Performance Modelling of Computer Systems

Mirco Tribastone

Institut für Informatik
Ludwig-Maximilians-Universität München

Stochastic Process Algebra

Overview

- Overview of classic (untimed) process algebra
- Associating exponential distributions to activities
- Introduction to the stochastic process algebra PEPA

Bibliographic references:

- J. Hillston. **A Compositional Approach to Performance Modelling**. Cambridge University Press, 1996.
- A. Clark, J. Hillston, and M. Tribastone. **Stochastic Process Algebras**. In *Formal Methods for Performance Evaluation: the 7th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM 2007*, LNCS 4486, Springer-Verlag.

- Overview of classic (untimed) process algebra
- Associating exponential distributions to activities
- Introduction to the stochastic process algebra PEPA

Bibliographic references:

- J. Hillston. **A Compositional Approach to Performance Modelling**. Cambridge University Press, 1996.
- A. Clark, J. Hillston, and M. Tribastone. **Stochastic Process Algebras**. In *Formal Methods for Performance Evaluation: the 7th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM 2007*, LNCS 4486, Springer-Verlag.

For classic process algebra, see previous lecture. The Greek symbols in our model can be seen as *labels*. For instance

$$C_{off} \stackrel{def}{=} \alpha_1.C_{on}$$

is just a process that can do an α_1 action to become C_{on}.
Associating exponential distributions was discussed in the previous lecture. We had activities of type $(break, \alpha_1)$, where α_1 can be seen as a *capacity* associated with the label *break*.

A copy of an introduction to process algebra (without rates) is available as a set of lecture notes from Prof. De Nicola at:
`http://www.pst.ifi.lmu.de/Lehre/sose-2013/formale-spezifikation-und-verifikation/intro-to-pa.pdf`

A copy of the CUP book is available online at:
`http://www.dcs.ed.ac.uk/pepa/book.pdf`

Features of Stochastic Process Algebra

- A high-level description technique for continuous-time Markov chains. . .
- . . . but not only:
  - hybrid systems;
  - continuous-state systems;
  - . . .
- A formal method: a textual language with a precise syntax and semantics.
- A compositional approach to performance evaluation: the modelling and reasoning is modular.

Features of Stochastic Process Algebra

- A high-level description technique for continuous-time Markov chains. . .
- . . . but not only:
  - hybrid systems;
  - continuous-state systems;
  - . . .
- A formal method: a textual language with a precise syntax and semantics.
- A compositional approach to performance evaluation: the modelling and reasoning is modular.

A high-level formal language such as a stochastic process algebra is to Markov chains what a high-level programming language such as Java is to machine code: A more concise description that hydes possibly very large Markov chains.

Modularity is important. In the example we considered a system with a customer (U_h and U_s) and a cashier (C_{on} and C_{off}).

$$U_h \stackrel{def}{=} (sleep, \lambda).U_s \qquad U_s \stackrel{def}{=} (serve, \mu).U_h$$

$$C_{on} \stackrel{def}{=} (serve, \mu).C_{on} + (break, \alpha_1).C_{off} \qquad C_{off} \stackrel{def}{=} (smoke, \alpha_2).C_{on}$$

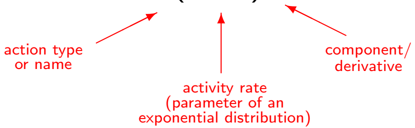
The composed model is written as

$$C_{on} \underset{\{serve\}}{\bowtie} U_H \tag{1}$$

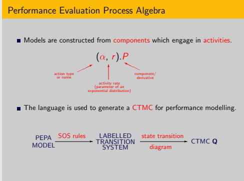
Performance Evaluation Process Algebra

- Models are constructed from components which engage in activities.

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



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



Discussed at length in the previous lecture. . .

PEPA

BNF Syntax

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

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

PEPA

BNF Syntax

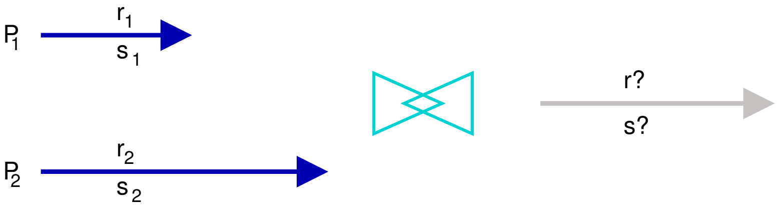
$$S ::= (\alpha, r).S \mid S + S \mid A$$
$$P ::= S \mid P \bowtie_L P \mid P/L$$

PREFIX:	$(\alpha, r).S$	designated first action
CHOICE:	$S + S$	competing components (race policy)
CONSTANT:	$A \stackrel{def}{=} S$	assigning names
COOPERATION:	$P \bowtie_L P$	$\alpha \notin L$ concurrent activity (*individual actions*) $\alpha \in L$ cooperative activity (*shared actions*)
HIDING:	P/L	abstraction $\alpha \in L \Rightarrow \alpha \to \tau$

Hiding is the only operator that was not discussed in the last lecture.

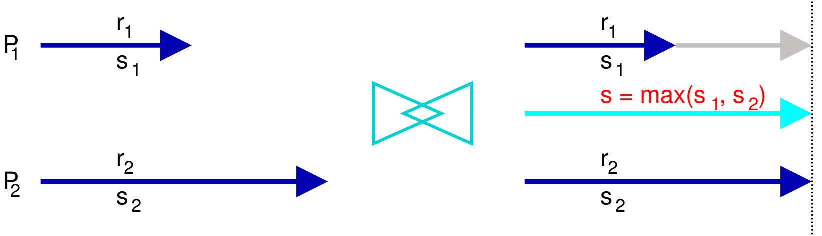
Timed Synchronisation

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

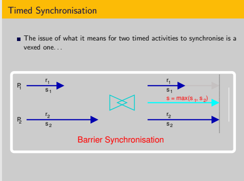


Timed Synchronisation

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



Barrier Synchronisation



Important point: the maximum of two exponential distributions **is not** an exponential distribution.
Consider for instance

$$P \stackrel{def}{=} (a, \lambda_1).P' \qquad Q \stackrel{def}{=} (a, \lambda_2).Q'$$

