Fluid Flow Approximation of PEPA Models

Jane Hillston.
LFCS, University of Edinburgh

20th September 2005
Outline

Introduction
  Background and Motivation
  PEPA

Continuous State Space Models
  Deriving Differential Equations
  Analysis based on Continuous-time Markov Chains
  Analysis based on Ordinary Differential Equations

Case Study in Web Services
  The model
  Analysis

Conclusions
Outline

Introduction
  Background and Motivation
  PEPA

Continuous State Space Models
  Deriving Differential Equations
  Analysis based on Continuous-time Markov Chains
  Analysis based on Ordinary Differential Equations

Case Study in Web Services
  The model
  Analysis

Conclusions
Performance evaluation: new mathematical structures

When constructing a model performance modellers usually make a choice between:

- Closed form analytical models (*queueing networks*);
Performance evaluation: new mathematical structures

When constructing a model performance modellers usually make a choice between:

- Closed form analytical models (queueing networks);
- Simulations; or
Performance evaluation: new mathematical structures

When constructing a model performance modellers usually make a choice between:

- Closed form analytical models (*queueing networks*);
- Simulations; or
- Numerical solution of continuous time Markov chains (CTMC) (*Stochastic Petri nets, Stochastic process algebras etc.*)
Performance evaluation: new mathematical structures

When constructing a model performance modellers usually make a choice between:

- Closed form analytical models (queueing networks);
- Simulations; or
- Numerical solution of continuous time Markov chains (CTMC) (Stochastic Petri nets, Stochastic process algebras etc.)

In each case the model has a discrete state space and continuous time.
Performance evaluation: new mathematical structures

When constructing a model performance modellers usually make a choice between:

- Closed form analytical models (*queueing networks*);
- Simulations; or
- Numerical solution of continuous time Markov chains (CTMC) (*Stochastic Petri nets, Stochastic process algebras etc.*)

In each case the model has a discrete state space and continuous time.

The major limitations of the CTMC approach are the state space explosion problem and the reliance on exponential distributions.
Performance evaluation: new mathematical structures

State space explosion becomes an ever more challenging problem as the scale and complexity of modern systems increase.
Performance evaluation: new mathematical structures

State space explosion becomes an ever more challenging problem as the scale and complexity of modern systems increase.

Is there an alternative?
Performance evaluation: new mathematical structures

State space explosion becomes an ever more challenging problem as the scale and complexity of modern systems increase.

Is there an alternative?

Use continuous state variables to approximate the discrete state space.
Performance evaluation: new mathematical structures

State space explosion becomes an ever more challenging problem as the scale and complexity of modern systems increase.

Is there an alternative?

Use continuous state variables to approximate the discrete state space.

Use ordinary differential equations to represent the evolution of those variables over time.
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.

\[(\alpha, r).P\]

- \(\alpha\): action type or name
- \(r\): activity rate (parameter of an exponential distribution)
- \(P\): component/derivative

Jane Hillston. LFCS, University of Edinburgh.

Fluid Flow Approximation of PEPA Models
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

Jane Hillston. LFCS, University of Edinburgh.

Fluid Flow Approximation of PEPA Models
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

Jane Hillston. LFCS, University of Edinburgh.

Fluid Flow Approximation of PEPA Models
Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.

\[
(\alpha, r).P
\]

The language may be 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
- component/derivative
- activity rate (parameter of an exponential distribution)

The language may be 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 may be used to generate a CTMC for performance modelling.

PEPA MODEL \rightarrow 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 may be used to generate a CTMC for performance modelling.

PEPA MODEL \rightarrow \text{SOS rules} \rightarrow \text{LABELLED TRANSITION SYSTEM}
Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.

\[(\alpha, r)P\]

The language may be used to generate a CTMC for performance modelling.
Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.

\[(\alpha, r).P\]

The language may be used to generate a CTMC for performance modelling.
Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.

\[(\alpha, r).P\]

The language may be used to generate a system of ordinary differential equations (ODEs) 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
- component/derivative

The language may be used to generate a system of ordinary differential equations (ODEs) for performance modelling.
Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.

\[ (\alpha, r).P \]

- The language may be used to generate a system of ordinary differential equations (ODEs) for performance modelling.
Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.

\[(\alpha, r).P\]

The language may be used to generate a system of ordinary differential equations (ODEs) for performance modelling.

PEPA model -> syntactic analysis -> activity matrix

Jane Hillston. LFCS, University of Edinburgh.

Fluid Flow Approximation of PEPA Models
Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.

\[(\alpha, r).P\]

The language may be used to generate a system of ordinary differential equations (ODEs) for performance modelling.

PEPA MODEL → syntactic analysis → ACTIVITY MATRIX → continuous interpretation

Jane Hillston. LFCS, University of Edinburgh.

Fluid Flow Approximation of PEPA Models
Performance Evaluation Process Algebra

Models are constructed from components which engage in activities.

\[(\alpha, r).P\]

- action type or name
- activity rate
- component/derivative

The language may be used to generate a system of ordinary differential equations (ODEs) for performance modelling.

PEPA MODEL \(\xrightarrow{\text{syntactic analysis}}\) ACTIVITY MATRIX \(\xrightarrow{\text{continuous interpretation}}\) ODEs
PEPA

\[ S ::= (\alpha, r).S \mid S + S \mid A \]
\[ P ::= S \mid P \Join P \mid P/L \]
PEPA

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

\[ P ::= S \mid P \otimes P \mid P/L \]

**PREFIX:**

\[(\alpha, r).S \] designated first action
PEPA

\[ S ::= (\alpha, r).S | S + S | A \]
\[ P ::= S | P \oplus P | P/L \]

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

**CHOICE:** \(S + S\) competing components (race policy)
PEPA

\[
S ::= (\alpha, r).S \mid S + S \mid A \\
P ::= S \mid P \Downarrow \ P \mid P/L
\]

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

**CHOICE:** \(S + S\) competing components (race policy)

**CONSTANT:** \(A \overset{\text{def}}{=} S\) assigning names
PEPA

\[ S ::= (\alpha, r).S \mid S + S \mid A \]
\[ P ::= S \mid P \otimes P \mid P/L \]

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

CHOICE: \(S + S\) competing components (race policy)

CONSTANT: \(A \overset{\text{def}}{=} S\) assigning names

COOPERATION: \(P \otimes_P P\) \(\alpha \notin L\) concurrent activity
  (individual actions)
\(\alpha \in L\) cooperative activity
  (shared actions)
Introduction
Continuous State Space Models
Case Study in Web Services
Conclusions

PEPA

\[ S ::= (\alpha, r).S \mid S + S \mid A \]
\[ P ::= S \mid P \notimes P \mid P/L \]

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

CHOICE: \( S + S \) competing components (race policy)

CONSTANT: \( A \overset{\text{def}}{=} S \) assigning names

COOPERATION: \( P \notimes P \) \( \alpha \notin L \) concurrent activity (individual actions)
\( \alpha \in L \) cooperative activity (shared actions)

HIDING: \( P/L \) abstraction \( \alpha \in L \Rightarrow \alpha \rightarrow \tau \)

Jane Hillston. LFCS, University of Edinburgh.

Fluid Flow Approximation of PEPA Models
Outline

Introduction
  Background and Motivation
  PEPA

Continuous State Space Models
  Deriving Differential Equations
  Analysis based on Continuous-time Markov Chains
  Analysis based on Ordinary Differential Equations

Case Study in Web Services
  The model
  Analysis

Conclusions
New mathematical structures: differential equations

- Use a more abstract state representation rather than the CTMC complete state space.
New mathematical structures: differential equations

- Use a more abstract state representation rather than the CTMC complete state space.
- Assume that these state variables are subject to continuous rather than discrete change.
New mathematical structures: differential equations

- Use a more abstract state representation rather than the CTMC complete state space.
- Assume that these state variables are subject to continuous rather than discrete change.
- No longer aim to calculate the probability distribution over the entire state space of the model.
New mathematical structures: differential equations

- Use a more abstract state representation rather than the CTMC complete state space.
- Assume that these state variables are subject to continuous rather than discrete change.
- No longer aim to calculate the probability distribution over the entire state space of the model.

