Stochastic Probes

Jeremy Bradley

jb@doc.ic.ac.uk

Imperial College London

LFCS, Edinburgh – 28 October 2003

Stochastic Probes

Stochastic Probes are:

- fragments of process algebra
- used to measure observed timed and probabilistic behaviour in stochastic process algebra models

Stochastic Probes

Stochastic Probes are:

- specified using action regular expressions
- currently implemented in PEPA, but are formalism independent
- not able to enforce behaviour – must be *behaviour neutral*

Stochastic Analysis

- Steady-state analysis
  - long-run average probability of ending up in set of states
- Transient-state analysis
  - Probability of being in a given set of states at a time, t
- Passage-time analysis
  - How long does it take to arrive at a set of states

PEPA: Process Algebra

- Syntax:

$$P \ ::= \ (a, \lambda).P \ \Big| \ P + P \ \Big| \ P \underset{S}{\bowtie} P \ \Big| \ P/L \ \Big| \ A$$

- $(a, \lambda).P$: prefix operation $P + P$: competitive choice

- $P \underset{S}{\bowtie} P$: component cooperation P/L: action hiding

PEPA: A Transmitter-Receiver

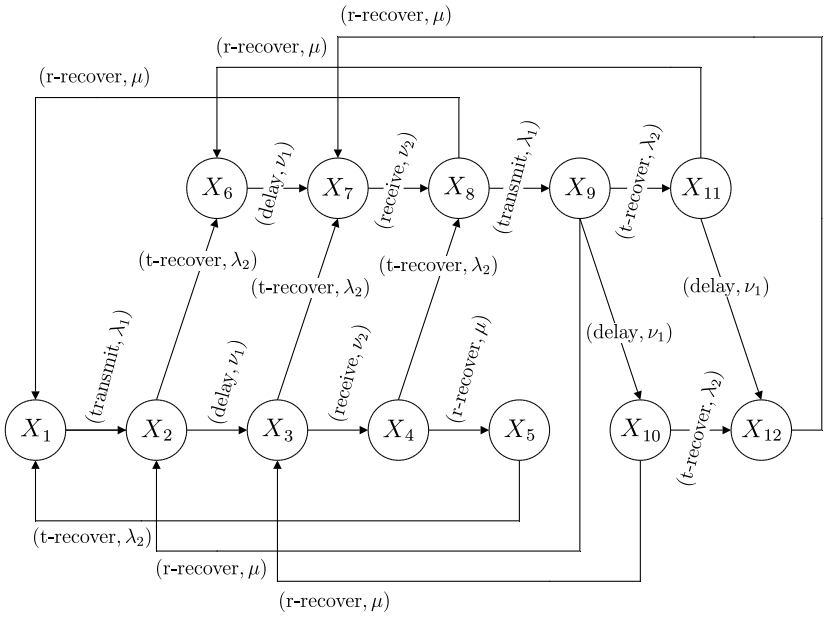
$$\text{Transmitter} \stackrel{\text{def}}{=} (\text{transmit}, \lambda_1).(\text{t_recover}, \lambda_2).\text{Transmitter}$$

$$\text{Receiver} \stackrel{\text{def}}{=} (\text{receive}, \top).(\text{r_recover}, \mu).\text{Receiver}$$

$$\text{Network} \stackrel{\text{def}}{=} (\text{transmit}, \top).(\text{delay}, \nu_1).(\text{receive}, \nu_2).\text{Network}$$

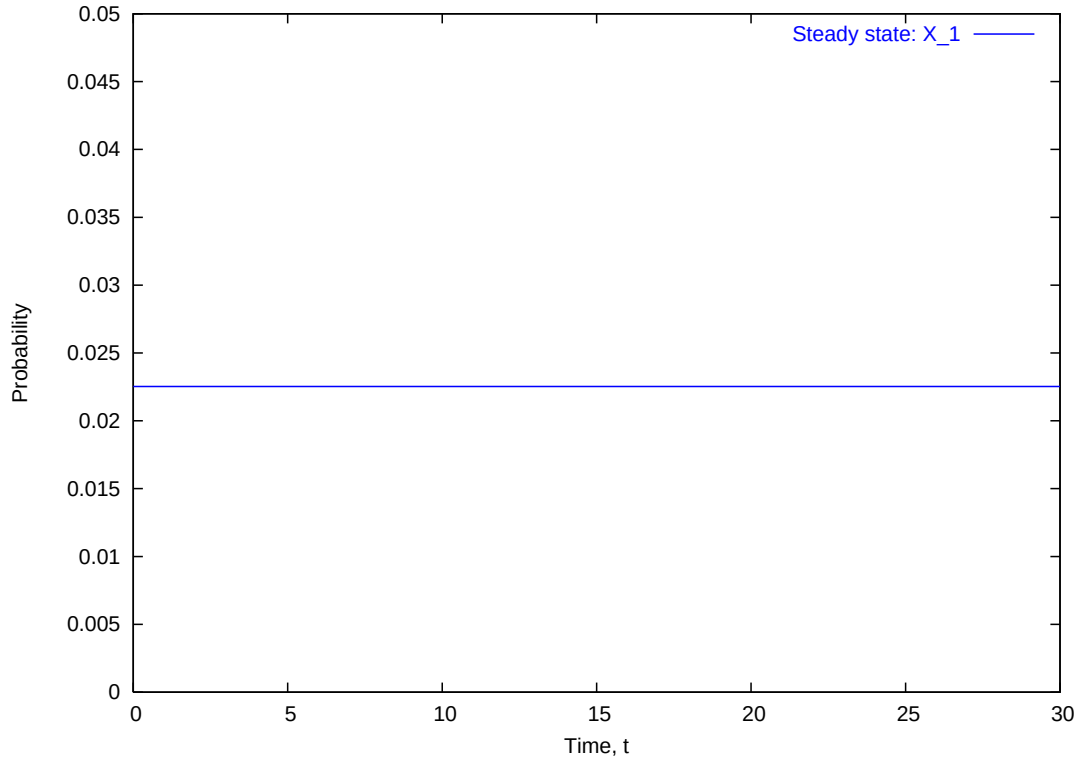
$$\text{System} \stackrel{\text{def}}{=} (\text{Transmitter} \underset{\emptyset}{\bowtie} \text{Receiver}) \underset{\{\text{transmit,receive}\}}{\bowtie} \text{Network}$$