The synchronisation

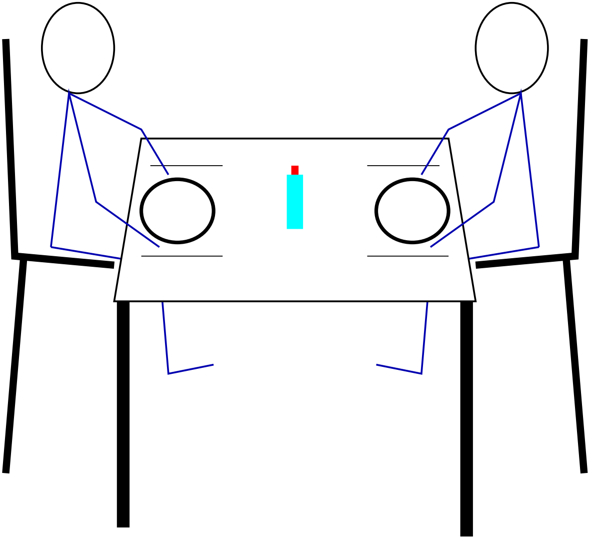
$$P \underset{\{a\}}{\bowtie} Q$$

will do a transition

$$P \underset{\{a\}}{\bowtie} Q \xrightarrow{(a,\min\{\lambda_1,\lambda_2\})} P' \underset{\{a\}}{\bowtie} Q'.$$

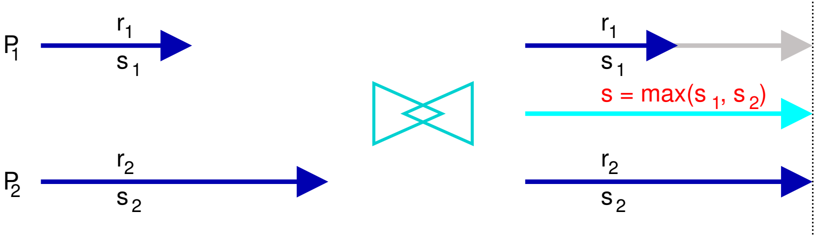
λ_1 and λ_2 are interpreted as capacities (e.g., bandwidths). There is a handshaking where the processes jointly decide the rate at which they perform the action, i.e., $\min\{\lambda_1, \lambda_2\}$ and then draw a sample from that distribution.

Timed Synchronisation



Timed Synchronisation

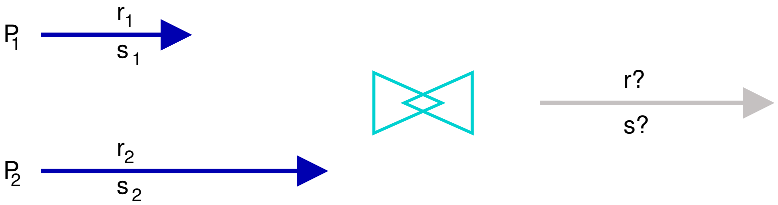
- The issue of what it means for two timed activities to synchronise is a vexed one...



s is no longer exponentially distributed

Timed Synchronisation

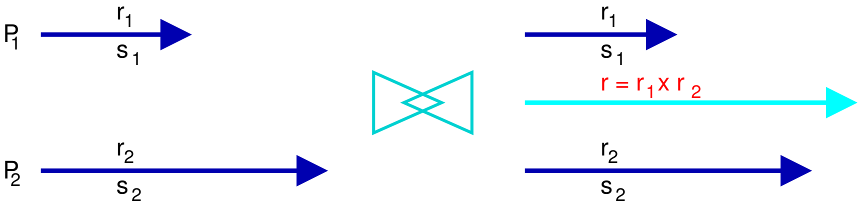
- The issue of what it means for two timed activities to synchronise is a vexed one. . .



algebraic considerations limit choices

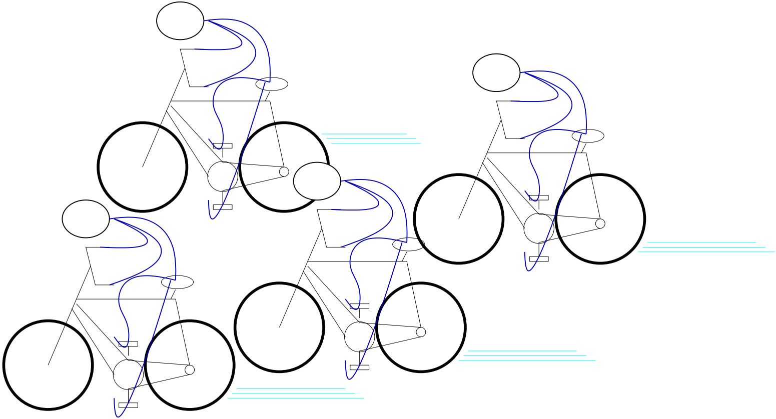
Timed Synchronisation

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

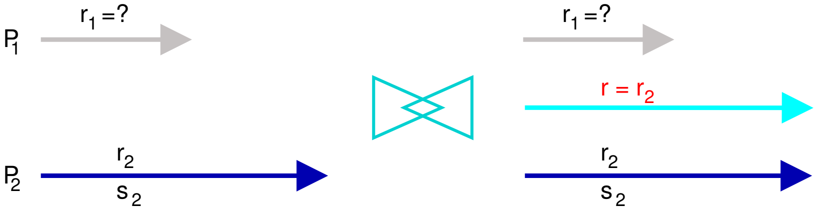


TIPP: new rate is product of individual rates

Timed Synchronisation

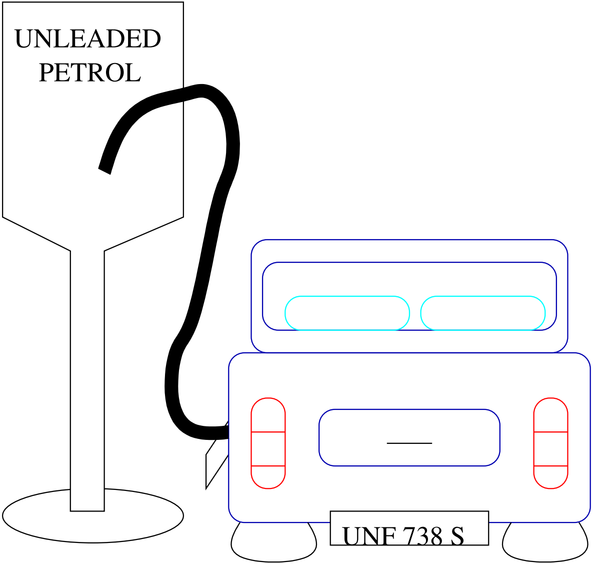


- The issue of what it means for two timed activities to synchronise is a vexed one. . .



