CICADA Seminar

Continuous Interpretations of Process Algebra Models

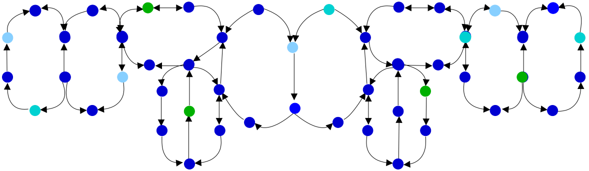
Jane Hillston

Laboratory for Foundations of Computer Science
School of Informatics,
University of Edinburgh

8th April 2008

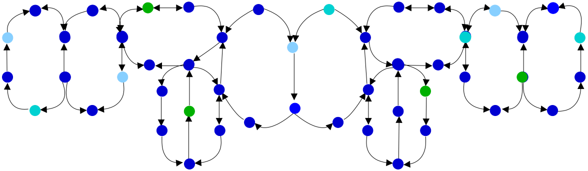
Collective Dynamics

The behaviour of many systems can be interpreted as the result of the collective behaviour of a large number of interacting entities.



Collective Dynamics

The behaviour of many systems can be interpreted as the result of the collective behaviour of a large number of interacting entities.



For such systems we are often as interested in the population level behaviour as we are in the behaviour of the individual entities.

Process Algebra and Collective Dynamics

Process algebras are well-suited to modelling such systems

Process Algebra and Collective Dynamics

Process algebras are well-suited to modelling such systems

- Developed to represent concurrent behaviour compositionally;

Process Algebra and Collective Dynamics

Process algebras are well-suited to modelling such systems

- Developed to represent concurrent behaviour compositionally;
- Capture the interactions between individuals explicitly;

Process Algebra and Collective Dynamics

Process algebras are well-suited to modelling such systems

- Developed to represent concurrent behaviour compositionally;
- Capture the interactions between individuals explicitly;
- Incorporate formal apparatus for reasoning about the behaviour of systems;

Process Algebra and Collective Dynamics

Process algebras are well-suited to modelling such systems

- Developed to represent concurrent behaviour compositionally;
- Capture the interactions between individuals explicitly;
- Incorporate formal apparatus for reasoning about the behaviour of systems;
- Stochastic extensions, such as PEPA, enable quantified behaviour of the dynamics of systems.

Process Algebra and Collective Dynamics

Process algebras are well-suited to modelling such systems

- Developed to represent concurrent behaviour compositionally;
- Capture the interactions between individuals explicitly;
- Incorporate formal apparatus for reasoning about the behaviour of systems;
- Stochastic extensions, such as PEPA, enable quantified behaviour of the dynamics of systems.

In the CODA project we are developing stochastic process algebras and associated theory, tailored to the construction and evaluation of the collective dynamics of large systems of interacting entities.

Novelty

The novelty in this project is twofold:

Novelty

The novelty in this project is twofold:

- Linking process algebra and continuous mathematical models for dynamic evaluation represents a paradigm shift in how such systems are studied.

Novelty

The novelty in this project is twofold:

- Linking process algebra and continuous mathematical models for dynamic evaluation represents a paradigm shift in how such systems are studied.
- The prospect of formally-based quantified evaluation of dynamic behaviour could have significant impact in application domains such as:
  - Large scale software systems;
  - Biochemical signalling pathways;
  - Epidemiological systems.

Outline

1. Continuous Approximation of PEPA models
   - Stochastic Process Algebra
   - Continuous Approximation
   - Numerical illustration
2. A Process Algebra for Hybrid Systems
   - HYPE definition
   - Semantics – operational, hybrid, equivalence
3. Conclusions

Outline

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

Process Algebra

- Models consist of agents which engage in actions.

$$\alpha.P$$

action type or name

agent/ component

Process Algebra

- Models consist of agents which engage in actions.

$$\alpha.P$$

action type or name

agent/ component

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

Process Algebra

- Models consist of agents which engage in actions.

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

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

Process algebra model

Process Algebra

- Models consist of agents which engage in actions.

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

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

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

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type
or name

activity rate
(parameter of an
exponential distribution)

component/
derivative

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type
or name

activity rate
(parameter of an
exponential distribution)

component/
derivative

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type
or name

activity rate
(parameter of an
exponential distribution)

component/
derivative

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

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

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

PEPA
MODEL

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

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

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

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

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

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

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

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

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

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

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

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

Performance Evaluation Process Algebra

PEPA components perform activities either independently or in co-operation with other components.

Performance Evaluation Process Algebra

PEPA components perform activities either independently or in co-operation with other components.

$(\alpha, r).P$	Prefix
$P_1 + P_2$	Choice
$P_1 \underset{L}{\bowtie} P_2$	Co-operation
P/L	Hiding
X	Variable

Performance Evaluation Process Algebra

PEPA components perform activities either independently or in co-operation with other components.

$(\alpha, r).P$	Prefix
$P_1 + P_2$	Choice
$P_1 \underset{L}{\bowtie} P_2$	Co-operation
P/L	Hiding
X	Variable

Performance Evaluation Process Algebra

PEPA components perform activities either independently or in co-operation with other components.

$(\alpha, r).P$	Prefix
$P_1 + P_2$	Choice
$P_1 \bowtie_{L} P_2$	Co-operation
P/L	Hiding
X	Variable

Performance Evaluation Process Algebra

PEPA components perform activities either independently or in co-operation with other components.

$(\alpha, r).P$	Prefix
$P_1 + P_2$	Choice
$P_1 \underset{L}{\bowtie} P_2$	Co-operation
P/L	Hiding
X	Variable

Performance Evaluation Process Algebra

PEPA components perform activities either independently or in co-operation with other components.

$(\alpha, r).P$	Prefix
$P_1 + P_2$	Choice
$P_1 \underset{L}{\bowtie} P_2$	Co-operation
P/L	Hiding
X	Variable

Performance Evaluation Process Algebra

PEPA components perform activities either independently or in co-operation with other components.

$(\alpha, r).P$	Prefix
$P_1 + P_2$	Choice
$P_1 \underset{L}{\bowtie} P_2$	Co-operation
P/L	Hiding
X	Variable

Performance Evaluation Process Algebra

PEPA components perform activities either independently or in co-operation with other components.

$(\alpha, r).P$	Prefix
$P_1 + P_2$	Choice
$P_1 \underset{L}{\bowtie} P_2$	Co-operation
P/L	Hiding
X	Variable

$P_1 \parallel P_2$ is a derived form for $P_1 \underset{\emptyset}{\bowtie} P_2$.

Performance Evaluation Process Algebra

PEPA components perform activities either independently or in co-operation with other components.

$(\alpha, r).P$	Prefix
$P_1 + P_2$	Choice
$P_1 \bowtie_{L} P_2$	Co-operation
P/L	Hiding
X	Variable

$P_1 \parallel P_2$ is a derived form for $P_1 \bowtie_{\emptyset} P_2$.

When working with large numbers of entities, we write $P[n]$ to denote an array of n copies of P executing in parallel.

Performance Evaluation Process Algebra

PEPA components perform activities either independently or in co-operation with other components.

$(\alpha, r).P$	Prefix
$P_1 + P_2$	Choice
$P_1 \underset{L}{\bowtie} P_2$	Co-operation
P/L	Hiding
X	Variable

$P_1 \parallel P_2$ is a derived form for $P_1 \underset{\emptyset}{\bowtie} P_2$.

When working with large numbers of entities, we write $P[n]$ to denote an array of n copies of P executing in parallel.

$$P[5] \equiv (P \parallel P \parallel P \parallel P \parallel P)$$

Solving discrete state models

Under the SOS semantics a PEPA model is mapped to a Continuous Time Markov Chain (CTMC) with global states determined by the local states of all the participating components.

Solving discrete state models

Under the SOS semantics a PEPA model is mapped to a Continuous Time Markov Chain (CTMC) with global states determined by the local states of all the participating components.

When the size of the state space is not too large they are amenable to numerical solution (linear algebra) to determine a steady state or transient probability distribution.

Solving discrete state models

Under the SOS semantics a PEPA model is mapped to a Continuous Time Markov Chain (CTMC) with global states determined by the local states of all the participating components.

When the size of the state space is not too large they are amenable to numerical solution (linear algebra) to determine a steady state or transient probability distribution.

Alternatively they may be studied using stochastic simulation. Each run generates a single trajectory through the state space. Many runs are needed in order to obtain average behaviours.

Solving discrete state models

Under the SOS semantics a PEPA model is mapped to a Continuous Time Markov Chain (CTMC) with global states determined by the local states of all the participating components.

When the size of the state space is not too large they are amenable to numerical solution (linear algebra) to determine a steady state or transient probability distribution.

Alternatively they may be studied using stochastic simulation. Each run generates a single trajectory through the state space. Many runs are needed in order to obtain average behaviours.

As the size of the state space becomes large it becomes infeasible to carry out numerical solution and extremely time-consuming to conduct stochastic simulation.

Continuous Approximation

The major limitation of the CTMC approach is the state space explosion problem.

Continuous Approximation

The major limitation of the CTMC approach is the state space explosion problem.

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

Continuous Approximation

The major limitation of the CTMC approach is the state space explosion problem.

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

Instead we can use continuous state variables to approximate the discrete state space.

Continuous Approximation

The major limitation of the CTMC approach is the state space explosion problem.

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

Instead we can use continuous state variables to approximate the discrete state space.

Continuous Approximation

The major limitation of the CTMC approach is the state space explosion problem.

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

Instead we can use continuous state variables to approximate the discrete state space.

Continuous Approximation

The major limitation of the CTMC approach is the state space explosion problem.

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

Instead we can use continuous state variables to approximate the discrete state space.

Continuous Approximation

The major limitation of the CTMC approach is the state space explosion problem.

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

Instead we can use continuous state variables to approximate the discrete state space.

Continuous Approximation

The major limitation of the CTMC approach is the state space explosion problem.

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

Instead we can use continuous state variables to approximate the discrete state space.

Continuous Approximation

The major limitation of the CTMC approach is the state space explosion problem.

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

Instead we can use continuous state variables to approximate the discrete state space.

Continuous Approximation

The major limitation of the CTMC approach is the state space explosion problem.

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

Instead we can use continuous state variables to approximate the discrete state space.

Continuous Approximation

The major limitation of the CTMC approach is the state space explosion problem.

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

Instead we can use continuous state variables to approximate the discrete state space.

Continuous Approximation

The major limitation of the CTMC approach is the state space explosion problem.

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

Instead we can use continuous state variables to approximate the discrete state space.

Continuous Approximation

The major limitation of the CTMC approach is the state space explosion problem.