Global state space

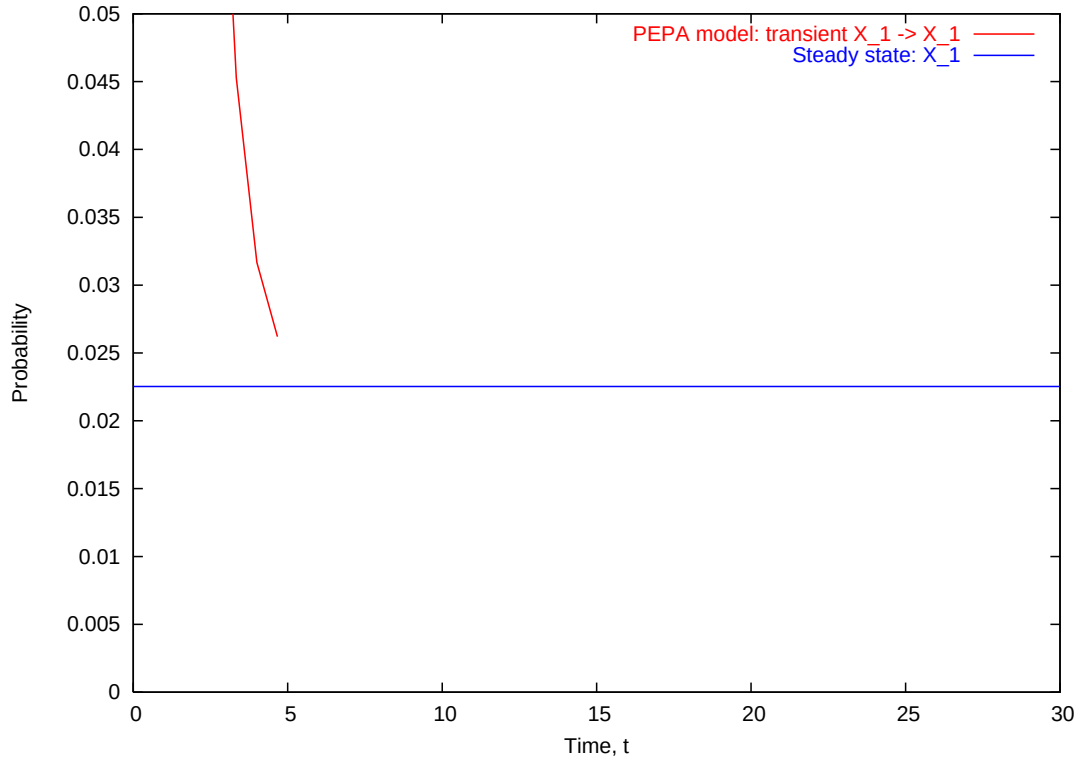


Look at probability of being in state X_1 at time, t

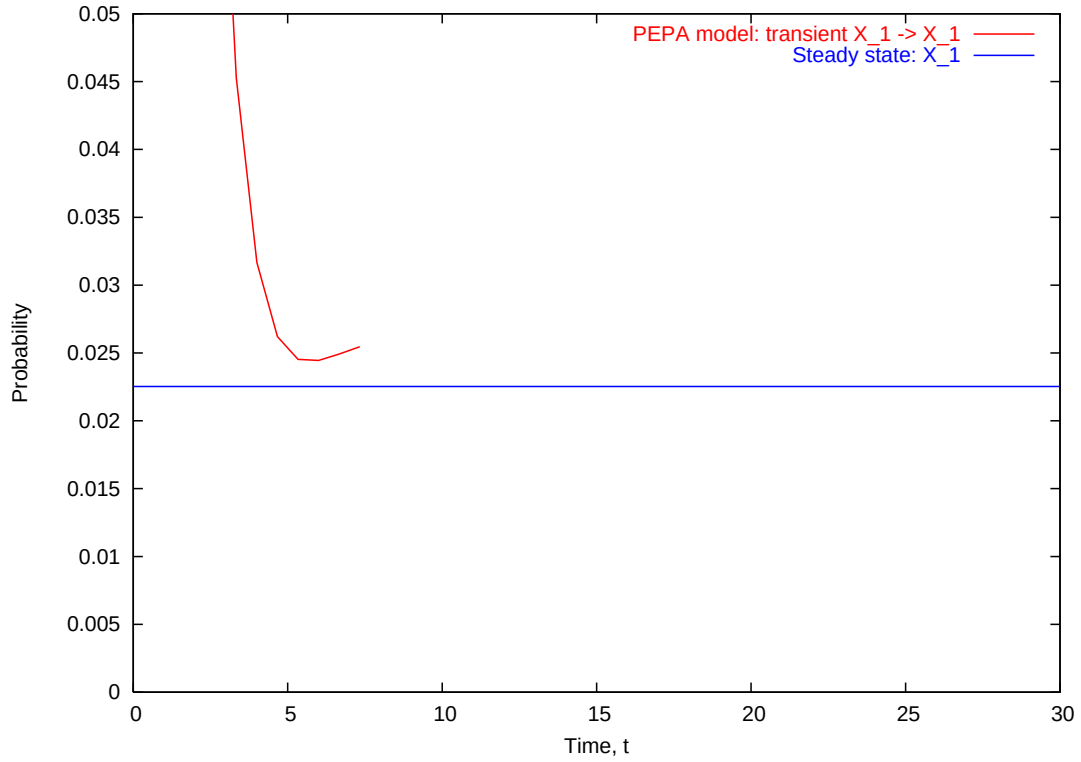
Transient distribution



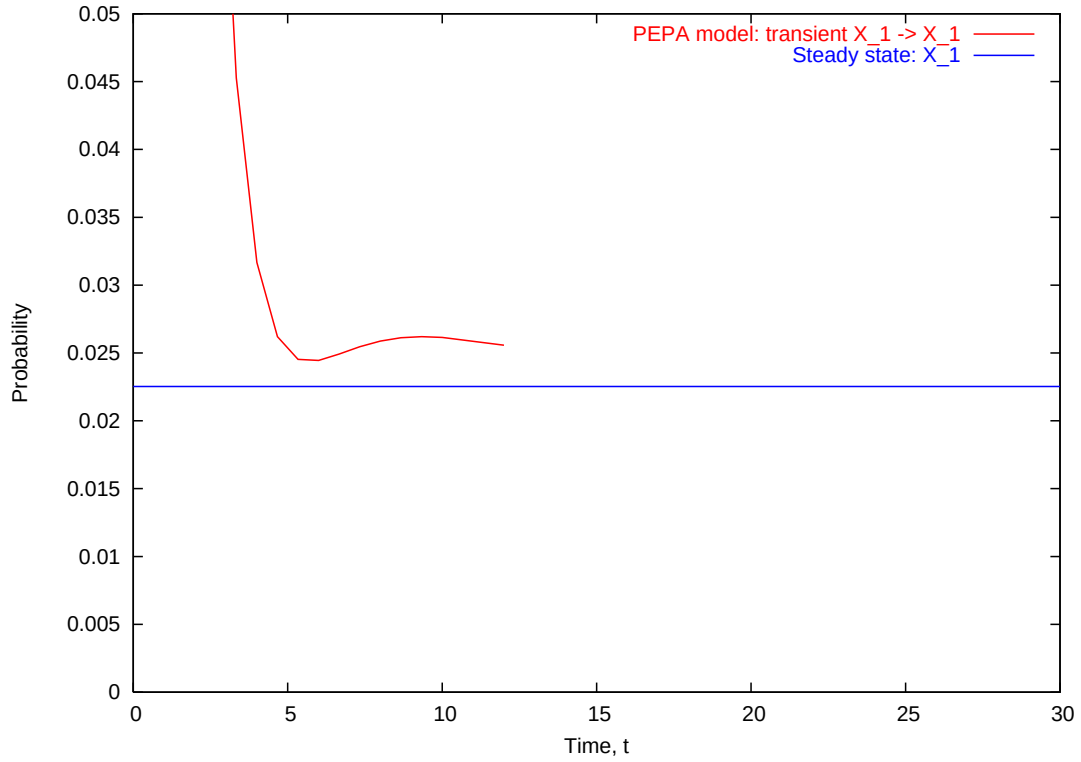
Transient distribution



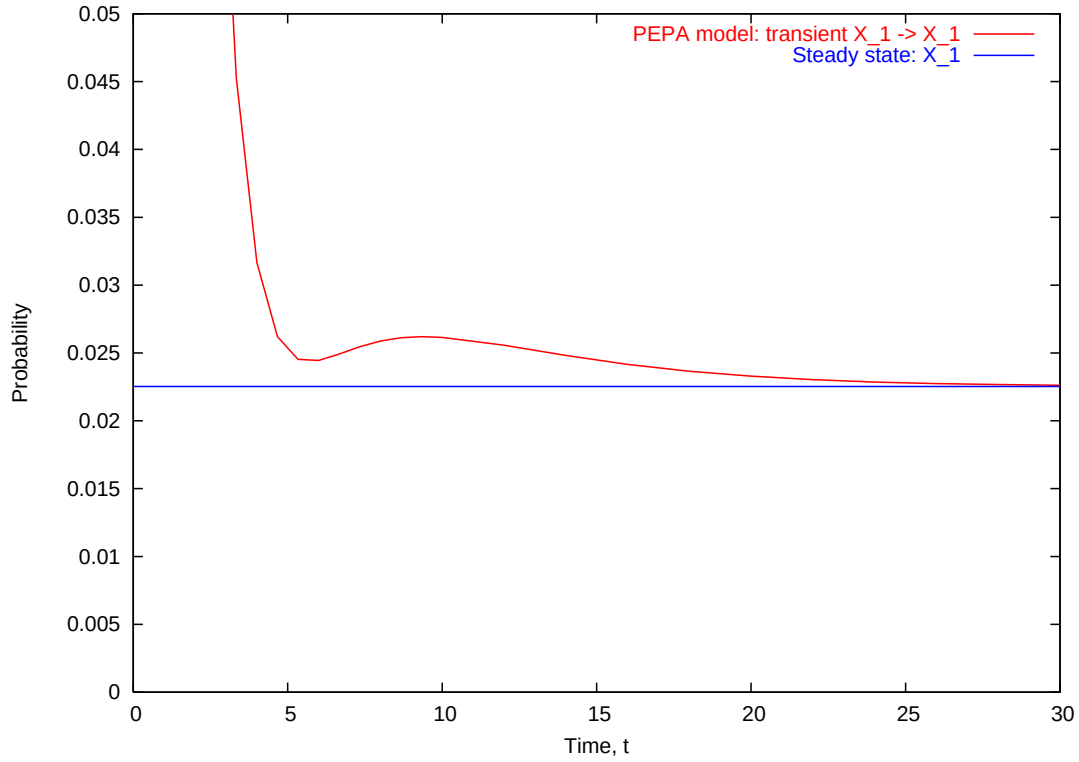
Transient distribution



Transient distribution



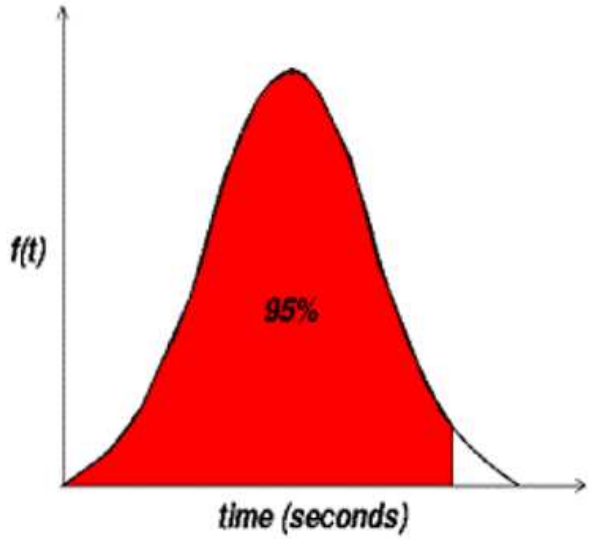
Transient distribution



Passage-time analysis

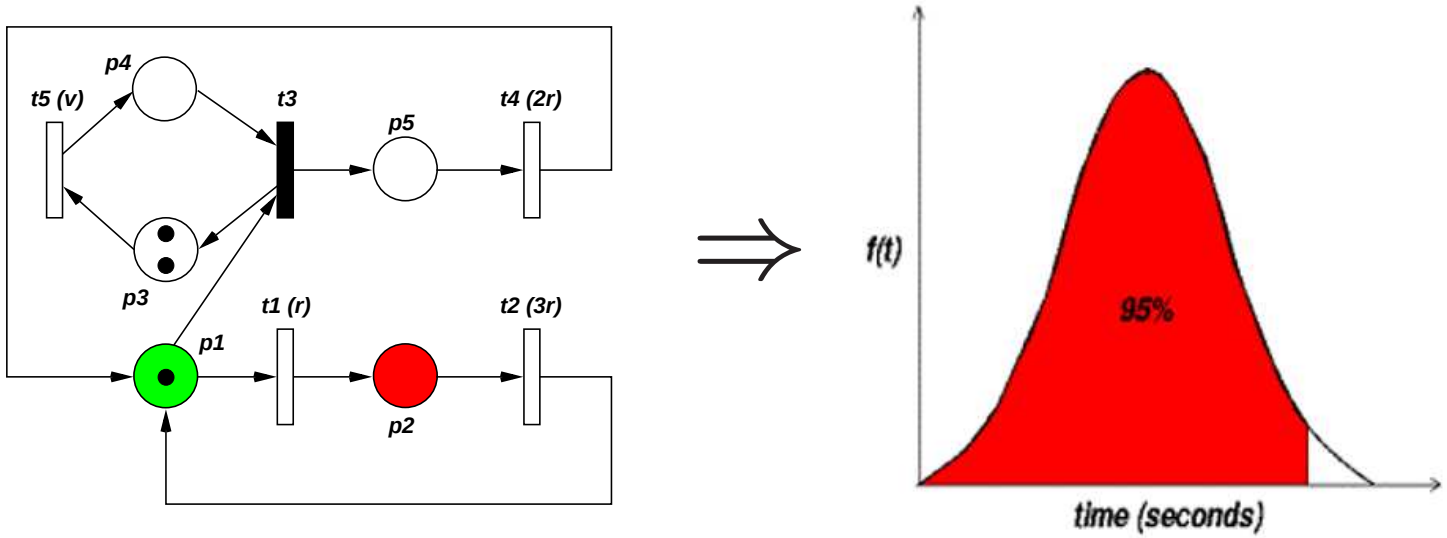
For example, allow us to extract passage-time density from a PEPA model:

$$
\begin{array}{lcl}
\mathrm{A1} & \stackrel{\mathrm{def}}{=} & (\mathrm{start}, r_1).\mathrm{A2} + (\mathrm{pause}, r_2).\mathrm{A3} \\
\mathrm{A2} & \stackrel{\mathrm{def}}{=} & (\mathrm{run}, r_3).\mathrm{A1} + (\mathrm{fail}, r_4).\mathrm{A3} \\
\mathrm{A3} & \stackrel{\mathrm{def}}{=} & (\mathrm{recover}, r_1).\mathrm{A1} \\
\mathrm{AA} & \stackrel{\mathrm{def}}{=} & (\mathrm{run}, \top).(\mathrm{alert}, r_5).\mathrm{AA} \\
\mathrm{Sys} & \stackrel{\mathrm{def}}{=} & \mathrm{AA} \underset{\{run\}}{\bowtie} \mathrm{A1}
\end{array}
\quad \Rightarrow \quad
$$



What is DNAmaca/HYDRA?

Extracting a passage-time density from a GSPN:



PEPA: Example

$$\begin{aligned}
\mathsf{A1} &\stackrel{\text{def}}{=} (\mathsf{start}, r_1).\mathsf{A2} + (\mathsf{pause}, r_2).\mathsf{A3} \\
\mathsf{A2} &\stackrel{\text{def}}{=} (\mathsf{run}, r_3).\mathsf{A1} + (\mathsf{fail}, r_4).\mathsf{A3} \\
\mathsf{A3} &\stackrel{\text{def}}{=} (\mathsf{recover}, r_1).\mathsf{A1} \\
\mathsf{AA} &\stackrel{\text{def}}{=} (\mathsf{run}, \top).(\mathsf{alert}, r_5).\mathsf{AA}
\end{aligned}$$

PEPA: Example

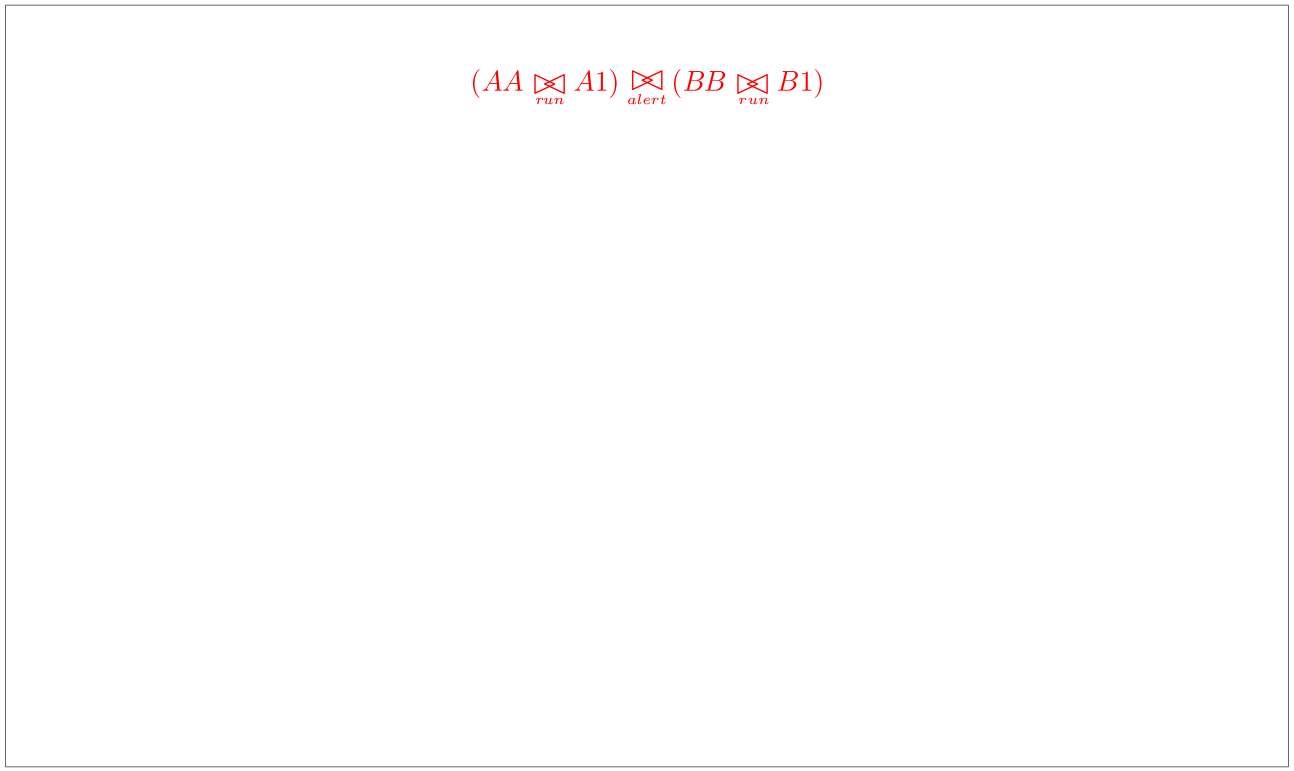
$$
\begin{aligned}
\mathsf{A1} &\stackrel{\text{def}}{=} (\mathsf{start}, r_1).\mathsf{A2} + (\mathsf{pause}, r_2).\mathsf{A3} \\
\mathsf{A2} &\stackrel{\text{def}}{=} (\mathsf{run}, r_3).\mathsf{A1} + (\mathsf{fail}, r_4).\mathsf{A3} \\
\mathsf{A3} &\stackrel{\text{def}}{=} (\mathsf{recover}, r_1).\mathsf{A1} \\
\mathsf{AA} &\stackrel{\text{def}}{=} (\mathsf{run}, \top).(\mathsf{alert}, r_5).\mathsf{AA}
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{B1} &\stackrel{\text{def}}{=} (\mathsf{start}, r_1).\mathsf{B2} + (\mathsf{pause}, r_2).\mathsf{B1} \\
\mathsf{B2} &\stackrel{\text{def}}{=} (\mathsf{run}, r_3).\mathsf{B1} \\
\mathsf{BB} &\stackrel{\text{def}}{=} (\mathsf{run}, \top).(\mathsf{alert}, r_5).\mathsf{BB}
\end{aligned}
$$

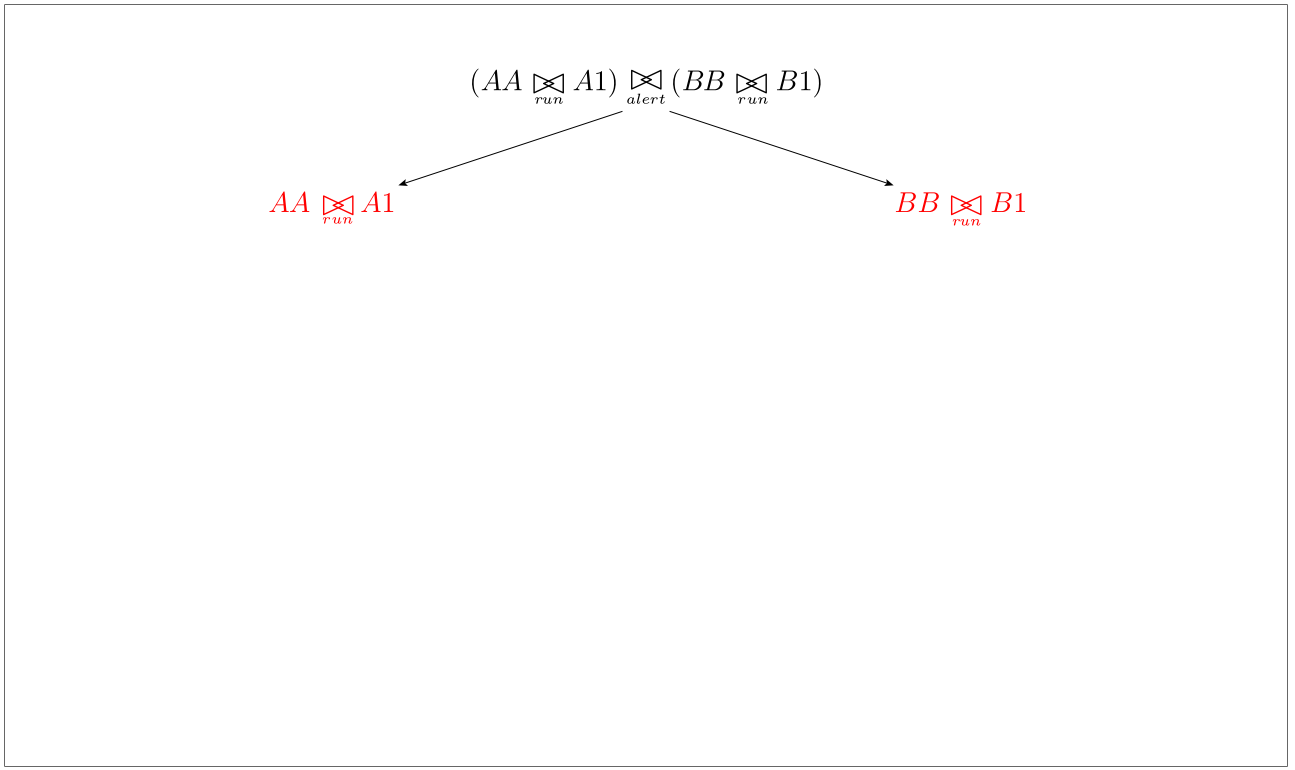
PEPA: Example

$$\begin{aligned}
\mathsf{A1} &\stackrel{\text{def}}{=} (\mathsf{start}, r_1).\mathsf{A2} + (\mathsf{pause}, r_2).\mathsf{A3} \\
\mathsf{A2} &\stackrel{\text{def}}{=} (\mathsf{run}, r_3).\mathsf{A1} + (\mathsf{fail}, r_4).\mathsf{A3} \\
\mathsf{A3} &\stackrel{\text{def}}{=} (\mathsf{recover}, r_1).\mathsf{A1} \\
\mathsf{AA} &\stackrel{\text{def}}{=} (\mathsf{run}, \top).(\mathsf{alert}, r_5).\mathsf{AA} \\
\\
\mathsf{B1} &\stackrel{\text{def}}{=} (\mathsf{start}, r_1).\mathsf{B2} + (\mathsf{pause}, r_2).\mathsf{B1} \\
\mathsf{B2} &\stackrel{\text{def}}{=} (\mathsf{run}, r_3).\mathsf{B1} \\
\mathsf{BB} &\stackrel{\text{def}}{=} (\mathsf{run}, \top).(\mathsf{alert}, r_5).\mathsf{BB} \\
\\
\mathsf{Sys} &\stackrel{\text{def}}{=} (\mathsf{AA} \underset{\mathsf{run}}{\bowtie} \mathsf{A1}) \underset{\mathsf{alert}}{\bowtie} (\mathsf{BB} \underset{\mathsf{run}}{\bowtie} \mathsf{B1})
\end{aligned}$$