Appropriate for models in which there are large numbers of components of the same type.
Differential equations from PEPA models

- In a PEPA model the state at any current time is the local derivative or state of each component of the model.
- We can represent the state of the system as the count of the current number of each possible local derivative or component type.
- We can approximate the behaviour of the model by treating each count as a continuous variable, and the state of the model as a whole as the set of such variables.
- The evolution of each count variable can then be described by an ordinary differential equation.
Differential equations from PEPA models

- In a PEPA model the state at any current time is the local derivative or state of each component of the model.
- We can represent the state of the system as the count of the current number of each possible local derivative or component type.
- We can approximate the behaviour of the model by treating each count as a continuous variable, and the state of the model as a whole as the set of such variables.
- The evolution of each count variable can then be described by an ordinary differential equation.
Differential equations from PEPA models

- In a PEPA model the state at any current time is the local derivative or state of each component of the model.
- We can represent the state of the system as the count of the current number of each possible local derivative or component type.
- We can approximate the behaviour of the model by treating each count as a continuous variable, and the state of the model as a whole as the set of such variables.
- The evolution of each count variable can then be described by an ordinary differential equation.

Jane Hillston. LFCS, University of Edinburgh.
Differential equations from PEPA models

- In a PEPA model the state at any current time is the local derivative or state of each component of the model.
- We can represent the state of the system as the count of the current number of each possible local derivative or component type.
- We can approximate the behaviour of the model by treating each count as a continuous variable, and the state of the model as a whole as the set of such variables.
- The evolution of each count variable can then be described by an ordinary differential equation.
Differential equations from PEPA models

- In a PEPA model the state at any current time is the local derivative or state of each component of the model.
- We can represent the state of the system as the count of the current number of each possible local derivative or component type.
- We can approximate the behaviour of the model by treating each count as a continuous variable, and the state of the model as a whole as the set of such variables.
- The evolution of each count variable can then be described by an ordinary differential equation (assuming rates are deterministic).
Differential equations from PEPA models

- The PEPA definitions of the component specify the activities which can increase or decrease the number of components exhibited in the current state.
- The cooperations show when the number of instances of another component will have an influence on the evolution of this component.
Differential equations from PEPA models

- The PEPA definitions of the component specify the activities which can increase or decrease the number of components exhibited in the current state.

- The **cooperations** show when the number of instances of another component will have an **influence** on the evolution of this component.
Differential equations from PEPA models

Let $N(C_{ij}, t)$ denote the number of $C_{ij}$ type components at time $t$. 
Differential equations from PEPA models

Let $N(C_{ij}, t)$ denote the number of $C_{ij}$ type components at time $t$.

Consider the change in a small time $\delta t$:

$$N(C_{ij}, t + \delta t) - N(C_{ij}, t) =$$

$$- \sum_{(\alpha,r) \in Ex(C_{ij})} r \times \min_{C_{kl} \in Ex(\alpha,r)} (N(C_{kl}, t)) \delta t$$

\underbrace{\text{exit activities}}

$$+ \sum_{(\alpha,r) \in En(C_{ij})} r \times \min_{C_{kl} \in Ex(\alpha,r)} (N(C_{kl}, t)) \delta t$$

\underbrace{\text{entry activities}}
Differential equations from PEPA models

Let $N(C_{ij}, t)$ denote the number of $C_{ij}$ type components at time $t$.

Consider the change in a small time $\delta t$:

$$
N(C_{ij}, t + \delta t) - N(C_{ij}, t) = - \sum_{(\alpha, r) \in Ex(C_{ij})} r \times \min_{C_{kl} \in Ex(\alpha, r)} (N(C_{kl}, t)) \delta t
$$

exit activities

$$
+ \sum_{(\alpha, r) \in En(C_{ij})} r \times \min_{C_{kl} \in Ex(\alpha, r)} (N(C_{kl}, t)) \delta t
$$

entry activities
Differential equations from PEPA models

Let $N(C_{ij}, t)$ denote the number of $C_{ij}$ type components at time $t$.

Consider the change in a small time $\delta t$:

$$N(C_{ij}, t + \delta t) - N(C_{ij}, t) =$$

$$- \sum_{(\alpha,r) \in \text{Ex}(C_{ij})} r \times \min_{C_{k_l} \in \text{Ex}(\alpha, r)} (N(C_{k_l}, t)) \delta t$$

exit activities

$$+ \sum_{(\alpha,r) \in \text{En}(C_{ij})} r \times \min_{C_{k_l} \in \text{Ex}(\alpha, r)} (N(C_{k_l}, t)) \delta t$$

entry activities
Differential equations from PEPA models

Let $N(C_{ij}, t)$ denote the number of $C_{ij}$ type components at time $t$.

Consider the change in a small time $\delta t$:

$$N(C_{ij}, t + \delta t) - N(C_{ij}, t) =$$

$$- \sum_{(\alpha, r) \in \text{Ex}(C_{ij})} r \times \min_{C_{kl} \in \text{Ex}(\alpha, r)} (N(C_{kl}, t)) \delta t$$

exit activities

$$+ \sum_{(\alpha, r) \in \text{En}(C_{ij})} r \times \min_{C_{kl} \in \text{Ex}(\alpha, r)} (N(C_{kl}, t)) \delta t$$

entry activities
Differential equations from PEPA models

Let $N(C_{ij}, t)$ denote the number of $C_{ij}$ type components at time $t$.

Consider the change in a small time $\delta t$:

$$N(C_{ij}, t + \delta t) - N(C_{ij}, t) =$$

$$- \sum_{(\alpha,r) \in Ex(C_{ij})} r \times \min_{C_{kl} \in Ex(\alpha,r)} (N(C_{kl}, t)) \delta t$$

exit activities

$$+ \sum_{(\alpha,r) \in En(C_{ij})} r \times \min_{C_{kl} \in Ex(\alpha,r)} (N(C_{kl}, t)) \delta t$$

entry activities
Differential equations from PEPA models

Let $N(C_{ij}, t)$ denote the number of $C_{ij}$ type components at time $t$.

Dividing by $\delta t$ and taking the limit, $\delta t \rightarrow 0$:

$$\frac{dN(C_{ij}, t)}{dt} = - \sum_{(\alpha,r) \in Ex(C_{ij})} r \times \min_{C_{kl} \in Ex(\alpha,r)} (N(C_{kl}, t))$$

$$+ \sum_{(\alpha,r) \in En(C_{ij})} r \times \min_{C_{kl} \in Ex(\alpha,r)} (N(C_{kl}, t))$$
Activity matrix

Derivation of the system of ODEs representing the PEPA model then proceeds via an activity matrix which records the influence of each activity on each component type/derivative.

The matrix has one row for each component type and one column for each activity type.

One ODE is generated corresponding to each row of the matrix, taking into account the negative entries in the non-zero columns as these are the components for which this is an exit activity.
Activity matrix

Derivation of the system of ODEs representing the PEPA model then proceeds via an activity matrix which records the influence of each activity on each component type/derivative.

The matrix has one row for each component type and one column for each activity type.

One ODE is generated corresponding to each row of the matrix, taking into account the negative entries in the non-zero columns as these are the components for which this is an exit activity.
Activity matrix

Derivation of the system of ODEs representing the PEPA model then proceeds via an activity matrix which records the influence of each activity on each component type/derivative.

The matrix has one row for each component type and one column for each activity type.

One ODE is generated corresponding to each row of the matrix, taking into account the negative entries in the non-zero columns as these are the components for which this is an exit activity.
Activity matrix

Derivation of the system of ODEs representing the PEPA model then proceeds via an activity matrix which records the influence of each activity on each component type/derivative.

The matrix has one row for each component type and one column for each activity type.

