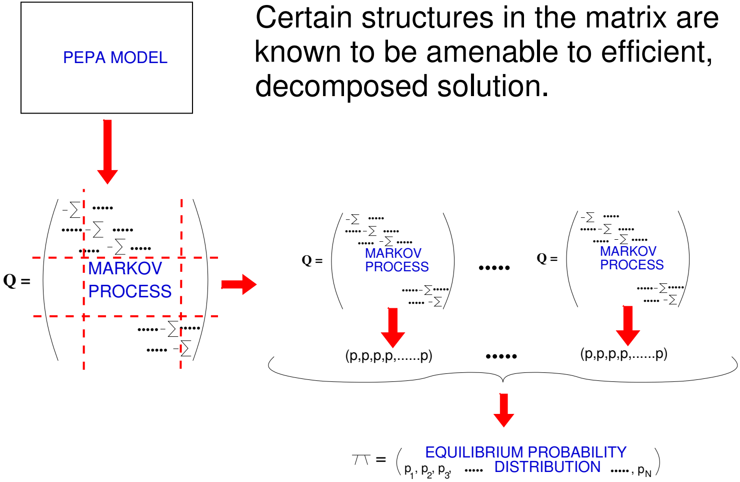
Characterising efficient solution



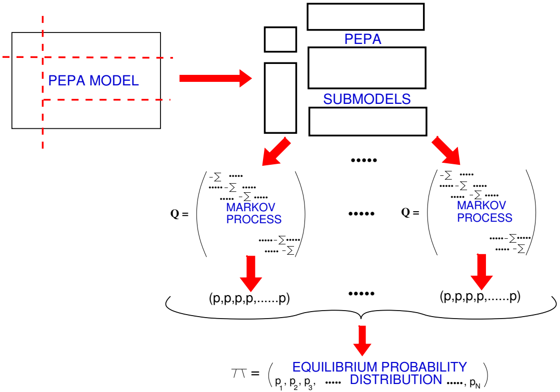
Storing and manipulating the matrix which represents the Markov process places limitations on the size of model which can be analysed.

Characterising efficient solution

Certain structures in the matrix are known to be amenable to efficient, decomposed solution.



Characterising efficient solution



Finding the corresponding structures in the process algebra means that these techniques can be applied automatically, before the monolithic matrix is formed.

Outline

Introduction

Interplay: Process Algebra and Markov Process

Applications

Conclusions

Applications

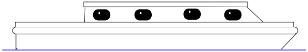
- Developing models of real applications has always been an integral part of the PEPA project.
- This allows us to demonstrate to ourselves and others that the theory we have developed is useful.
- It is also a valuable source of inspiration for new theory and future directions.

PEPA Case Studies (1)

- Multiprocessor access-contention protocols (Gilmore, Hillston and Ribaudo, Edinburgh and Turin)
- Protocols for fault-tolerant systems (Clark, Gilmore, Hillston and Ribaudo, Edinburgh and Turin)
- Multimedia traffic characteristics (Bowman et al, Kent)
- Database systems (The STEADY group, Heriot-Watt University)
- Software Architectures (Pooley, Bradley and Thomas, Heriot-Watt and Durham)
- Switch behaviour in active networks (Hillston, Kloul and Mokhtari, Edinburgh and Versailles)

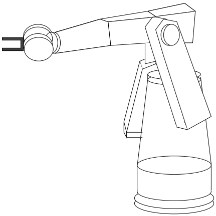
PEPA Case Studies (2)

- Locks and movable bridges in inland shipping in Belgium (Knapen, Hasselt)



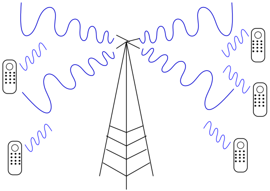
PEPA Case Studies (2)

- Locks and movable bridges in inland shipping in Belgium (Knapen, Hasselt)
- Robotic workcells (Holton, Gilmore and Hillston, Bradford and Edinburgh)