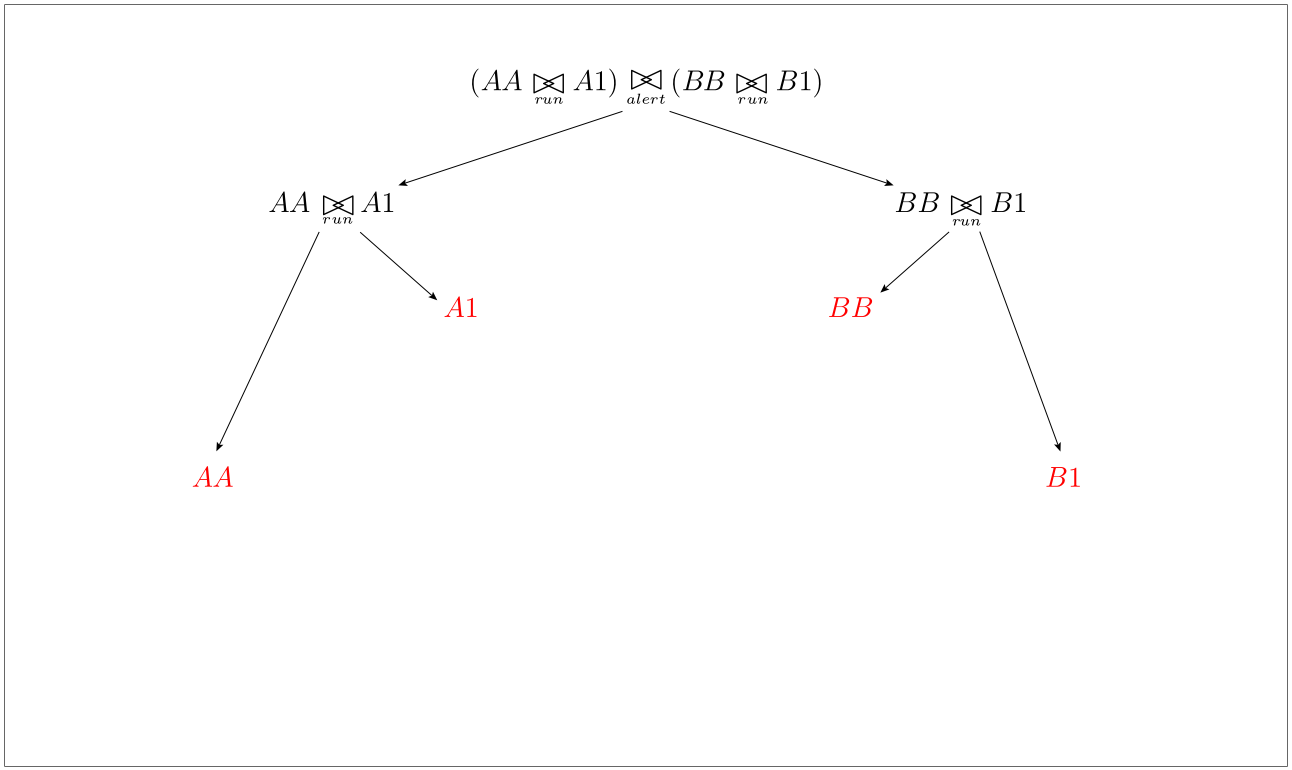
Example ipc Compilation



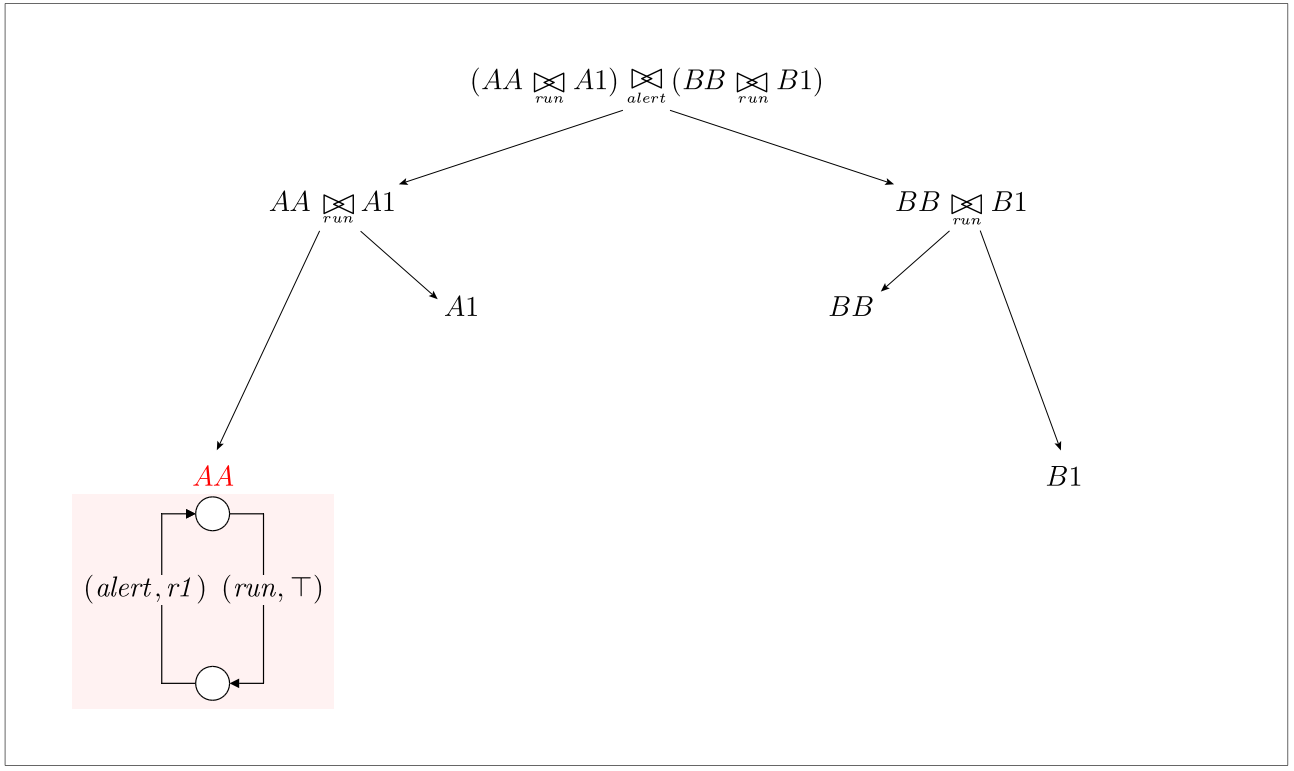
Example ipc Compilation



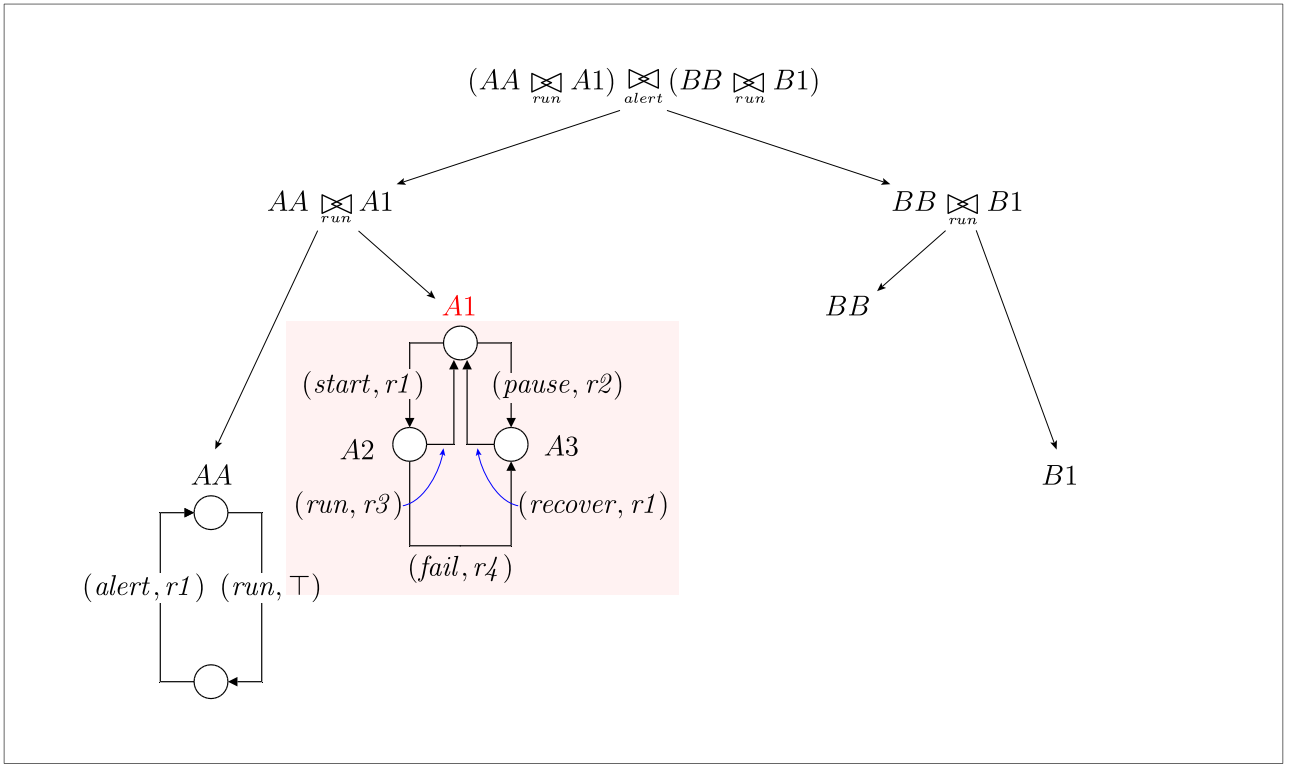
Example ipc Compilation



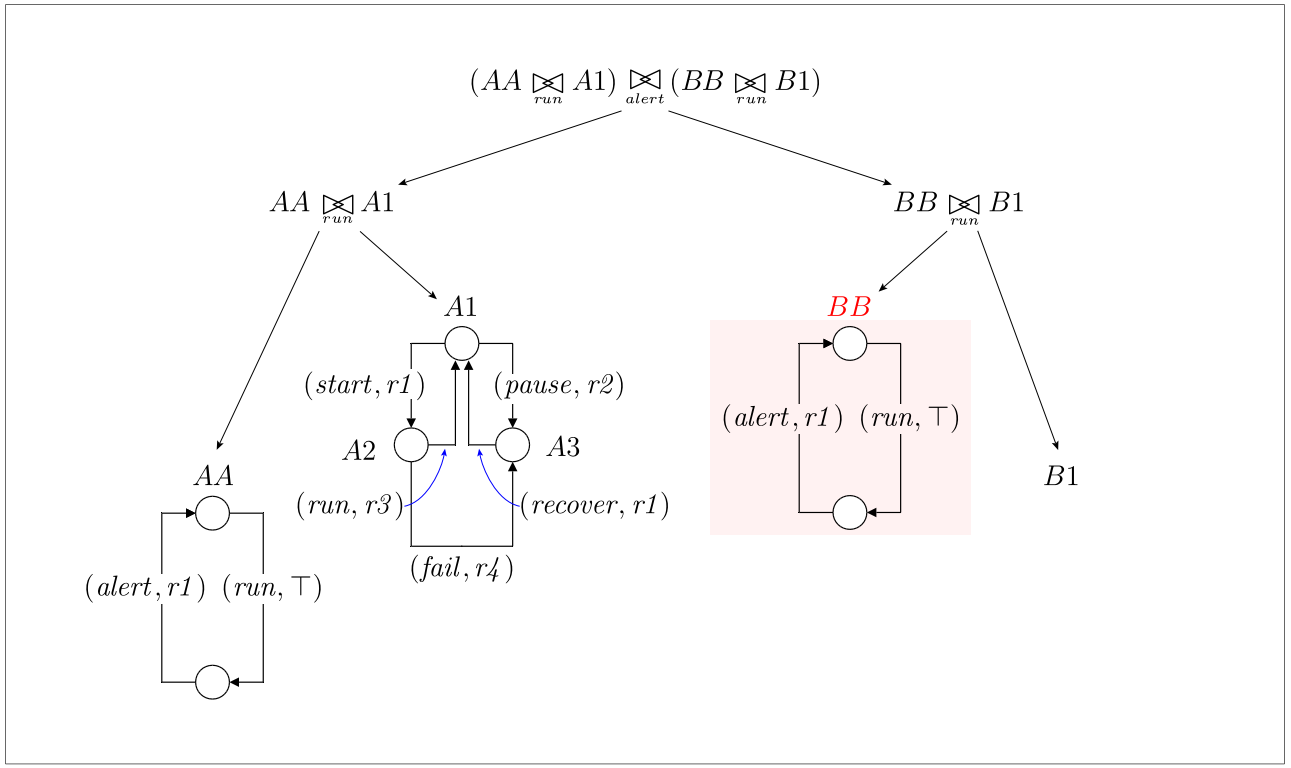
Example ipc Compilation



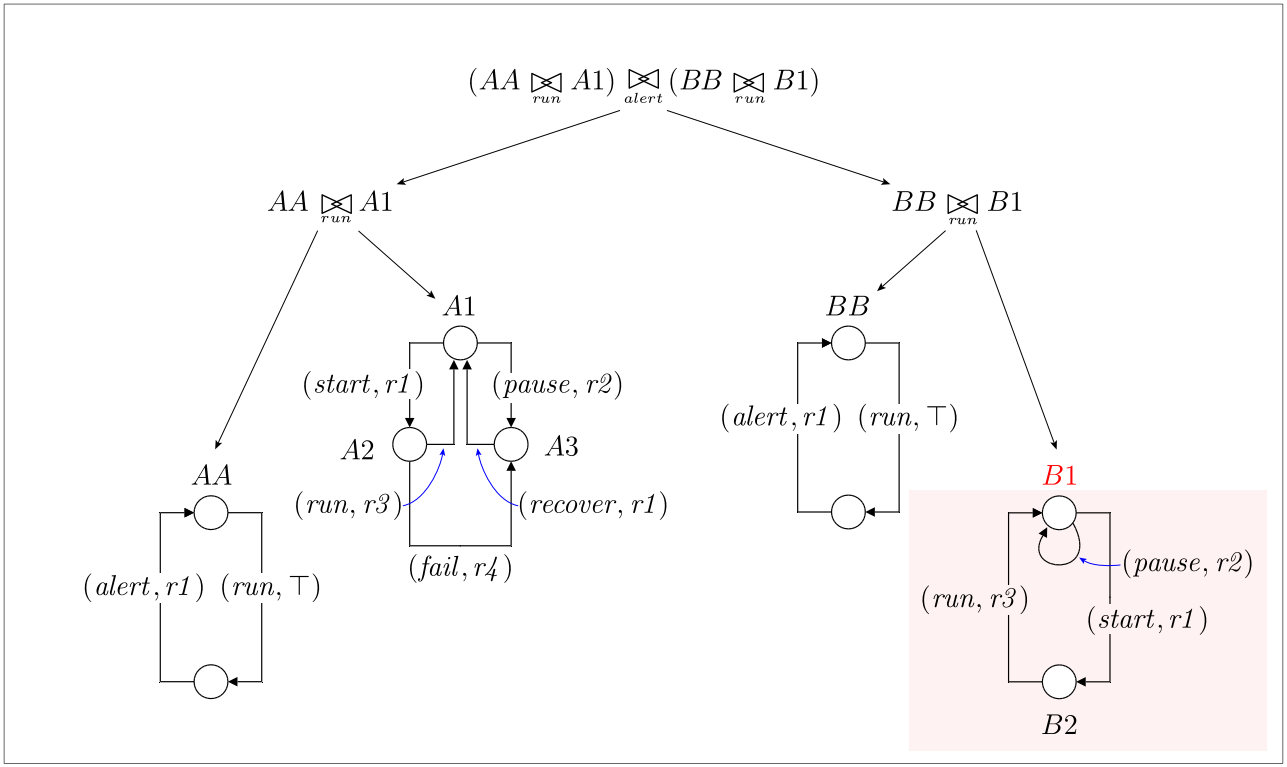
Example ipc Compilation



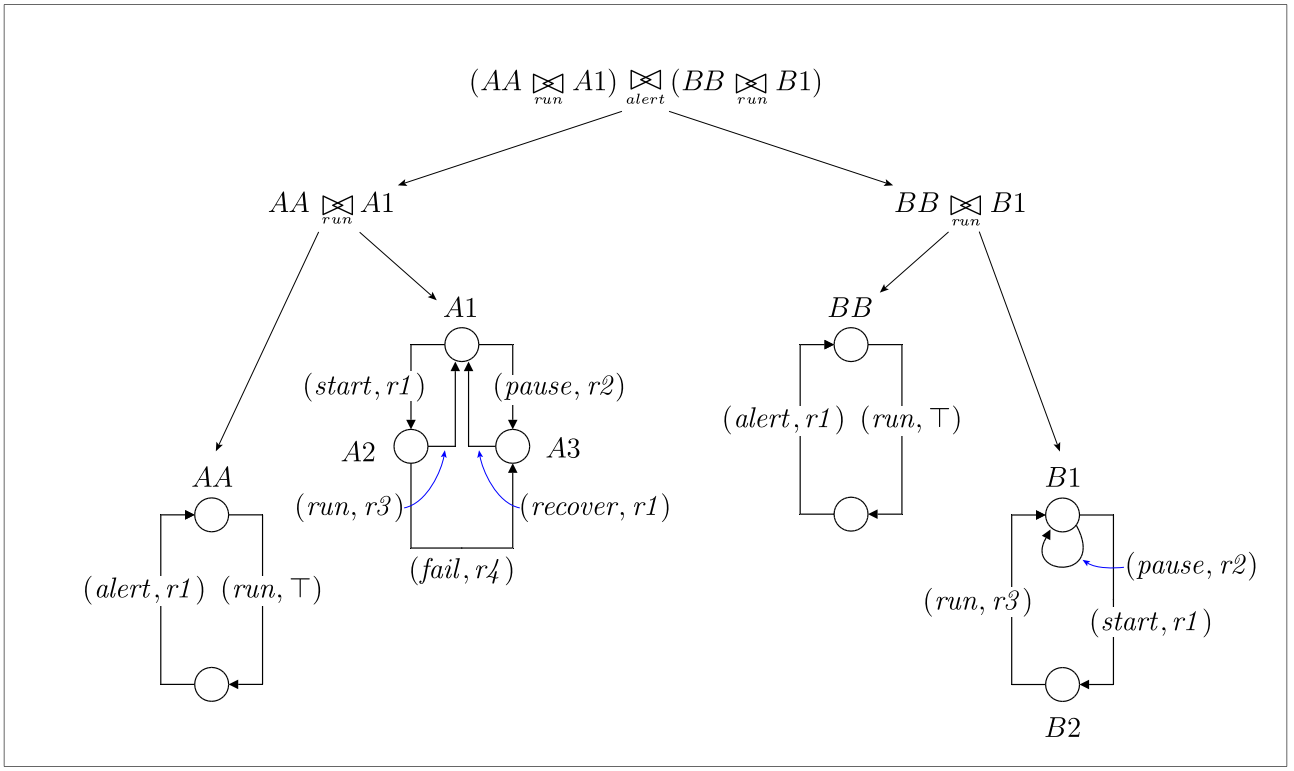
Example ipc Compilation



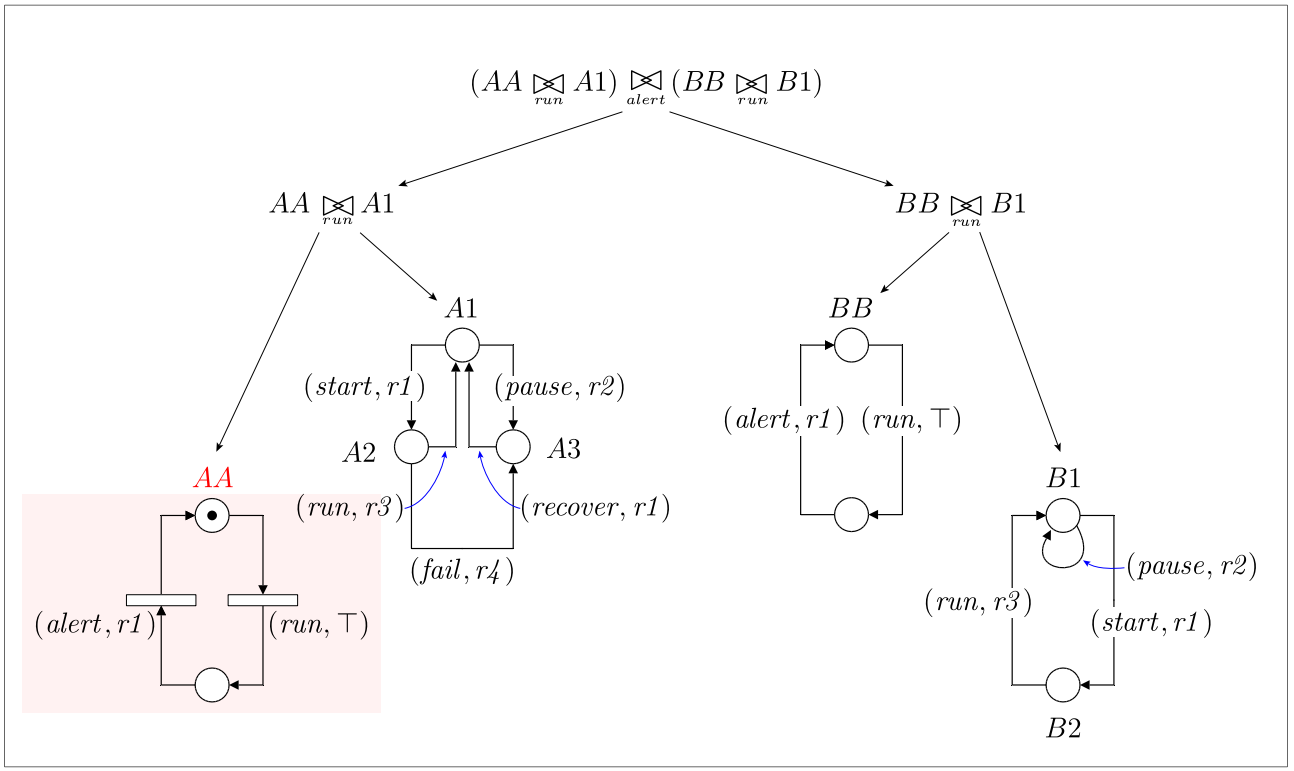
Example ipc Compilation



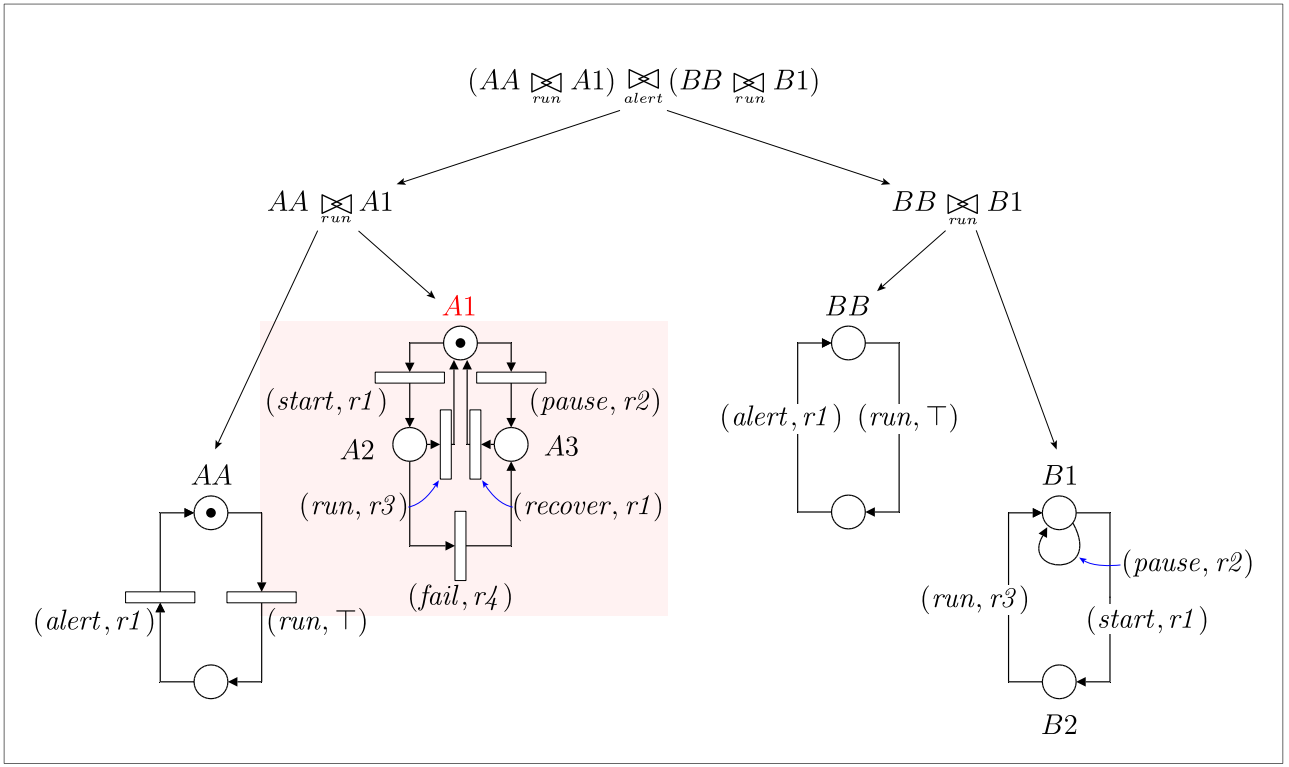
Example ipc Compilation



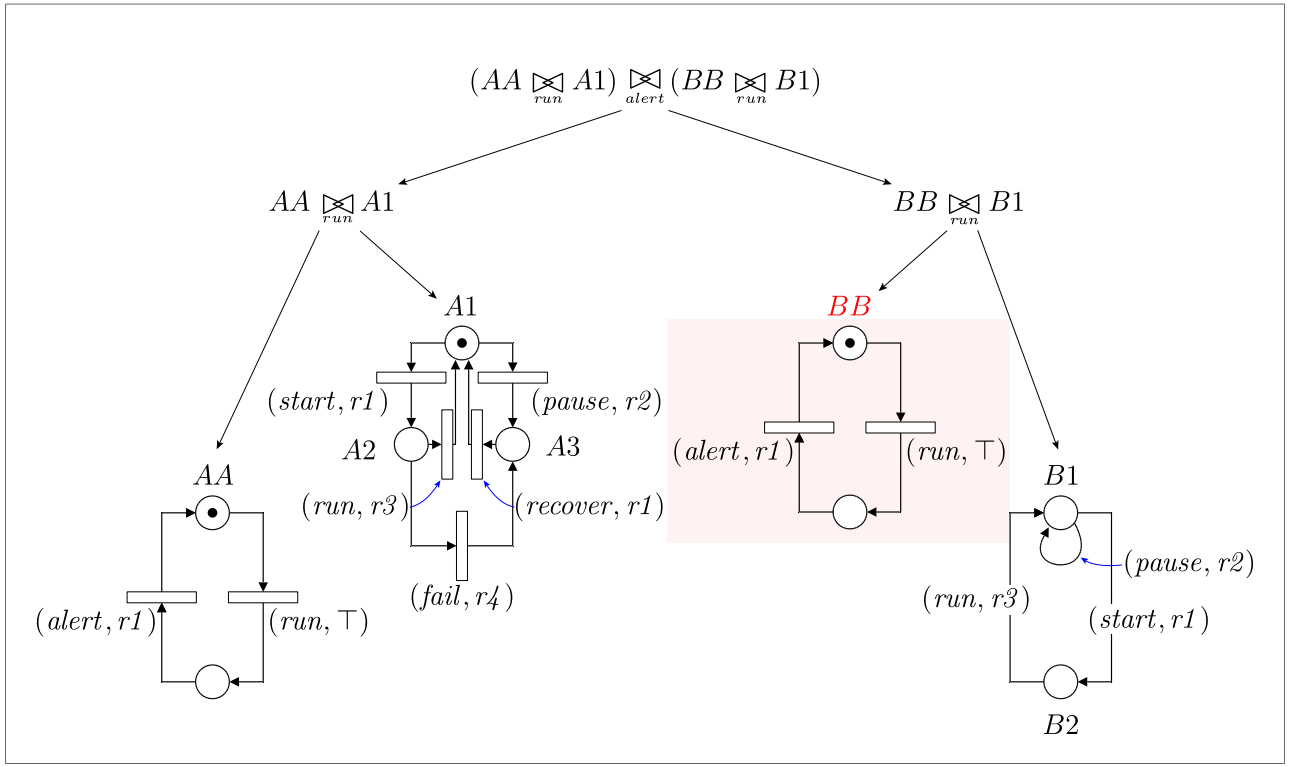
Example ipc Compilation



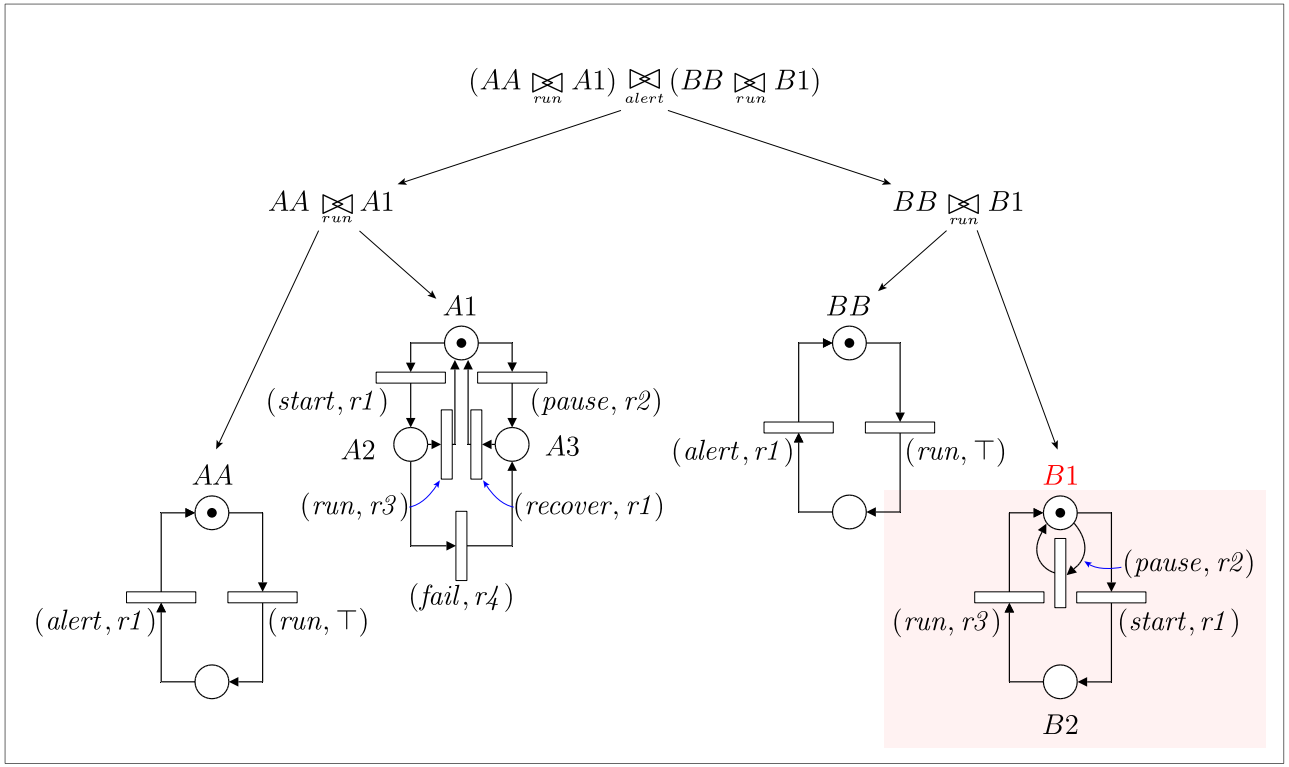
Example ipc Compilation



