Contextual Lumpability

Jane Hillston

School of Informatics
The University of Edinburgh
Scotland

joint work with Carla Piazza (Udine),
Andrea Marin and Sabina Rossi (Ca' Foscari),

Overview

- Lumbability (Kemeny and Snell, 1960) has long been established as an exact aggregation techniques for Markov chains.

Overview

- Lumbability (Kemeny and Snell, 1960) has long been established as an exact aggregation techniques for Markov chains.
- In the context of stochastic process algebras, Markovian bisimulations, have been shown to characterise lumpability in the sense that partitioning the state space on the basis of such an equivalence relation generates a lumpable partition of the underlying state space.

Overview

- Lumbability (Kemeny and Snell, 1960) has long been established as an exact aggregation techniques for Markov chains.
- In the context of stochastic process algebras, Markovian bisimulations, have been shown to characterise lumpability in the sense that partitioning the state space on the basis of such an equivalence relation generates a lumpable partition of the underlying state space.
- In the stochastic process algebra PEPA, this equivalence relation is termed strong equivalence.

Overview

- In this paper we introduced a relaxed form of lumpability for PEPA models, termed contextual lumpability.

Overview

- In this paper we introduced a relaxed form of lumpability for PEPA models, termed contextual lumpability.
- We also define a new equivalence relation for PEPA models, termed lumpable bisimilarity.

Overview

- In this paper we introduced a relaxed form of lumpability for PEPA models, termed contextual lumpability.
- We also define a new equivalence relation for PEPA models, termed lumpable bisimilarity.
- We show that lumpable bisimilarity is a characterisation of contextual lumpability and induces the largest contextual lumping of the underlying Markov chain.

Overview

- In this paper we introduced a relaxed form of lumpability for PEPA models, termed contextual lumpability.
- We also define a new equivalence relation for PEPA models, termed lumpable bisimilarity.
- We show that lumpable bisimilarity is a characterisation of contextual lumpability and induces the largest contextual lumping of the underlying Markov chain.
- Finally we provide an algorithm for lumpable bisimilarity.

Outline

Equivalence relations and state space explosion

It is well-known that Markovian based modelling techniques are prone to state space explosion.

Equivalence relations and state space explosion

It is well-known that Markovian based modelling techniques are prone to state space explosion.

Model aggregation: use a state-state equivalence to establish a partition of the state space of a model, and replace each set of states by one macro-state.

Equivalence relations and state space explosion

It is well-known that Markovian based modelling techniques are prone to state space explosion.

Model aggregation: use a state-state equivalence to establish a partition of the state space of a model, and replace each set of states by one macro-state.

Equivalence relations and state space explosion

It is well-known that Markovian based modelling techniques are prone to state space explosion.

Model aggregation: use a state-state equivalence to establish a partition of the state space of a model, and replace each set of states by one macro-state.

Models based on model aggregation/state-state equivalence will not in general be exact, because the macro states will not in general preserve the Markov property.

Equivalence relations and state space explosion

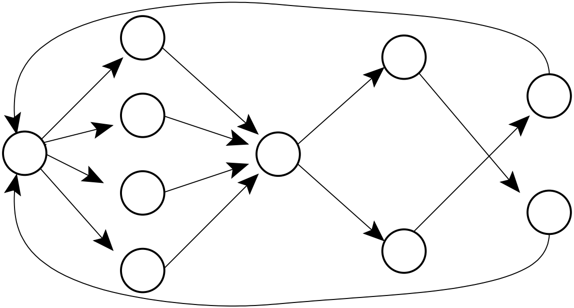
It is well-known that Markovian based modelling techniques are prone to state space explosion.

Model aggregation: use a state-state equivalence to establish a partition of the state space of a model, and replace each set of states by one macro-state.

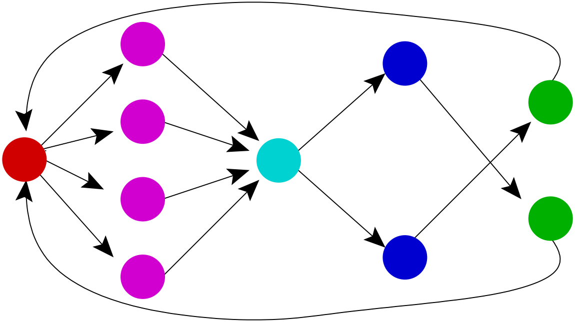
Models based on model aggregation/state-state equivalence will not in general be exact, because the macro states will not in general preserve the Markov property.