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

Instead we can use continuous state variables to approximate the discrete state space.

Continuous Approximation

The major limitation of the CTMC approach is the state space explosion problem.

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

Instead we can use continuous state variables to approximate the discrete state space.

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

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, i.e. models of populations.

Differential equations from PEPA models

Let $N(\mathcal{C}_{i_j}, t)$ denote the number of $\mathcal{C}_{i_j}$ type components at time t.

Differential equations from PEPA models

Let $N(\mathcal{C}_{i_j}, t)$ denote the number of $\mathcal{C}_{i_j}$ type components at time t.

Consider the change in a small time δt:

$$
\begin{aligned}
N(\mathcal{C}_{i_j}, t+\delta t) \;-\; N(\mathcal{C}_{i_j}, t) = \\
& - \underbrace{\sum_{(\alpha,r)\in Ex(\mathcal{C}_{i_j})} r \times \min_{\mathcal{C}_{k_l}\in Ex(\alpha,r)} (N(\mathcal{C}_{k_l}, t))\, \delta t}_{\text{exit activities}} \\
& + \underbrace{\sum_{(\alpha,r)\in En(\mathcal{C}_{i_j})} r \times \min_{\mathcal{C}_{k_l}\in Ex(\alpha,r)} (N(\mathcal{C}_{k_l}, t))\, \delta t}_{\text{entry activities}}
\end{aligned}
$$

Differential equations from PEPA models

Let $N(\mathcal{C}_{i_j}, t)$ denote the number of $\mathcal{C}_{i_j}$ type components at time t.

Consider the change in a small time δt:

$$
\begin{aligned}
N(\mathcal{C}_{i_j}, t+\delta t) \; - \; N(\mathcal{C}_{i_j}, t) = \\
&\underbrace{- \sum_{(\alpha,r)\in Ex(\mathcal{C}_{i_j})} r \times \min_{\mathcal{C}_{k_l}\in Ex(\alpha,r)} (N(\mathcal{C}_{k_l}, t))\, \delta t}_{\text{exit activities}} \\
&+ \underbrace{\sum_{(\alpha,r)\in En(\mathcal{C}_{i_j})} r \times \min_{\mathcal{C}_{k_l}\in Ex(\alpha,r)} (N(\mathcal{C}_{k_l}, t))\, \delta t}_{\text{entry activities}}
\end{aligned}
$$

Differential equations from PEPA models

Let $N(\mathcal{C}_{i_j}, t)$ denote the number of $\mathcal{C}_{i_j}$ type components at time t.

Consider the change in a small time δt:

$$
\begin{aligned}
N(\mathcal{C}_{i_j}, t+\delta t) \; - \; N(\mathcal{C}_{i_j}, t) = \\
& - \underbrace{\sum_{(\alpha,r)\in Ex(\mathcal{C}_{i_j})} r \times \min_{\mathcal{C}_{k_l}\in Ex(\alpha,r)} (N(\mathcal{C}_{k_l}, t))\, \delta t}_{\text{exit activities}} \\
& + \underbrace{\sum_{(\alpha,r)\in En(\mathcal{C}_{i_j})} r \times \min_{\mathcal{C}_{k_l}\in Ex(\alpha,r)} (N(\mathcal{C}_{k_l}, t))\, \delta t}_{\text{entry activities}}
\end{aligned}
$$

Differential equations from PEPA models

Let $N(\mathcal{C}_{i_j}, t)$ denote the number of $\mathcal{C}_{i_j}$ type components at time t.

Consider the change in a small time δt:

$$N(\mathcal{C}_{i_j}, t+\delta t) \; - \; N(\mathcal{C}_{i_j}, t) =$$

$$\underbrace{- \sum_{(\alpha,r)\in Ex(\mathcal{C}_{i_j})} r \times \min_{\mathcal{C}_{k_l}\in Ex(\alpha,r)} (N(\mathcal{C}_{k_l}, t))\, \delta t}_{\text{exit activities}}$$

$$+ \underbrace{\sum_{(\alpha,r)\in En(\mathcal{C}_{i_j})} r \times \min_{\mathcal{C}_{k_l}\in Ex(\alpha,r)} (N(\mathcal{C}_{k_l}, t))\, \delta t}_{\text{entry activities}}$$

Differential equations from PEPA models

Let $N(\mathcal{C}_{i_j}, t)$ denote the number of $\mathcal{C}_{i_j}$ type components at time t.

Consider the change in a small time δt:

$$N(\mathcal{C}_{i_j}, t+\delta t) - N(\mathcal{C}_{i_j}, t) = \underbrace{-\sum_{(\alpha,r)\in Ex(\mathcal{C}_{i_j})} r \times \min_{\mathcal{C}_{k_l}\in Ex(\alpha,r)} (N(\mathcal{C}_{k_l}, t))\,\delta t}_{\text{exit activities}} \underbrace{+\sum_{(\alpha,r)\in En(\mathcal{C}_{i_j})} r \times \min_{\mathcal{C}_{k_l}\in Ex(\alpha,r)} (N(\mathcal{C}_{k_l}, t))\,\delta t}_{\text{entry activities}}$$

Differential equations from PEPA models

Let $N(\mathcal{C}_{i_j}, t)$ denote the number of $\mathcal{C}_{i_j}$ type components at time t.

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

$$\frac{dN(C_{i_j}, t)}{dt} = - \sum_{(\alpha,r) \in Ex(\mathcal{C}_{i_j})} r \times \min_{\mathcal{C}_{k_l} \in Ex(\alpha,r)} (N(C_{k_l}, t)) + \sum_{(\alpha,r) \in En(\mathcal{C}_{i_j})} r \times \min_{\mathcal{C}_{k_l} \in Ex(\alpha,r)} (N(C_{k_l}, t))$$

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.

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

$$(\underbrace{\alpha}_{\text{action type or name}}, \underbrace{r}_{\text{activity rate}}).\underbrace{P}_{\text{component/ derivative}}$$

Stochastic 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 CTMC.

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

Stochastic 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)

Stochastic 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)

PEPA
MODEL

Stochastic 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)

PEPA MODEL $\xrightarrow[\text{analysis}]{\text{syntactic}}$

Stochastic 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)

PEPA MODEL $\xrightarrow[\text{analysis}]{\text{syntactic}}$ ACTIVITY MATRIX

Stochastic 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)

$$\text{PEPA MODEL} \xrightarrow[\text{analysis}]{\text{syntactic}} \text{ACTIVITY MATRIX} \xrightarrow[\text{interpretation}]{\text{continuous}}$$

Stochastic 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)

$$\text{PEPA MODEL} \xrightarrow[\text{analysis}]{\text{syntactic}} \text{ACTIVITY MATRIX} \xrightarrow[\text{interpretation}]{\text{continuous}} \text{ODEs}$$

A simple example: processors and resources

$$
\begin{aligned}
Proc_0 &\stackrel{def}{=} (task1, \top).Proc_1 \\
Proc_1 &\stackrel{def}{=} (task2, r_2).Proc_0 \\
Res_0 &\stackrel{def}{=} (task1, r_1).Res_1 \\
Res_1 &\stackrel{def}{=} (reset, s).Res_0
\end{aligned}
$$

$$
Proc_0[P] \underset{\{task1\}}{\bowtie} Res_0[R]
$$

A simple example: processors and resources

$$
\begin{aligned}
Proc_0 &\stackrel{def}{=} (task1, \top).Proc_1 \\
Proc_1 &\stackrel{def}{=} (task2, r_2).Proc_0 \\
Res_0 &\stackrel{def}{=} (task1, r_1).Res_1 \\
Res_1 &\stackrel{def}{=} (reset, s).Res_0
\end{aligned}
$$

$$
Proc_0[P] \underset{\{task1\}}{\bowtie} Res_0[R]
$$

CTMC interpretation

Processors (P)	Resources (R)	States (2^{P+R})
1	1	4
2	1	8
2	2	16
3	2	32
3	3	64
4	3	128
4	4	256
5	4	512
5	5	1024
6	5	2048
6	6	4096
7	6	8192
7	7	16384
8	7	32768
8	8	65536
9	8	131072
9	9	262144
10	9	524288
10	10	1048576

A simple example: processors and resources

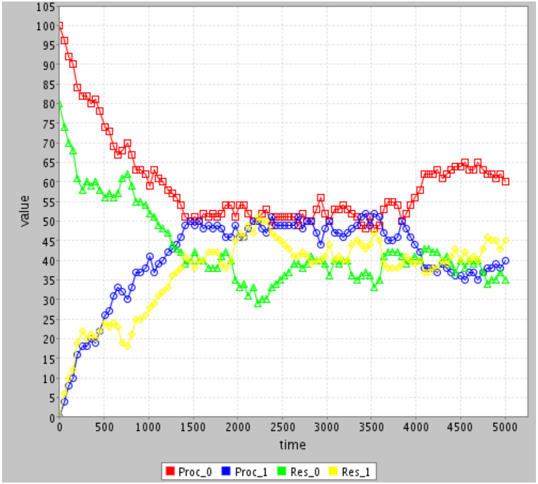
$$
\begin{aligned}
Proc_0 &\stackrel{def}{=} (task1, \top).Proc_1 \\
Proc_1 &\stackrel{def}{=} (task2, r_2).Proc_0 \\
Res_0 &\stackrel{def}{=} (task1, r_1).Res_1 \\
Res_1 &\stackrel{def}{=} (reset, s).Res_0
\end{aligned}
$$

$$Proc_0[P] \underset{\{task1\}}{\bowtie} Res_0[R]$$

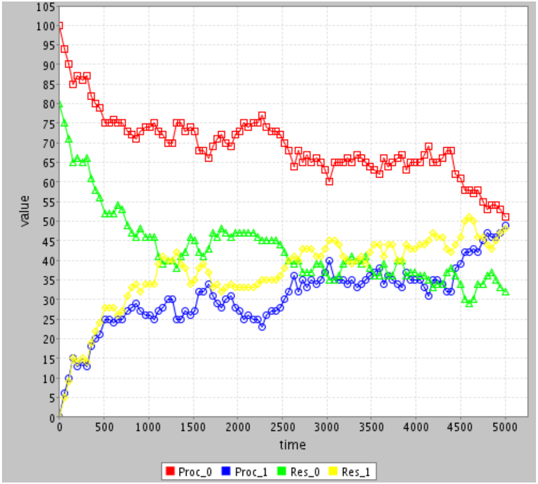
ODE interpretation

$$
\begin{aligned}
\frac{\mathrm{d}Proc_0}{\mathrm{d}t} &= -r_1 \min(Proc_0, Res_0) \\
&\quad + r_2\, Proc_1 \\
\frac{\mathrm{d}Proc_1}{\mathrm{d}t} &= r_1 \min(Proc_0, Res_0) \\
&\quad - r_2\, Proc_1 \\
\frac{\mathrm{d}Res_0}{\mathrm{d}t} &= -r_1 \min(Proc_0, Res_0) \\
&\quad + s\, Res_1 \\
\frac{\mathrm{d}Res_1}{\mathrm{d}t} &= r_1 \min(Proc_0, Res_0) \\
&\quad - s\, Res_1
\end{aligned}
$$