One ODE is generated corresponding to each row of the matrix, taking into account the negative entries in the non-zero columns as these are the components for which this is an exit activity.
Analysis based on Continuous-time Markov Chains

Modelling with quantified process algebras

Tiny example

\[ P_1 \overset{\text{def}}{=} (start, r).P_2 \quad P_2 \overset{\text{def}}{=} (run, r).P_3 \quad P_3 \overset{\text{def}}{=} (stop, r).P_1 \]

\[ System \overset{\text{def}}{=} (P_1 \parallel P_1) \]
Modelling with quantified process algebras

Tiny example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1) \]

This example defines a system with nine reachable states:

1. \( P_1 \parallel P_1 \)
2. \( P_1 \parallel P_2 \)
3. \( P_1 \parallel P_3 \)
4. \( P_2 \parallel P_1 \)
5. \( P_2 \parallel P_2 \)
6. \( P_2 \parallel P_3 \)
7. \( P_3 \parallel P_1 \)
8. \( P_3 \parallel P_2 \)
9. \( P_3 \parallel P_3 \)

The transitions between states have quantified duration \( r \) which can be evaluated against a CTMC or ODE interpretation.
Analysis based on Continuous-time Markov Chains

Tiny example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1) \]

Using transient analysis we can evaluate the probability of each state with respect to time. For \( t = 0 \):

1. 1.0000  
2. 0.0000  
3. 0.0000  
4. 0.0000  
5. 0.0000  
6. 0.0000  
7. 0.0000  
8. 0.0000  
9. 0.0000
Analysis based on Continuous-time Markov Chains

Tiny example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1) \]

Using transient analysis we can evaluate the probability of each state with respect to time. For \( t = 1 \):

1. 0.1642
2. 0.1567
3. 0.0842
4. 0.1567
5. 0.1496
6. 0.0804
7. 0.0842
8. 0.0804
9. 0.0432
Analysis based on Continuous-time Markov Chains

Tiny example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1) \]

Using transient analysis we can evaluate the probability of each state with respect to time. For \( t = 2 \):

1. 0.1056        4. 0.1159        7. 0.1034
2. 0.1159        5. 0.1272        8. 0.1135
3. 0.1034        6. 0.1135        9. 0.1012
Analysis based on Continuous-time Markov Chains

Tiny example

\[ P_1 \overset{\text{def}}{=} (\text{start, } r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run, } r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop, } r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1) \]

Using transient analysis we can evaluate the probability of each state with respect to time. For \( t = 3 \):

1. 0.1082
2. 0.1106
3. 0.1100
4. 0.1106
5. 0.1132
6. 0.1125
7. 0.1100
8. 0.1125
9. 0.1119
Analysis based on Continuous-time Markov Chains

Tiny example

\[ P_1 \stackrel{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \stackrel{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \stackrel{\text{def}}{=} (\text{stop}, r).P_1 \]

\[ \text{System} \stackrel{\text{def}}{=} (P_1 \parallel P_1) \]

Using transient analysis we can evaluate the probability of each state with respect to time. For \( t = 4 \):

1. 0.1106
2. 0.1108
3. 0.1111
4. 0.1108
5. 0.1110
6. 0.1113
7. 0.1111
8. 0.1113
9. 0.1116
Analysis based on Continuous-time Markov Chains

Tiny example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1) \]

Using transient analysis we can evaluate the probability of each state with respect to time. For \( t = 5 \):

1. 0.1111  
2. 0.1110  
3. 0.1111  
4. 0.1110  
5. 0.1110  
6. 0.1111  
7. 0.1111  
8. 0.1111  
9. 0.1111
Analysis based on Continuous-time Markov Chains

Tiny example

\[ P_1 \overset{\text{def}}{=} (\text{start, } r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run, } r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop, } r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1) \]

Using transient analysis we can evaluate the probability of each state with respect to time. For \( t = 6 \):

1. 0.1111  
2. 0.1111  
3. 0.1111  
4. 0.1111  
5. 0.1110  
6. 0.1111  
7. 0.1111  
8. 0.1111  
9. 0.1111
Analysis based on Continuous-time Markov Chains

Tiny example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1) \]

Using transient analysis we can evaluate the probability of each state with respect to time. For \( t = 7 \):

1. 0.1111  
2. 0.1111  
3. 0.1111  
4. 0.1111  
5. 0.1111  
6. 0.1111  
7. 0.1111  
8. 0.1111  
9. 0.1111
Analysis based on Ordinary Differential Equations

Tiny example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1) \]

Using the ordinary differential equation semantics we can compute the expected number of each type of component.

For \( t = 0 \):

\[
\begin{align*}
P_1 &= 2.0000 \\
P_2 &= 0.0000 \\
P_3 &= 0.0000
\end{align*}
\]
Analysis based on Ordinary Differential Equations

Tiny example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1) \]

Using the ordinary differential equation semantics we can compute the expected number of each type of component.

For \( t = 1 \):

\begin{align*}
P_1 &= 0.8121 \\
P_2 &= 0.7734 \\
P_3 &= 0.4144
\end{align*}
Analysis based on Ordinary Differential Equations

Tiny example

\[ P_1 \overset{\text{def}}{=} (start, r).P_2 \quad P_2 \overset{\text{def}}{=} (run, r).P_3 \quad P_3 \overset{\text{def}}{=} (stop, r).P_1 \]

System \( \overset{\text{def}}{=} (P_1 \parallel P_1) \)

Using the ordinary differential equation semantics we can compute the expected number of each type of component.

For \( t = 2 \):

| \( P_1 \) | 0.6490 |
| \( P_2 \) | 0.7051 |
| \( P_3 \) | 0.6457 |
Analysis based on Ordinary Differential Equations

Tiny example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1) \]

Using the ordinary differential equation semantics we can compute the expected number of each type of component.

For \( t = 3 \):

\[ \begin{align*}
P_1 & \quad 0.6587 \\
P_2 & \quad 0.6719 \\
P_3 & \quad 0.6692
\end{align*} \]
Analysis based on Ordinary Differential Equations

Tiny example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1) \]

Using the ordinary differential equation semantics we can compute the expected number of each type of component.

For \( t = 4 \):

\[ P_1 \quad 0.6648 \]
\[ P_2 \quad 0.6665 \]
\[ P_3 \quad 0.6685 \]
Analysis based on Ordinary Differential Equations

Tiny example

\[
P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \\
P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \\
P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1
\]

System \overset{\text{def}}{=} (P_1 \parallel P_1)

Using the ordinary differential equation semantics we can compute the expected number of each type of component.

For \( t = 5 \):

\[
\begin{align*}
P_1 &\; 0.6666 \\
P_2 &\; 0.6663 \\
P_3 &\; 0.6669
\end{align*}
\]
Analysis based on Ordinary Differential Equations

Tiny example
\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1) \]

Using the ordinary differential equation semantics we can compute the expected number of each type of component.

For \( t = 6 \):

<table>
<thead>
<tr>
<th>Component</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P_1 )</td>
<td>0.6666</td>
</tr>
<tr>
<td>( P_2 )</td>
<td>0.6666</td>
</tr>
<tr>
<td>( P_3 )</td>
<td>0.6666</td>
</tr>
</tbody>
</table>
Analysis based on Ordinary Differential Equations

Tiny example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1) \]

Using the ordinary differential equation semantics we can compute the expected number of each type of component.

For \( t = 7 \):

- \( P_1 = 0.6666 \)
- \( P_2 = 0.6666 \)
- \( P_3 = 0.6666 \)
Analysis based on Ordinary Differential Equations

Slightly larger example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

System \( \overset{\text{def}}{=} (P_1 \parallel P_1 \parallel P_1) \)

A slightly larger example with a third copy of the process also initiated in state \( P_1 \).

For \( t = 0 \):

\[
\begin{align*}
P_1 & \quad 3.0000 \\
P_2 & \quad 0.0000 \\
P_3 & \quad 0.0000
\end{align*}
\]
Analysis based on Ordinary Differential Equations

Slightly larger example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1 \parallel P_1) \]

A slightly larger example with a third copy of the process also initiated in state \( P_1 \).