But it has long been established that if the aggregation is based on a lumpable partition then the Markov property is preserved [Kemeny and Snell, 1960].

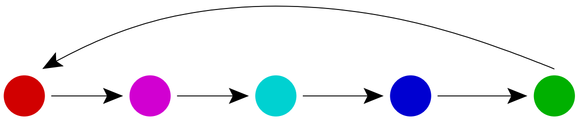
Reducing by lumpability



Reducing by lumpability



Reducing by lumpability



Lumpability

Lumpability

A CTMC, $\{X_i\}$, is (ordinarily) lumpable with respect to some partition $\chi = \{X_{[k]}\}$ if for any $X_{[k]}, X_{[l]} \in \chi$ with $k \neq l$ and $X_i, X_j \in X_{[k]}$,

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

where $q(X_i, X_{[l]})$ is the aggregated transition rate from X_i to all states in $X_{[l]}$, i.e., $q(X_i, X_{[l]}) = \sum_{X_m \in X_{[l]}} q(X_i, X_m)$.

Lumpable Relation

Lumpable relation

A relation $\mathcal{R} \subseteq \mathcal{C} \times \mathcal{C}$ over PEPA components is lumpable if for any component P, $ds(P)/\mathcal{R}$ induces a lumpable partition on the state space of the CTMC corresponding to P.

Lumpable Relation

Lumpable relation

A relation $\mathcal{R} \subseteq \mathcal{C} \times \mathcal{C}$ over PEPA components is lumpable if for any component P, $ds(P)/\mathcal{R}$ induces a lumpable partition on the state space of the CTMC corresponding to P.

Proposition

Let I be a set of indices and $\mathcal{R}_i$ be a lumpable relation for all $i \in I$. Then the union, $\mathcal{R} = \cup_{i \in I} \mathcal{R}_i$, is also a lumpable relation.

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

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

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

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

SPA MODEL

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

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

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

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

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

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

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

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

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

Stochastic Process Algebra

- Models are constructed from components which engage in activities.

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

action type or name

activity rate (parameter of an exponential distribution)

component/ derivative

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

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

PEPA

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

PEPA

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

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

PEPA

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

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

PEPA

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

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

PEPA

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

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

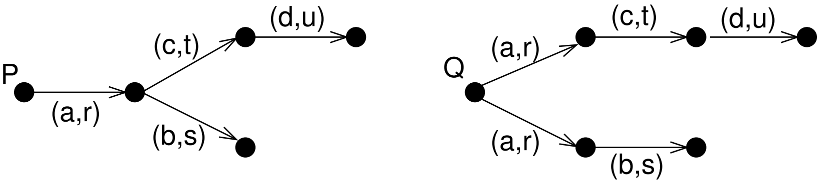
PEPA

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

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

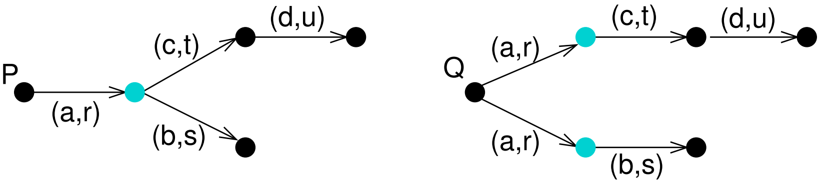
Equivalence Relations

Process algebras often define equivalence relations in terms of bisimulations, based on the notion of observability.



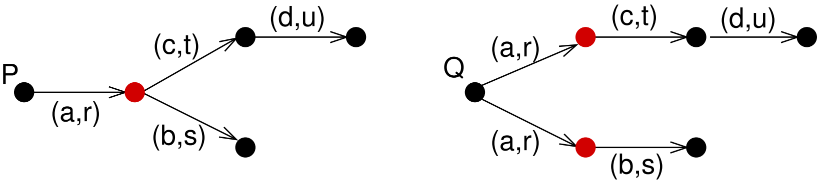
Equivalence Relations

Process algebras often define equivalence relations in terms of bisimulations, based on the notion of observability.