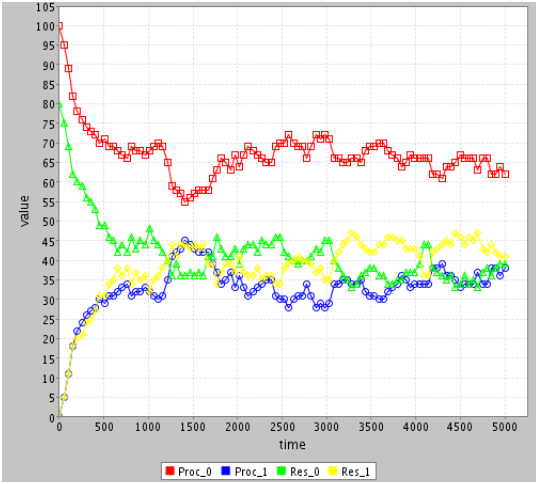
Processors and resources (simulation run A)



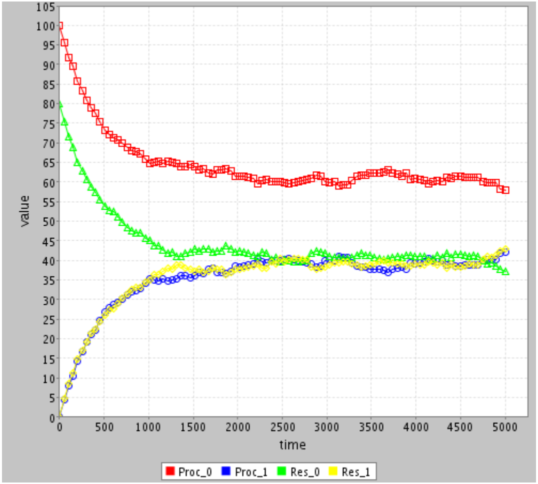
Processors and resources (simulation run B)



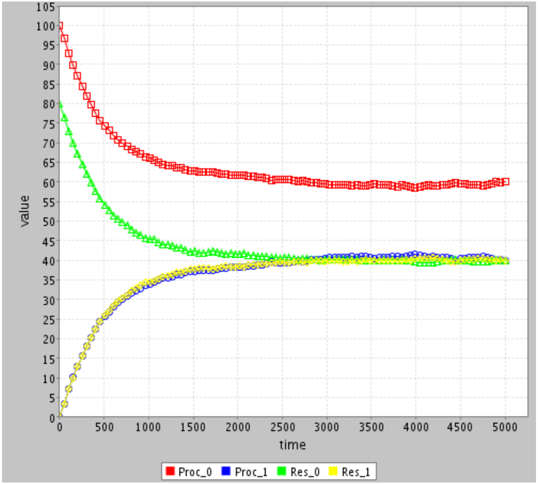
Processors and resources (simulation run C)



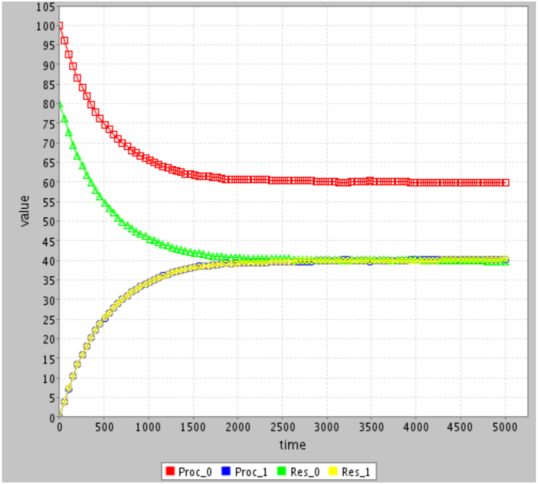
Processors and resources (average of 10 runs)



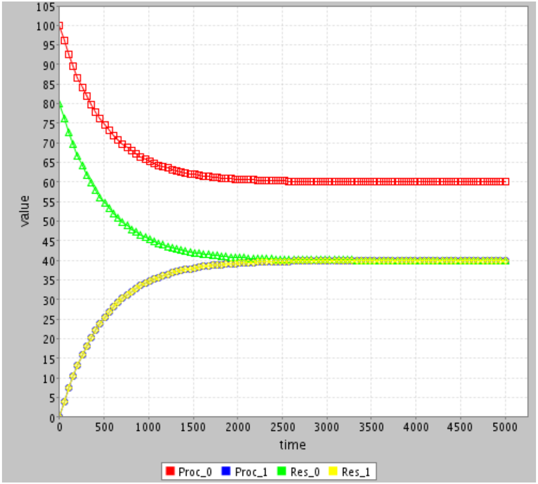
Processors and resources (average of 100 runs)



Processors and resources (average of 1000 runs)



Processors and resources (ODE solution)



Outline

1 Continuous Approximation of PEPA models
- Stochastic Process Algebra
- Continuous Approximation
- Numerical illustration

2 A Process Algebra for Hybrid Systems
- HYPE definition
- Semantics – operational, hybrid, equivalence

3 Conclusions

Introduction

Hybrid systems, combining continuous and discrete behaviour, arise in several application domains e.g. manufacturing systems, genetic networks etc.

Introduction

Hybrid systems, combining continuous and discrete behaviour, arise in several application domains e.g. manufacturing systems, genetic networks etc.

In part our motivation was consideration of PEPA models in which only some components could be legitimately subjected to continuous approximation.

Introduction

Hybrid systems, combining continuous and discrete behaviour, arise in several application domains e.g. manufacturing systems, genetic networks etc.

In part our motivation was consideration of PEPA models in which only some components could be legitimately subjected to continuous approximation.

For this and other reasons we were keen to develop an approach in which ODEs describing continuous behaviour could be derived in a fairly natural way from the process algebra model which captured both the discrete events and continuous changes.

Other formal approaches to hybrid systems

Hybrid automata are a well-established approach to modelling hybrid systems which are supported by a number of tools and analysis techniques. Their drawbacks are that they are graphical rather than textual, and the approach is not very compositional.

Other formal approaches to hybrid systems

Hybrid automata are a well-established approach to modelling hybrid systems which are supported by a number of tools and analysis techniques. Their drawbacks are that they are graphical rather than textual, and the approach is not very compositional.

There have also been a number of other process algebras for hybrid systems:

- ACP^{srt}_{hs} — Bergstra and Middelburg
- HyPA — Cuijpers and Reniers
- hybrid χ — van Beek *et al*
- ϕ-calculus — Rounds and Song

Other formal approaches to hybrid systems

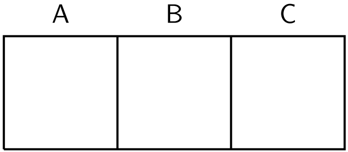
Hybrid automata are a well-established approach to modelling hybrid systems which are supported by a number of tools and analysis techniques. Their drawbacks are that they are graphical rather than textual, and the approach is not very compositional.

There have also been a number of other process algebras for hybrid systems:

- ACP^{srt}_{hs} — Bergstra and Middelburg
- HyPA — Cuijpers and Reniers
- hybrid χ — van Beek *et al*
- ϕ-calculus — Rounds and Song

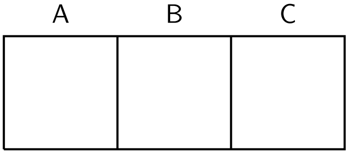
These take a coarse-grained approach, often with ODEs embedded within the syntax.

Heater example I



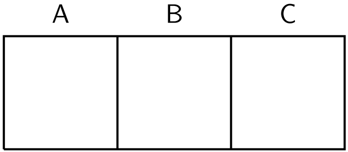
- three adjacent rooms

Heater example I



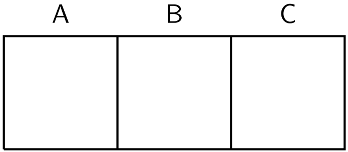
- three adjacent rooms
- fan heaters can be placed in each room

Heater example I



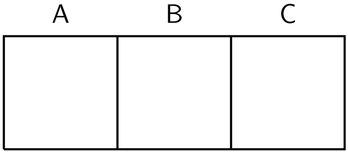
- three adjacent rooms
- fan heaters can be placed in each room
  - full effect on room

Heater example I



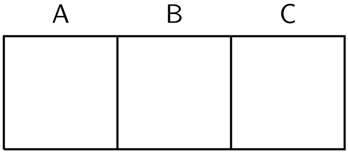
- three adjacent rooms
- fan heaters can be placed in each room
  - full effect on room
  - reduced effect on adjacent room

Heater example I



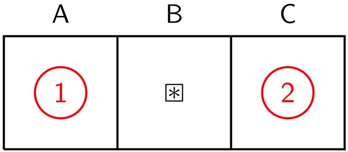
- three adjacent rooms
- fan heaters can be placed in each room
  - full effect on room
  - reduced effect on adjacent room
  - no affect on non-adjacent room

Heater example I



- three adjacent rooms
- fan heaters can be placed in each room
  - full effect on room
  - reduced effect on adjacent room
  - no affect on non-adjacent room
- heaters can be switched on and off, no thermostat

Heater example I



- three adjacent rooms
- fan heaters can be placed in each room
  - full effect on room
  - reduced effect on adjacent room
  - no affect on non-adjacent room
- heaters can be switched on and off, no thermostat
- How does the temperature in Room B change if there is one heater in Room A and one in Room C?

HYPE definition

We distinguish two types of actions in a system:

HYPE definition

We distinguish two types of actions in a system:

- events — instantaneous, discrete changes

$$\underline{\mathsf{a}} \in \mathcal{E}$$

Each event is associated with an event condition: activation conditions and variable resets.

HYPE definition

We distinguish two types of actions in a system:

- events — instantaneous, discrete changes

$$\underline{\mathsf{a}} \in \mathcal{E}$$

  Each event is associated with an event condition: activation conditions and variable resets.

- activities — influences on a continuous aspect of system, sometimes termed flows

$$\alpha \in \mathcal{A} \qquad \alpha(\vec{X}) = (\iota, r, I(\vec{X}))$$

influence name   rate   influence type with $[\![I(\vec{X})]\!] = f(\vec{X})$

where $\vec{X}$ is a formal parameter.