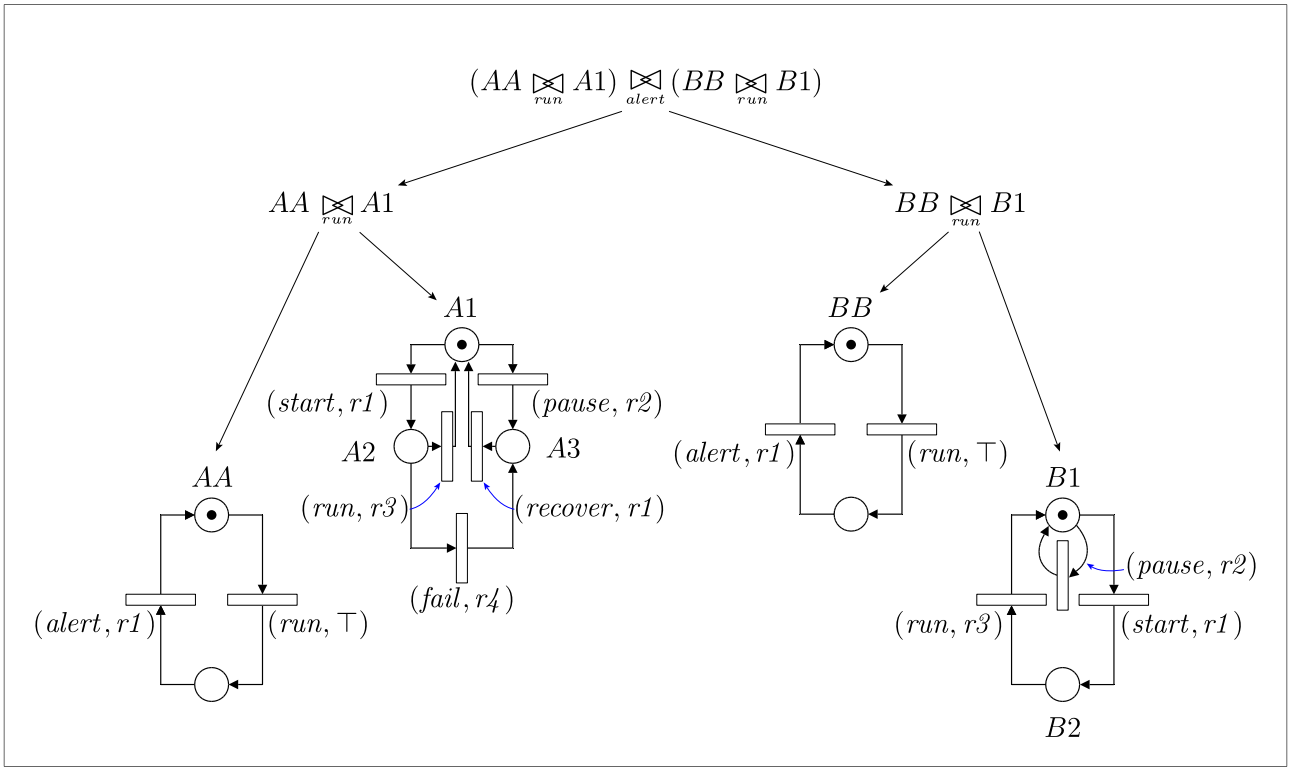
Example ipc Compilation



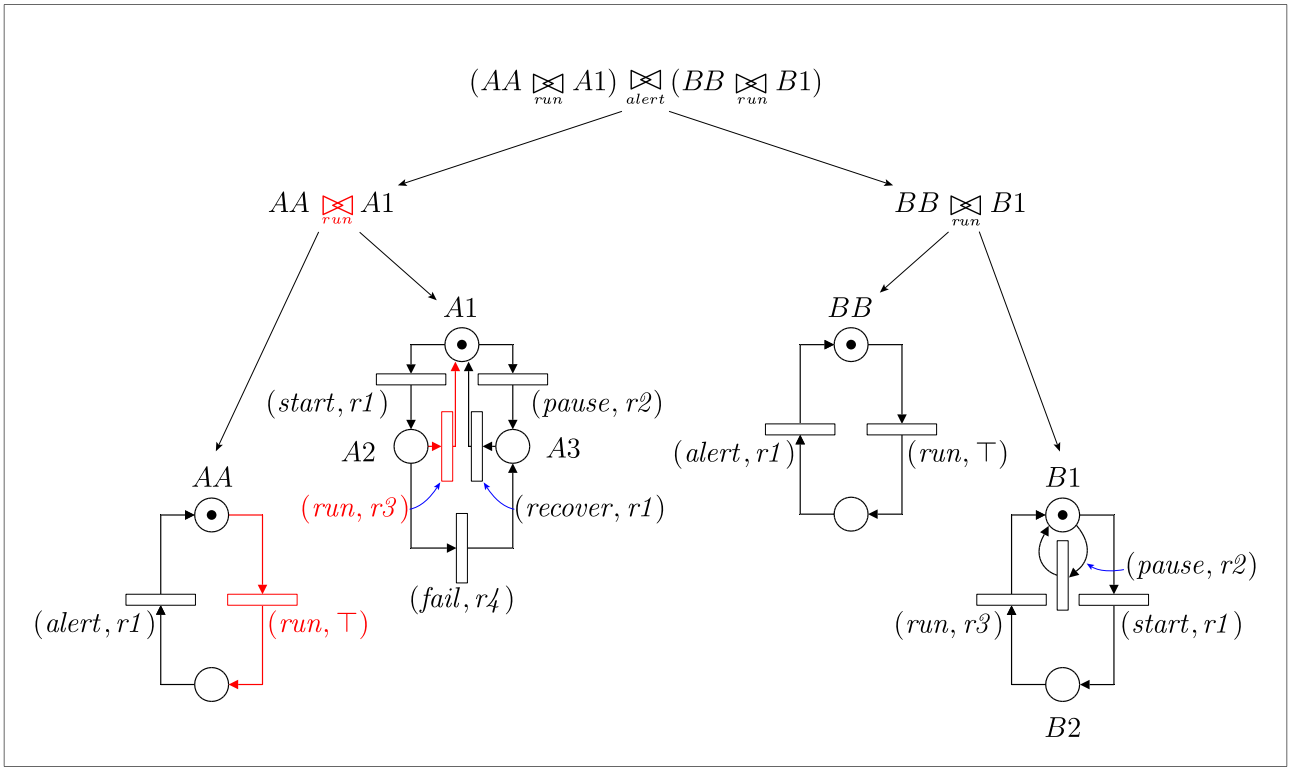
Example ipc Compilation



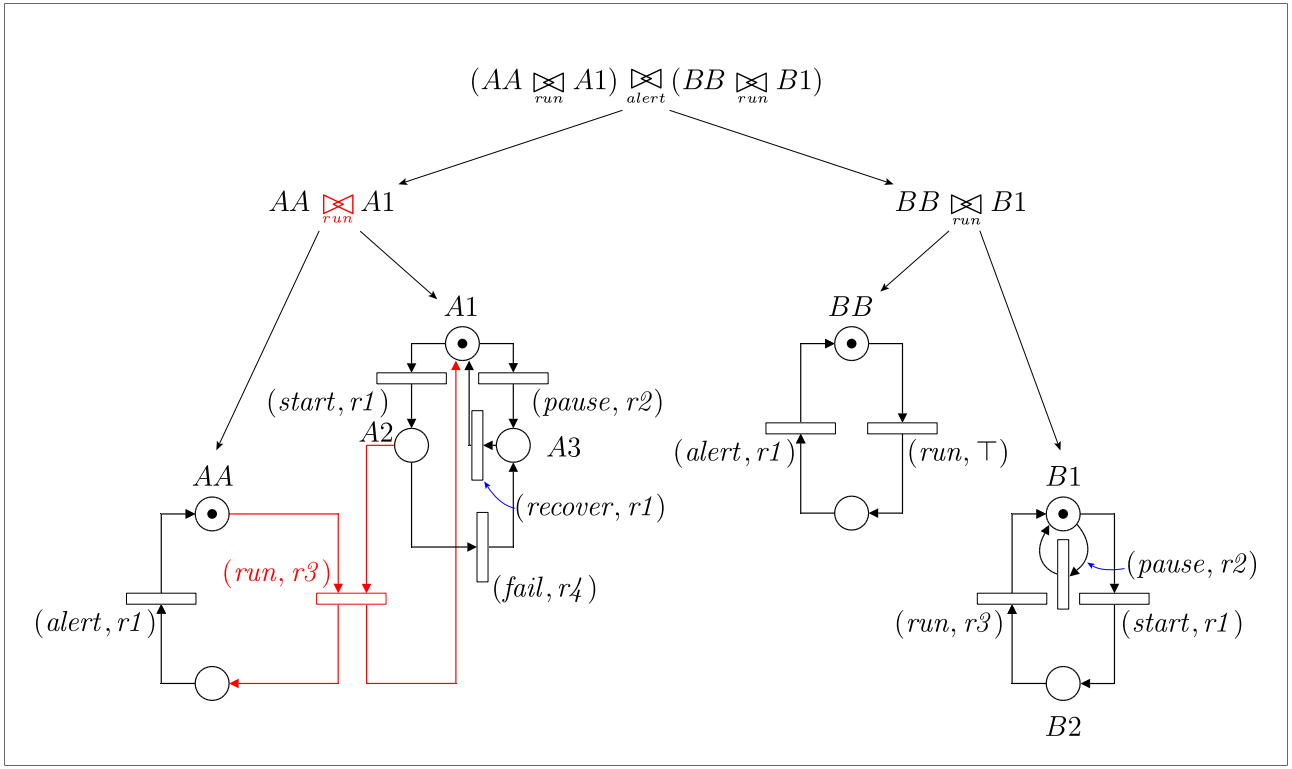
Example ipc Compilation



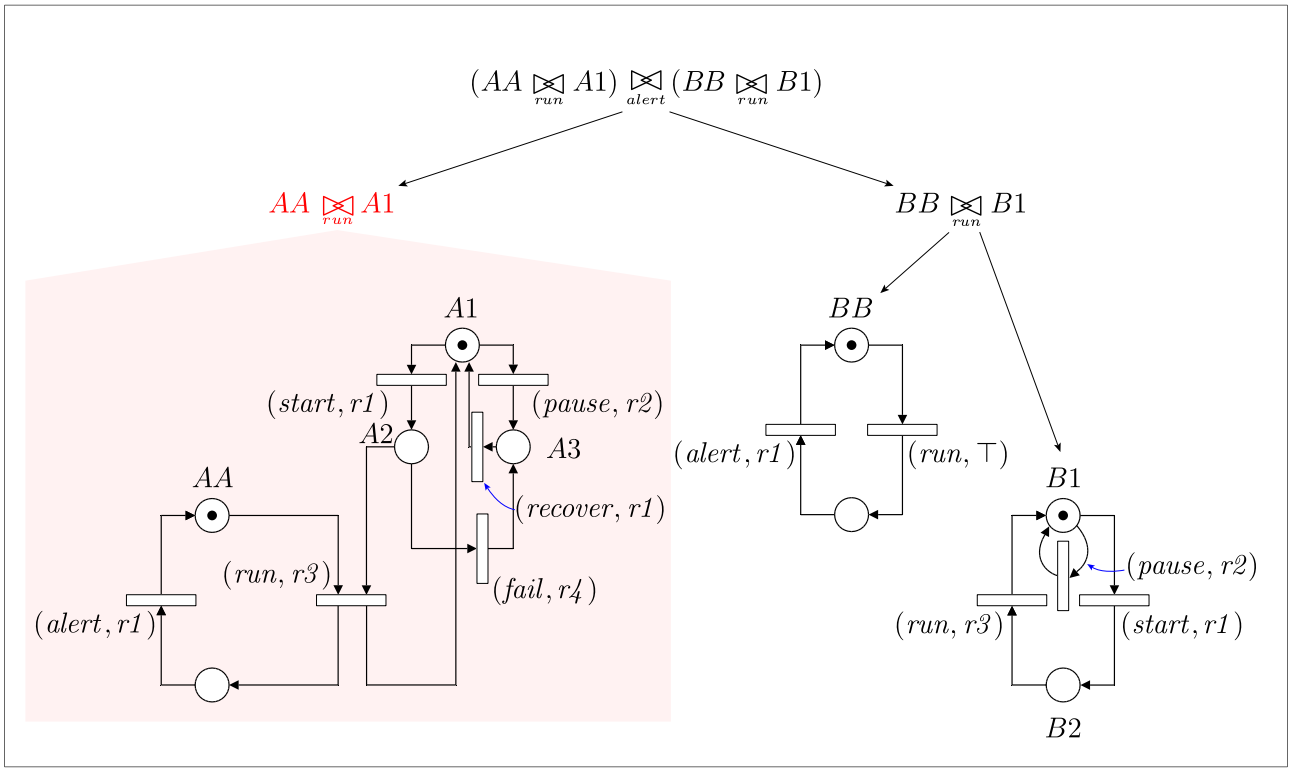
Example ipc Compilation



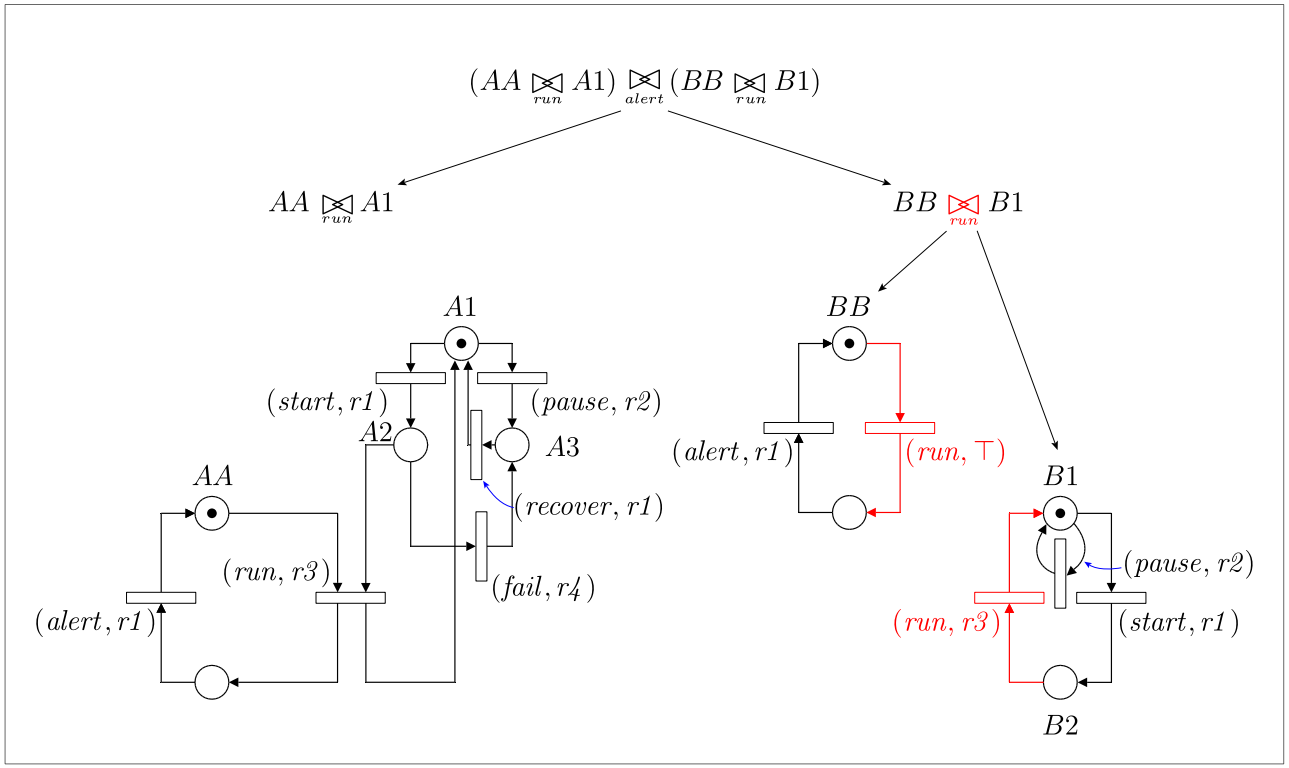
Example ipc Compilation



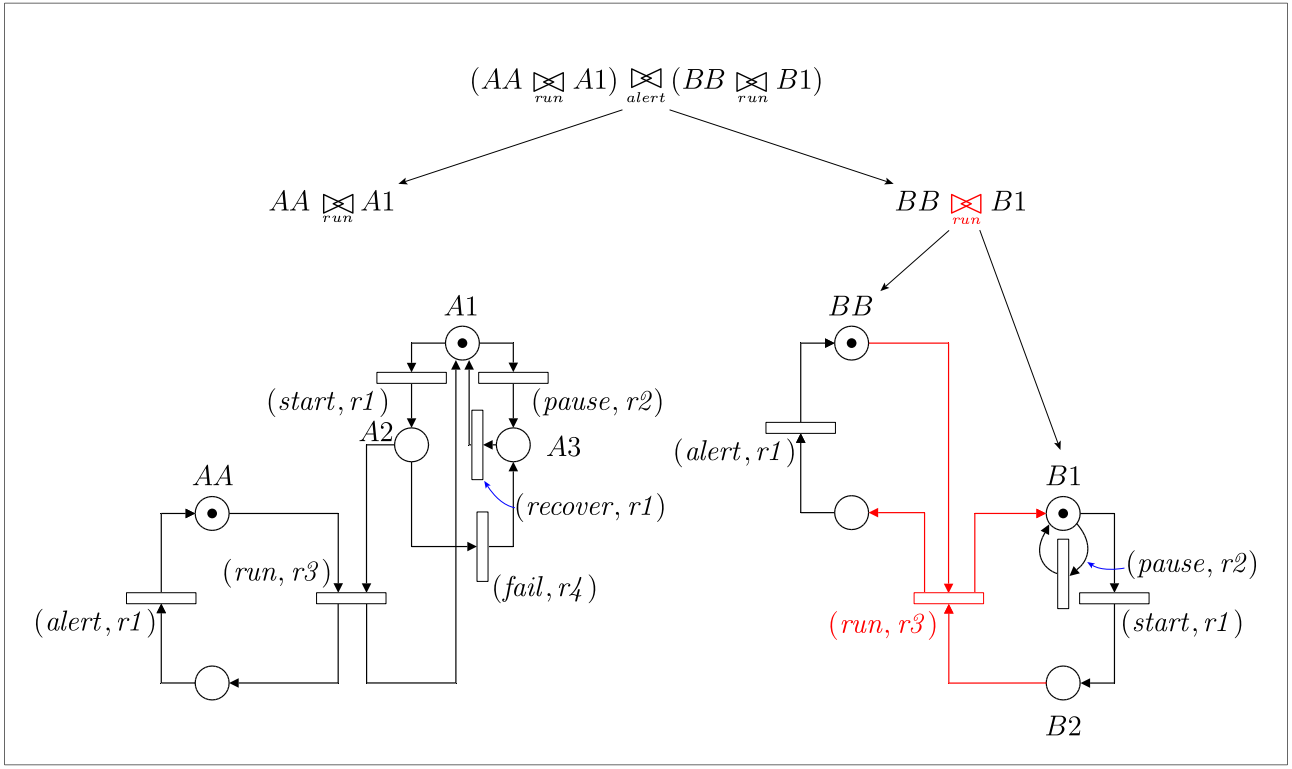
Example ipc Compilation



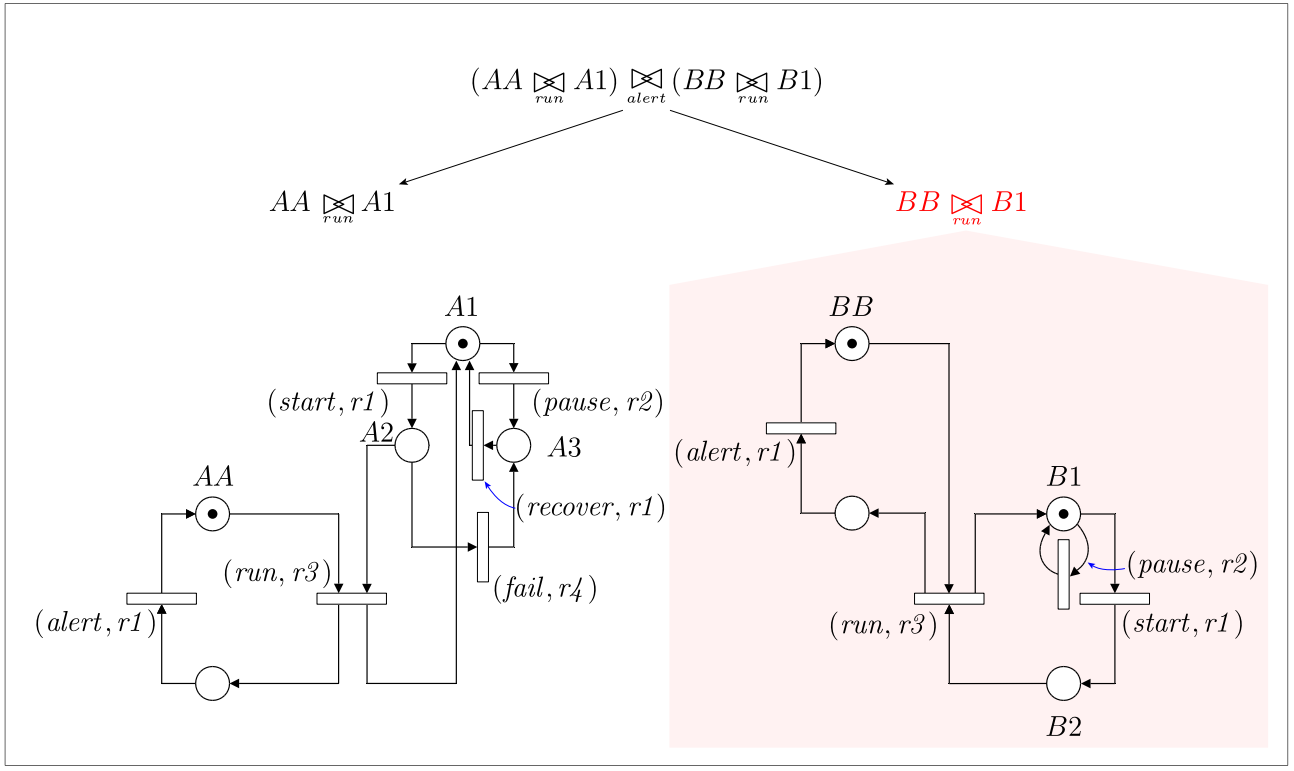
Example ipc Compilation



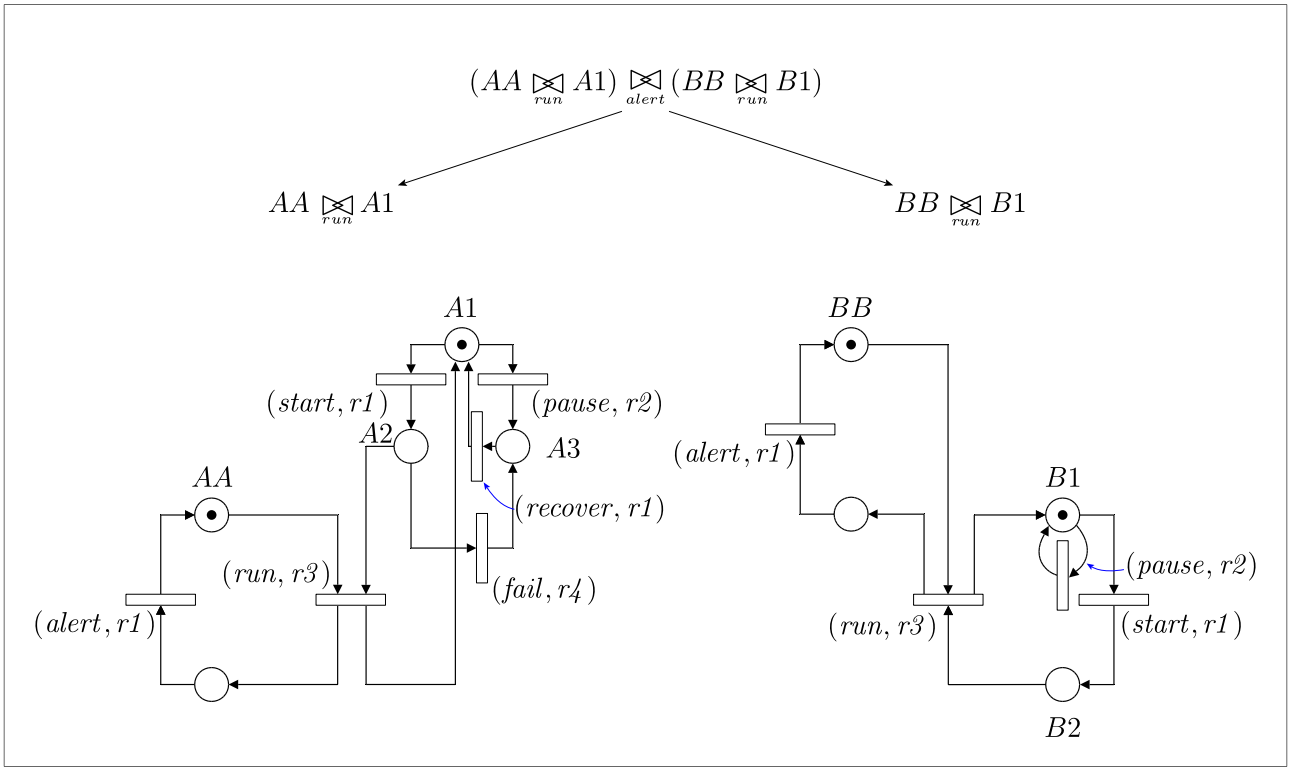
Example ipc Compilation



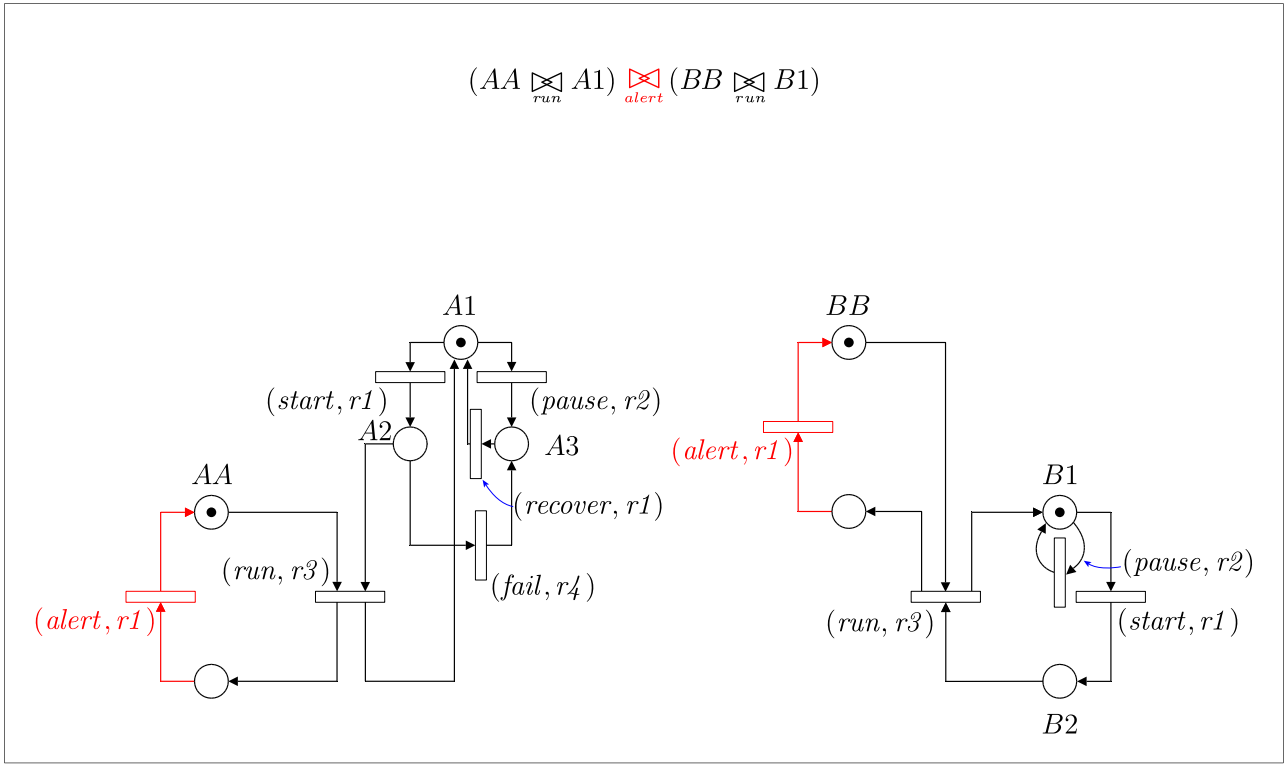
Example ipc Compilation



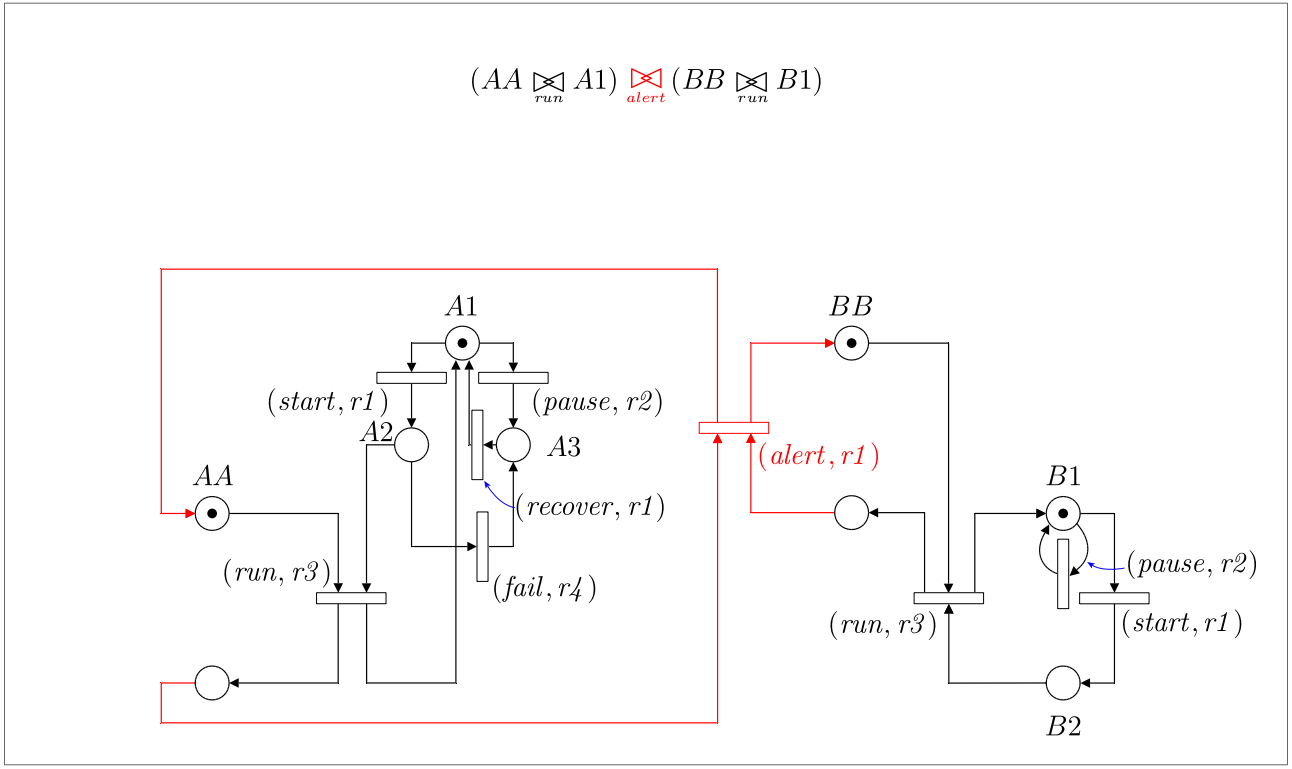