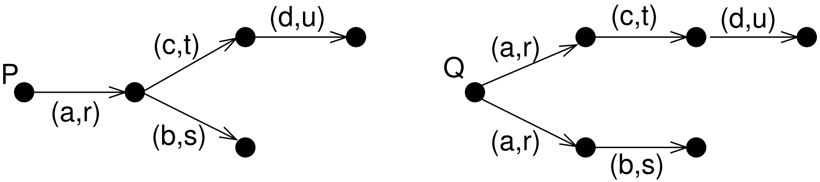
Equivalence Relations

Process algebras often define equivalence relations in terms of bisimulations, based on the notion of observability.



Equivalence Relations

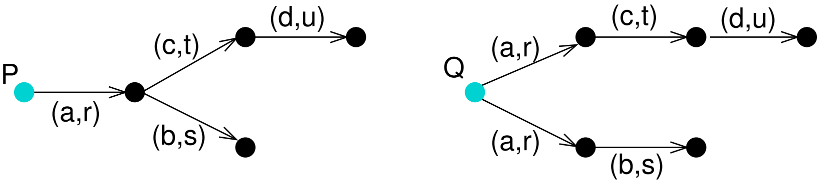
Process algebras often define equivalence relations in terms of bisimulations, based on the notion of observability.



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

Equivalence Relations

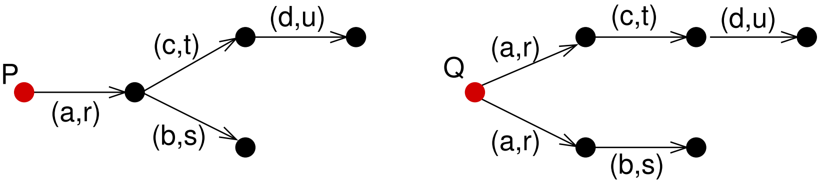
Process algebras often define equivalence relations in terms of bisimulations, based on the notion of observability.



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

Equivalence Relations

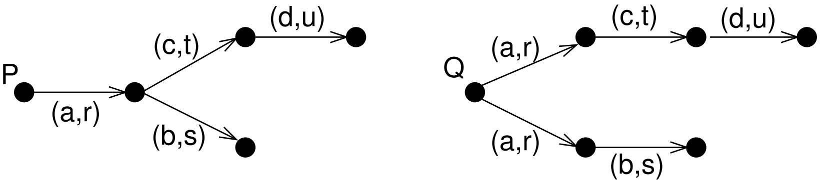
Process algebras often define equivalence relations in terms of bisimulations, based on the notion of observability.



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

Equivalence Relations

Process algebras often define equivalence relations in terms of bisimulations, based on the notion of observability.



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

Strong Equivalence in PEPA

Strong Equivalence

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

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

where

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

Strong Equivalence in PEPA

Strong Equivalence

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

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

where

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

Current action type preservation

A relation $\mathcal{R} \subseteq \mathcal{C} \times \mathcal{C}$ over PEPA components is current action type preserving if for all PEPA components P, Q such that $(P, Q) \in \mathcal{R}$, $\mathcal{A}(P) = \mathcal{A}(Q)$.

Contextuality

Contextuality

A relation $\mathcal{R} \subseteq \mathcal{C} \times \mathcal{C}$ over PEPA components is contextual if for all PEPA components P, Q such that $(P, Q) \in \mathcal{R}$ and for all contexts $C[\cdot]$,

$$(C[P], C[Q]) \in \mathcal{R}$$

where a context is a term with a hole $[\cdot]$ defined by the grammar:

$$C[\cdot] ::= [\cdot] \;\mid\; [\cdot] \underset{L}{\bowtie} P \;\mid\; P \underset{L}{\bowtie} [\cdot] \;\mid\; [\cdot]/L$$

Strong Equivalence and Lumpability

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

Strong Equivalence and Lumpability

- Given this definition it is fairly straightforward to show that if we consider strong equivalence of states within a single model, it induces an ordinarily lumpable partition on the state space of the underlying Markov chain.
- Moreover it can be shown that strong equivalence is contextual. (Indeed, it is a congruence with respect to all operators of PEPA including the dynamic ones.)