For \( t = 1 \):

\[
\begin{align*}
P_1 & \quad 1.1782 \\
P_2 & \quad 1.1628 \\
P_3 & \quad 0.6590
\end{align*}
\]
Analysis based on Ordinary Differential Equations

Slightly larger example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

System \( \overset{\text{def}}{=} (P_1 \parallel P_1 \parallel P_1) \)

A slightly larger example with a third copy of the process also initiated in state \( P_1 \).

For \( t = 2 \):

\[ \begin{array}{lcl}
   P_1 & 0.9766 \\
   P_2 & 1.0754 \\
   P_3 & 0.9479 \\
\end{array} \]
Analysis based on Ordinary Differential Equations

Slightly larger example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1 \parallel P_1) \]

A slightly larger example with a third copy of the process also initiated in state \( P_1 \).

For \( t = 3 \):

\[
\begin{align*}
P_1 & : 0.9838 \\
P_2 & : 1.0142 \\
P_3 & : 1.0020
\end{align*}
\]
Analysis based on Ordinary Differential Equations

Slightly larger example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

System \( \overset{\text{def}}{=} (P_1 \parallel P_1 \parallel P_1) \)

A slightly larger example with a third copy of the process also initiated in state \( P_1 \).

For \( t = 4 \):

\[
\begin{align*}
P_1 & \quad 0.9981 \\
P_2 & \quad 0.9995 \\
P_3 & \quad 1.0023
\end{align*}
\]
Analysis based on Ordinary Differential Equations

Slightly larger example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

\[ \text{System} \overset{\text{def}}{=} (P_1 \parallel P_1 \parallel P_1) \]

A slightly larger example with a third copy of the process also initiated in state \( P_1 \).

For \( t = 5 \):

\[ P_1 \quad 1.0001 \]
\[ P_2 \quad 0.9996 \]
\[ P_3 \quad 1.0003 \]
Analysis based on Ordinary Differential Equations

Slightly larger example

\[ P_1 \overset{\text{def}}{=} (\text{start, } r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run, } r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop, } r).P_1 \]

System \overset{\text{def}}{=} (P_1 \parallel P_1 \parallel P_1)

A slightly larger example with a third copy of the process also initiated in state \( P_1 \).

For \( t = 6 \):

\[
\begin{align*}
P_1 &= 1.0001 \\
P_2 &= 0.9999 \\
P_3 &= 1.0000
\end{align*}
\]
Analysis based on Ordinary Differential Equations

Slightly larger example

\[
P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1
\]

\[
\text{System} \overset{\text{def}}{=} (P_1 \parallel P_1 \parallel P_1)
\]

A slightly larger example with a third copy of the process also initiated in state \( P_1 \).

For \( t = 7 \):

\[
\begin{align*}
P_1 & \quad 1.0000 \\
P_2 & \quad 0.9999 \\
P_3 & \quad 0.9999
\end{align*}
\]
Analysis based on Ordinary Differential Equations

Slightly larger example

\[ P_1 \overset{\text{def}}{=} (\text{start}, r).P_2 \quad P_2 \overset{\text{def}}{=} (\text{run}, r).P_3 \quad P_3 \overset{\text{def}}{=} (\text{stop}, r).P_1 \]

System \( \overset{\text{def}}{=} (P_1 \parallel P_1 \parallel P_1) \)

A slightly larger example with a third copy of the process also initiated in state \( P_1 \).

For \( t = 8 \):

\[
\begin{align*}
  P_1 & \quad 1.0000 \\
  P_2 & \quad 1.0000 \\
  P_3 & \quad 1.0000
\end{align*}
\]
Isn’t this just the Chapman-Kolmogorov equations?

It is possible to perform transient analysis of a continuous-time Markov chain by solving the Chapman-Kolmogorov differential equations:

\[
\frac{d\pi(t)}{dt} = \pi(t)Q
\]

[Stewart, 1994]
Isn’t this just the Chapman-Kolmogorov equations?

It is possible to perform transient analysis of a continuous-time Markov chain by solving the Chapman-Kolmogorov differential equations:

$$\frac{d\pi(t)}{dt} = \pi(t)Q$$

[Stewart, 1994]

That’s not what we’re doing. We go directly to ODEs.
Outline

Introduction
  Background and Motivation
  PEPA

Continuous State Space Models
  Deriving Differential Equations
  Analysis based on Continuous-time Markov Chains
  Analysis based on Ordinary Differential Equations

Case Study in Web Services
  The model
  Analysis

Conclusions
The example which we consider is a **Web service** which has two types of clients:

- first party application clients which access the web service across a secure intranet, and
- second party browser clients which access the Web service across the Internet.

Second party clients route their service requests via trusted brokers.
The model

Example: Secure Web Service use

- The example which we consider is a Web service which has two types of clients:
  - first party application clients which access the web service across a secure intranet, and
  - second party browser clients which access the Web service across the Internet.

- Second party clients route their service requests via trusted brokers.
The example which we consider is a Web service which has two types of clients:

- first party application clients which access the web service across a secure intranet, and
- second party browser clients which access the Web service across the Internet.

Second party clients route their service requests via trusted brokers.
The example which we consider is a Web service which has two types of clients:

- first party application clients which access the web service across a secure intranet, and
- second party browser clients which access the Web service across the Internet.

Second party clients route their service requests via trusted brokers.
PEPA model: Second party clients and Brokers

- A second party client composes service requests, encrypts these and sends them to its broker.
PEPA model: Second party clients and Brokers

- A second party client composes service requests, encrypts these and sends them to its broker.
- It then waits for a response from the broker.
The model

**PEPA model: Second party clients and Brokers**

- A second party client composes service requests, encrypts these and sends them to its broker.
- It then waits for a response from the broker.
- Brokers add decryption and encryption steps to build end-to-end security from point-to-point security.
PEPA model: Second party clients and Brokers

- A second party client composes service requests, encrypts these and sends them to its broker.
- It then waits for a response from the broker.
- Brokers add decryption and encryption steps to build end-to-end security from point-to-point security.
- The broker forwards the request to the Web service and then waits for a response.

Jane Hillston. LFCS, University of Edinburgh.
Fluid Flow Approximation of PEPA Models
PEPA model: Second party clients and Brokers

- A second party client composes service requests, encrypts these and sends them to its broker.
- It then waits for a response from the broker.
- Brokers add decryption and encryption steps to build end-to-end security from point-to-point security.
- The broker forwards the request to the Web service and then waits for a response.
- Decryption and re-encryption are performed before returning the response to the client.
PEPA model: Second party clients

\[
\begin{align*}
\text{SPC}_{idle} & \quad \overset{\text{def}}{=} \quad (\text{compose}_{sp}, \ r_{sp\_cmp}) \cdot \text{SPC}_{enc} \\
\text{SPC}_{enc} & \quad \overset{\text{def}}{=} \quad (\text{encrypt}_{b}, \ r_{sp\_encb}) \cdot \text{SPC}_{sending} \\
\text{SPC}_{sending} & \quad \overset{\text{def}}{=} \quad (\text{request}_{b}, \ r_{sp\_req}) \cdot \text{SPC}_{waiting} \\
\text{SPC}_{waiting} & \quad \overset{\text{def}}{=} \quad (\text{response}_{b}, \top) \cdot \text{SPC}_{dec} \\
\text{SPC}_{dec} & \quad \overset{\text{def}}{=} \quad (\text{decrypt}_{b}, \ r_{sp\_decb}) \cdot \text{SPC}_{idle}
\end{align*}
\]
The model