HYPE definition

We distinguish two types of actions in a system:

- events — instantaneous, discrete changes

$$\underline{\mathrm{a}} \in \mathcal{E}$$

  Each event is associated with an event condition: activation conditions and variable resets.

- activities — influences on a continuous aspect of system, sometimes termed flows

$$\alpha \in \mathcal{A} \qquad \alpha(\vec{X}) = (\iota, r, I(\vec{X}))$$

influence name rate influence type with $[\![I(\vec{X})]\!] = f(\vec{X})$

where $\vec{X}$ is a formal parameter.

HYPE definition

We distinguish two types of actions in a system:

- events — instantaneous, discrete changes

$$\underline{\mathrm{a}} \in \mathcal{E}$$

  Each event is associated with an event condition: activation conditions and variable resets.

- activities — influences on a continuous aspect of system, sometimes termed flows

$$\alpha \in \mathcal{A} \qquad \alpha(\vec{X}) = (\iota, r, I(\vec{X}))$$

influence name

rate

influence type with $[\![I(\vec{X})]\!] = f(\vec{X})$

where $\vec{X}$ is a formal parameter.

HYPE definition

We distinguish two types of actions in a system:

- events — instantaneous, discrete changes

$$\underline{\mathsf{a}} \in \mathcal{E}$$

Each event is associated with an event condition: activation conditions and variable resets.

- activities — influences on a continuous aspect of system, sometimes termed flows

$$\alpha \in \mathcal{A} \qquad \alpha(\vec{X}) = (\iota, r, I(\vec{X}))$$

influence name   rate   influence type with $[\![I(\vec{X})]\!] = f(\vec{X})$

where $\vec{X}$ is a formal parameter.

HYPE definition (cont.)

- subcomponents

$$S ::= \underline{\mathsf{a}} : \alpha . C_s \mid S + S \qquad \underline{\mathsf{a}} \in \mathcal{E}, \alpha \in \mathcal{A}$$

  - subcomponent names: $C_s(\vec{X}) \stackrel{def}{=} S$

HYPE definition (cont.)

- subcomponents

$$S ::= \underline{\mathrm{a}} : \alpha . C_s \mid S + S \qquad \underline{\mathrm{a}} \in \mathcal{E}, \alpha \in \mathcal{A}$$

  - subcomponent names: $C_s(\vec{X}) \stackrel{\mathit{def}}{=} S$

- components

$$P ::= C(\vec{X}) \mid P \underset{L}{\bowtie} P \qquad L \subseteq \mathcal{E}$$

  - component names: $C(\vec{X}) \stackrel{\mathit{def}}{=} P$ or subcomponent name

HYPE definition (cont.)

- subcomponents

$$S ::= \underline{\mathrm{a}} : \alpha . C_s \mid S + S \qquad \underline{\mathrm{a}} \in \mathcal{E}, \alpha \in \mathcal{A}$$

  - subcomponent names: $C_s(\vec{X}) \stackrel{\text{def}}{=} S$

- components

$$P ::= C(\vec{X}) \mid P \underset{L}{\bowtie} P \qquad L \subseteq \mathcal{E}$$

  - component names: $C(\vec{X}) \stackrel{\text{def}}{=} P$ or subcomponent name

- uncontrolled system

$$\Sigma ::= C(\vec{V}) \mid \Sigma \underset{L}{\bowtie} \Sigma \qquad L \subseteq \mathcal{E}$$

Heater example II

- room: $Room_x(T) \stackrel{def}{=} \underline{\mathrm{init}}:(t_{0,x}, -1, linear(T)).Room_x(T)$

Heater example II

- room: $Room_x(T) \stackrel{def}{=} \underline{\text{init}} : (t_{0,x}, -1, linear(T)).Room_x(T)$
- fan i in Room x affecting Room y:

$$\begin{aligned} Fan_{i,x,y} &\stackrel{def}{=} \underline{\text{init}} : (t_{i,y}, 0, const).Fan_{i,x,y} + \\ &\quad \underline{\text{off}}_i : (t_{i,y}, 0, const).Fan_{i,x,y} + \\ &\quad \underline{\text{on}}_i : (t_{i,y}, r_i, const_{\psi(x,y)}).Fan_{i,x,y} \\ \psi(x,y) &= \begin{cases} in & \text{if } x = y \\ adj & \text{if } x \text{ and } y \text{ are adjacent} \\ far & \text{otherwise} \end{cases} \end{aligned}$$

Heater example II

- room: $Room_x(T) \stackrel{def}{=} \underline{\text{init}} : (t_{0,x}, -1, linear(T)).Room_x(T)$
- fan i in Room x affecting Room y:

$$
\begin{aligned}
Fan_{i,x,y} &\stackrel{def}{=} \underline{\text{init}} : (t_{i,y}, 0, const).Fan_{i,x,y} + \\
&\quad \underline{\text{off}}_i : (t_{i,y}, 0, const).Fan_{i,x,y} + \\
&\quad \underline{\text{on}}_i : (t_{i,y}, r_i, const_{\psi(x,y)}).Fan_{i,x,y} \\
\psi(x,y) &= \begin{cases} in & \text{if } x = y \\ adj & \text{if } x \text{ and } y \text{ are adjacent} \\ far & \text{otherwise} \end{cases}
\end{aligned}
$$

$t_{i,y}$ represents influence of fan i on Room y

Heater example II

- room: $Room_x(T) \stackrel{def}{=} \underline{\text{init}}:(t_{0,x}, -1, linear(T)).Room_x(T)$
- fan i in Room x affecting Room y:

$$
\begin{aligned}
Fan_{i,x,y} & \stackrel{def}{=} & \underline{\text{init}}:(t_{i,y}, 0, const).Fan_{i,x,y} + \\
& & \underline{\text{off}}_i:(t_{i,y}, 0, const).Fan_{i,x,y} + \\
& & \underline{\text{on}}_i:(t_{i,y}, r_i, const_{\psi(x,y)}).Fan_{i,x,y} \\
\psi(x,y) & = & \begin{cases} in & \text{if } x = y \\ adj & \text{if } x \text{ and } y \text{ are adjacent} \\ far & \text{otherwise} \end{cases}
\end{aligned}
$$

$t_{i,y}$ represents influence of fan i on Room y

- uncontrolled system:

$$Sys \stackrel{def}{=} (Fan_{1,A,B} \underset{\{\underline{\text{init}}\}}{\bowtie} Fan_{2,C,B}) \underset{\{\underline{\text{init}}\}}{\bowtie} Room_B(T_B)$$

HYPE definition (cont.)

We assume, additionally, that the system may be subject to a controller which can impose orderings on events:

$$M ::= \underline{\mathrm{a}}.M \mid 0 \mid M + M \qquad \underline{\mathrm{a}} \in \mathcal{E}$$

$$Con ::= M \mid Con \underset{L}{\bowtie} Con \qquad L \subseteq \mathcal{E}$$

HYPE definition (cont.)

We assume, additionally, that the system may be subject to a controller which can impose orderings on events:

$$M ::= \underline{\mathrm{a}}.M \mid 0 \mid M + M \qquad \underline{\mathrm{a}} \in \mathcal{E}$$

$$Con ::= M \mid Con \underset{L}{\bowtie} Con \qquad L \subseteq \mathcal{E}$$

When the controller is composed with the uncontrolled system we obtain a controlled system:

$$ConSys ::= \Sigma \underset{L}{\bowtie} \underline{\mathrm{init}}.Con \qquad L \subseteq \mathcal{E}$$

Heater example II

- controller:

$$Con \stackrel{def}{=} Con_1 \underset{\emptyset}{\bowtie} Con_2 \qquad Con_i \stackrel{def}{=} \underline{\mathrm{on}}_i.\underline{\mathrm{off}}_i.Con_i$$

Heater example II

- controller:

$$Con \stackrel{def}{=} Con_1 \underset{\emptyset}{\bowtie} Con_2 \qquad Con_i \stackrel{def}{=} \underline{\mathrm{on}}_i.\underline{\mathrm{off}}_i.Con_i$$

- controlled system:

$$MF \stackrel{def}{=} Sys \underset{K}{\bowtie} \underline{\mathrm{init}}.Con \qquad K = \{\underline{\mathrm{init}}, \underline{\mathrm{on}}_1, \underline{\mathrm{off}}_1, \underline{\mathrm{on}}_2, \underline{\mathrm{off}}_2\}$$

Heater example II

- controller:

$$Con \stackrel{\mathit{def}}{=} Con_1 \bowtie_{\emptyset} Con_2 \qquad Con_i \stackrel{\mathit{def}}{=} \underline{\mathrm{on}}_i.\underline{\mathrm{off}}_i.Con_i$$

- controlled system:

$$MF \stackrel{\mathit{def}}{=} Sys \bowtie_{K} \underline{\mathrm{init}}.Con \qquad K = \{\underline{\mathrm{init}}, \underline{\mathrm{on}}_1, \underline{\mathrm{off}}_1, \underline{\mathrm{on}}_2, \underline{\mathrm{off}}_2\}$$

- variable: $\mathcal{V} = \{T_B\}$ with $\mathrm{iv}(t_{i,B}) = T_B$

Heater example II

- controller:

$$Con \stackrel{\text{def}}{=} Con_1 \underset{\emptyset}{\bowtie} Con_2 \qquad Con_i \stackrel{\text{def}}{=} \underline{\mathsf{on}}_i.\underline{\mathsf{off}}_i.Con_i$$

- controlled system:

$$MF \stackrel{\text{def}}{=} Sys \underset{K}{\bowtie} \underline{\mathsf{init}}.Con \qquad K = \{\underline{\mathsf{init}}, \underline{\mathsf{on}}_1, \underline{\mathsf{off}}_1, \underline{\mathsf{on}}_2, \underline{\mathsf{off}}_2\}$$

- variable: $\mathcal{V} = \{T_B\}$ with $\mathrm{iv}(t_{i,B}) = T_B$
- event conditions:
  $\mathrm{ec}(\underline{\mathsf{init}}) = (true, (T_B' = T_0))$, $\mathrm{ec}(\underline{\mathsf{off}}_i) = (\bot, (T_B' = T_B))$,
  $\mathrm{ec}(\underline{\mathsf{on}}_i) = (\bot, true)$

Heater example II

- controller:

$$Con \stackrel{def}{=} Con_1 \underset{\emptyset}{\bowtie} Con_2 \qquad Con_i \stackrel{def}{=} \underline{\mathsf{on}}_i.\underline{\mathsf{off}}_i.Con_i$$

- controlled system:

$$MF \stackrel{def}{=} Sys \underset{K}{\bowtie} \underline{\mathsf{init}}.Con \qquad K = \{\underline{\mathsf{init}}, \underline{\mathsf{on}}_1, \underline{\mathsf{off}}_1, \underline{\mathsf{on}}_2, \underline{\mathsf{off}}_2\}$$

- variable: $\mathcal{V} = \{T_B\}$ with $\mathrm{iv}(t_{i,B}) = T_B$
- event conditions:
  $\mathrm{ec}(\underline{\mathsf{init}}) = (true, (T'_B = T_0))$, $\mathrm{ec}(\underline{\mathsf{off}}_i) = (\perp, (T'_B = T_B))$,
  $\mathrm{ec}(\underline{\mathsf{on}}_i) = (\perp, true)$
- influence definitions:
  $[\![const_{in}]\!] = 1$, $[\![const_{adj}]\!] = 0.5$, $[\![const_{far}]\!] = [\![const]\!] = 0$,
  $[\![linear(X)]\!] = X$

HYPE definition (cont.)