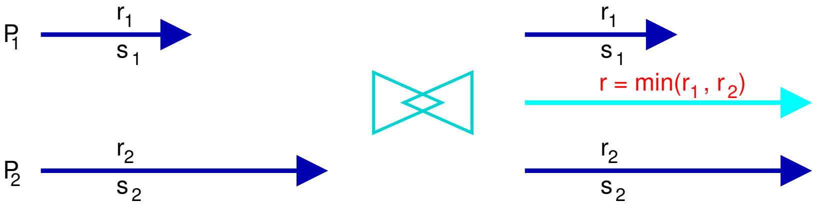
EMPA: one participant is passive

Timed Synchronisation



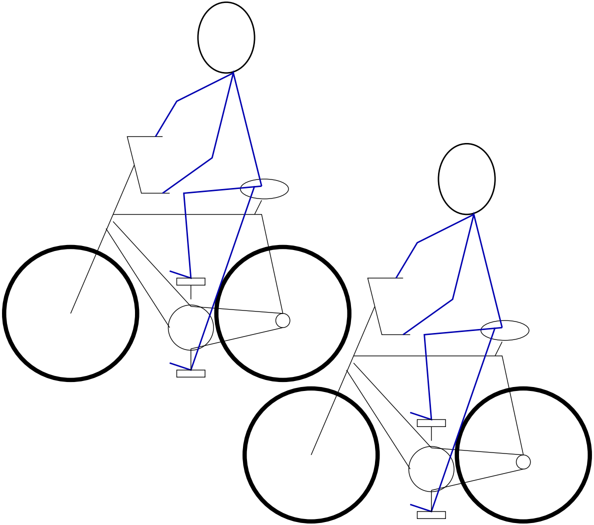
Timed Synchronisation

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



bounded capacity: new rate is the minimum of the rates

Timed Synchronisation



Cooperation in PEPA

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

Operational Semantics of PEPA

$$\mathrm{S}_0: \quad \frac{}{(\alpha,r).P \xrightarrow{(\alpha,r)} P}$$

$$\mathrm{A}_0: \quad \frac{P \xrightarrow{(\alpha,r)} P'}{A \xrightarrow{(\alpha,r)} P'},\ A \stackrel{def}{=} P$$

$$\mathrm{S}_1: \quad \frac{P \xrightarrow{(\alpha,r)} P'}{P+Q \xrightarrow{(\alpha,r)} P'}$$

$$\mathrm{S}_2: \quad \frac{Q \xrightarrow{(\alpha,r)} Q'}{P+Q \xrightarrow{(\alpha,r)} Q'}$$

$$\mathrm{C}_0: \quad \frac{P \xrightarrow{(\alpha,r)} P'}{P \underset{L}{\bowtie} Q \xrightarrow{(\alpha,r)} P' \underset{L}{\bowtie} Q},\ \alpha \notin L$$

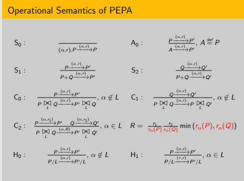
$$\mathrm{C}_1: \quad \frac{Q \xrightarrow{(\alpha,r)} Q'}{P \underset{L}{\bowtie} Q \xrightarrow{(\alpha,r)} P \underset{L}{\bowtie} Q'},\ \alpha \notin L$$

$$\mathrm{C}_2: \quad \frac{P \xrightarrow{(\alpha,r_1)} P' \qquad Q \xrightarrow{(\alpha,r_2)} Q'}{P \underset{L}{\bowtie} Q \xrightarrow{(\alpha,R)} P' \underset{L}{\bowtie} Q'},\ \alpha \in L$$

$$R = \frac{r_1}{r_\alpha(P)} \frac{r_2}{r_\alpha(Q)} \min\left(r_\alpha(P), r_\alpha(Q)\right)$$

$$\mathrm{H}_0: \quad \frac{P \xrightarrow{(\alpha,r)} P'}{P/L \xrightarrow{(\alpha,r)} P'/L},\ \alpha \notin L$$

$$\mathrm{H}_1: \quad \frac{P \xrightarrow{(\alpha,r)} P'}{P/L \xrightarrow{(\tau,r)} P'/L},\ \alpha \in L$$



We introduced part of the operational semantics in the previous lecture, where we covered S_0, A_0, S_1, and S_2. We called the terms involved sequential components.

The rules C_0 and C_1 are for the parallel operator of PEPA $\bowtie_L$. If an operand can do an action which is not synchronised, the the overall composite process can do the same action, but only the operand makes progress. As an exercise, derive such a transition from (1).

Rule C_2 is the crucial operator of the language and considers a synchronised action. If both processes do the same action, so does the composite process with a rate which is a function of the two rates of the operands. The calculation involves a function, the apparent rate, $r_\alpha(P)$ which has not been defined yet.

Before doing that, we see a few examples of patterns of communication that can be captured in PEPA.

Multiway Synchronisation

$$F \stackrel{def}{=} (fork, r_f).(join, r_j).F'$$

$$W_1 \stackrel{def}{=} (fork, r_{f_1}).(doWork_1, r_1).W_1'$$

$$W_2 \stackrel{def}{=} (fork, r_{f_2}).(doWork_2, r_2).W_2'$$

$$F' \stackrel{def}{=} \ldots, \quad W_1' \stackrel{def}{=} \ldots, \quad W_2' \stackrel{def}{=} \ldots$$

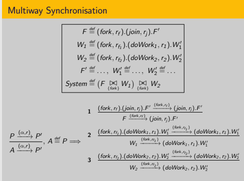
$$System \stackrel{def}{=} \left(F \underset{\{fork\}}{\bowtie} W_1\right) \underset{\{fork\}}{\bowtie} W_2$$

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

1 $$\frac{(fork, r_f).(join, r_j).F' \xrightarrow{(fork,r_f)} (join, r_j).F'}{F \xrightarrow{(fork,r_f)} (join, r_j).F'}$$

2 $$\frac{(fork, r_{f_1}).(doWork_1, r_1).W_1' \xrightarrow{(fork,r_{f_1})} (doWork_1, r_1).W_1'}{W_1 \xrightarrow{(fork,r_{f_1})} (doWork_1, r_1).W_1'}$$

3 $$\frac{(fork, r_{f_2}).(doWork_2, r_2).W_2' \xrightarrow{(fork,r_{f_2})} (doWork_2, r_2).W_2'}{W_2 \xrightarrow{(fork,r_{f_2})} (doWork_2, r_2).W_2'}$$



The primed processes are not shown. As an exercise, you may want to complete this model with their definition.