**PEPA model: Second party clients**

\[
\begin{align*}
SPC_{idle} & \overset{\text{def}}{=} (\text{compose}_{sp}, r_{sp\_cmp}) \cdot SPC_{enc} \\
SPC_{enc} & \overset{\text{def}}{=} (\text{encrypt}_b, r_{sp\_encb}) \cdot SPC_{sending} \\
SPC_{sending} & \overset{\text{def}}{=} (\text{request}_b, r_{sp\_req}) \cdot SPC_{waiting} \\
SPC_{waiting} & \overset{\text{def}}{=} (\text{response}_b, \top) \cdot SPC_{dec} \\
SPC_{dec} & \overset{\text{def}}{=} (\text{decrypt}_b, r_{sp\_decb}) \cdot SPC_{idle}
\end{align*}
\]
The model

**PEPA model: Second party clients**

- **Second party**
- **Broker**
- **Web service**
- **First party**

\[
\begin{align*}
SPC_{idle} & \overset{\text{def}}{=} (\text{compose}_{sp}, r_{sp\_cmp}).SPC_{enc} \\
SPC_{enc} & \overset{\text{def}}{=} (\text{encrypt}_{b}, r_{sp\_encb}).SPC_{sending} \\
SPC_{sending} & \overset{\text{def}}{=} (\text{request}_{b}, r_{sp\_req}).SPC_{waiting} \\
SPC_{waiting} & \overset{\text{def}}{=} (\text{response}_{b}, \top).SPC_{dec} \\
SPC_{dec} & \overset{\text{def}}{=} (\text{decrypt}_{b}, r_{sp\_decb}).SPC_{idle}
\end{align*}
\]
The model

**PEPA model: Brokers**

\[
\begin{align*}
\text{Broker}_{idle} & \define (\text{request}_b, \top).\text{Broker}_{dec\_input} \\
\text{Broker}_{dec\_input} & \define (\text{decrypt}_{sp}, r_{b\_dec\_sp}).\text{Broker}_{enc\_input} \\
\text{Broker}_{enc\_input} & \define (\text{encrypt}_{ws}, r_{b\_enc\_ws}).\text{Broker}_{sending} \\
\text{Broker}_{sending} & \define (\text{request}_{ws}, r_{b\_req}).\text{Broker}_{waiting} \\
\text{Broker}_{waiting} & \define (\text{response}_{ws}, \top).\text{Broker}_{dec\_resp} \\
\text{Broker}_{dec\_resp} & \define (\text{decrypt}_{ws}, r_{b\_dec\_ws}).\text{Broker}_{enc\_resp} \\
\text{Broker}_{enc\_resp} & \define (\text{encrypt}_{sp}, r_{b\_enc\_sp}).\text{Broker}_{replying} \\
\text{Broker}_{replying} & \define (\text{response}_b, r_{b\_resp}).\text{Broker}_{idle}
\end{align*}
\]
The model

PEPA model: Brokers

Broker\_idle \quad \text{def} = \quad (\text{request}_b, \top).\text{Broker}_\text{dec\_input}

Broker\_dec\_input \quad \text{def} = \quad (\text{decrypt}_{sp}, r_{b\_dec\_sp}).\text{Broker}_\text{enc\_input}

Broker\_enc\_input \quad \text{def} = \quad (\text{encrypt}_{ws}, r_{b\_enc\_ws}).\text{Broker}_\text{sending}

Broker\_sending \quad \text{def} = \quad (\text{request}_{ws}, r_{b\_req}).\text{Broker}_\text{waiting}

Broker\_waiting \quad \text{def} = \quad (\text{response}_{ws}, \top).\text{Broker}_\text{dec\_resp}

Broker\_dec\_resp \quad \text{def} = \quad (\text{decrypt}_{ws}, r_{b\_dec\_ws}).\text{Broker}_\text{enc\_resp}

Broker\_enc\_resp \quad \text{def} = \quad (\text{encrypt}_{sp}, r_{b\_enc\_sp}).\text{Broker}_\text{replying}

Broker\_replying \quad \text{def} = \quad (\text{response}_b, r_{b\_resp}).\text{Broker}_{\text{idle}}
The model

PEPA model: Brokers

\[
\begin{align*}
\text{Broker}_{\text{idle}} & \overset{\text{def}}{=} (\text{request}_b, \top).\text{Broker}_{\text{dec\_input}} \\
\text{Broker}_{\text{dec\_input}} & \overset{\text{def}}{=} (\text{decrypt}_{\text{sp}}, r_{b\_\text{dec\_sp}}).\text{Broker}_{\text{enc\_input}} \\
\text{Broker}_{\text{enc\_input}} & \overset{\text{def}}{=} (\text{encrypt}_{\text{ws}}, r_{b\_\text{enc\_ws}}).\text{Broker}_{\text{sending}} \\
\text{Broker}_{\text{sending}} & \overset{\text{def}}{=} (\text{request}_{\text{ws}}, r_{b\_\text{req}}).\text{Broker}_{\text{waiting}} \\
\text{Broker}_{\text{waiting}} & \overset{\text{def}}{=} (\text{response}_{\text{ws}}, \top).\text{Broker}_{\text{dec\_resp}} \\
\text{Broker}_{\text{dec\_resp}} & \overset{\text{def}}{=} (\text{decrypt}_{\text{ws}}, r_{b\_\text{dec\_ws}}).\text{Broker}_{\text{enc\_resp}} \\
\text{Broker}_{\text{enc\_resp}} & \overset{\text{def}}{=} (\text{encrypt}_{\text{sp}}, r_{b\_\text{enc\_sp}}).\text{Broker}_{\text{replying}} \\
\text{Broker}_{\text{replying}} & \overset{\text{def}}{=} (\text{response}_{\text{b}}, r_{b\_\text{resp}}).\text{Broker}_{\text{idle}}
\end{align*}
\]
The model

**PEPA model: Brokers**

![Diagram of the PEPA model: Brokers]

- **Broker\textsubscript{idle}**
  \[
  \text{def} \quad (\text{request}_b, \top).\text{Broker}_{\text{dec}\_\text{input}}
  \]

- **Broker\textsubscript{dec\_input}**
  \[
  \text{def} \quad (\text{decrypt}_{sp}, r_{b\_\text{dec\_sp}}).\text{Broker}_{\text{enc\_input}}
  \]

- **Broker\textsubscript{enc\_input}**
  \[
  \text{def} \quad (\text{encrypt}_{ws}, r_{b\_\text{enc\_ws}}).\text{Broker}_{\text{sending}}
  \]

- **Broker\textsubscript{sending}**
  \[
  \text{def} \quad (\text{request}_{ws}, r_{b\_\text{req}}).\text{Broker}_{\text{waiting}}
  \]

- **Broker\textsubscript{waiting}**
  \[
  \text{def} \quad (\text{response}_{ws}, \top).\text{Broker}_{\text{dec\_resp}}
  \]

- **Broker\textsubscript{dec\_resp}**
  \[
  \text{def} \quad (\text{decrypt}_{ws}, r_{b\_\text{dec\_ws}}).\text{Broker}_{\text{enc\_resp}}
  \]

- **Broker\textsubscript{enc\_resp}**
  \[
  \text{def} \quad (\text{encrypt}_{sp}, r_{b\_\text{enc\_sp}}).\text{Broker}_{\text{replying}}
  \]

- **Broker\textsubscript{replying}**
  \[
  \text{def} \quad (\text{response}_b, r_{b\_\text{resp}}).\text{Broker}_{\text{idle}}
  \]
**PEPA model: First party clients**

- The lifetime of a first party client mirrors that of a second party client except that encryption need not be used when all of the communication is conducted across a secure intranet.
The model

PEPA model: First party clients

- The lifetime of a first party client mirrors that of a second party client except that encryption need not be used when all of the communication is conducted across a secure intranet.
- Also the service may be invoked by a remote method invocation to the host machine instead of via HTTP.
The model

PEPA model: First party clients

The lifetime of a first party client mirrors that of a second party client except that encryption need not be used when all of the communication is conducted across a secure intranet.

Also the service may be invoked by a remote method invocation to the host machine instead of via HTTP.