Strong Equivalence and Lumpability

- Given this definition it is fairly straightforward to show that if we consider strong equivalence of states within a single model, it induces an ordinarily lumpable partition on the state space of the underlying Markov chain.
- Moreover it can be shown that strong equivalence is contextual. (Indeed, it is a congruence with respect to all operators of PEPA including the dynamic ones.)
- This means that aggregation based on lumpability can be applied component by component.

Producer-Consumer Example

- We consider a system consisting of a Producer process and a Consumer process.
- The Producer thinks, computes and transfers results to the Consumer.
- Periodically errors occur during the computation and the process has to undergo recovery before it is ready to redo the computation.
- The Consumer enqueues the transferred jobs and sends them in batches.
- Jobs waiting in the queue can spontaneously generate further jobs.
- A maximum buffer size of N jobs is set and, in case of saturation, further arrivals are lost.

Producer-Consumer Example: PEPA

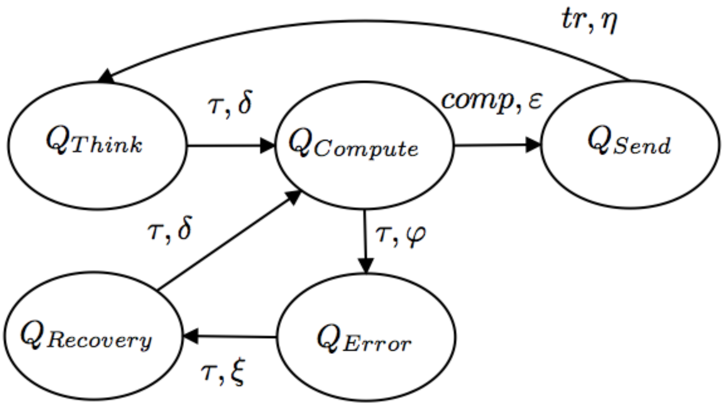
Producer

$$\begin{aligned}
Q_{Think} &\stackrel{def}{=} (\tau, \delta).Q_{Compute} \\
Q_{Compute} &\stackrel{def}{=} (comp, \varepsilon).Q_{Send} + (\tau, \varphi).Q_{Error} \\
Q_{Send} &\stackrel{def}{=} (tr, \eta).Q_{Think} \\
Q_{Error} &\stackrel{def}{=} (\tau, \xi).Q_{Recovery} \\
Q_{Recovery} &\stackrel{def}{=} (\tau, \delta).Q_{Compute}
\end{aligned}$$

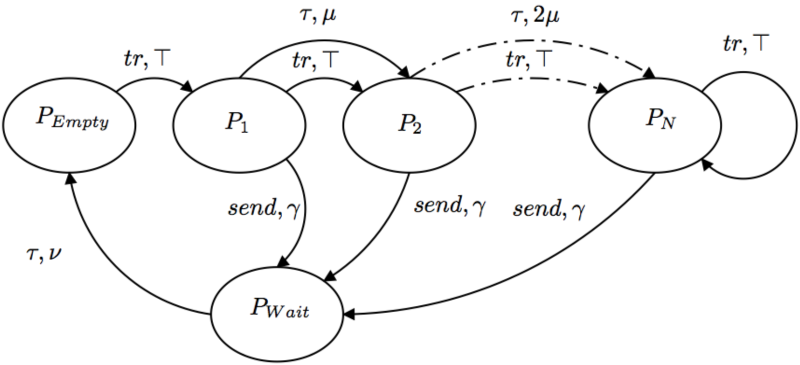
Consumer

$$\begin{aligned}
P_{Empty} &\stackrel{def}{=} (tr, \top).P_1 \\
P_i &\stackrel{def}{=} (\tau, i\mu).P_{i+1} + (tr, \top).P_{i+1} + (send, \gamma).P_{Wait} \quad 1 \leq i < N \\
P_N &\stackrel{def}{=} (tr, \top).P_N + (send, \gamma).P_{Wait} \\
P_{Wait} &\stackrel{def}{=} (\tau, \nu).P_{Empty}
\end{aligned}$$