For the curious, you may want to check out this tool for PEPA:

- The PEPA Eclipse Plug-in
  `http://www.dcs.ed.ac.uk/pepa/tools/plugin/index.html`
- and some tutorial documentation
  `http://homepages.inf.ed.ac.uk/stg/pepa_eclipse/`

Multiway Synchronisation

$$F \stackrel{def}{=} (fork, r_f).(join, r_j).F'$$

$$W_1 \stackrel{def}{=} (fork, r_{f_1}).(doWork_1, r_1).W_1'$$

$$W_2 \stackrel{def}{=} (fork, r_{f_2}).(doWork_2, r_2).W_2'$$

$$F' \stackrel{def}{=} \ldots, \quad W_1' \stackrel{def}{=} \ldots, \quad W_2' \stackrel{def}{=} \ldots$$

$$System \stackrel{def}{=} \left(F \underset{\{fork\}}{\bowtie} W_1\right) \underset{\{fork\}}{\bowtie} W_2$$

$$\frac{F \xrightarrow{(fork, r_f)} (join, r_j)F' \qquad W_1 \xrightarrow{(fork, r_{f_1})} (doWork_1, r_1).W_1'}{F \underset{\{fork\}}{\bowtie} W_1 \xrightarrow{(fork, r')} (join, r_j).F' \underset{\{fork\}}{\bowtie} (doWork_1, r_1).W_1' \equiv \color{red}{LHS}}$$

$$\frac{\color{red}{LHS} \qquad W_2 \xrightarrow{(fork, r_{f_2})} (doWork_2, r_2).W_2'}{F \underset{\{fork\}}{\bowtie} W_1 \underset{\{fork\}}{\bowtie} W_2 \xrightarrow{(fork, r'')} (join, r_j).F' \underset{\{fork\}}{\bowtie} (doWork_1, r_1)W_1' \underset{\{fork\}}{\bowtie} (doWork_2, r_2).W_2'}$$

Other Communication Patterns

$$\begin{aligned}
Premium &\stackrel{def}{=} (dwn, r_p).Premium' \\
Basic &\stackrel{def}{=} (dwn, r_b).Basic' \\
S &\stackrel{def}{=} (dwn, r_s).S' \\
&\dots \\
System &\stackrel{def}{=} (Premium \parallel Basic) \underset{L}{\bowtie} S, \\
L &= \{dwn\}
\end{aligned}$$

$$\frac{P \xrightarrow{(\alpha,r)} P'}{P \underset{L}{\bowtie} Q \xrightarrow{(\alpha,r)} P' \underset{L}{\bowtie} Q}, \alpha \notin L$$

$$\frac{Q \xrightarrow{(\alpha,r)} Q'}{P \underset{L}{\bowtie} Q \xrightarrow{(\alpha,r)} P \underset{L}{\bowtie} Q'}, \alpha \notin L$$

$$\frac{P \xrightarrow{(\alpha,r_1)} P' \quad Q \xrightarrow{(\alpha,r_2)} Q'}{P \underset{L}{\bowtie} Q \xrightarrow{(\alpha,R)} P' \underset{L}{\bowtie} Q'}, \alpha \in L$$

$$R = \frac{r_1}{r_\alpha(P)} \frac{r_2}{r_\alpha(Q)} \min(r_\alpha(P), r_\alpha(Q))$$

$$\frac{\dfrac{Premium \xrightarrow{(dwn,r_p)} Premium'}{Premium \parallel Basic \xrightarrow{(dwn,r_p)} Premium' \parallel Basic} \qquad \overline{S \xrightarrow{(dwn,r_s)} S'}}{\dfrac{Premium \parallel Basic \underset{L}{\bowtie} S \xrightarrow{(dwn,r_{ps})} Premium' \parallel Basic \underset{L}{\bowtie} S'}{System \xrightarrow{(dwn,r_{ps})} Premium' \parallel Basic \underset{L}{\bowtie} S'}}$$

Rates in PEPA

PEPA supports the notion of infinite capacity:

$$(\alpha, r).P, \qquad \text{with } r \in \mathbb{R}_{>0} \cup \{n\top, n \in \mathbb{N}\}.$$

- A positive real denotes the rate of the exponential distribution associated with the activity.
- The top symbol $\top$ denotes an unspecified (or passive) rate. The rate will be assigned by other cooperating components in the system.
- Passive rates are given weights (naturals) which are useful to determine the relative probabilities of distinct passive activities to occur. ($1\top$ is usually written $\top$ for short.)

Arithmetic for Passive Rates

$$
\begin{aligned}
m\top + n\top &= (m+n)\top, && \text{for any } m,n \in \mathbb{N} \\
\frac{m\top}{n\top} &= \frac{m}{n}, && \text{for any } m,n \in \mathbb{N} \\
\min(r, n\top) &= r, && \text{for any } r \in \mathbb{R}_{>0} \text{ and } n \in \mathbb{N} \\
\min(m\top, n\top) &= \min(m,n)\top, && \text{for any } m,n \in \mathbb{N}
\end{aligned}
$$

- Summation and division between active and passive rates are not allowed.
- For expression of the following kind:

$$\frac{r}{s} \times \frac{m\top}{n\top}, \qquad r, s \in \mathbb{R}_{>0}, m, n \in \mathbb{N}$$

we assume that the two divisions have precedence over the multiplication.