HYPE model

$$(\mathit{ConSys}, \mathcal{V}, \mathcal{X}, \mathit{IN}, \mathit{IT}, \mathcal{E}, \mathcal{A}, \mathrm{ec}, \mathrm{iv}, \mathit{EC}, \mathit{ID})$$

- event conditions

$$\mathrm{ec} : \mathcal{E} \rightarrow \mathit{EC}$$

EC consists of activation conditions and resets associated with events

HYPE definition (cont.)

HYPE model

$$(\mathit{ConSys}, \mathcal{V}, \mathcal{X}, \mathit{IN}, \mathit{IT}, \mathcal{E}, \mathcal{A}, \mathrm{ec}, \mathrm{iv}, \mathit{EC}, \mathit{ID})$$

- event conditions

$$\mathrm{ec} : \mathcal{E} \rightarrow \mathit{EC}$$

  EC consists of activation conditions and resets associated with events

- influences and variables

$$\mathrm{iv} : \mathit{IN} \rightarrow \mathcal{V}$$

  each influence is associated with one variable

Well-defined HYPE model

We impose a number of (mild) conditions on the formation of a HYPE model in order for it to be considered well-defined:

- for each subcomponent $\mathcal{C}_s(\vec{X}) \stackrel{def}{=} S$, only $\mathcal{C}_s(\vec{X})$ can appear in S, $\underline{a}$ can only appear once and each ι must also appear in a prefix with $\underline{init}$.

Well-defined HYPE model

We impose a number of (mild) conditions on the formation of a HYPE model in order for it to be considered well-defined:

- for each subcomponent $\mathcal{C}_s(\vec{X}) \stackrel{\mathit{def}}{=} S$, only $\mathcal{C}_s(\vec{X})$ can appear in S, $\underline{\mathsf{a}}$ can only appear once and each ι must also appear in a prefix with $\underline{\mathsf{init}}$.
- each pair $\underline{\mathsf{a}}$ and ι can appear at most once together in a prefix.

Well-defined HYPE model

We impose a number of (mild) conditions on the formation of a HYPE model in order for it to be considered well-defined:

- for each subcomponent $\mathcal{C}_s(\vec{X}) \stackrel{def}{=} S$, only $\mathcal{C}_s(\vec{X})$ can appear in S, $\underline{a}$ can only appear once and each ι must also appear in a prefix with $\underline{init}$.
- each pair $\underline{a}$ and ι can appear at most once together in a prefix.
- in any component, any event in more than one subcomponent must be synchronised on.

Well-defined HYPE model

We impose a number of (mild) conditions on the formation of a HYPE model in order for it to be considered well-defined:

- for each subcomponent $\mathcal{C}_s(\vec{X}) \stackrel{def}{=} S$, only $\mathcal{C}_s(\vec{X})$ can appear in S, $\underline{a}$ can only appear once and each ι must also appear in a prefix with $\underline{\mathrm{init}}$.
- each pair $\underline{a}$ and ι can appear at most once together in a prefix.
- in any component, any event in more than one subcomponent must be synchronised on.
- in *ConSys*, Σ and *Con* must have the same events and these must be synchronised on.

Well-defined HYPE model

We impose a number of (mild) conditions on the formation of a HYPE model in order for it to be considered well-defined:

- for each subcomponent $\mathcal{C}_s(\vec{X}) \stackrel{def}{=} S$, only $\mathcal{C}_s(\vec{X})$ can appear in S, $\underline{a}$ can only appear once and each ι must also appear in a prefix with $\underline{init}$.
- each pair $\underline{a}$ and ι can appear at most once together in a prefix.
- in any component, any event in more than one subcomponent must be synchronised on.
- in *ConSys*, Σ and *Con* must have the same events and these must be synchronised on.

The heater example is well-defined.

Operational semantics

We define the state of the system to be the set of currently acting influences: $\sigma : IN \rightarrow (\mathbb{R}^{+} \times IT)$ and for convenience, we write it as a list of triples $(\iota, r, I(\vec{X}))$

Operational semantics

We define the state of the system to be the set of currently acting influences: $\sigma : IN \rightarrow (\mathbb{R}^+ \times IT)$ and for convenience, we write it as a list of triples $(\iota, r, I(\vec{X}))$

A configuration is then a controlled system together with its state: $\langle ConSys, \sigma \rangle$

Operational semantics

We define the state of the system to be the set of currently acting influences: $\sigma : IN \rightarrow (\mathbb{R}^+ \times IT)$ and for convenience, we write it as a list of triples $(\iota, r, I(\vec{X}))$

A configuration is then a controlled system together with its state: $\langle ConSys, \sigma \rangle$

We define the semantics of the system as a set of operational rules which tell us how states evolve through actions, and give rise to a labelled transition system: $(\mathcal{F}, \mathcal{E}, \rightarrow \subseteq \mathcal{F} \times \mathcal{E} \times \mathcal{F})$

Auxilliary functions

We require two auxilliary functions to define the rules:

- updating function: $\sigma[\iota \mapsto (r, I)]$

$$\sigma[\iota \mapsto (r, I)](x) = \begin{cases} (r, I) & \text{if } x = \iota \\ \sigma(x) & \text{otherwise} \end{cases}$$

Auxilliary functions

We require two auxilliary functions to define the rules:

- updating function: $\sigma[\iota \mapsto (r, I)]$

$$\sigma[\iota \mapsto (r, I)](x) = \begin{cases} (r, I) & \text{if } x = \iota \\ \sigma(x) & \text{otherwise} \end{cases}$$

- change identifying function: $\Gamma : \mathcal{S} \times \mathcal{S} \times \mathcal{S} \to \mathcal{S}$

$$(\Gamma(\sigma, \tau, \tau'))(\iota) = \begin{cases} \tau(\iota) & \text{if } \sigma(\iota) = \tau'(\iota), \\ \tau'(\iota) & \text{if } \sigma(\iota) = \tau(\iota), \\ \text{undefined} & \text{otherwise} \end{cases}$$

Operational semantics (cont.)

Prefix with influence:

$$\frac{}{\langle \underline{a}:(\iota, r, I).E, \sigma\rangle \xrightarrow{\underline{a}} \langle E, \sigma[\iota \mapsto (r, I)]\rangle}$$

Prefix without influence:

$$\frac{}{\langle \underline{a}.E, \sigma\rangle \xrightarrow{\underline{a}} \langle E, \sigma\rangle}$$

Choice:

$$\frac{\langle E, \sigma\rangle \xrightarrow{\underline{a}} \langle E', \sigma'\rangle}{\langle E + F, \sigma\rangle \xrightarrow{\underline{a}} \langle E', \sigma'\rangle} \qquad \frac{\langle F, \sigma\rangle \xrightarrow{\underline{a}} \langle F', \sigma'\rangle}{\langle E + F, \sigma\rangle \xrightarrow{\underline{a}} \langle F', \sigma'\rangle}$$

Constant:

$$\frac{\langle E, \sigma\rangle \xrightarrow{\underline{a}} \langle E', \sigma'\rangle}{\langle A, \sigma\rangle \xrightarrow{\underline{a}} \langle E', \sigma'\rangle}(A \stackrel{def}{=} E)$$

Operational semantics (cont.)

Parallel without synchronisation:

$$\frac{\langle E, \sigma \rangle \xrightarrow{\underline{a}} \langle E', \sigma' \rangle}{\langle E \underset{K}{\bowtie} F, \sigma \rangle \xrightarrow{\underline{a}} \langle E' \underset{K}{\bowtie} F, \sigma' \rangle} \quad \underline{a} \notin K$$

$$\frac{\langle F, \sigma \rangle \xrightarrow{\underline{a}} \langle F', \sigma' \rangle}{\langle E \underset{K}{\bowtie} F, \sigma \rangle \xrightarrow{\underline{a}} \langle E \underset{K}{\bowtie} F', \sigma' \rangle} \quad \underline{a} \notin K$$

Parallel with synchronisation:

$$\frac{\langle E, \sigma \rangle \xrightarrow{\underline{a}} \langle E', \tau \rangle \quad \langle F, \sigma \rangle \xrightarrow{\underline{a}} \langle F', \tau' \rangle}{\langle E \underset{K}{\bowtie} F, \sigma \rangle \xrightarrow{\underline{a}} \langle E' \underset{K}{\bowtie} F', \Gamma(\sigma, \tau, \tau') \rangle}$$

$$\underline{a} \in K, \ \Gamma \text{ defined}$$

Heater example III

- transition derivation

$$\frac{\langle Fan_{1,A,B}, \tau \rangle \xrightarrow{\text{init}} \langle Fan_{1,A,B}, \tau_1 \rangle \qquad \langle Fan_{2,C,B}, \tau \rangle \xrightarrow{\text{init}} \langle Fan_{2,C,B}, \tau_2 \rangle}{\langle Fan_{1,A,B} \underset{\underline{\text{init}}}{\bowtie} Fan_{2,C,B}, \tau \rangle \xrightarrow{\text{init}} \langle Fan_{1,A,B} \underset{\underline{\text{init}}}{\bowtie} F_{2,C,B}, \tau_3 \rangle}$$

$$\begin{aligned} \tau &= \{t_{0,B} \mapsto *, t_{1,B} \mapsto *, t_{2,B} \mapsto *\} \\ \tau_1 &= \tau[t_{1,B} \mapsto (0, c)] = \{t_{0,B} \mapsto *, t_{1,B} \mapsto (0, c), t_{2,B} \mapsto *\} \\ \tau_2 &= \tau[t_{2,B} \mapsto (0, c)] = \{t_{0,B} \mapsto *, t_{1,B} \mapsto *, t_{2,B} \mapsto (0, c)\} \\ \tau_3 &= \Gamma(\tau, \tau_1, \tau_2) = \{t_{0,B} \mapsto *, t_{1,B} \mapsto (0, c), t_{2,B} \mapsto (0, c)\} \end{aligned}$$

Heater example III (cont.)