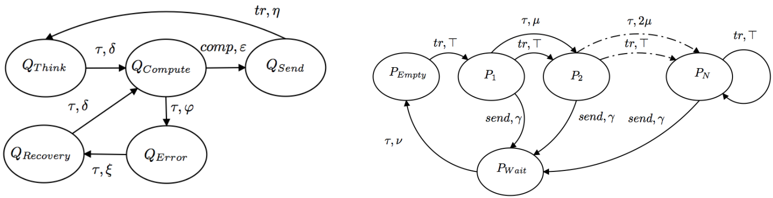
Producer-Consumer Example: Producer



Producer-Consumer Example: Consumer



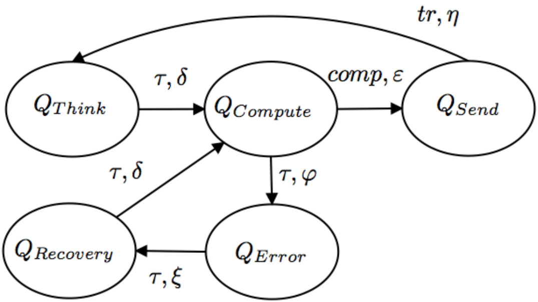
Producer-Consumer Example: System



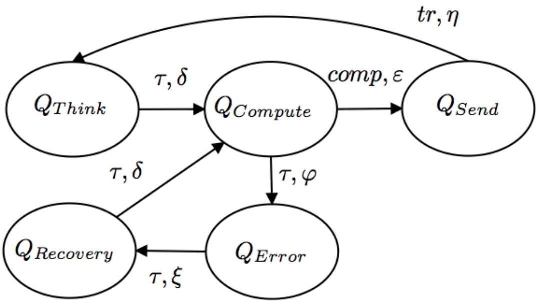
The whole system is modelled by:

$$S \stackrel{def}{=} P_{Empty} \underset{\{tr\}}{\bowtie} Q_{Think} \,.$$

Example revisited: Original producer

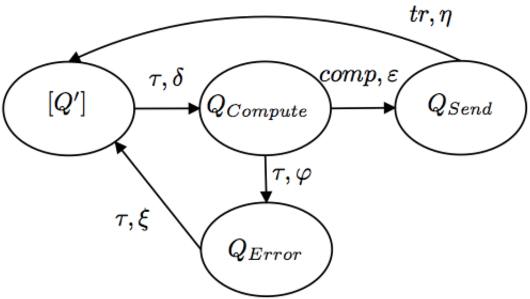


Example revisited: Original producer



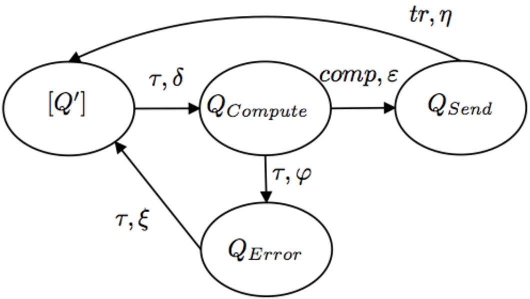
We can see that Q_{Think} and $Q_{Recovery}$ are strongly equivalent and consequently reduce the representation of the producer.

Example revisited: Reduced producer



We can see that Q_{Think} and $Q_{Recovery}$ are strongly equivalent and consequently reduce the representation of the producer.

Example revisited: Reduced producer



We can see that Q_{Think} and $Q_{Recovery}$ are strongly equivalent and consequently reduce the representation of the producer.

No such reduction is possible for the Consumer.

Contextual Lumpability

We know that any partition generated by strong equivalence at the process algebra level gives rise to a lumpable partition in the underlying Markov chain.

Contextual Lumpability

We know that any partition generated by strong equivalence at the process algebra level gives rise to a lumpable partition in the underlying Markov chain.

But we also know that this is not always the largest lumping that could be achieved if we worked directly at the level of the Markov chain.

Contextual Lumpability

We know that any partition generated by strong equivalence at the process algebra level gives rise to a lumpable partition in the underlying Markov chain.