Apparent Rate Calculation

$$\frac{P \xrightarrow{(\alpha,r_1)} P' \quad Q \xrightarrow{(\alpha,r_2)} Q'}{P \underset{L}{\bowtie} Q \xrightarrow{(\alpha,R)} P' \underset{L}{\bowtie} Q'}, \ \alpha \in L, \qquad R = \frac{r_1}{r_\alpha(P)} \frac{r_2}{r_\alpha(Q)} \min\left(r_\alpha(P), r_\alpha(Q)\right)$$

$$r_\alpha\left((\beta, r).P \right) = \begin{cases} r & \text{if} \quad \beta = \alpha \\ 0 & \text{if} \quad \beta \neq \alpha \end{cases}$$

$$r_\alpha(P + Q) = r_\alpha(P) + r_\alpha(Q)$$

$$r_\alpha(P \underset{L}{\bowtie} Q) = \begin{cases} \min\left(r_\alpha(P), r_\alpha(Q)\right) & \text{if } \alpha \in L \\ r_\alpha(P) + r_\alpha(Q) & \text{if } \alpha \notin L \end{cases}$$

$$r_\alpha(P/L) = \begin{cases} r_\alpha(P) & \text{if} \quad \alpha \notin L \\ 0 & \text{if} \quad \alpha \in L \end{cases}$$

Components which are both active and passive with respect to some action type are not allowed, e.g. $(\alpha, 1.0).P + (\alpha, \top).P$.

Examples

For r_1, r_2 positive reals,

$$\frac{(\alpha, r_1).P_1 \xrightarrow{(\alpha,r_1)} P_1 \qquad (\alpha, r_2).P_2 \xrightarrow{(\alpha,r_2)} P_2}{(\alpha, r_1).P_1 \underset{\{\alpha\}}{\bowtie} (\alpha, r_2).P_2 \xrightarrow{(\alpha,R)} P_1 \underset{\{\alpha\}}{\bowtie} P_2},$$

where

$$R = \frac{r_1}{r_\alpha\big((\alpha, r_1).P_1\big)} \frac{r_2}{r_\alpha\big((\alpha, r_2).P_2\big)} \min\Big(r_\alpha\big((\alpha, r_1).P_1\big), r_\alpha\big((\alpha, r_2).P_2\big)\Big)$$
$$= \frac{r_1}{r_1}\frac{r_2}{r_2}\min(r_1, r_2) = \min(r_1, r_2).$$

We recover the intuitive definition of the minimum between the two rates.

Examples

For r a positive real,

$$\frac{(\alpha, r).P_1 \xrightarrow{(\alpha,r)} P_1 \qquad (\alpha, \top).P_2 \xrightarrow{(\alpha,\top)} P_2}{(\alpha, r).P_1 \underset{\{\alpha\}}{\bowtie} (\alpha, \top).P_2 \xrightarrow{(\alpha,R)} P_1 \underset{\{\alpha\}}{\bowtie} P_2},$$

where

$$\begin{aligned} R &= \frac{r}{r_\alpha\big((\alpha, r).P_1\big)} \frac{\top}{r_\alpha\big((\alpha, \top).P_2\big)} \min\Big(r_\alpha\big((\alpha, r).P_1\big), r_\alpha\big((\alpha, \top).P_2\big)\Big) \\ &= \frac{r}{r}\frac{\top}{\top}\min(r, \top) = r. \end{aligned}$$

We recover the intuitive definition of infinite capacity — the rate of synchronisation is determined by the active component.

Examples

For r a positive real and any natural n,

$$\frac{(\alpha, r).P_1 \xrightarrow{(\alpha,r)} P_1 \qquad (\alpha, n\top).P_2 \xrightarrow{(\alpha,n\top)} P_2}{(\alpha, r).P_1 \underset{\{\alpha\}}{\bowtie} (\alpha, n\top).P_2 \xrightarrow{(\alpha,R)} P_1 \underset{\{\alpha\}}{\bowtie} P_2},$$

where

$$\begin{aligned} R &= \frac{r}{r_\alpha\big((\alpha, r).P_1\big)} \frac{n\top}{r_\alpha\big((\alpha, n\top).P_2\big)} \min\Big(r_\alpha\big((\alpha, r).P_1\big), r_\alpha\big((\alpha, n\top).P_2\big)\Big) \\ &= \frac{r}{r}\frac{n\top}{n\top} \min(r, n\top) = r. \end{aligned}$$

Passive weights may not affect the overall rate if only one passive component is present.

(Slightly More Complicated) Examples

$$\boxed{\begin{array}{l} Act \stackrel{def}{=} (\alpha, r).Act' \\ Pas \stackrel{def}{=} (\alpha, 1\top).Pas' + (\alpha, 2\top).Pas'' \\ Sys \stackrel{def}{=} Act \underset{\{\alpha\}}{\bowtie} Pas \end{array}}$$

$$\dfrac{\dfrac{(\alpha, r).Act' \xrightarrow{(\alpha,r)} Act'}{Act \xrightarrow{(\alpha,r)} Act'} \qquad \dfrac{\dfrac{\overline{(\alpha, 1\top).Pas' \xrightarrow{(\alpha,1\top)} Pas'}}{(\alpha, 1\top).Pas' + (\alpha, 2\top).Pas'' \xrightarrow{(\alpha,1\top)} Pas'}}{Pas \xrightarrow{(\alpha,1\top)} Pas'}}{\dfrac{Act \underset{\{\alpha\}}{\bowtie} Pas \xrightarrow{(\alpha,R')} Act' \underset{\{\alpha\}}{\bowtie} Pas'}{Sys \xrightarrow{(\alpha,R')} Act' \underset{\{\alpha\}}{\bowtie} Pas'}},$$

$$R' = \frac{r}{r_\alpha(Act)} \frac{1\top}{r_\alpha(Pas)} \min\left(r_\alpha(Act), r_\alpha(Pas)\right) = \frac{r}{r} \frac{1\top}{1\top + 2\top} \min(r, 1\top{+}2\top) = \frac{1}{3} r.$$

(Slightly More Complicated) Examples

$$
\begin{aligned}
Act &\stackrel{def}{=} (\alpha, r).Act' \\
Pas &\stackrel{def}{=} (\alpha, 1\top).Pas' + (\alpha, 2\top).Pas'' \\
Sys &\stackrel{def}{=} Act \underset{\{\alpha\}}{\bowtie} Pas
\end{aligned}
$$