- labelled transition system

$$
\begin{tikzcd}
& \langle Sys \underset{K}{\bowtie} \underline{\mathrm{init}}.Con, \_ \rangle \arrow[d, "\underline{\mathrm{init}}"'] & \\
& \langle Sys \underset{K}{\bowtie} (Con_1 \parallel Con_2), \sigma_0 \rangle \arrow[dl, shift right, "\underline{\mathrm{on}}_1"'] \arrow[dr, shift left, "\underline{\mathrm{on}}_2"] & \\
\langle Sys \underset{K}{\bowtie} (\underline{\mathrm{off}}_1.Con_1 \parallel Con_2), \sigma_1 \rangle \arrow[ur, shift right, "\underline{\mathrm{off}}_1"'] \arrow[dr, shift right, "\underline{\mathrm{on}}_2"'] & & \langle Sys \underset{K}{\bowtie} (Con_1 \parallel \underline{\mathrm{off}}_2.Con_2), \sigma_2 \rangle \arrow[ul, shift left, "\underline{\mathrm{off}}_2"] \arrow[dl, shift left, "\underline{\mathrm{on}}_1"] \\
& \langle Sys \underset{K}{\bowtie} (\underline{\mathrm{off}}_1.Con_1 \parallel \underline{\mathrm{off}}_2.Con_2), \sigma_3 \rangle \arrow[ul, shift right, "\underline{\mathrm{off}}_2"'] \arrow[ur, shift left, "\underline{\mathrm{off}}_1"] &
\end{tikzcd}
$$

Heater example III (cont.)

- for MF, there are four states

$$\sigma_0 = \{t_{0,B} \mapsto (-1, linear(T_B)), t_{1,B} \mapsto (0, const), t_{2,B} \mapsto (0, const)\}$$

$$\sigma_1 = \{t_{0,B} \mapsto (-1, linear(T_B)), t_{1,B} \mapsto (r_1, const_{adj}), t_{2,B} \mapsto (0, const)\}$$

$$\sigma_2 = \{t_{0,B} \mapsto (-1, linear(T_B)), t_{1,B} \mapsto (0, const), t_{2,B} \mapsto (r_2, const_{adj})\}$$

$$\sigma_3 = \{t_{0,B} \mapsto (-1, linear(T_B)), t_{1,B} \mapsto (r_1, const_{adj}), t_{2,B} \mapsto (r_2, const_{adj})\}$$

Hybrid semantics

- extract ODEs from each state σ in the labelled transition system

$$CS_\sigma = \left\{ \text{ODE for variable } V \;\middle|\; V \in \mathcal{V} \right\} \text{ where}$$

$$\frac{dV}{dt} = \sum \left\{ r[\![I(\vec{W})]\!] \mid \mathrm{iv}(\iota) = V \text{ and } \sigma(\iota) = (r, I(\vec{W})) \right\}$$

Hybrid semantics

- extract ODEs from each state σ in the labelled transition system

$$CS_\sigma = \left\{ \text{ODE for variable } V \;\middle|\; V \in \mathcal{V} \right\} \text{ where}$$

$$\frac{dV}{dt} = \sum \left\{ r[\![I(\vec{W})]\!] \;\middle|\; \mathrm{iv}(\iota) = V \text{ and } \sigma(\iota) = (r, I(\vec{W})) \right\}$$

  - for any influence name associated with V

Hybrid semantics

- extract ODEs from each state σ in the labelled transition system

$$CS_\sigma = \left\{ \text{ODE for variable } V \;\middle|\; V \in \mathcal{V} \right\} \text{ where}$$

$$\frac{dV}{dt} = \sum \left\{ r[\![I(\vec{W})]\!] \;\middle|\; \mathrm{iv}(\iota) = V \text{ and } \sigma(\iota) = (r, I(\vec{W})) \right\}$$

  - for any influence name associated with V
  - determine from σ its rate and influence type

Hybrid semantics

- extract ODEs from each state σ in the labelled transition system

$$CS_\sigma = \left\{ \text{ODE for variable } V \;\middle|\; V \in \mathcal{V} \right\} \text{ where}$$

$$\frac{dV}{dt} = \sum \left\{ r[\![I(\vec{W})]\!] \mid \mathrm{iv}(\iota) = V \text{ and } \sigma(\iota) = (r, I(\vec{W})) \right\}$$

  - for any influence name associated with V
  - determine from σ its rate and influence type
  - multiply its rate and influence function together

Hybrid semantics

- extract ODEs from each state σ in the labelled transition system

$$CS_\sigma = \left\{ \text{ODE for variable } V \;\middle|\; V \in \mathcal{V} \right\} \text{ where}$$

$$\frac{dV}{dt} = \sum \left\{ r[\![I(\vec{W})]\!] \;\middle|\; \mathrm{iv}(\iota) = V \text{ and } \sigma(\iota) = (r, I(\vec{W})) \right\}$$

  - for any influence name associated with V
  - determine from σ its rate and influence type
  - multiply its rate and influence function together
  - sum these over all associated influence names

Heater example IV

- state σ_0 occurs when both fans are off

$$MF_{\sigma_0} = \left\{ \frac{dT_B}{dt} = -T_B \right\}$$

Heater example IV

- state σ_0 occurs when both fans are off

$$MF_{\sigma_0} = \left\{ \frac{dT_B}{dt} = -T_B \right\}$$

- state σ_1 occurs when fan 1 is on

$$MF_{\sigma_1} = \left\{ \frac{dT_B}{dt} = -T_B + 0.5 r_1 \right\}$$

Heater example IV

- state σ_0 occurs when both fans are off

$$MF_{\sigma_0} = \left\{ \frac{dT_B}{dt} = -T_B \right\}$$

- state σ_1 occurs when fan 1 is on

$$MF_{\sigma_1} = \left\{ \frac{dT_B}{dt} = -T_B + 0.5r_1 \right\}$$

- state σ_2 occurs when fan 2 is on

$$MF_{\sigma_2} = \left\{ \frac{dT_B}{dt} = -T_B + 0.5r_2 \right\}$$

Heater example IV

- state σ_0 occurs when both fans are off

$$MF_{\sigma_0} = \left\{ \frac{dT_B}{dt} = -T_B \right\}$$

- state σ_1 occurs when fan 1 is on

$$MF_{\sigma_1} = \left\{ \frac{dT_B}{dt} = -T_B + 0.5r_1 \right\}$$

- state σ_2 occurs when fan 2 is on

$$MF_{\sigma_2} = \left\{ \frac{dT_B}{dt} = -T_B + 0.5r_2 \right\}$$

- state σ_3 occurs when both fans are on

$$MF_{\sigma_3} = \left\{ \frac{dT_B}{dt} = -T_B + 0.5(r_1 + r_2) \right\}$$

Hybrid automata

- $(V, E, \mathbf{X}, \mathcal{E}, \mathit{flow}, \mathit{init}, \mathit{inv}, \mathit{event}, \mathit{jump}, \mathit{reset}, \mathit{urgent})$

Hybrid automata

- $(V, E, \mathbf{X}, \mathcal{E}, \mathit{flow}, \mathit{init}, \mathit{inv}, \mathit{event}, \mathit{jump}, \mathit{reset}, \mathit{urgent})$
- $\mathbf{X} = \{X_1, \ldots, X_n\}$, $\dot{X}_j$, X'_j

Hybrid automata

- $(V, E, \mathbf{X}, \mathcal{E}, \mathit{flow}, \mathit{init}, \mathit{inv}, \mathit{event}, \mathit{jump}, \mathit{reset}, \mathit{urgent})$
- $\mathbf{X} = \{X_1, \ldots, X_n\}$, $\dot{X}_j$, X_j'
- control graph: $G = (V, E)$

Hybrid automata

- $(V, E, \mathbf{X}, \mathcal{E}, \mathit{flow}, \mathit{init}, \mathit{inv}, \mathit{event}, \mathit{jump}, \mathit{reset}, \mathit{urgent})$
- $\mathbf{X} = \{X_1, \ldots, X_n\}$, $\dot{X}_j$, X'_j
- control graph: $G = (V, E)$
- (control) modes: $v \in V$
  - associated ODEs: $\dot{\mathbf{X}} = \mathit{flow}(v)$
  - initial conditions: $\mathit{init}(v)$
  - invariants: $\mathit{inv}(v)$

Hybrid automata

- $(V, E, \mathbf{X}, \mathcal{E}, \mathit{flow}, \mathit{init}, \mathit{inv}, \mathit{event}, \mathit{jump}, \mathit{reset}, \mathit{urgent})$
- $\mathbf{X} = \{X_1, \ldots, X_n\}$, $\dot{X}_j$, X'_j
- control graph: $G = (V, E)$
- (control) modes: $v \in V$
    - associated ODEs: $\dot{\mathbf{X}} = \mathit{flow}(v)$
    - initial conditions: $\mathit{init}(v)$
    - invariants: $\mathit{inv}(v)$
- (control) switches: $e \in E$
    - events: $\mathit{event}(e) \in \mathcal{E}$
    - predicate on $\mathbf{X}$: $\mathit{jump}(e)$
    - predicate on $\mathbf{X} \cup \mathbf{X}'$: $\mathit{reset}(e)$
    - boolean: $\mathit{urgent}(e)$

HYPE model to hybrid automaton

- modes V: set of reachable configurations

HYPE model to hybrid automaton

- modes V: set of reachable configurations
- edges E: transitions between configurations

HYPE model to hybrid automaton

- modes V: set of reachable configurations
- edges E: transitions between configurations
- variables $\mathbf{X}$: variables $\mathcal{V}$

HYPE model to hybrid automaton

- modes V: set of reachable configurations
- edges E: transitions between configurations
- variables $\mathbf{X}$: variables $\mathcal{V}$
- if $v_j = \langle P_j, \sigma_j \rangle$ then

$$\mathit{flow}(v_j)[X_i] \;=\; \sum\{r[\![I(\vec{W})]\!] \mid iv(\iota) = X_i \text{ and } \sigma_j(\iota) = (r, I(\vec{W}))\}$$

HYPE model to hybrid automaton

- modes V: set of reachable configurations
- edges E: transitions between configurations
- variables $\mathbf{X}$: variables $\mathcal{V}$
- if $v_j = \langle P_j, \sigma_j \rangle$ then

$$\mathit{flow}(v_j)[X_i] = \sum\{r[\![I(\vec{W})]\!] \mid iv(\iota) = X_i \text{ and } \sigma_j(\iota) = (r, I(\vec{W}))\}$$