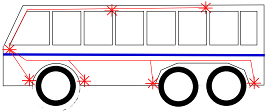
PEPA Case Studies (2)

- Locks and movable bridges in inland shipping in Belgium (Knapen, Hasselt)
- Robotic workcells (Holton, Gilmore and Hillston, Bradford and Edinburgh)
- Cellular telephone networks (Kloul, Fourneau and Valois, Versailles)



PEPA Case Studies (2)

- Locks and movable bridges in inland shipping in Belgium (Knapen, Hasselt)
- Robotic workcells (Holton, Gilmore and Hillston, Bradford and Edinburgh)
- Cellular telephone networks (Kloul, Fourneau and Valois, Versailles)
- Automotive diagnostic expert systems (Console, Picardi and Ribaudo, Turin)



New application domains: biochemical signalling pathways

- Biological advances mean that much more is now known about the components of cells and the interactions between them.

New application domains: biochemical signalling pathways

- Biological advances mean that much more is now known about the components of cells and the interactions between them.
- Systems biology aims to develop a better understanding of the processes involved.

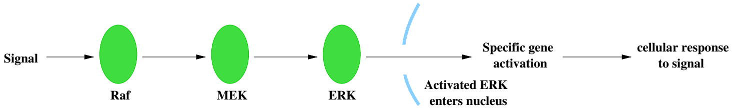
New application domains: biochemical signalling pathways

- Biological advances mean that much more is now known about the components of cells and the interactions between them.
- Systems biology aims to develop a better understanding of the processes involved.
- Stochastic process algebras have found a new role in developing models for systems biology, allowing biologists to test hypotheses and prioritise experiments.

Extracellular signalling

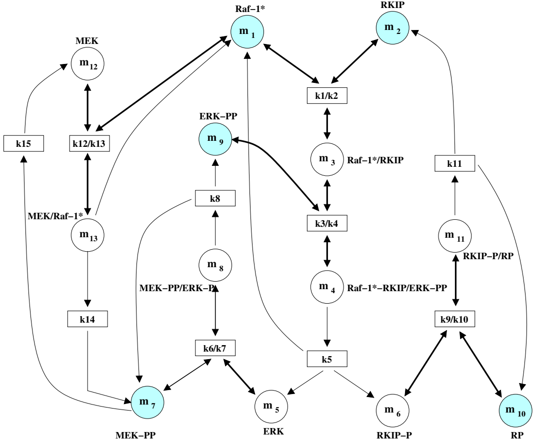
Extracellular signalling — communication between cells.

- signalling molecules released by one cell migrate to another;
- these molecules enter the cell and instigate a pathway, or series of reactions, which carries the information from the membrane to the nucleus;
- the Ras/Raf-1/MEK/ERK pathway conveys differentiation signals to the nucleus of a cell.



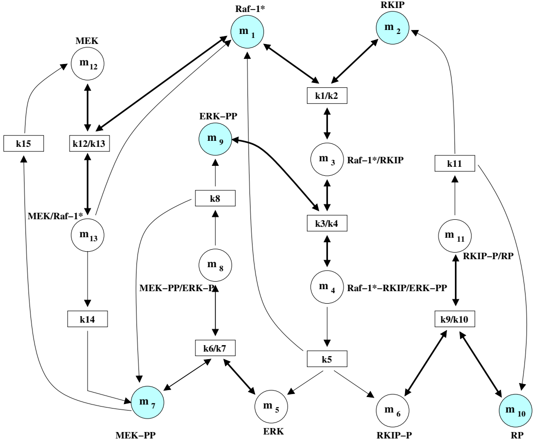
Special relevance to cancer research because when pathways operate abnormally cells divide uncontrollably.

The Ras/Raf-1/MEK/ERK pathway



We have constructed two, complementary, PEPA models of the pathway.