Thus the first party client experiences the Web service as a blocking remote method invocation.
The model

**PEPA model: First party clients**

\[
\begin{align*}
FPC_{idle} & \overset{\text{def}}{=} (\text{compose}_{fp}, r_{fp\_cmp}).FPC_{calling} \\
FPC_{calling} & \overset{\text{def}}{=} (\text{invoke}_{ws}, r_{fp\_inv}).FPC_{blocked} \\
FPC_{blocked} & \overset{\text{def}}{=} (\text{result}_{ws}, \top).FPC_{idle}
\end{align*}
\]
The model

**PEPA model: First party clients**

\[
FPC_{idle} \overset{\text{def}}{=} (\text{compose}_{fp}, r_{fp\_cmp}).FPC_{calling}
\]
\[
FPC_{calling} \overset{\text{def}}{=} (\text{invoke}_{ws}, r_{fp\_inv}).FPC_{blocked}
\]
\[
FPC_{blocked} \overset{\text{def}}{=} (\text{result}_{ws}, \top).FPC_{idle}
\]
The model

PEPA model: First party clients

- **FPC\text{idle}** defined as $(\text{compose}_{fp}, r_{fp\_cmp}).FPC_{\text{calling}}$
- **FPC\text{calling}** defined as $(\text{invoke}_{ws}, r_{fp\_inv}).FPC_{\text{blocked}}$
- **FPC\text{blocked}** defined as $(\text{result}_{ws}, \top).FPC_{\text{idle}}$
The model

PEPA model: Web service

\[
\begin{align*}
WS_{idle} & \overset{\text{def}}{=} (request_{ws}, \top) \cdot WS_{decoding} \\
& \quad + (invoke_{ws}, \top) \cdot WS_{method} \\
WS_{decoding} & \overset{\text{def}}{=} (decryptReq_{ws}, r_{ws_{-}dec_{-}b}) \cdot WS_{execution} \\
WS_{execution} & \overset{\text{def}}{=} (execute_{ws}, r_{ws_{-}exec}) \cdot WS_{securing} \\
WS_{securing} & \overset{\text{def}}{=} (encryptResp_{ws}, r_{ws_{-}enc_{-}b}) \cdot WS_{responding} \\
WS_{responding} & \overset{\text{def}}{=} (response_{ws}, r_{ws_{-}resp_{-}b}) \cdot WS_{idle} \\
WS_{method} & \overset{\text{def}}{=} (execute_{ws}, r_{ws_{-}exec}) \cdot WS_{returning} \\
WS_{returning} & \overset{\text{def}}{=} (result_{ws}, r_{ws_{-}res}) \cdot WS_{idle}
\end{align*}
\]
The model

PEPA model: Web service

\[
egin{align*}
WS_{idle} & \overset{\text{def}}{=} (\text{request}_{ws}, \top).WS_{decoding} + (\text{invoke}_{ws}, \top).WS_{method} \\
WS_{decoding} & \overset{\text{def}}{=} (\text{decryptReq}_{ws}, r_{ws_{dec_b}}).WS_{execution} \\
WS_{execution} & \overset{\text{def}}{=} (\text{execute}_{ws}, r_{ws_{exec}}).WS_{securing} \\
WS_{securing} & \overset{\text{def}}{=} (\text{encryptResp}_{ws}, r_{ws_{enc_b}}).WS_{responding} \\
WS_{responding} & \overset{\text{def}}{=} (\text{response}_{ws}, r_{ws_{resp_b}}).WS_{idle} \\
WS_{method} & \overset{\text{def}}{=} (\text{execute}_{ws}, r_{ws_{exec}}).WS_{returning} \\
WS_{returning} & \overset{\text{def}}{=} (\text{result}_{ws}, r_{ws_{res}}).WS_{idle}
\end{align*}
\]
PEPA model: Web service

\[
\begin{align*}
WS_{idle} & \overset{\text{def}}{=} (request_{ws}, T).WS_{decoding} \\
& \quad + (\text{invoke}_{ws}, T).WS_{method} \\
WS_{decoding} & \overset{\text{def}}{=} (\text{decryptReq}_{ws}, r_{ws\_dec\_b}).WS_{execution} \\
WS_{execution} & \overset{\text{def}}{=} (\text{execute}_{ws}, r_{ws\_exec}).WS_{securing} \\
WS_{securing} & \overset{\text{def}}{=} (\text{encryptResp}_{ws}, r_{ws\_enc\_b}).WS_{responding} \\
WS_{responding} & \overset{\text{def}}{=} (\text{response}_{ws}, r_{ws\_resp\_b}).WS_{idle} \\
WS_{method} & \overset{\text{def}}{=} (\text{execute}_{ws}, r_{ws\_exec}).WS_{returning} \\
WS_{returning} & \overset{\text{def}}{=} (\text{result}_{ws}, r_{ws\_res}).WS_{idle}
\end{align*}
\]
The model

**PEPA model: Web service**

![Diagram of the model with nodes labeled Second party, Broker, Web service, and First party, connected with arrows.]

\[
WS_{idle} \equiv \left( request_{ws}, \top \right).WS_{decoding} \\
+ \left( invoke_{ws}, \top \right).WS_{method} \\
WS_{decoding} \equiv \left( decryptReq_{ws}, r_{ws\_dec\_b} \right).WS_{execution} \\
WS_{execution} \equiv \left( execute_{ws}, r_{ws\_exec} \right).WS_{securing} \\
WS_{securing} \equiv \left( encryptResp_{ws}, r_{ws\_enc\_b} \right).WS_{responding} \\
WS_{responding} \equiv \left( response_{ws}, r_{ws\_resp\_b} \right).WS_{idle} \\
WS_{method} \equiv \left( execute_{ws}, r_{ws\_exec} \right).WS_{returning} \\
WS_{returning} \equiv \left( result_{ws}, r_{ws\_res} \right).WS_{idle}
\]
The model

PEPA model: System composition

In the initial state of the system model we represent each of the four component types being initially in their idle state.

\[
\text{System} \overset{\text{def}}{=} \left( SPC_{idle} \parallel_{\mathcal{K}} \text{Broker}_{idle} \right) \parallel_{\mathcal{L}} \left( WS_{idle} \parallel_{\mathcal{M}} FPC_{idle} \right)
\]

where

\[
\mathcal{K} = \{ \text{request}_b, \text{response}_b \}
\]

\[
\mathcal{L} = \{ \text{request}_{ws}, \text{response}_{ws} \}
\]

\[
\mathcal{M} = \{ \text{invoke}_{ws}, \text{result}_{ws} \}
\]
The model

**PEPA model: System composition**

In the initial state of the system model we represent each of the four component types being initially in their idle state.

\[
\text{System} \overset{\text{def}}{=} (\text{SPC}_{\text{idle}} \circlearrowleft_{\mathcal{K}} \text{Broker}_{\text{idle}}) \circlearrowright_{\mathcal{L}} (\text{WS}_{\text{idle}} \circlearrowright_{\mathcal{M}} \text{FPC}_{\text{idle}})
\]

where
\[
\begin{align*}
\mathcal{K} &= \{ \text{request}_b, \text{response}_b \} \\
\mathcal{L} &= \{ \text{request}_{ws}, \text{response}_{ws} \} \\
\mathcal{M} &= \{ \text{invoke}_{ws}, \text{result}_{ws} \}
\end{align*}
\]

This model represents the smallest possible instance of the system, where there is one instance of each component type. We evaluate the system as the number of clients, brokers, and copies of the service increase.

Jane Hillston. LFCS, University of Edinburgh.