Example ipc Compilation



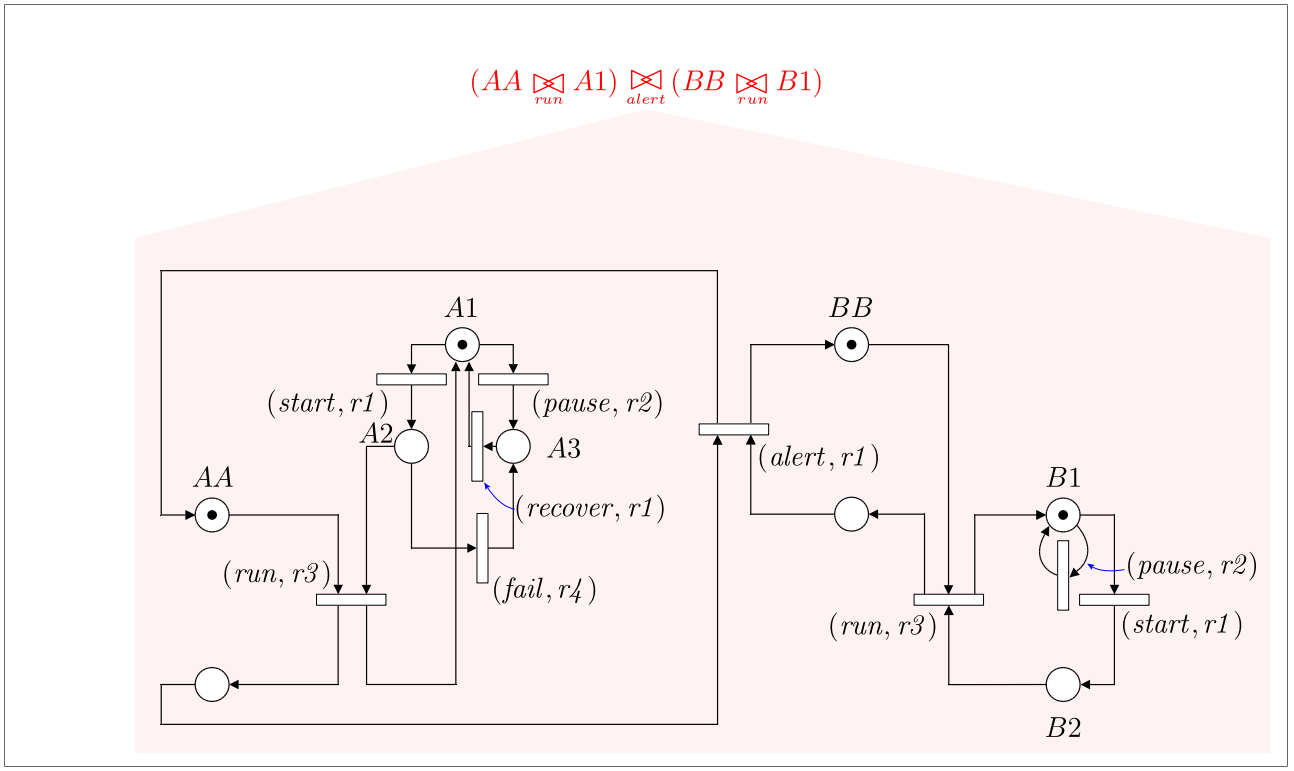
Example ipc Compilation



Example ipc Compilation

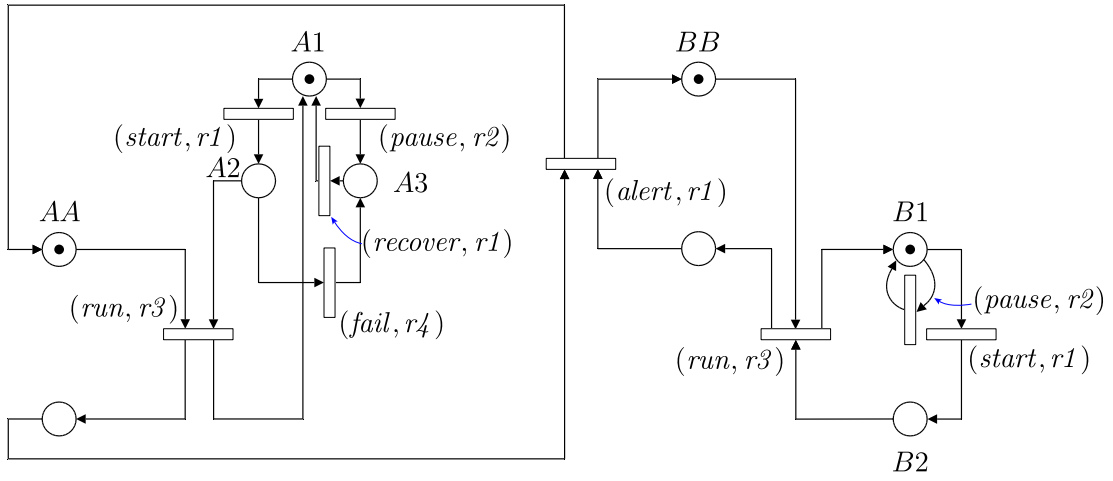


Example ipc Compilation



Example ipc Compilation

$$(AA \underset{run}{\bowtie} A1) \underset{alert}{\bowtie} (BB \underset{run}{\bowtie} B1)$$



DNAmaca: Example

A transition definition in DNAmaca:

```
\transition{A2_fail} {
        %% PEPAaction type { fail }
        \condition{ A2 > 0 }
        \action {
                next -> A2 = A2 - 1;
                next -> A3 = A3 + 1;
        }
        \priority{1}
        \rate{ PEPA_r4 }
}
```

Stochastic Probe Syntax