But we also know that this is not always the largest lumping that could be achieved if we worked directly at the level of the Markov chain.

On the other hand, the contextually of the relation — the fact that it is preserved by the static combinators of the language — has considerable advantages when working on model reduction.

Contextual Lumpability

We know that any partition generated by strong equivalence at the process algebra level gives rise to a lumpable partition in the underlying Markov chain.

But we also know that this is not always the largest lumping that could be achieved if we worked directly at the level of the Markov chain.

On the other hand, the contextually of the relation — the fact that it is preserved by the static combinators of the language — has considerable advantages when working on model reduction.

Therefore we seek to define a relation which maintains contextually but achieves greater lumping.

Contextual Lumpability: formal definition

Contextual lumpability

Contextual lumpability, denoted $\cong_l$, is the largest contextual, current action type preserving, lumpable relation over PEPA terms.

Contextual Lumpability: formal definition

Contextual lumpability

Contextual lumpability, denoted $\cong_l$, is the largest contextual, current action type preserving, lumpable relation over PEPA terms.

Contextual Lumpability: formal definition

Contextual lumpability

Contextual lumpability, denoted $\cong_l$, is the largest contextual, current action type preserving, lumpable relation over PEPA terms.

Contextual Lumpability: formal definition

Contextual lumpability

Contextual lumpability, denoted $\cong_l$, is the largest contextual, current action type preserving, lumpable relation over PEPA terms.

Lumpable Bisimulation

We can consider contextual lumpability as a target or template for what we would like to achieve, but it is not the definition of equivalence in the usual process algebra style of bisimulation based on observability.

Lumpable Bisimulation

We can consider contextual lumpability as a target or template for what we would like to achieve, but it is not the definition of equivalence in the usual process algebra style of bisimulation based on observability.

Thus we seek to define a bisimulation that can be seen to fit the template of contextual lumpability.

Lumpable Bisimulation: formal definition

Lumpable bisimulation

An equivalence relation over PEPA components, $\mathcal{R} \subseteq \mathcal{C} \times \mathcal{C}$, is a lumpable bisimulation if whenever $(P, Q) \in \mathcal{R}$ then for all $\alpha \in \mathcal{A}$ and for all $S \in \mathcal{C}/\mathcal{R}$ such that

- either $\alpha \neq \tau$,
- or $\alpha = \tau$ and $P, Q \notin S$,

it holds

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

Remarks

- Note that this definition allows arbitrary τ activities between components belonging to the same equivalence class.

Remarks

- Note that this definition allows arbitrary τ activities between components belonging to the same equivalence class.
- Thus it can be regarded as coarser than strong equivalence.

Remarks

- Note that this definition allows arbitrary τ activities between components belonging to the same equivalence class.
- Thus it can be regarded as coarser than strong equivalence.
- However, it has weaker congruence properties because although it is contextual, unlike strong equivalence it is not preserved by choice.

Remarks

- Note that this definition allows arbitrary τ activities between components belonging to the same equivalence class.
- Thus it can be regarded as coarser than strong equivalence.
- However, it has weaker congruence properties because although it is contextual, unlike strong equivalence it is not preserved by choice.
- Nevertheless for practical purposes this is sufficient for our needs.

Lumpable Bisimilarity

Proposition

Let I be a set of indices and $\mathcal{R}_i$ be a lumpable bisimulation for all $i \in I$. Then the transitive closure of their union, $\mathcal{R} = (\cup_{i \in I} \mathcal{R}_i)^*$, is also a lumpable bisimulation.

Lumpable Bisimilarity

Proposition

Let I be a set of indices and $\mathcal{R}_i$ be a lumpable bisimulation for all $i \in I$. Then the transitive closure of their union, $\mathcal{R} = (\cup_{i \in I} \mathcal{R}_i)^*$, is also a lumpable bisimulation.

Lumpable bisimilarity

Two PEPA components P and Q are lumpably bisimilar, written $P \approx_l Q$, if $(P, Q) \in \mathcal{R}$ for some lumpable bisimulation $\mathcal{R}$, i.e.,

$$\approx_l = \bigcup \{\mathcal{R} \mid \mathcal{R} \text{ is a lumpable bisimulation}\}.$$

$\approx_l$ is called lumpable bisimilarity and it is the largest symmetric lumpable bisimulation over PEPA components.