It is also possible to prove the following derivation tree:

$$
\dfrac{\dfrac{\dfrac{}{(\alpha, r).Act' \xrightarrow{(\alpha,r)} Act'}}{Act \xrightarrow{(\alpha,r)} Act'} \qquad \dfrac{\dfrac{\dfrac{}{(\alpha, 2\top).Pas'' \xrightarrow{(\alpha,2\top)} Pas''}}{(\alpha, 1\top).Pas' + (\alpha, 2\top).Pas'' \xrightarrow{(\alpha,2\top)} Pas''}}{Pas \xrightarrow{(\alpha,2\top)} Pas''}}{\dfrac{Act \underset{\{\alpha\}}{\bowtie} Pas \xrightarrow{(\alpha,R'')} Act' \underset{\{\alpha\}}{\bowtie} Pas''}{Sys \xrightarrow{(\alpha,R'')} Act' \underset{\{\alpha\}}{\bowtie} Pas''}},
$$

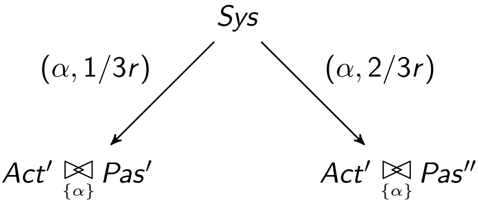
$$
R'' = \frac{r}{r_\alpha(Act)} \frac{2\top}{r_\alpha(Pas)} \min\left(r_\alpha(Act), r_\alpha(Pas)\right) = \frac{r}{r} \frac{2\top}{1\top + 2\top} \min(r, 1\top + 2\top) = \frac{2}{3} r.
$$

(Slightly More Complicated) Examples

$$Act \stackrel{\text{def}}{=} (\alpha, r).Act'$$

$$Pas \stackrel{\text{def}}{=} (\alpha, 1\top).Pas' + (\alpha, 2\top).Pas''$$

$$Sys \stackrel{\text{def}}{=} Act \underset{\{\alpha\}}{\bowtie} Pas$$



Apparent Rates in Active Cooperation

$$Cli \stackrel{def}{=} (\alpha, r_d).Cli'$$
$$Ser \stackrel{def}{=} (\alpha, r_u).Ser'$$
$$Sys \stackrel{def}{=} (Cli \parallel Cli) \underset{\{\alpha\}}{\bowtie} Ser$$

$$\frac{\dfrac{\dfrac{}{(\alpha, r_d).Cli' \xrightarrow{(\alpha, r_d)} Cli'}}{Cli \xrightarrow{(\alpha, r_d)} Cli'}}{Cli \parallel Cli \xrightarrow{(\alpha, r_d)} Cli' \parallel Cli} \quad \frac{(\alpha, r_u).Ser' \xrightarrow{(\alpha, r_u)} Ser'}{Ser \xrightarrow{(\alpha, r_u)} Ser'}$$

$$\frac{}{Cli \parallel Cli \underset{\{\alpha\}}{\bowtie} Ser \xrightarrow{(\alpha, R')} Cli' \parallel Cli \underset{\{\alpha\}}{\bowtie} Ser'},$$

$$R' = \frac{r_d}{r_d + r_d} \frac{r_u}{r_u} \min(r_d + r_d, r_u) = \frac{1}{2} \min(r_d + r_d, r_u)$$

Apparent Rates in Active Cooperation

$$
\begin{array}{|l|}
\hline
Cli \stackrel{def}{=} (\alpha, r_d).Cli' \\
Ser \stackrel{def}{=} (\alpha, r_u).Ser' \\
Sys \stackrel{def}{=} (Cli \parallel Cli) \underset{\{\alpha\}}{\bowtie} Ser \\
\hline
\end{array}
$$

The following derivation tree can also be proven:

$$
\dfrac{
\dfrac{\dfrac{\dfrac{}{(\alpha, r_d).Cli' \xrightarrow{(\alpha, r_d)} Cli'}}{Cli \xrightarrow{(\alpha, r_d)} Cli'}}{\color{red}{Cli \parallel Cli \xrightarrow{(\alpha, r_d)} Cli \parallel Cli'}}
\qquad
\dfrac{(\alpha, r_u).Ser' \xrightarrow{(\alpha, r_u)} Ser'}{Ser \xrightarrow{(\alpha, r_u)} Ser'}
}{
Cli \parallel Cli \underset{\{\alpha\}}{\bowtie} Ser \xrightarrow{(\alpha, R'')} Cli \parallel Cli' \underset{\{\alpha\}}{\bowtie} Ser'
},
$$

$$
R'' = \frac{r_d}{r_d + r_d} \frac{r_u}{r_u} \min(r_d + r_d, r_u) = \frac{1}{2} \min(r_d + r_d, r_u) = R'
$$

Apparent Rates in Active Cooperation

$$Cli \stackrel{def}{=} (\alpha, r_d).Cli'$$

$$Ser \stackrel{def}{=} (\alpha, r_u).Ser'$$

$$Sys \stackrel{def}{=} (Cli \parallel Cli) \underset{\{\alpha\}}{\bowtie} Ser$$

$$Cli \parallel Cli \underset{\{\alpha\}}{\bowtie} Ser$$

$$\xrightarrow{(\alpha, 1/2 \min(2\, r_d, r_u))} Cli' \parallel Cli \underset{\{\alpha\}}{\bowtie} Ser'$$

$$\xrightarrow{(\alpha, 1/2 \min(2\, r_d, r_u))} Cli \parallel Cli' \underset{\{\alpha\}}{\bowtie} Ser'$$

Labelled Transition System: Details

Derivative Set

Given a PEPA component P, the derivative set of P, denoted by $ds(P)$ is defined as the smallest set of components such that

- $P \in ds(P)$;
- if $P \xrightarrow{(\alpha,r)} P'$ then $P' \in ds(P)$.

Derivation Graph

Let $\mathcal{A}$ be a set of action labels and $\mathcal{A}ct = \{| \, (\alpha, r) : \alpha \in \mathcal{A}, r \in \mathbb{R}_{>0} |\}$. The derivation graph of a component P has $ds(P)$ as the set of nodes.
The multiset of arcs $A \in ds(P) \times ds(P) \times \mathcal{A}ct$ is such that

$$P \xrightarrow{(\alpha,r)} P' \implies \big(P, P', (\alpha, r)\big) \in A,$$

with multiplicity equal to the number of distinct derivations $P \xrightarrow{(\alpha,r)} P'$.