Probes consist of action regular expressions based on sequences, branching and ranges.

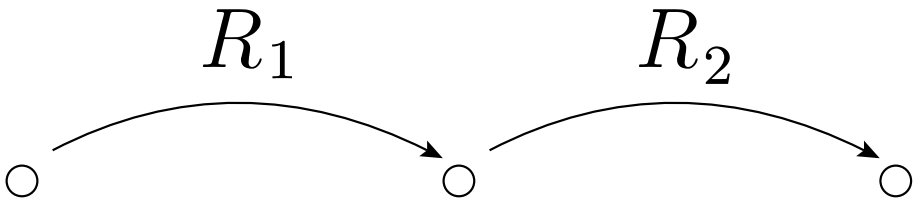
$$
\begin{array}{rcl}
R & ::= & a \mid T, T \mid (S) \\
S & ::= & T \mid T|S \\
T & ::= & R \mid R\{n\} \mid R\{m,n\} \mid R^{+} \mid R^{*} \mid R? \\
a & ::= & act \mid act : \mathsf{start} \mid act : \mathsf{stop}
\end{array}
$$

Require to specify start/stop actions to start and stop the measurement over the model

regexp Probes

Sequence operator:

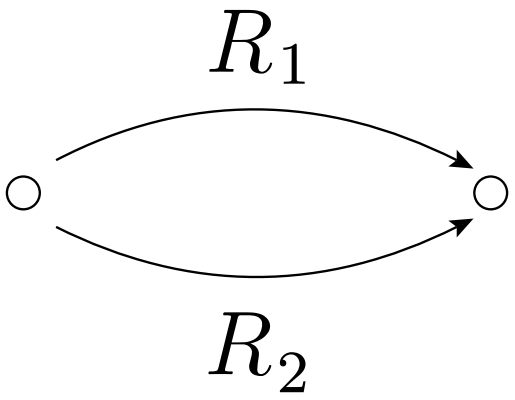
R_1, R_2



regexp Probes

Choice operator:

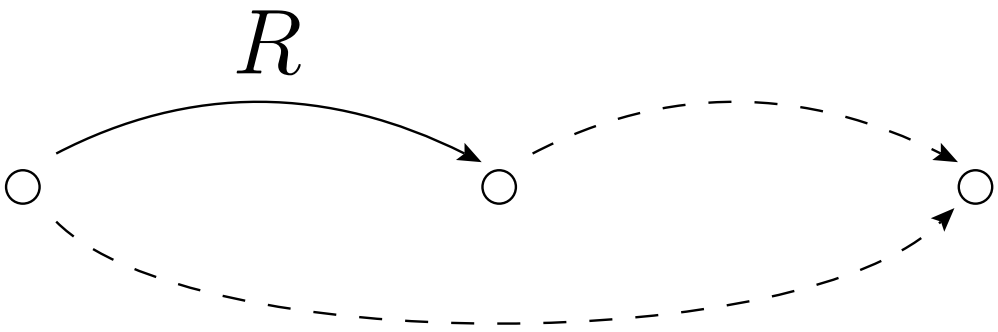
$$R_1 | R_2$$



regexp Probes

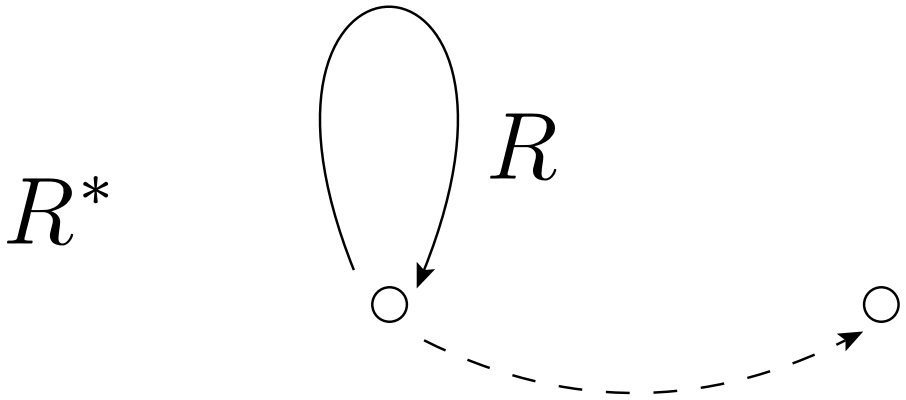
Zero-or-one operator:

$R?$



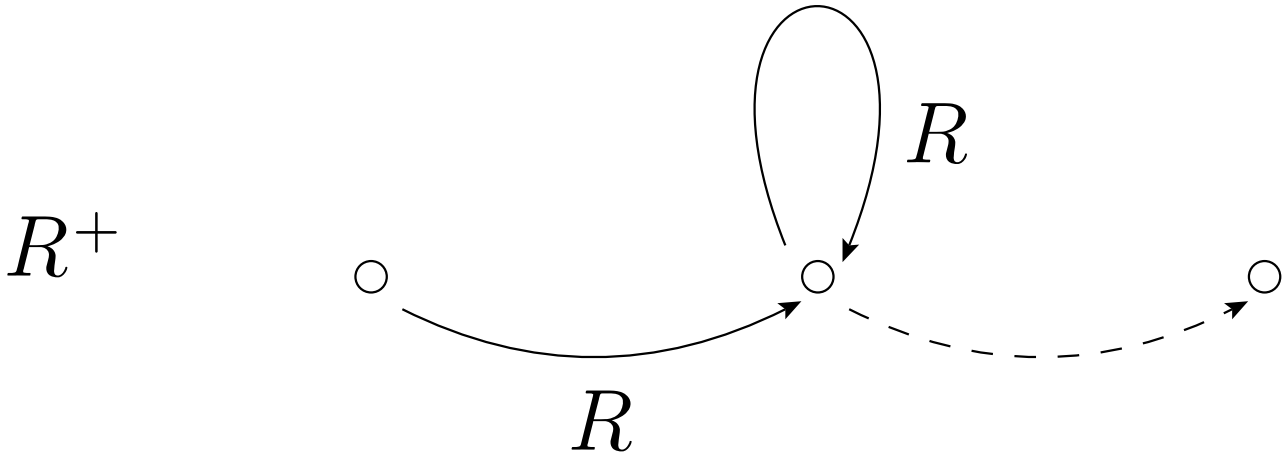
regexp Probes

Closure operator:



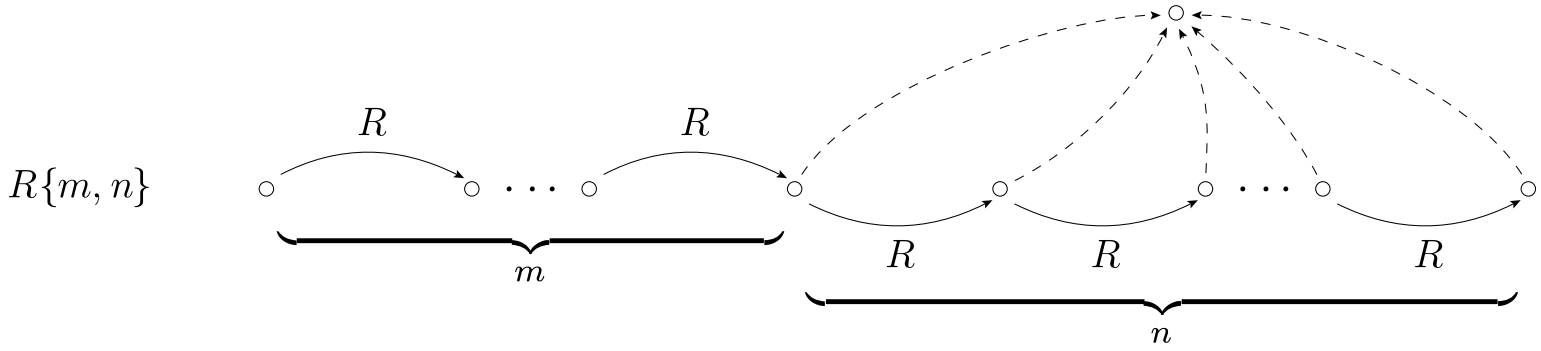
regexp Probes

Positive closure operator:



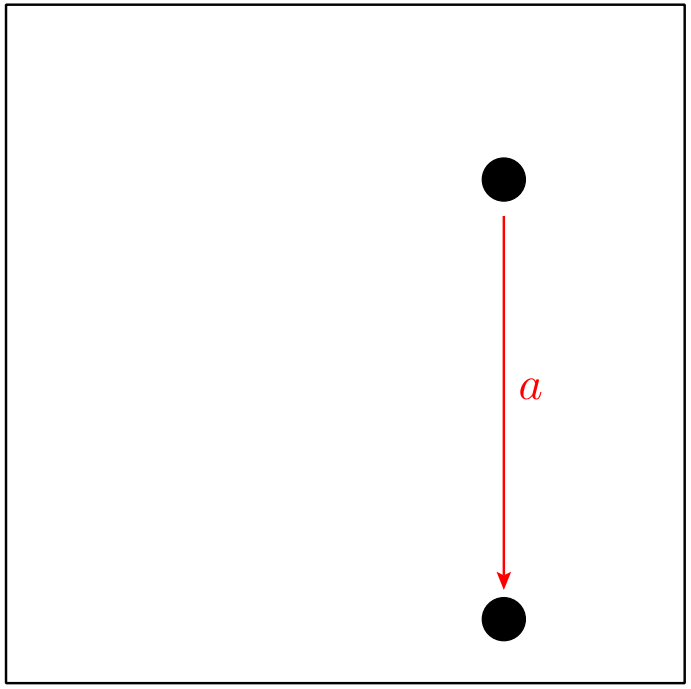
regexp Probes

Range operator:



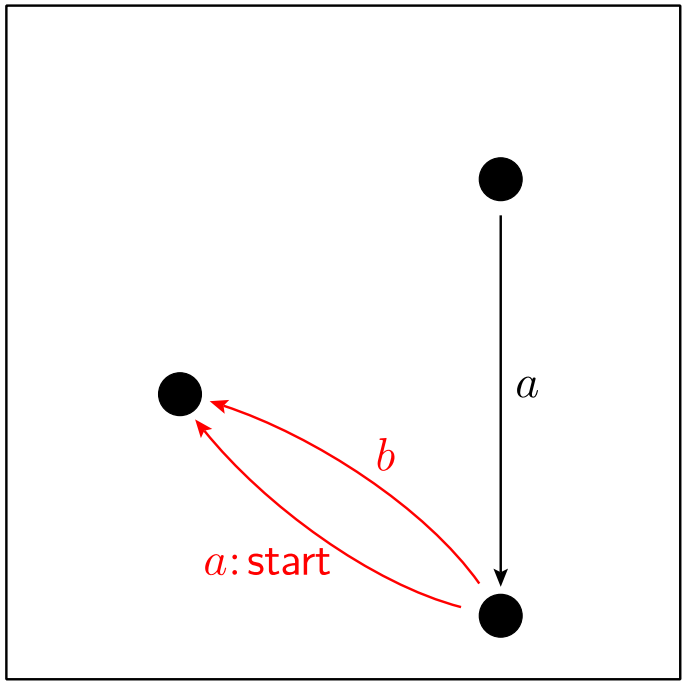
Probe Example

Probe definition: $(a, (a{:}\,\mathsf{start} \mid b), (a{:}\,\mathsf{stop} \mid b))$



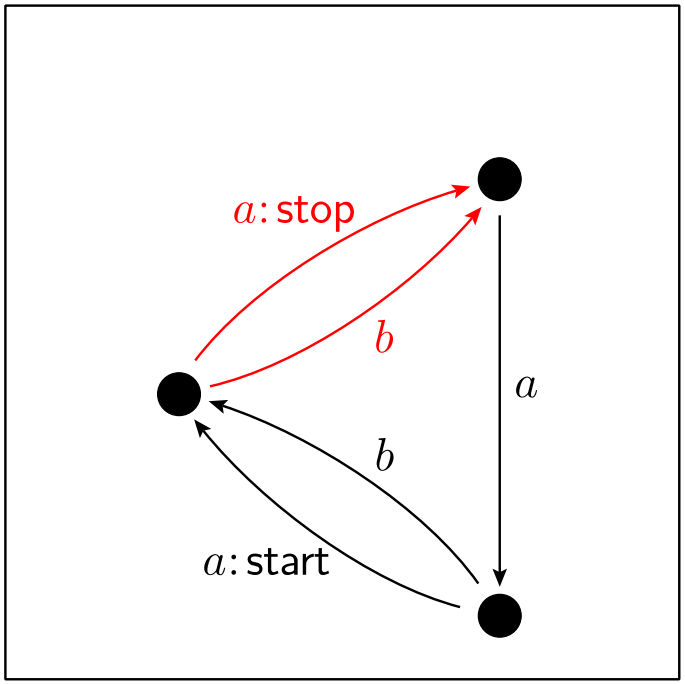
Probe Example

Probe definition: (a, (a: start | b), (a: stop | b))



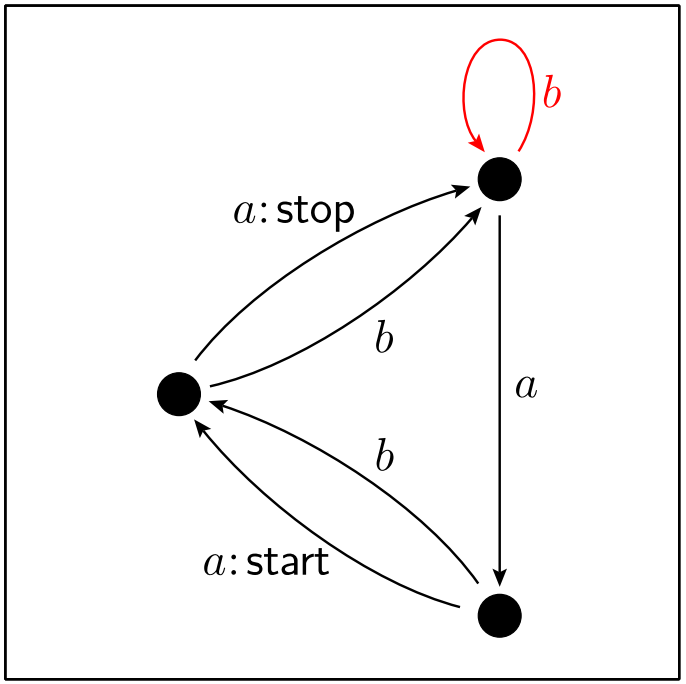
Probe Example

Probe definition: (a, (a: start | b), (a: stop | b))



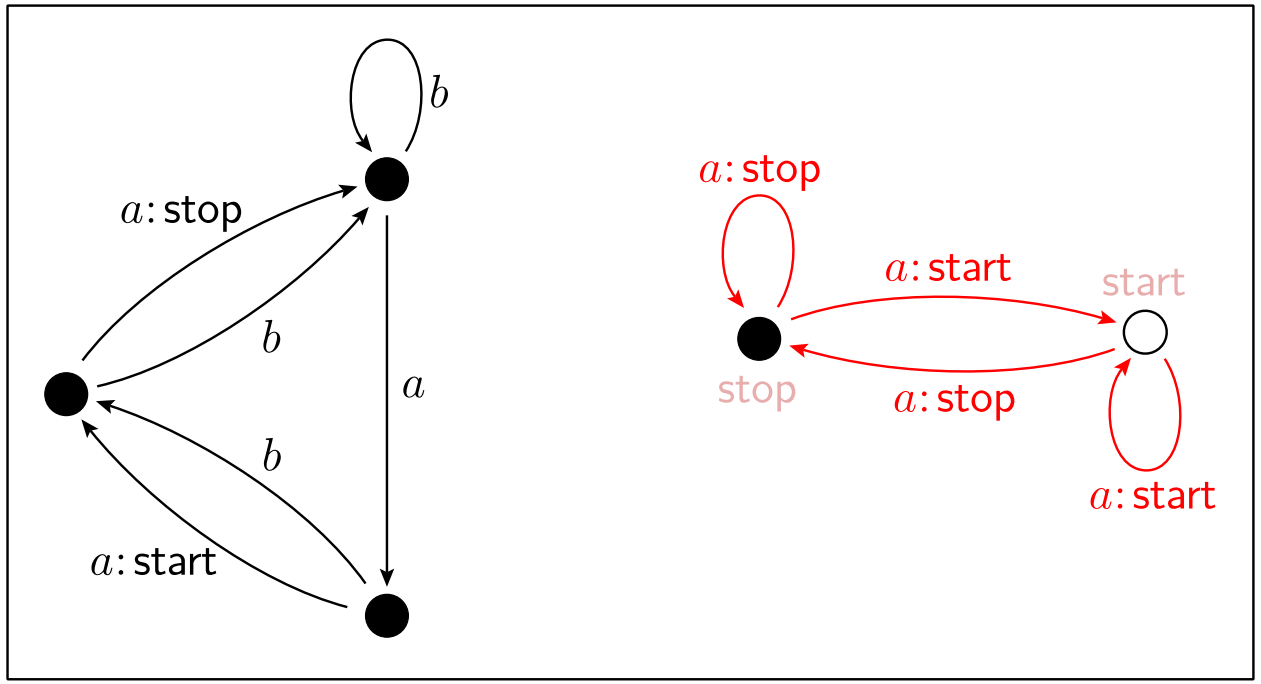
Probe Example

Probe definition: (a, (a:start | b), (a:stop | b))



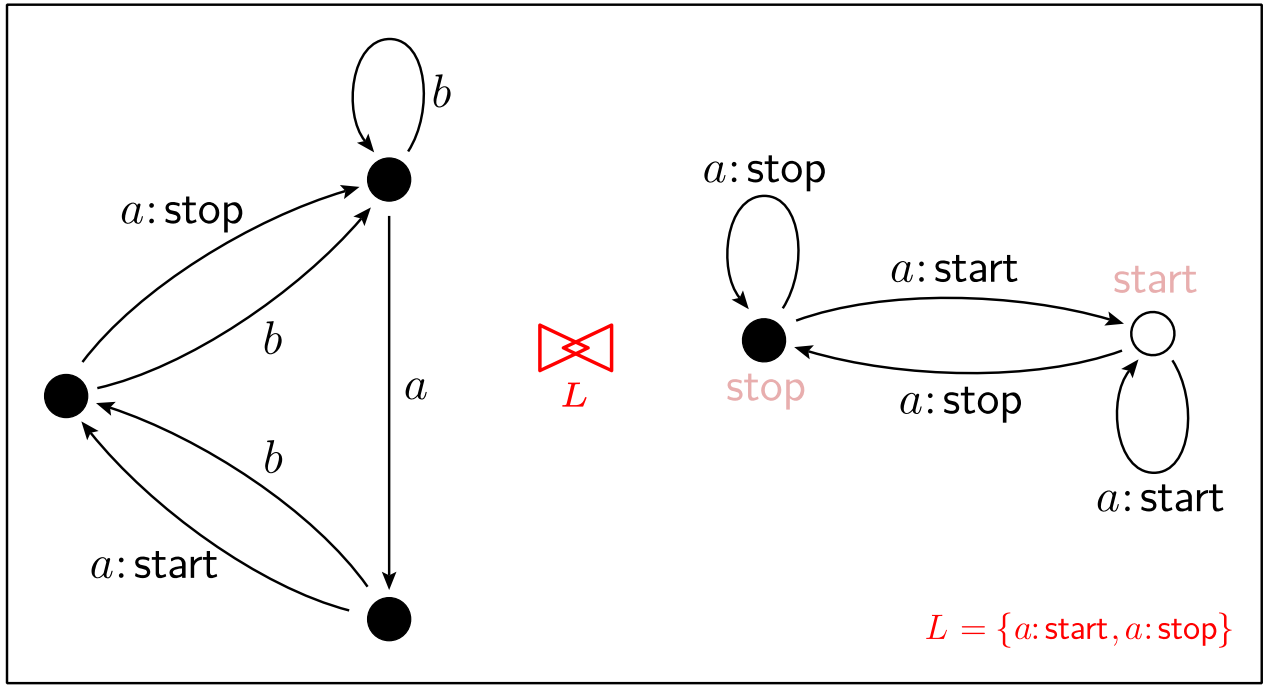
Probe Example

Probe definition: (a, (a: start | b), (a: stop | b))



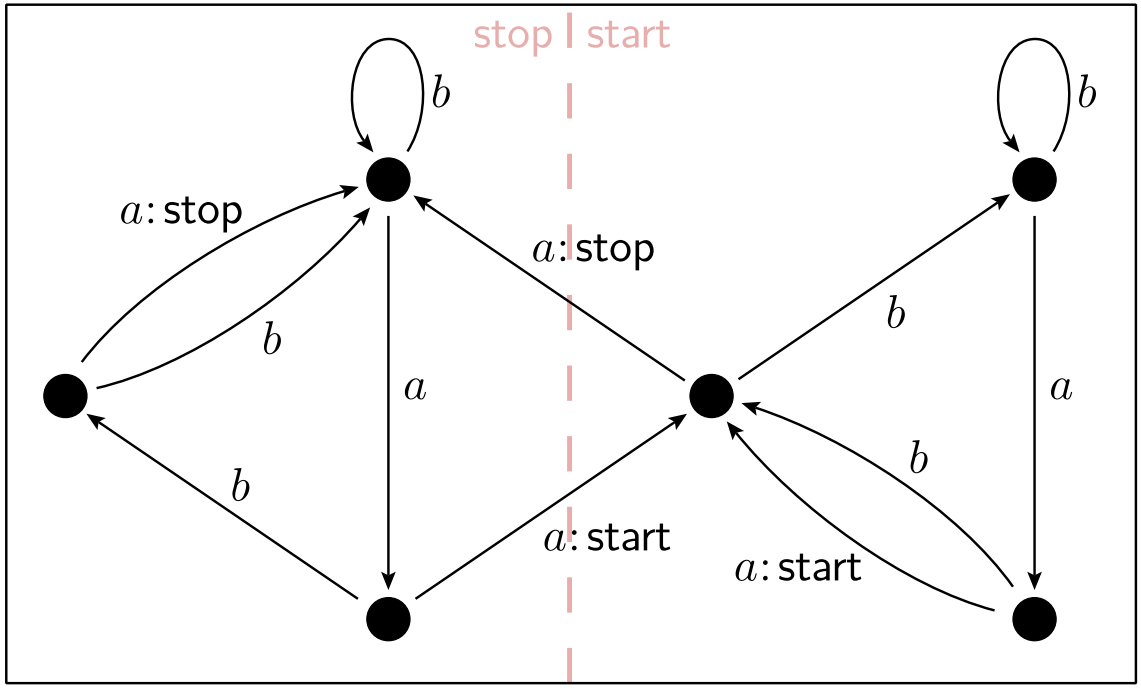
Probe Example

Probe definition: (a, (a:start | b), (a:stop | b))