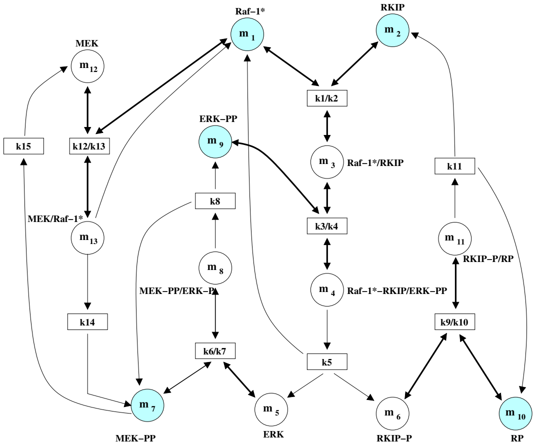
The Ras/Raf-1/MEK/ERK pathway



We have constructed two, complementary, PEPA models of the pathway.

- Reagents-centric

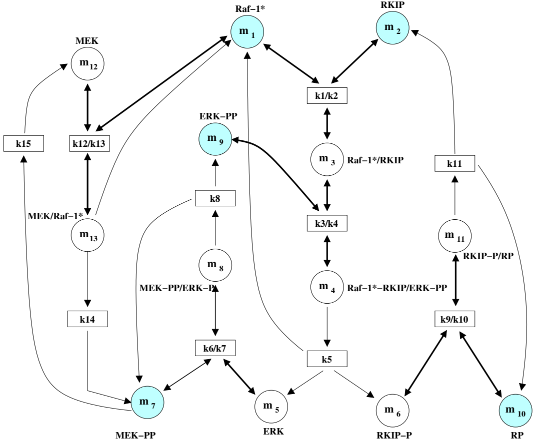
The Ras/Raf-1/MEK/ERK pathway



We have constructed two, complementary, PEPA models of the pathway.

- Reagents-centric
- Pathway-centric

The Ras/Raf-1/MEK/ERK pathway

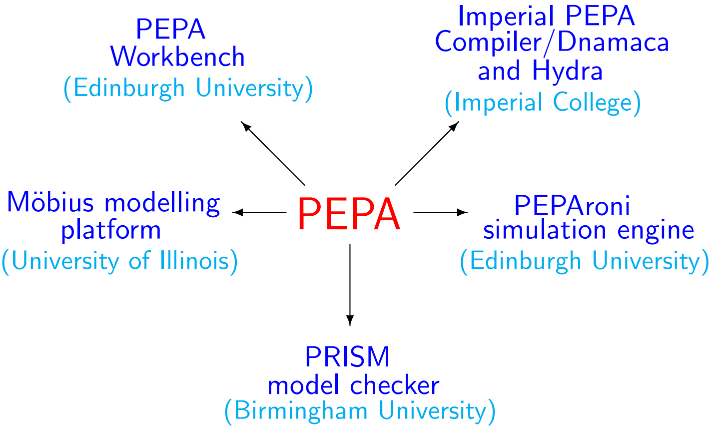


We have constructed two, complementary, PEPA models of the pathway.

- Reagents-centric
- Pathway-centric

and shown them to be equivalent.

Tool Support



Outline

Introduction

Interplay: Process Algebra and Markov Process

Applications

Conclusions

Some concluding remarks...

The incorporation of stochastic quantitative information into process algebras has been extremely fruitful:

Some concluding remarks...

The incorporation of stochastic quantitative information into process algebras has been extremely fruitful:

- For performance modelling:
  - the rigour of the formal description technique has had benefits for both practice and theory;
  - enhanced analysis capabilities.

Some concluding remarks...

The incorporation of stochastic quantitative information into process algebras has been extremely fruitful:

- For performance modelling:
  - the rigour of the formal description technique has had benefits for both practice and theory;
  - enhanced analysis capabilities.
- For theoretical computer science:
  - interesting new class of languages has been introduced, stimulating much new research.

Acknowledgements

The PEPA project has been funded by SERC, EPSRC and the CEC.

Thank you