Lumpable Bisimilarity and Contextual Lumpability

It remains to show that lumpable bisimilarity is a characterisation of contextual lumpability.

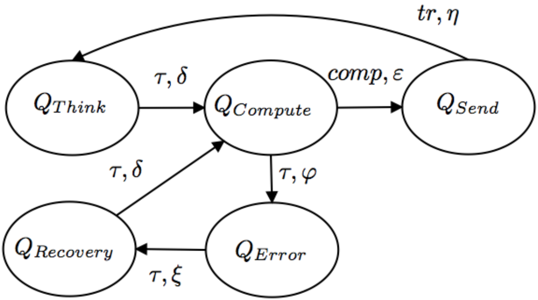
Lumpable Bisimilarity and Contextual Lumpability

It remains to show that lumpable bisimilarity is a characterisation of contextual lumpability.

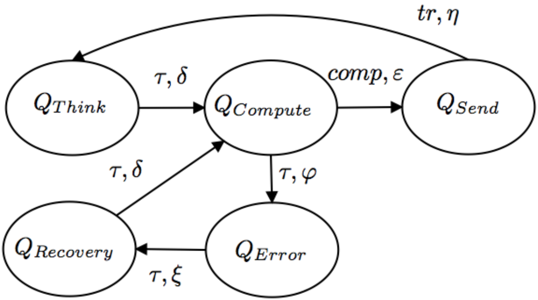
Theorem

Let P and Q be two PEPA components.
It holds that: $P \approx_l Q$ if and only if $P \cong_l Q$.

Example revisited: Original producer

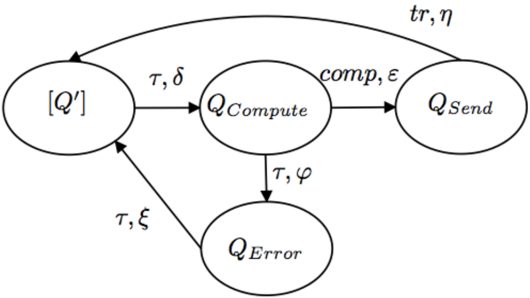


Example revisited: Original producer



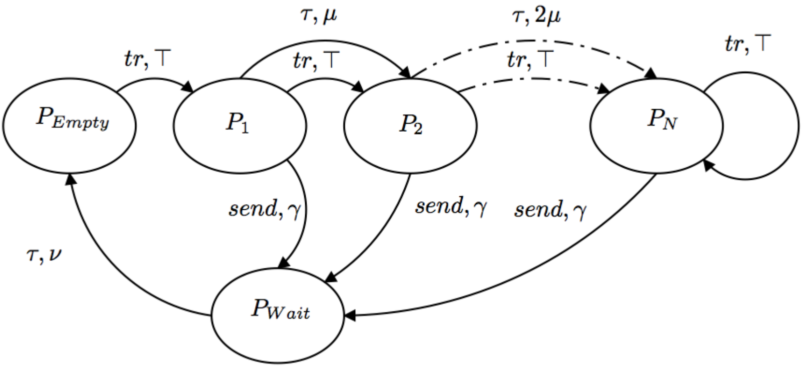
We can see that $Q_{Think} \approx_l Q_{Recovery}$ and hence they belong to the same equivalence class, say $[Q']$.

Example revisited: Reduced producer

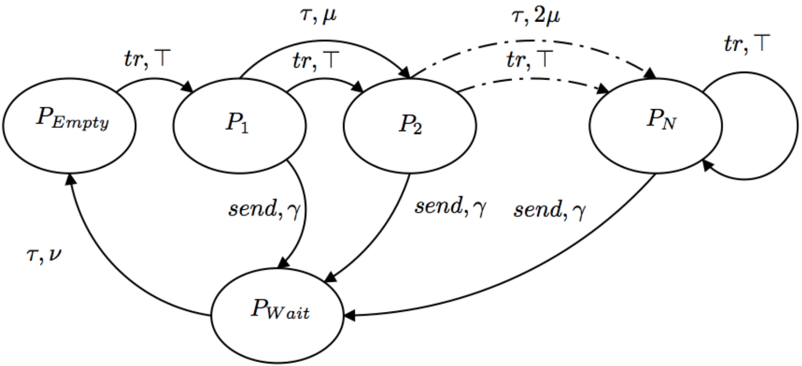


We can see that $Q_{Think} \approx_I Q_{Recovery}$ and hence they belong to the same equivalence class, say $[Q']$.