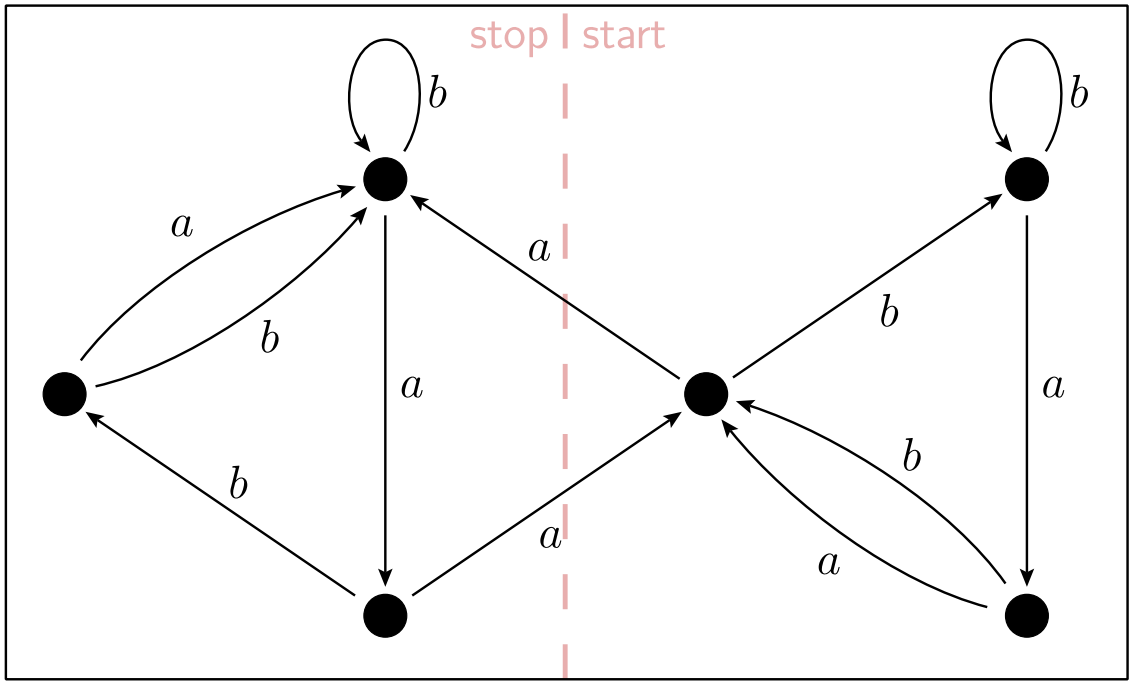
Probe Example

Probe definition: (a, (a: start | b), (a: stop | b))



Probe Example

Probe definition: (a, (a:start | b), (a:stop | b))



Example: Web server

- Multiple redundant web servers
- Servers subject to failure and synchronised recovery if necessary
- Buffered web page updates
- Priority given to read access

Server definition

$$
\begin{aligned}
\mathrm{Server} &\stackrel{def}{=} (\mathrm{s_read_request}, \top).\mathrm{Server_read} \\
&+ (\mathrm{s_fail}, r_{s_f}).\mathrm{Server_fail} \\
&+ (\mathrm{s_write}, r_{s_w}).\mathrm{Server} \\
\mathrm{Server_read} &\stackrel{def}{=} (\mathrm{s_read_lookup}, r_{s_{rl}}).\mathrm{Server} \\
\mathrm{Server_fail} &\stackrel{def}{=} (\mathrm{s_fail_recover}, r_{s_{fr}}).\mathrm{Server} \\
&+ (\mathrm{s_fail_recover_all}, \top).\mathrm{Server} \\
&+ (\mathrm{s_write}, r_{s_{fsw}}).\mathrm{Server_fail}
\end{aligned}
$$

Server group definition

$$
\begin{aligned}
\mathsf{Server_group}_0 &\stackrel{def}{=} (\mathsf{s_fail}, \top).\mathsf{Server_group}_1 \\
\mathsf{Server_group}_i &\stackrel{def}{=} (\mathsf{s_fail}, \top).\mathsf{Server_group}_{i+1} \\
&\quad + \quad (\mathsf{s_fail_recover}, \top) \\
&\quad\quad .\mathsf{Server_group}_{i-1} : 1 \leq i < S \\
\mathsf{Server_group}_S &\stackrel{def}{=} (\mathsf{s_fail_recover_all}, r_{s_{gsfra}}) \\
&\quad\quad .\mathsf{Server_group}_0
\end{aligned}
$$

Server cluster definition

$$\text{Servers} \stackrel{def}{=} (\text{Server} \underset{\mathcal{L}}{\bowtie} \text{Server} \underset{\mathcal{L}}{\bowtie} \text{Server} \underset{\mathcal{L}}{\bowtie} \text{Server}) \underset{\mathcal{L}'}{\bowtie} \text{Server_group}_0$$

where:

- $\mathcal{L} = \{\text{s_write}, \text{s_fail_recover_all}\}$
- $\mathcal{L}' = \mathcal{L} \cup \{\text{s_fail_recover}\}$

Write buffer definition

$$
\begin{aligned}
\mathrm{Write_buffer}_0 &\stackrel{def}{=} (\mathrm{b_write}, \top).\mathrm{Write_buffer}_1 \\
\mathrm{Write_buffer}_i &\stackrel{def}{=} (\mathrm{b_write}, \top).\mathrm{Write_buffer}_{i+1} \\
&\quad + \quad (\mathrm{s_write}, \top).\mathrm{Write_buffer}_0 \\
&\qquad\qquad : 1 \leq i < B \\
\mathrm{Write_buffer}_B &\stackrel{def}{=} (\mathrm{s_write}, \top).\mathrm{Write_buffer}_0
\end{aligned}
$$

Writer definition

$$\text{Writer} \stackrel{def}{=} (\text{b_write}, r_{w_{bw}}).\text{Writer_writ}$$
$$\text{Writer_writ} \stackrel{def}{=} (\text{w_reset}, r_{w_r}).\text{Writer}$$

and an arbitrary group writers:

$$\text{Writers} \stackrel{def}{=} \text{Writer} \underset{\emptyset}{\bowtie} \text{Writer} \underset{\emptyset}{\bowtie} \text{Writer}$$

Reader defintion

$$\begin{aligned} \mathsf{Reader} &\stackrel{def}{=} (\mathsf{s_read_request}, r_{r_{srr}}) \\ &\quad .(\mathsf{s_read_lookup}, \top).\mathsf{Reader_read} \\ \mathsf{Reader_read} &\stackrel{def}{=} (\mathsf{r_reset}, r_{r_r}).\mathsf{Reader} \end{aligned}$$

and an arbitrary number or readers:

$$\mathsf{Readers} \stackrel{def}{=} \mathsf{Reader} \underset{\emptyset}{\bowtie} \mathsf{Reader} \underset{\emptyset}{\bowtie} \mathsf{Reader}$$

Overall System

$$\text{Environment} \stackrel{def}{=} \text{Writers} \underset{\emptyset}{\bowtie} \text{Readers}$$

$$\text{Web_cluster} \stackrel{def}{=} \text{Servers} \underset{\{\text{s_write}\}}{\bowtie} \text{Write_buffer}_0$$

$$\text{System} \stackrel{def}{=} \text{Environment} \underset{\mathcal{N}}{\bowtie} \text{Web_cluster}$$

where $\mathcal{N} = \{\text{b_write}, \text{s_read_request}, \text{s_read_lookup}\}$.

Probe Query A

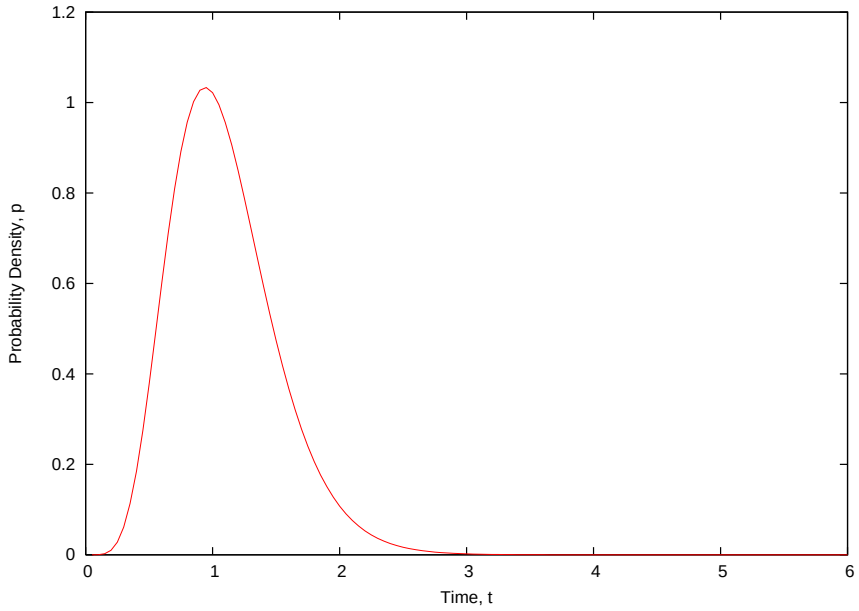
Probe A: measures the length of time to process at least 4 read events:.

$$\Pi_A = \text{s_read_request}: \text{start}, (\text{s_read_lookup}\{4\}), \text{r_reset}: \text{stop}$$

Model + probe stats: 362,240 states; 4,061,440 transitions

Probe Query A

Probability density function for probe A:



Probe Query B

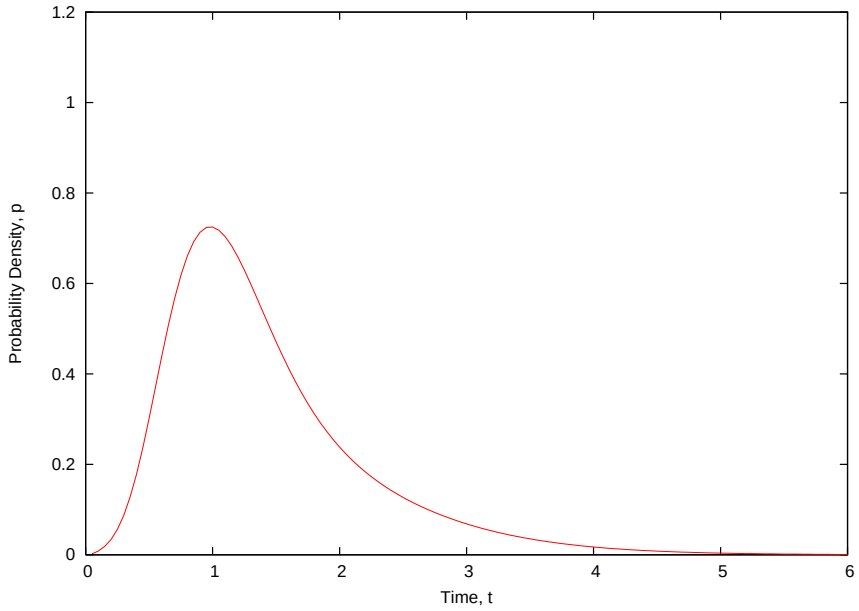
Probe B: observes the length of time it takes either to observe 4 completed reads or 2 completed writes:

$$\Pi_B = (\text{s_read_request}: \text{start}, (\text{s_read_lookup}\{4\}), \text{r_reset}: \text{stop}) \mid (\text{b_write}: \text{start}, \text{b_write}\{2\}, \text{w_reset}: \text{stop})$$

Model + probe stats: 581,200 states; 6,516,356 transitions

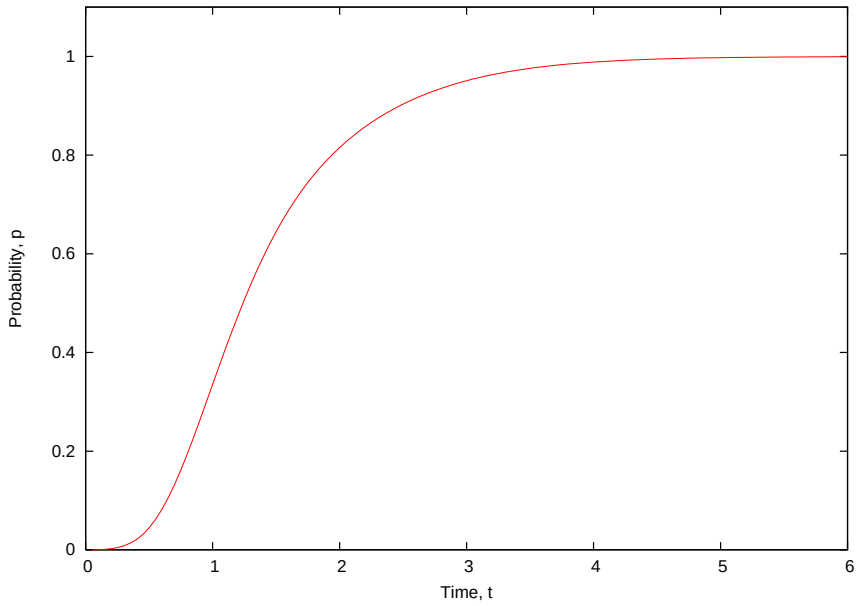
Probe Query B

Probability density function for probe B:



Probe Query B

Cumulative distribution function for probe B:



Probe Queries

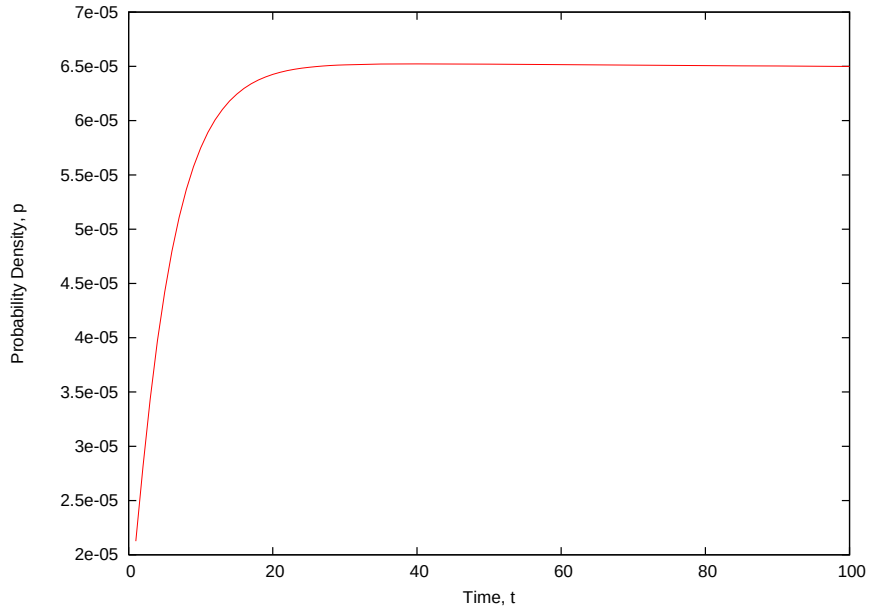
Probe C: extracts the passage-time to first full system failure:

$$\Pi_C = (\text{s_read_request}:\text{start} \mid \text{b_write}:\text{start}), \text{s_fail_recover}:\text{stop}$$

Model + probe stats: 661,960 states; 9,440,988 transitions

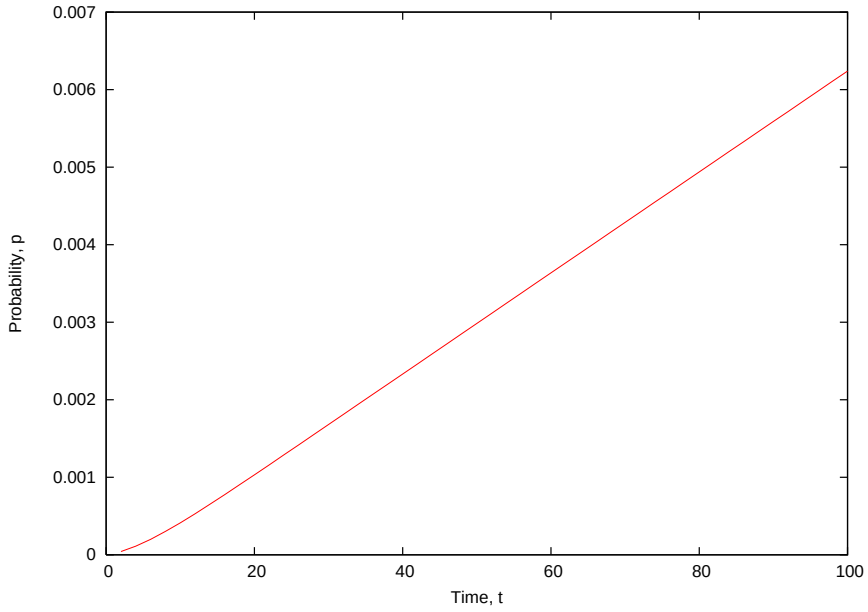
Probe Query C

Probability density function for probe C:



Probe Query C

Cumulative distribution function for probe C:



Conclusion

Stochastic probes:

- use additional SPA components to define performance measures
- unify three forms of performance analysis with a single specification mechanism
- use action regular expressions to specify probe behaviour
- cannot enforce behaviour – the next step?