- $\mathit{inv}(v) = \mathit{true}$

HYPE model to hybrid automaton

- modes V: set of reachable configurations
- edges E: transitions between configurations
- variables $\mathbf{X}$: variables $\mathcal{V}$
- if $v_j = \langle P_j, \sigma_j \rangle$ then

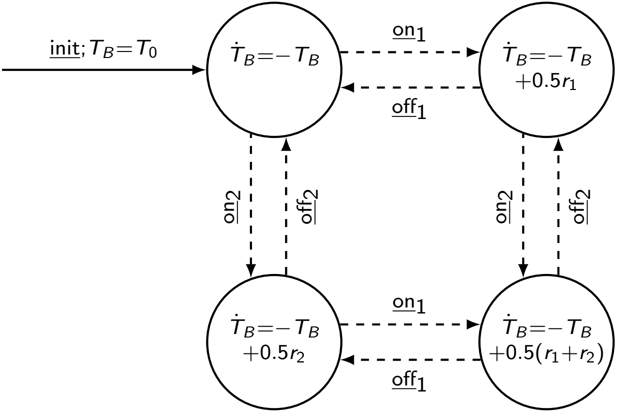
  $$\mathit{flow}(v_j)[X_i] \; = \; \sum\{r[\![I(\vec{W})]\!] \mid iv(\iota) = X_i \text{ and } \sigma_j(\iota) = (r, I(\vec{W}))\}$$
- $\mathit{inv}(v) = \mathit{true}$
- let e be an edge associated with $\underline{a}$ and let $ec(\underline{a}) = (\mathit{act}_{\underline{a}}, \mathit{res}_{\underline{a}})$
  - $\mathit{event}(e) = \underline{a}$ and $\mathit{reset}(e) = \mathit{res}_{\underline{a}}$
  - if $\mathit{act}_{\underline{a}} \neq \perp$ then $\mathit{jump}(e) = \mathit{act}_{\underline{a}}$ and $\mathit{urgent}(e) = \mathit{true}$
    else $\mathit{jump}(e) = \mathit{true}$ and $\mathit{urgent}(e) = \mathit{false}$

HYPE model to hybrid automaton

- modes V: set of reachable configurations
- edges E: transitions between configurations
- variables $\mathbf{X}$: variables $\mathcal{V}$
- if $v_j = \langle P_j, \sigma_j \rangle$ then

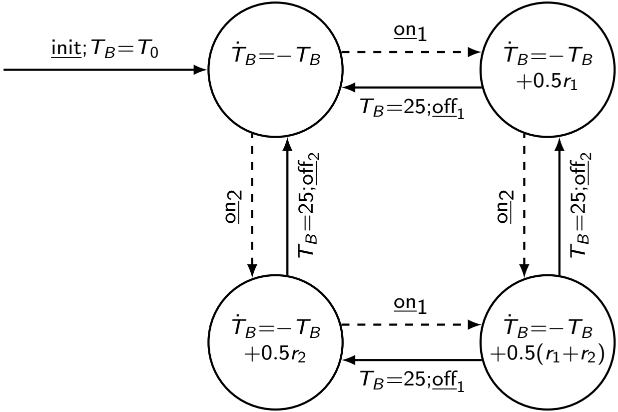
  $$flow(v_j)[X_i] \;=\; \sum\{r[\![I(\vec{W})]\!] \mid iv(\iota) = X_i \text{ and } \sigma_j(\iota) = (r, I(\vec{W}))\}$$
- $inv(v) = true$
- let e be an edge associated with $\underline{a}$ and let $ec(\underline{a}) = (act_{\underline{a}}, res_{\underline{a}})$
  - $event(e) = \underline{a}$ and $reset(e) = res_{\underline{a}}$
  - if $act_{\underline{a}} \neq \perp$ then $jump(e) = act_{\underline{a}}$ and $urgent(e) = true$
    else $jump(e) = true$ and $urgent(e) = false$
- $init(v) = \begin{cases} res_{\underline{init}} & \text{if } v = \langle P, \sigma \rangle \text{ with primes removed} \\ false & otherwise \end{cases}$

Heater example V: Hybrid automaton



Heater example VI: Hybrid automaton

- max temp of $25^\circ C$: $\mathrm{ec}(\underline{\mathsf{off}}_i) = ((T_B = 25), \mathit{true})$



Equivalence semantics

- system bisimulation: relation B if for all $(P, Q) \in B$ whenever
  1. $\langle P, \sigma \rangle \xrightarrow{\mathsf{a}} \langle P', \sigma' \rangle$, there exists $\langle Q', \sigma' \rangle$ with $\langle Q, \sigma \rangle \xrightarrow{\mathsf{a}} \langle Q', \sigma' \rangle$ and $(P', Q') \in B$.
  2. $\langle Q, \sigma \rangle \xrightarrow{\mathsf{a}} \langle Q', \sigma' \rangle$, there exists $\langle P', \sigma' \rangle$ with $\langle P, \sigma \rangle \xrightarrow{\mathsf{a}} \langle P', \sigma' \rangle$ and $(P', Q') \in B$.

Equivalence semantics

- system bisimulation: relation B if for all $(P, Q) \in B$ whenever
  1. $\langle P, \sigma \rangle \xrightarrow{\mathrm{a}} \langle P', \sigma' \rangle$, there exists $\langle Q', \sigma' \rangle$ with $\langle Q, \sigma \rangle \xrightarrow{\mathrm{a}} \langle Q', \sigma' \rangle$ and $(P', Q') \in B$.
  2. $\langle Q, \sigma \rangle \xrightarrow{\mathrm{a}} \langle Q', \sigma' \rangle$, there exists $\langle P', \sigma' \rangle$ with $\langle P, \sigma \rangle \xrightarrow{\mathrm{a}} \langle P', \sigma' \rangle$ and $(P', Q') \in B$.
- system bisimilar: $P \sim_s Q$ if in a system bisimulation

Equivalence semantics

- system bisimulation: relation B if for all $(P, Q) \in B$ whenever
  1. $\langle P, \sigma \rangle \xrightarrow{\text{a}} \langle P', \sigma' \rangle$, there exists $\langle Q', \sigma' \rangle$ with $\langle Q, \sigma \rangle \xrightarrow{\text{a}} \langle Q', \sigma' \rangle$ and $(P', Q') \in B$.
  2. $\langle Q, \sigma \rangle \xrightarrow{\text{a}} \langle Q', \sigma' \rangle$, there exists $\langle P', \sigma' \rangle$ with $\langle P, \sigma \rangle \xrightarrow{\text{a}} \langle P', \sigma' \rangle$ and $(P', Q') \in B$.
- system bisimilar: $P \sim_s Q$ if in a system bisimulation
- *Theorem 1*: $\sim_s$ is a congruence for all operators

Equivalence semantics

- system bisimulation: relation B if for all $(P, Q) \in B$ whenever
  1. $\langle P, \sigma \rangle \xrightarrow{\underline{a}} \langle P', \sigma' \rangle$, there exists $\langle Q', \sigma' \rangle$ with $\langle Q, \sigma \rangle \xrightarrow{\underline{a}} \langle Q', \sigma' \rangle$ and $(P', Q') \in B$.
  2. $\langle Q, \sigma \rangle \xrightarrow{\underline{a}} \langle Q', \sigma' \rangle$, there exists $\langle P', \sigma' \rangle$ with $\langle P, \sigma \rangle \xrightarrow{\underline{a}} \langle P', \sigma' \rangle$ and $(P', Q') \in B$.
- system bisimilar: $P \sim_s Q$ if in a system bisimulation
- *Theorem 1*: $\sim_s$ is a congruence for all operators
- *Theorem 2*: if Σ_1 and Σ_2 have the same prefixes then $\Sigma_1 \underset{L}{\bowtie} \underline{\text{init}}.Con \sim_s \Sigma_2 \underset{L}{\bowtie} \underline{\text{init}}.Con$, assuming well-defined systems

Equivalence semantics

- system bisimulation: relation B if for all $(P, Q) \in B$ whenever
  1. $\langle P, \sigma\rangle \xrightarrow{\text{a}} \langle P', \sigma'\rangle$, there exists $\langle Q', \sigma'\rangle$ with $\langle Q, \sigma\rangle \xrightarrow{\text{a}} \langle Q', \sigma'\rangle$ and $(P', Q') \in B$.
  2. $\langle Q, \sigma\rangle \xrightarrow{\text{a}} \langle Q', \sigma'\rangle$, there exists $\langle P', \sigma'\rangle$ with $\langle P, \sigma\rangle \xrightarrow{\text{a}} \langle P', \sigma'\rangle$ and $(P', Q') \in B$.
- system bisimilar: $P \sim_s Q$ if in a system bisimulation
- *Theorem 1*: $\sim_s$ is a congruence for all operators
- *Theorem 2*: if Σ_1 and Σ_2 have the same prefixes then $\Sigma_1 \underset{L}{\bowtie} \underline{\text{init}}.Con \sim_s \Sigma_2 \underset{L}{\bowtie} \underline{\text{init}}.Con$, assuming well-defined systems
- *Theorem 3*: if $P \sim_s Q$ then $P_\sigma = Q_\sigma$ for all σ, assuming well-defined systems

Heater example VII

- Consider two fans in Room C and none in Room A

$$Sys' \stackrel{def}{=} (Fan_{1,C,B} \underset{\{\underline{\text{init}}\}}{\bowtie} Fan_{2,C,B}) \underset{\{\underline{\text{init}}\}}{\bowtie} Room_B(T_B)$$

$$MF' \stackrel{def}{=} Sys' \underset{K}{\bowtie} \underline{\text{init}}.Con \qquad K = \{\underline{\text{init}}, \underline{\text{on}}_1, \underline{\text{off}}_1, \underline{\text{on}}_2, \underline{\text{off}}_2\}$$

Heater example VII

- Consider two fans in Room C and none in Room A

$$Sys' \stackrel{def}{=} (Fan_{1,C,B} \underset{\{\underline{\mathrm{init}}\}}{\bowtie} Fan_{2,C,B}) \underset{\{\underline{\mathrm{init}}\}}{\bowtie} Room_B(T_B)$$

$$MF' \stackrel{def}{=} Sys' \underset{K}{\bowtie} \underline{\mathrm{init}}.Con \qquad K = \{\underline{\mathrm{init}}, \underline{\mathrm{on}}_1, \underline{\mathrm{off}}_1, \underline{\mathrm{on}}_2, \underline{\mathrm{off}}_2\}$$