Example revisited: Original consumer

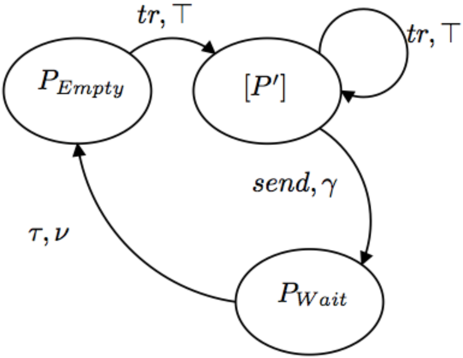


Example revisited: Original consumer



Here we observe that $P_1 \approx_l P_2 \approx_l \cdots \approx_l P_N$ and let $[P']$ be the associated equivalence class.

Example revisited: Reduced consumer



Here we observe that $P_1 \approx_l P_2 \approx_l \cdots \approx_l P_N$ and let $[P']$ be the associated equivalence class.

Algorithm

- We take advantage of an existing algorithm $\textsc{Lumpability}(G = (D, \Delta, W), I)$, developed by Valmari and Franceschinis in 2010 (TACAS 2010).

Algorithm

- We take advantage of an existing algorithm $\textsc{Lumpability}(G = (D, \Delta, W), I)$, developed by Valmari and Franceschinis in 2010 (TACAS 2010).
- $\textsc{Lumpability}(G = (D, \Delta, W), I)$ computes the largest equivalence relation $R \subseteq I$ over the derivative set D compatible with the derivation graph G.

Algorithm

- We take advantage of an existing algorithm $\textsc{Lumpability}(G = (D, \Delta, W), I)$, developed by Valmari and Franceschinis in 2010 (TACAS 2010).
- $\textsc{Lumpability}(G = (D, \Delta, W), I)$ computes the largest equivalence relation $R \subseteq I$ over the derivative set D compatible with the derivation graph G.
- $\textsc{Contextual\_Lumpability}(\{G_\alpha^{\mathcal{D}}\}_{\alpha \in \mathcal{A} \setminus \{\tau\}}, \widehat{G}_\tau^{\mathcal{D}})$ returns the relation $\approx_I \cap (\mathcal{D} \times \mathcal{D})$, i.e., the largest lumpable bisimulation over $\mathcal{D}$.

Algorithm

$\textsc{Contextual\_Lumpability}(\{G_\alpha^{\mathcal{D}}\}_{\alpha \in \mathcal{A} \setminus \{\tau\}}, \widehat{G}_\tau^{\mathcal{D}})$

Require: $\mathcal{D}$ finite and $\mathcal{D} = ds(\mathcal{D})$;
$R = \emptyset$;
$\mathcal{R} = \mathcal{D} \times \mathcal{D}$;
while $R \neq \mathcal{R}$ **do**
  $R = \mathcal{R}$;
  for all $\alpha \in \mathcal{A} \setminus \{\tau\}$ **do**
    $\mathcal{R} = \textsc{Lumpability}(G_\alpha^{\mathcal{D}}, \mathcal{R})$;
  end for
  $\mathcal{R} = \textsc{Lumpability}(\widehat{G}_\tau^{\mathcal{D}}, \mathcal{R})$;
end while
return $\mathcal{R}$;

Conclusions

- We have introduced a relaxed form of lumpability for PEPA models, termed contextual lumpability.
- We also defined a new equivalence relation for PEPA models, termed lumpable bisimilarity.
- We showed that lumpable bisimilarity is a characterisation of contextual lumpability and induces the largest contextual lumping of the underlying Markov chain of a PEPA model.
- Finally we provided an algorithm for lumpable bisimilarity.