Why Multisets

$$P \stackrel{\text{def}}{=} (\alpha, r).P' \;\Big|\; P \stackrel{\text{def}}{=} (\alpha, r).P' + (\alpha, r).P' \;\Big|\; \ldots \;\Big|\; P \stackrel{\text{def}}{=} \textstyle\sum_n (\alpha, r).P'$$

- If distinct inference trees were not taken into account, then the derivation graph would have only one transition $P \xrightarrow{(\alpha,r)} P'$.
- With a multiset, we have one, two, ..., n such transitions, respectively.
- Intuitively, this capture the fact that process P has different apparent rates in these cases.

An Algorithm for State-Space Derivation

```
ds(P_0) ⇐ {P_0}
push P_0 onto Stack
while Stack is not empty do
   pop P off Stack
   infer multiset (P, P', (α, r)) from P
   for all (P, P', (α, r)) do
      if P' ∉ ds(P_0) then
         push P' onto Stack
         add P' to ds(P_0)
      end if
   end for
end while
```

The Underlying Markov Process

- Let P_0 be the initial state of the system.
- Assign a state to each process in $\mathrm{ds}(P_0)$.
- For each triple $\big(P, P', (\alpha, r)\big)$ with multiplicity m, assign rate $m\,r$ to the transition between P and P'.

Well-Formedness

- Note that all leaves of the derivation trees must have rates in the (strictly) positive reals.
- This means that passive actions must eventually synchronise with an active ones.
- Models that do not satisfy this condition are rejected.
- For example,

$$(\alpha, \top).P \underset{\{\alpha\}}{\bowtie} (\alpha, \top).Q$$

will be rejected for any P and Q.

Consumer/Producer in PEPA

$$Cons_1 \stackrel{def}{=} (get, r_g).Cons_2$$
$$Cons_2 \stackrel{def}{=} (cons, r_c).Cons_1$$
$$Prod_1 \stackrel{def}{=} (make, r_m).Prod_2$$
$$Prod_2 \stackrel{def}{=} (put, r_p).Prod_1$$
$$Buf_2 \stackrel{def}{=} (get, \top).Buf_1$$
$$Buf_1 \stackrel{def}{=} (get, \top).Buf_0 + (put, \top).Buf_2$$
$$Buf_0 \stackrel{def}{=} (put, \top).Buf_1$$
$$Sys \stackrel{def}{=} Cons_1 \underset{\{get\}}{\bowtie} Buf_2 \underset{\{put\}}{\bowtie} Prod_1$$

Possible variants:

- A buffer with n places:

$$Buf_n \stackrel{def}{=} (get, \top).Buf_{n-1}$$
$$Buf_i \stackrel{def}{=} (get, \top).Buf_{i-1} + (put, \top).Buf_{i+1}, \quad \text{for } 1 \leq i \leq n-1$$
$$Buf_0 \stackrel{def}{=} (put, \top).Buf_1$$

- and k consumers:

$$\overbrace{Cons_1 \parallel Cons_1 \parallel \ldots \parallel Cons_1}^{k} \underset{\{get\}}{\bowtie} Buf_n \underset{\{put\}}{\bowtie} Prod_1$$

Consumer/Producer in PEPA

$$
\begin{array}{llllll}
Cons_1 & \stackrel{def}{=} & (get, r_g).Cons_2 & Prod_1 & \stackrel{def}{=} & (make, r_m).Prod_2 \\
Cons_2 & \stackrel{def}{=} & (cons, r_c).Cons_1 & Prod_2 & \stackrel{def}{=} & (put, r_p).Prod_1 \\
Buf_2 & \stackrel{def}{=} & (get, \top).Buf_1 & Buf_1 & \stackrel{def}{=} & (get, \top).Buf_0 + (put, \top).Buf_2 \\
Buf_0 & \stackrel{def}{=} & (put, \top).Buf_1 & Sys & \stackrel{def}{=} & Cons_1 \underset{\{get\}}{\bowtie} Buf_2 \underset{\{put\}}{\bowtie} Prod_1
\end{array}
$$

$$
\cfrac{\cfrac{\cfrac{Cons_1 \xrightarrow{(get,r_g)} Cons_2 \qquad Buf_2 \xrightarrow{(get,\top)} Buf_1}{Cons_1 \underset{\{get\}}{\bowtie} Buf_2 \xrightarrow{(get,r_g)} Cons_2 \underset{\{get\}}{\bowtie} Buf_1}}{Cons_1 \underset{\{get\}}{\bowtie} Buf_2 \underset{\{put\}}{\bowtie} Prod_1 \xrightarrow{(get,r_g)} Cons_2 \underset{\{get\}}{\bowtie} Buf_1 \underset{\{put\}}{\bowtie} Prod_1}}{Sys \xrightarrow{(get,r_g)} Cons_2 \underset{\{get\}}{\bowtie} Buf_1 \underset{\{put\}}{\bowtie} Prod_1}
$$

Can we prove anything else for *Sys*?

Consumer/Producer in PEPA

$$
\begin{array}{lclllcl}
Cons_1 & \stackrel{def}{=} & (get, r_g).Cons_2 & & Prod_1 & \stackrel{def}{=} & (make, r_m).Prod_2 \\
Cons_2 & \stackrel{def}{=} & (cons, r_c).Cons_1 & & Prod_2 & \stackrel{def}{=} & (put, r_p).Prod_1 \\
Buf_2 & \stackrel{def}{=} & (get, \top).Buf_1 & & Buf_1 & \stackrel{def}{=} & (get, \top).Buf_0 + (put, \top).Buf_2 \\
Buf_0 & \stackrel{def}{=} & (put, \top).Buf_1 & & Sys & \stackrel{def}{=} & Cons_1 \underset{\{get\}}{\bowtie} Buf_2 \underset{\{put\}}{\bowtie} Prod_1
\end{array}
$$

$$
\frac{\dfrac{Prod_1 \xrightarrow{(make, r_m)} Prod_2}{Cons_1 \underset{\{get\}}{\bowtie} Buf_2 \underset{\{get\}}{\bowtie} Prod_1 \xrightarrow{(make, r_m)} Cons_1 \underset{\{get\}}{\bowtie} Buf_2 \underset{\{put\}}{\bowtie} Prod_2}}{Sys \xrightarrow{(make, r_m)} Cons_1 \underset{\{get\}}{\bowtie} Buf_2 \underset{\{put\}}{\bowtie} Prod_2}
$$