Heater example VII

- Consider two fans in Room C and none in Room A

$$Sys' \stackrel{def}{=} (Fan_{1,C,B} \underset{\{\underline{\mathrm{init}}\}}{\bowtie} Fan_{2,C,B}) \underset{\{\underline{\mathrm{init}}\}}{\bowtie} Room_B(T_B)$$

$$MF' \stackrel{def}{=} Sys' \underset{K}{\bowtie} \underline{\mathrm{init}}.Con \qquad K = \{\underline{\mathrm{init}}, \underline{\mathrm{on}}_1, \underline{\mathrm{off}}_1, \underline{\mathrm{on}}_2, \underline{\mathrm{off}}_2\}$$

- Sys and Sys' have the same prefixes

$$\bigcup_{i=1,2} \{\underline{\mathrm{init}}:(t_{i,y}, 0, const), \underline{\mathrm{on}}_i:(t_{i,y}, r_i, const_{adj}), \underline{\mathrm{off}}_i:(t_{i,y}, 0, const)\}$$

$$\cup\ \{\underline{\mathrm{init}}:(t_{0,x}, -1, linear(T))\}$$

Heater example VII

- Consider two fans in Room C and none in Room A

$$Sys' \stackrel{def}{=} (Fan_{1,C,B} \underset{\{\underline{\text{init}}\}}{\bowtie} Fan_{2,C,B}) \underset{\{\underline{\text{init}}\}}{\bowtie} Room_B(T_B)$$

$$MF' \stackrel{def}{=} Sys' \underset{K}{\bowtie} \underline{\text{init}}.Con \qquad K = \{\underline{\text{init}}, \underline{\text{on}}_1, \underline{\text{off}}_1, \underline{\text{on}}_2, \underline{\text{off}}_2\}$$

- Sys and Sys' have the same prefixes

$$\bigcup_{i=1,2} \{\underline{\text{init}}:(t_{i,y}, 0, const), \underline{\text{on}}_i:(t_{i,y}, r_i, const_{adj}), \underline{\text{off}}_i:(t_{i,y}, 0, const)\}$$

$$\cup \ \{\underline{\text{init}}:(t_{0,x}, -1, linear(T))\}$$

- by Theorem 2, $MF \sim_s MF'$

Heater example VII

- Consider two fans in Room C and none in Room A

$$Sys' \stackrel{def}{=} (Fan_{1,C,B} \underset{\{\underline{\text{init}}\}}{\bowtie} Fan_{2,C,B}) \underset{\{\underline{\text{init}}\}}{\bowtie} Room_B(T_B)$$

$$MF' \stackrel{def}{=} Sys' \underset{K}{\bowtie} \underline{\text{init}}.Con \qquad K = \{\underline{\text{init}}, \underline{\text{on}}_1, \underline{\text{off}}_1, \underline{\text{on}}_2, \underline{\text{off}}_2\}$$

- Sys and Sys' have the same prefixes

$$\bigcup_{i=1,2} \{\underline{\text{init}} : (t_{i,y}, 0, const), \underline{\text{on}}_i : (t_{i,y}, r_i, const_{adj}), \underline{\text{off}}_i : (t_{i,y}, 0, const)\}$$

$$\cup \ \{\underline{\text{init}} : (t_{0,x}, -1, linear(T))\}$$

- by Theorem 2, $MF \sim_s MF'$
- by Theorem 3, MF and MF' have the same ODEs

Outline

Conclusions and further work

- Many interesting and important systems can be regarded as examples of collective dynamics and emergent behaviour.
- Process algebras, such as PEPA and HYPE, are well-suited to modelling the behaviour of such systems in terms of the individuals and their interactions.
- Continuous approximation of PEPA models allows an alternative mathematical analysis of the average behaviour of discrete event systems.
- HYPE is offering a fine-grained, flow-based process algebra approach to modelling hybrid systems.

Heater example VIII

- system with temperature limit in ACP^{srt}_{hs}
- $\theta \equiv (T_B^{\bullet} = {}^{\bullet}T_B) \quad \psi \equiv (T_B = 25)$

$$
\begin{aligned}
\text{Start} &\stackrel{def}{=} (T_B = T_0) \wedge\!\!\blacktriangle\ \text{Off12} \\
\text{Off12} &\stackrel{def}{=} (\dot{T}_B = -T_B) \curvearrowright\!\!\blacktriangledown\ \sigma^*_{\text{rel}}(\theta \ulcorner\!\!\blacktriangledown\ (on_1 \cdot \text{On1} + on_2 \cdot \text{On2})) \\
\text{On1} &\stackrel{def}{=} (T_B \leq 25 \wedge \dot{T}_B = -T_B + 0.5r_1) \\
&\quad \curvearrowright\!\!\blacktriangledown\ \sigma^*_{\text{rel}}((\theta \ulcorner\!\!\blacktriangledown\ on_2 \cdot \text{On12}) + (\psi :\longrightarrow (\theta \ulcorner\!\!\blacktriangledown\ off_1 \cdot \text{Off12}))) \\
\text{On2} &\stackrel{def}{=} (T_B \leq 25 \wedge \dot{T}_B = -T_B + 0.5r_2) \\
&\quad \curvearrowright\!\!\blacktriangledown\ \sigma^*_{\text{rel}}((\theta \ulcorner\!\!\blacktriangledown\ on_1 \cdot \text{On12}) + (\psi :\longrightarrow (\theta \ulcorner\!\!\blacktriangledown\ off_2 \cdot \text{Off12}))) \\
\text{On12} &\stackrel{def}{=} (T_B \leq 25 \wedge \dot{T}_B = -T_B + 0.5(r_1 + r_2)) \\
&\quad \curvearrowright\!\!\blacktriangledown\ \sigma^*_{\text{rel}}(\psi :\longrightarrow (\theta \ulcorner\!\!\blacktriangledown\ (off_1 \cdot \text{On2} + off_2 \cdot \text{On1})))
\end{aligned}
$$

Heater example VIII

- system with temperature limit in ACP^{srt}_{hs}
- $\theta \equiv (T_B^{\bullet} = {}^{\bullet}T_B) \quad \psi \equiv (T_B = 25)$

$$
\begin{aligned}
\mathrm{Start} &\stackrel{def}{=} (T_B = T_0) \wedge\!\!\blacktriangle\ \mathrm{Off12} \\
\mathrm{Off12} &\stackrel{def}{=} (\dot{T}_B = -T_B) \curvearrowright\!\blacktriangledown\ \sigma^*_{\mathrm{rel}}(\theta \ulcorner\!\blacktriangledown\ (on_1 \cdot \mathrm{On1} + on_2 \cdot \mathrm{On2})) \\
\mathrm{On1} &\stackrel{def}{=} (T_B \leq 25 \wedge \dot{T}_B = -T_B + 0.5r_1) \\
&\quad \curvearrowright\!\blacktriangledown\ \sigma^*_{\mathrm{rel}}((\theta \ulcorner\!\blacktriangledown\ on_2 \cdot \mathrm{On12}) + (\psi :\longrightarrow (\theta \ulcorner\!\blacktriangledown\ off_1 \cdot \mathrm{Off12}))) \\
\mathrm{On2} &\stackrel{def}{=} (T_B \leq 25 \wedge \dot{T}_B = -T_B + 0.5r_2) \\
&\quad \curvearrowright\!\blacktriangledown\ \sigma^*_{\mathrm{rel}}((\theta \ulcorner\!\blacktriangledown\ on_1 \cdot \mathrm{On12}) + (\psi :\longrightarrow (\theta \ulcorner\!\blacktriangledown\ off_2 \cdot \mathrm{Off12}))) \\
\mathrm{On12} &\stackrel{def}{=} (T_B \leq 25 \wedge \dot{T}_B = -T_B + 0.5(r_1 + r_2)) \\
&\quad \curvearrowright\!\blacktriangledown\ \sigma^*_{\mathrm{rel}}(\psi :\longrightarrow (\theta \ulcorner\!\blacktriangledown\ (off_1 \cdot \mathrm{On2} + off_2 \cdot \mathrm{On1})))
\end{aligned}
$$

Further Work

The relationship between the CTMC and the ODEs generated by the continuous approximation of a PEPA model has already been established (Kurtz's Theorem), but there is more to do. For example, analysing the approximation error for a given number of copies of components.

Further Work

The relationship between the CTMC and the ODEs generated by the continuous approximation of a PEPA model has already been established (Kurtz's Theorem), but there is more to do. For example, analysing the approximation error for a given number of copies of components.

HYPE is still a new language and we have many directions for further investigation. For example:

Further Work

The relationship between the CTMC and the ODEs generated by the continuous approximation of a PEPA model has already been established (Kurtz's Theorem), but there is more to do. For example, analysing the approximation error for a given number of copies of components.

HYPE is still a new language and we have many directions for further investigation. For example:

- Defining alternative forms of equivalence;

Further Work

The relationship between the CTMC and the ODEs generated by the continuous approximation of a PEPA model has already been established (Kurtz's Theorem), but there is more to do. For example, analysing the approximation error for a given number of copies of components.

HYPE is still a new language and we have many directions for further investigation. For example:

- Defining alternative forms of equivalence;
- Comparisons with other process algebra for hybrid systems;

Further Work

The relationship between the CTMC and the ODEs generated by the continuous approximation of a PEPA model has already been established (Kurtz's Theorem), but there is more to do. For example, analysing the approximation error for a given number of copies of components.

HYPE is still a new language and we have many directions for further investigation. For example:

- Defining alternative forms of equivalence;
- Comparisons with other process algebra for hybrid systems;
- Implementing the mapping to hybrid automata;

Further Work

The relationship between the CTMC and the ODEs generated by the continuous approximation of a PEPA model has already been established (Kurtz's Theorem), but there is more to do. For example, analysing the approximation error for a given number of copies of components.

HYPE is still a new language and we have many directions for further investigation. For example:

- Defining alternative forms of equivalence;
- Comparisons with other process algebra for hybrid systems;
- Implementing the mapping to hybrid automata;
- Further case studies.

Thanks!

Thanks!

Acknowledgements: collaborators
HYPE is joint work with Luca Bortolussi and Vashti Galpin.

Thanks!

Acknowledgements: collaborators
HYPE is joint work with Luca Bortolussi and Vashti Galpin.

Acknowledgements: funding
The CODA project: *Process Algebra for Collective Dynamics*, is funded by EPSRC.

Thanks!

Acknowledgements: collaborators
HYPE is joint work with Luca Bortolussi and Vashti Galpin.

Acknowledgements: funding
The CODA project: *Process Algebra for Collective Dynamics*, is funded by EPSRC.

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