**Fluid Flow Approximation of PEPA Models**
Cost of analysis

- We compare ODE-based evaluation against other techniques which could be used to analyse the model.
Cost of analysis

- We compare ODE-based evaluation against other techniques which could be used to analyse the model.
  - Steady-state and transient analysis as implemented by the PRISM probabilistic model-checker.
Cost of analysis

- We compare ODE-based evaluation against other techniques which could be used to analyse the model.
  - Steady-state and transient analysis as implemented by the PRISM probabilistic model-checker.
  - Monte Carlo Markov Chain simulation (a Java implementation of Gillespie’s Direct Method).
Running times from analyses (in seconds)

<table>
<thead>
<tr>
<th>Second party clients</th>
<th>Brokers</th>
<th>Web service instances</th>
<th>First party clients</th>
<th>Number of states in the full state-space</th>
<th>Number of states in the aggregated state-space</th>
<th>Sparse matrix steady-state</th>
<th>Matrix/MTBDD steady-state</th>
<th>Transient solution for time $t = 100$</th>
<th>MCMC simulation one run to $t = 100$</th>
<th>ODE solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>48</td>
<td>48</td>
<td>1.04</td>
<td>1.10</td>
<td>1.01</td>
<td>2.47</td>
<td>2.81</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>6,304</td>
<td>860</td>
<td>2.15</td>
<td>2.26</td>
<td>2.31</td>
<td>2.45</td>
<td>2.81</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1,130,496</td>
<td>161,296</td>
<td>172.48</td>
<td>255.48</td>
<td>588.80</td>
<td>2.48</td>
<td>2.83</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>&gt;234M</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
</tr>
</tbody>
</table>

Jane Hillston. LFCS, University of Edinburgh.

Fluid Flow Approximation of PEPA Models
## Running times from analyses (in seconds)

<table>
<thead>
<tr>
<th>Second party clients</th>
<th>Brokers</th>
<th>Web service instances</th>
<th>First party clients</th>
<th>Number of states in the full state-space</th>
<th>Number of states in the aggregated state-space</th>
<th>Sparse matrix steady-state</th>
<th>Matrix/MTBDD steady-state</th>
<th>Transient solution for time $t = 100$</th>
<th>MCMC simulation one run to $t = 100$</th>
<th>ODE solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>48</td>
<td>48</td>
<td>1.04</td>
<td>1.10</td>
<td>1.01</td>
<td>2.47</td>
<td>2.81</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>6,304</td>
<td>860</td>
<td>2.15</td>
<td>2.26</td>
<td>2.31</td>
<td>2.45</td>
<td>2.81</td>
</tr>
</tbody>
</table>
### Analysis

#### Running times from analyses (in seconds)

<table>
<thead>
<tr>
<th>Second party clients</th>
<th>Brokers</th>
<th>Web service instances</th>
<th>First party clients</th>
<th>Second party clients</th>
<th>Brokers</th>
<th>Web service instances</th>
<th>First party clients</th>
<th>Number of states in the full state-space</th>
<th>Number of states in the aggregated state-space</th>
<th>Sparse matrix steady-state</th>
<th>Matrix/MTBBDD steady-state</th>
<th>Transient solution for time $t = 100$</th>
<th>MCMC simulation one run to $t = 100$</th>
<th>ODE solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>48</td>
<td>48</td>
<td></td>
<td>1.04</td>
<td>1.10</td>
<td>1.01</td>
<td>2.47</td>
<td>2.81</td>
<td>2.78</td>
<td>2.78</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>6,304</td>
<td>860</td>
<td></td>
<td>2.15</td>
<td>2.26</td>
<td>2.31</td>
<td>2.45</td>
<td>2.81</td>
<td>3.72</td>
<td>2.77</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1,130,496</td>
<td>161,296</td>
<td></td>
<td>172.48</td>
<td>255.48</td>
<td>588.80</td>
<td>2.48</td>
<td>2.83</td>
<td>5.44</td>
<td>2.77</td>
</tr>
</tbody>
</table>

Jane Hillston. LFCS, University of Edinburgh.

Fluid Flow Approximation of PEPA Models
## Running times from analyses (in seconds)

<table>
<thead>
<tr>
<th>Second party clients</th>
<th>Brokers</th>
<th>Web service instances</th>
<th>First party clients</th>
<th>Number of states in the full state-space</th>
<th>Number of states in the aggregated state-space</th>
<th>Sparse matrix steady-state</th>
<th>Matrix/MTBDD steady-state</th>
<th>Transient solution for time $t = 100$</th>
<th>MCMC simulation one run to $t = 100$</th>
<th>ODE solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>48</td>
<td>48</td>
<td>1.04</td>
<td>1.10</td>
<td>1.01</td>
<td>2.47</td>
<td>2.81</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>6,304</td>
<td>860</td>
<td>2.15</td>
<td>2.26</td>
<td>2.31</td>
<td>2.45</td>
<td>2.81</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1,130,496</td>
<td>161,296</td>
<td>172.48</td>
<td>255.48</td>
<td>588.80</td>
<td>2.48</td>
<td>2.83</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>$&gt;234M$</td>
<td>$-$</td>
<td>$-$</td>
<td>$-$</td>
<td>$-$</td>
<td>2.44</td>
<td>2.85</td>
</tr>
</tbody>
</table>
## Running times from analyses (in seconds)

<table>
<thead>
<tr>
<th>Second party clients</th>
<th>Brokers</th>
<th>Web service instances</th>
<th>First party clients</th>
<th>Number of states in the full state-space</th>
<th>Number of states in the aggregated state-space</th>
<th>Sparse matrix steady-state</th>
<th>Matrix/MTBDD steady-state</th>
<th>Transient solution for time t = 100</th>
<th>MCMC simulation one run to t = 100</th>
<th>ODE solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>48</td>
<td>48</td>
<td>1.04</td>
<td>1.10</td>
<td>1.01</td>
<td>2.47</td>
<td>2.81</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>6,304</td>
<td>860</td>
<td>2.15</td>
<td>2.26</td>
<td>2.31</td>
<td>2.45</td>
<td>2.81</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1,130,496</td>
<td>161,296</td>
<td>172.48</td>
<td>255.48</td>
<td>588.80</td>
<td>2.48</td>
<td>2.83</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>&gt;234M</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>2.44</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>2.78</td>
</tr>
</tbody>
</table>

Jane Hillston. LFCS, University of Edinburgh.

Fluid Flow Approximation of PEPA Models
## Analysis

### Running times from analyses (in seconds)

<table>
<thead>
<tr>
<th></th>
<th>Second party clients</th>
<th>Brokers</th>
<th>Web service instances</th>
<th>First party clients</th>
<th>Number of states in the full state-space</th>
<th>Number of states in the aggregated state-space</th>
<th>Sparse matrix steady-state</th>
<th>Matrix/MTBDD steady-state</th>
<th>Transient solution for time $t = 100$</th>
<th>MCMC simulation one run to $t = 100$</th>
<th>ODE solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>48</td>
<td>48</td>
<td>1.04</td>
<td>1.10</td>
<td>1.01</td>
<td>2.47</td>
<td>2.81</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>6,304</td>
<td>860</td>
<td>2.15</td>
<td>2.26</td>
<td>2.31</td>
<td>2.45</td>
<td>2.81</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1,130,496</td>
<td>161,296</td>
<td>172.48</td>
<td>255.48</td>
<td>588.80</td>
<td>2.48</td>
<td>2.83</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>$&gt;234M$</td>
<td>$-$</td>
<td>$-$</td>
<td>$-$</td>
<td>$-$</td>
<td>2.44</td>
<td>2.85</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>$-$</td>
<td>$-$</td>
<td>$-$</td>
<td>$-$</td>
<td>$-$</td>
<td>2.78</td>
<td>2.78</td>
</tr>
<tr>
<td>1000</td>
<td>100</td>
<td>500</td>
<td>1000</td>
<td>1000</td>
<td>$-$</td>
<td>$-$</td>
<td>$-$</td>
<td>$-$</td>
<td>$-$</td>
<td>3.72</td>
<td>2.77</td>
</tr>
</tbody>
</table>

Jane Hillston. LFCS, University of Edinburgh.