Summarising, the following transitions were found:

$$
Sys \xrightarrow{(get, r_g)} Cons_2 \underset{\{get\}}{\bowtie} Buf_1 \underset{\{put\}}{\bowtie} Prod_1
$$

$$
Sys \xrightarrow{(make, r_m)} Cons_1 \underset{\{get\}}{\bowtie} Buf_2 \underset{\{put\}}{\bowtie} Prod_2
$$

Consumer/Producer in PEPA

$$
\begin{array}{rclrcl}
Cons_1 & \stackrel{def}{=} & (get, r_g).Cons_2 & Prod_1 & \stackrel{def}{=} & (make, r_m).Prod_2 \\
Cons_2 & \stackrel{def}{=} & (cons, r_c).Cons_1 & Prod_2 & \stackrel{def}{=} & (put, r_p).Prod_1 \\
Buf_2 & \stackrel{def}{=} & (get, \top).Buf_1 & Buf_1 & \stackrel{def}{=} & (get, \top).Buf_0 + (put, \top).Buf_2 \\
Buf_0 & \stackrel{def}{=} & (put, \top).Buf_1 & Sys & \stackrel{def}{=} & Cons_1 \underset{\{get\}}{\bowtie} Buf_2 \underset{\{put\}}{\bowtie} Prod_1
\end{array}
$$

Popping $Cons_2 \underset{\{get\}}{\bowtie} Buf_1 \underset{\{put\}}{\bowtie} Prod_1$ off the stack,

$$
\frac{Cons_2 \xrightarrow{(cons, r_c)} Cons_1}{Cons_2 \underset{\{get\}}{\bowtie} Buf_1 \underset{\{put\}}{\bowtie} Prod_1 \xrightarrow{(cons, r_c)} Cons_1 \underset{\{get\}}{\bowtie} Buf_1 \underset{\{put\}}{\bowtie} Prod_1},
$$

$$
\frac{Prod_1 \xrightarrow{(make, r_m)} Prod_2}{Cons_2 \underset{\{get\}}{\bowtie} Buf_1 \underset{\{put\}}{\bowtie} Prod_1 \xrightarrow{(make, r_m)} Cons_2 \underset{\{get\}}{\bowtie} Buf_1 \underset{\{put\}}{\bowtie} Prod_2}.
$$

Consumer/Producer in PEPA

$$
\begin{array}{rclrcl}
Cons_1 & \stackrel{def}{=} & (get, r_g).Cons_2 & Prod_1 & \stackrel{def}{=} & (make, r_m).Prod_2 \\
Cons_2 & \stackrel{def}{=} & (cons, r_c).Cons_1 & Prod_2 & \stackrel{def}{=} & (put, r_p).Prod_1 \\
Buf_2 & \stackrel{def}{=} & (get, \top).Buf_1 & Buf_1 & \stackrel{def}{=} & (get, \top).Buf_0 + (put, \top).Buf_2 \\
Buf_0 & \stackrel{def}{=} & (put, \top).Buf_1 & Sys & \stackrel{def}{=} & Cons_1 \underset{\{get\}}{\bowtie} Buf_2 \underset{\{put\}}{\bowtie} Prod_1
\end{array}
$$

Therefore, we still need to infer transitions for the following processes...

$$Cons_1 \underset{\{get\}}{\bowtie} Buf_2 \underset{\{put\}}{\bowtie} Prod_2$$

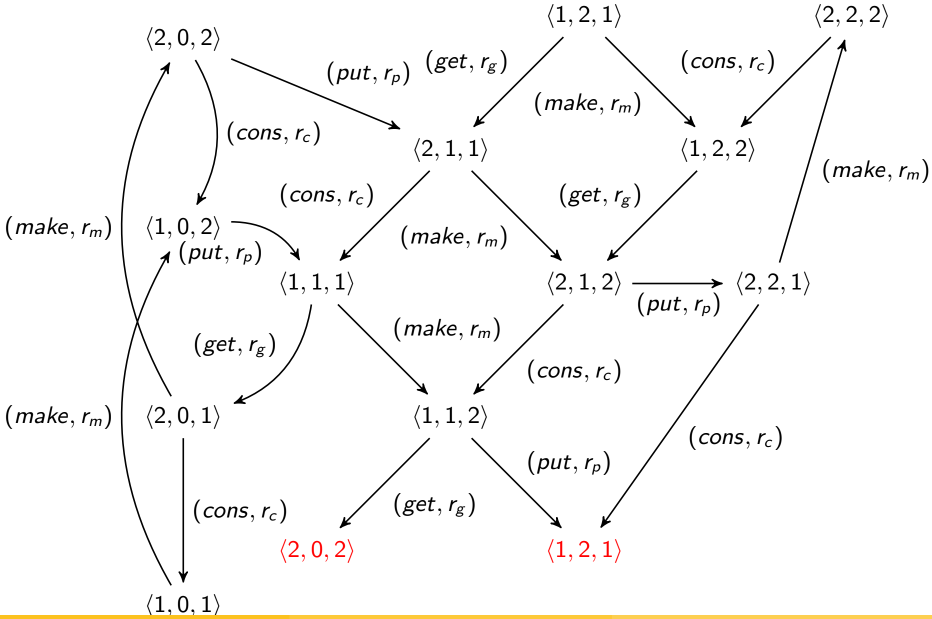
$$Cons_1 \underset{\{get\}}{\bowtie} Buf_1 \underset{\{put\}}{\bowtie} Prod_1$$

$$Cons_2 \underset{\{get\}}{\bowtie} Buf_1 \underset{\{put\}}{\bowtie} Prod_2$$

...and all those that are found along the way.

Notice that the cooperation structure is fixed across all processes. Thus, we may denote a state by $\langle i, j, k \rangle$ to indicate $Cons_i \underset{\{get\}}{\bowtie} Buf_j \underset{\{put\}}{\bowtie} Prod_k$.

Consumer/Producer in PEPA: Complete Derivation Graph



Consumer/Producer in PEPA: State-Transition Diagram