**Fluid Flow Approximation of PEPA Models**
## Analysis

### Running times from analyses (in seconds)

<table>
<thead>
<tr>
<th>Second party clients</th>
<th>Brokers</th>
<th>Web service instances</th>
<th>First party clients</th>
<th>Number of states in the full state-space</th>
<th>Number of states in the aggregated state-space</th>
<th>Sparse matrix steady-state</th>
<th>Matrix/MTBDD steady-state</th>
<th>Transient solution for time $t = 100$</th>
<th>MCMC simulation one run to $t = 100$</th>
<th>ODE solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>48</td>
<td>48</td>
<td>1.04</td>
<td>1.10</td>
<td>1.01</td>
<td>2.47</td>
<td>2.81</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>6,304</td>
<td>860</td>
<td>2.15</td>
<td>2.26</td>
<td>2.31</td>
<td>2.45</td>
<td>2.81</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1,130,496</td>
<td>161,296</td>
<td>172.48</td>
<td>255.48</td>
<td>588.80</td>
<td>2.48</td>
<td>2.83</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>$&gt;234M$</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>2.44</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>2.78</td>
</tr>
<tr>
<td>1000</td>
<td>100</td>
<td>500</td>
<td>1000</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>3.72</td>
<td>2.77</td>
</tr>
<tr>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>5.44</td>
<td>2.77</td>
</tr>
</tbody>
</table>

Jane Hillston. LFCS, University of Edinburgh.

Fluid Flow Approximation of PEPA Models
## Analysis

### Running times from analyses (in seconds)

<table>
<thead>
<tr>
<th>Second party clients</th>
<th>Brokers</th>
<th>Web service instances</th>
<th>First party clients</th>
<th>Number of states in the full state-space</th>
<th>Number of states in the aggregated state-space</th>
<th>Sparse matrix steady-state</th>
<th>Matrix/MTBDD steady-state</th>
<th>Transient solution for time $t = 100$</th>
<th>MCMC simulation one run to $t = 100$</th>
<th>ODE solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>48</td>
<td>48</td>
<td>1.04</td>
<td>1.10</td>
<td>1.01</td>
<td>2.47</td>
<td>2.81</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>6,304</td>
<td>860</td>
<td>2.15</td>
<td>2.26</td>
<td>2.31</td>
<td>2.45</td>
<td>2.81</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1,130,496</td>
<td>161,296</td>
<td>172.48</td>
<td>255.48</td>
<td>588.80</td>
<td>2.48</td>
<td>2.83</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>$&gt;234M$</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>2.44</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>2.44</td>
</tr>
<tr>
<td>1000</td>
<td>100</td>
<td>500</td>
<td>1000</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>3.72</td>
</tr>
<tr>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>2.77</td>
</tr>
</tbody>
</table>

Jane Hillston. LFCS, University of Edinburgh.

Fluid Flow Approximation of PEPA Models
Time series analysis via ODEs

- We assume a system in which the number of clients of both kinds, brokers, and web service instances are all 1000.
Time series analysis via ODEs

- We assume a system in which the number of clients of both kinds, brokers, and web service instances are all 1000.
- We present the results from our ODE integrator as time-series plots of the number of each type of component behaviour as a function of time, as time runs from $t = 0$ to $t = 100$. 
Time series analysis via ODEs

- We assume a system in which the number of clients of both kinds, brokers, and web service instances are all 1000.
- We present the results from our ODE integrator as time-series plots of the number of each type of component behaviour as a function of time, as time runs from $t = 0$ to $t = 100$.
- The graphs show fluctuations in the numbers of components with respect to time.
Time series analysis via ODEs

- We assume a system in which the number of clients of both kinds, brokers, and web service instances are all 1000.
- We present the results from our ODE integrator as time-series plots of the number of each type of component behaviour as a function of time, as time runs from $t = 0$ to $t = 100$.
- The graphs show fluctuations in the numbers of components with respect to time.
- We can observe an initial flurry of activity until the system stabilises into its steady-state equilibrium at time (around) $t = 50$. 
Second party clients

Jane Hillston. LFCS, University of Edinburgh.

Fluid Flow Approximation of PEPA Models
Brokers

[Graph showing the number of brokers over time with different states such as Decoding input, Decoding response, Encoding input, Encoding response, Idle, Replying, Sending, and Waiting.]

Jane Hillston. LFCS, University of Edinburgh.

Fluid Flow Approximation of PEPA Models
First party clients

![Graph showing the number of first party clients over time in different states: Blocked, Calling, and Idle.](image)
Web service

Jane Hillston. LFCS, University of Edinburgh.

Fluid Flow Approximation of PEPA Models
Comparison with Continuous-Time Markov Chain solution

- Direct comparison of results is not possible.
Comparison with Continuous-Time Markov Chain solution

- Direct comparison of results is not possible.
- For systems with the same blocking characteristics (1000 instances of all components vs 1 instance of each component) we compared values of performance measures derived via each of the two models.

Jane Hillston. LFCS, University of Edinburgh.

Fluid Flow Approximation of PEPA Models
Comparison with Continuous-Time Markov Chain solution

- Direct comparison of results is not possible.
- For systems with the same blocking characteristics (1000 instances of all components vs 1 instance of each component) we compared values of performance measures derived via each of the two models.
- We found good agreement between the CTMC results and those results obtained by ODE solution after the system stabilises into its steady-state equilibrium.
Comparison with Continuous-Time Markov Chain solution

- Direct comparison of results is not possible.

- For systems with the same blocking characteristics (1000 instances of all components vs 1 instance of each component) we compared values of performance measures derived via each of the two models.

- We found good agreement between the CTMC results and those results obtained by ODE solution after the system stabilises into its steady-state equilibrium. Less than 1% difference.
Outline

Introduction
  Background and Motivation
  PEPA

Continuous State Space Models
  Deriving Differential Equations
  Analysis based on Continuous-time Markov Chains
  Analysis based on Ordinary Differential Equations

Case Study in Web Services
  The model
  Analysis

Conclusions
Conclusions

- Unlike fluid queues and fluid stochastic Petri nets, these are wholly continuous models.
Conclusions

- Unlike fluid queues and fluid stochastic Petri nets, these are wholly continuous models.
- Closest to diffusion approximations in queueing networks or continuous Petri nets.
Conclusions

- Unlike fluid queues and fluid stochastic Petri nets, these are wholly continuous models.
- Closest to diffusion approximations in queueing networks or continuous Petri nets.
- Based on an assumption that there are large numbers of components of the same type within the model.
Conclusions

- Unlike fluid queues and fluid stochastic Petri nets, these are wholly continuous models.
- Closest to diffusion approximations in queueing networks or continuous Petri nets.
- Based on an assumption that there are large numbers of components of the same type within the model.
- Particularly good for when we want to model the behaviour of individuals but study the behaviour of populations.
Conclusions

Unlike fluid queues and fluid stochastic Petri nets, these are wholly continuous models.

Closest to diffusion approximations in queueing networks or continuous Petri nets.

Based on an assumption that there are large numbers of components of the same type within the model.

Particularly good for when we want to model the behaviour of individuals but study the behaviour of populations.

The models are no longer stochastic, since service rates are assumed to be deterministic.
Future Work

- Relax some of the restrictions which are placed on the model:
Future Work

- Relax some of the restrictions which are placed on the model:
  - Repeated components do not need to be independent of each other;
Future Work

- Relax some of the restrictions which are placed on the model:
  - Repeated components do not need to be independent of each other;
  - Partners in cooperations can have different rates for the shared activity;
Future Work

- Relax some of the restrictions which are placed on the model:
  - Repeated components do not need to be independent of each other;
  - Partners in cooperations can have different rates for the shared activity;
- Reintroduce a stochastic element:
Future Work

- Relax some of the restrictions which are placed on the model:
  - Repeated components do not need to be independent of each other;
  - Partners in cooperations can have different rates for the shared activity;
- Reintroduce a stochastic element:
  - Use of random or stochastic differential equations;
Future Work

- Relax some of the restrictions which are placed on the model:
  - Repeated components do not need to be independent of each other;
  - Partners in cooperations can have different rates for the shared activity;
- Reintroduce a stochastic element:
  - Use of random or stochastic differential equations;
  - Limit the continuous element to a few continuous component types, with others having usual CTMC semantics (c.f. fluid models)