# Derivation of Passage-time Densities in PEPA Models Using ipc: the Imperial PEPA Compiler

**Jeremy Bradley, Nick Dingle, Stephen Gilmore and William Knottenbelt**

{jb,njd200,wjk}@doc.ic.ac.uk

Imperial College London

stg@dcs.ed.ac.uk

University of Edinburgh

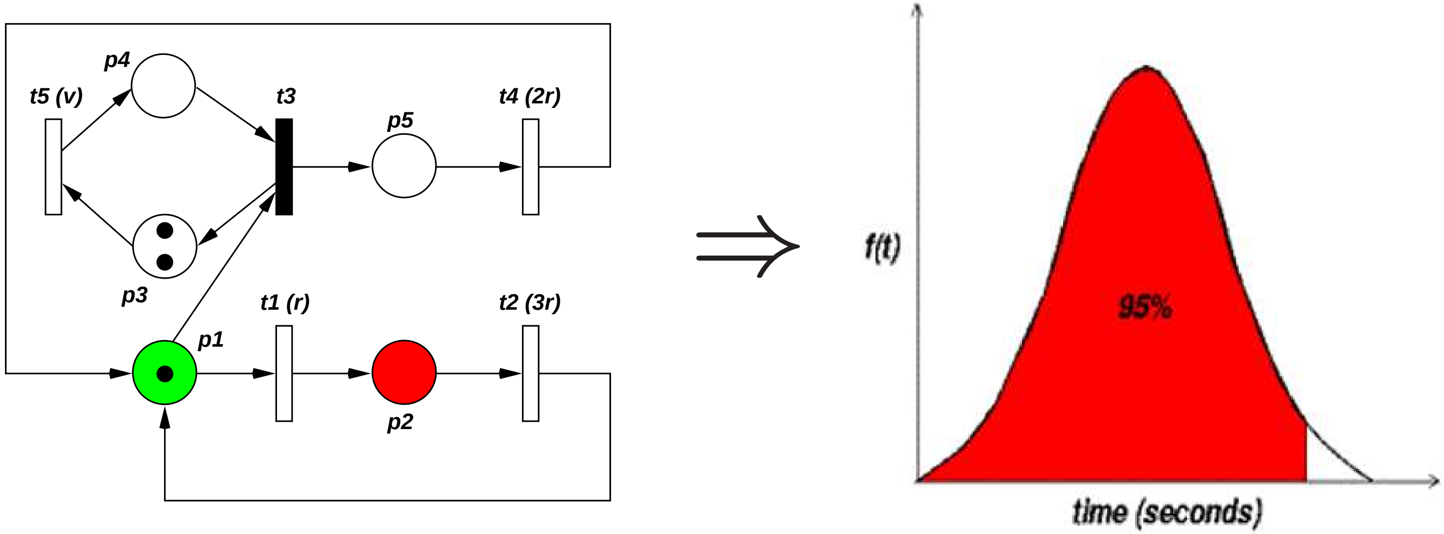
MASCOTS 2003 – 15 October 2003

# What is PEPA?

- A stochastic process algebra
- Markovian or exponential distributions
- Traditionally analysed with steady-state measures

# What is DNAmaca/HYDRA?

Extracting a passage-time density from a GSPN:

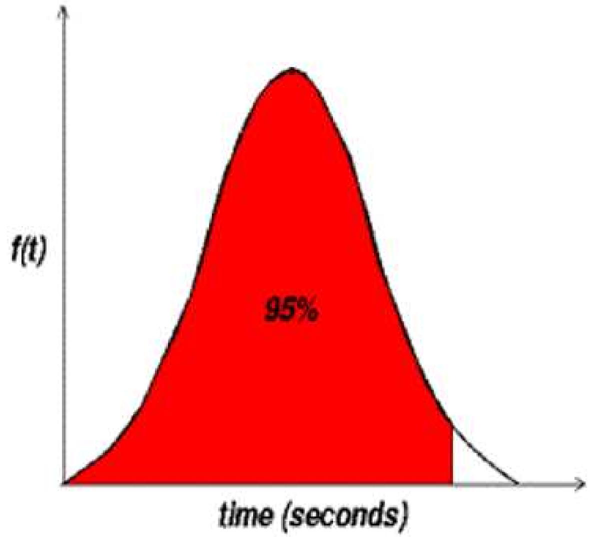


# Motivation

Extract a passage-time density from a PEPA model:

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

$\Longrightarrow$



# 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 $\quad$ $P+P$: competitive choice

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

# 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

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

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

# PEPA: Example

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

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

$$
\mathsf{Sys} \stackrel{\text{def}}{=} (\mathsf{AA} \underset{\mathsf{run}}{\bowtie} \mathsf{A1}) \underset{\mathsf{alert}}{\bowtie} (\mathsf{BB} \underset{\mathsf{run}}{\bowtie} \mathsf{B1})
$$

# Example ipc Compilation

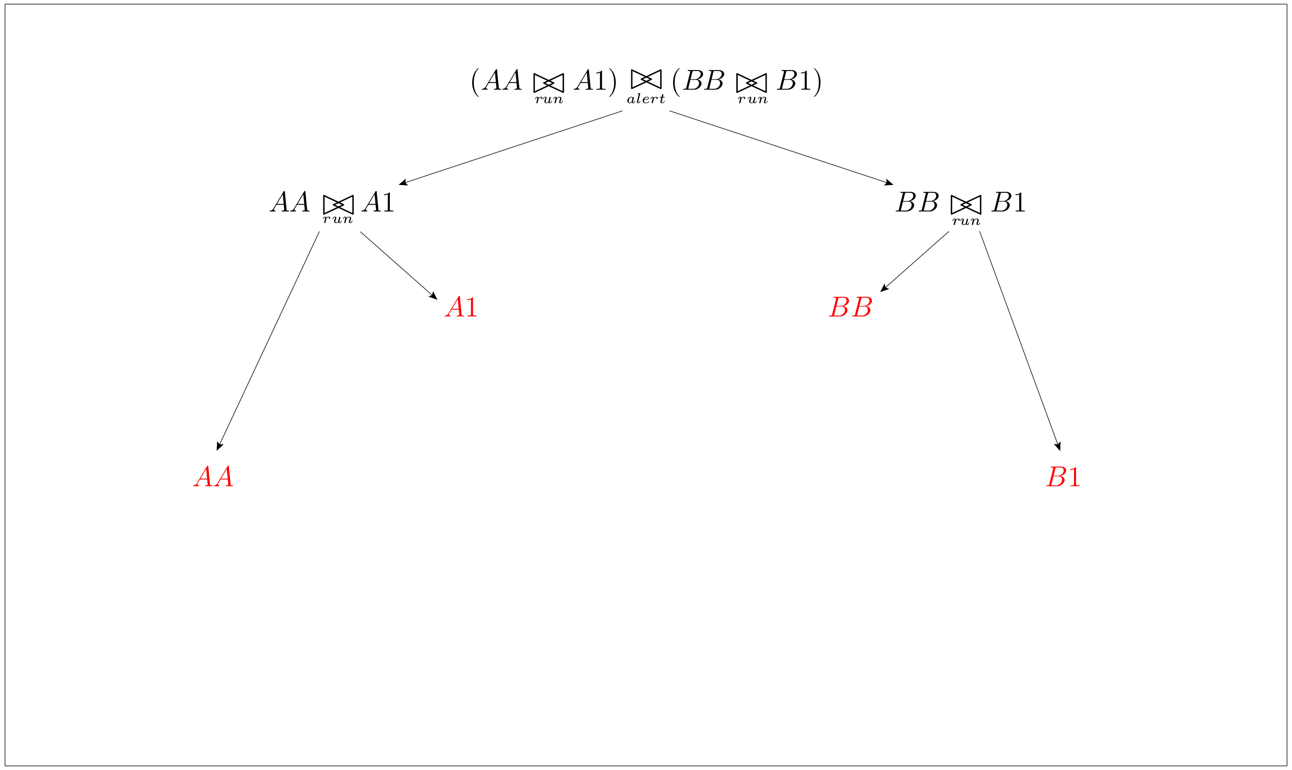
$$(AA \underset{run}{\bowtie} A1) \underset{alert}{\bowtie} (BB \underset{run}{\bowtie} B1)$$

# Example ipc Compilation

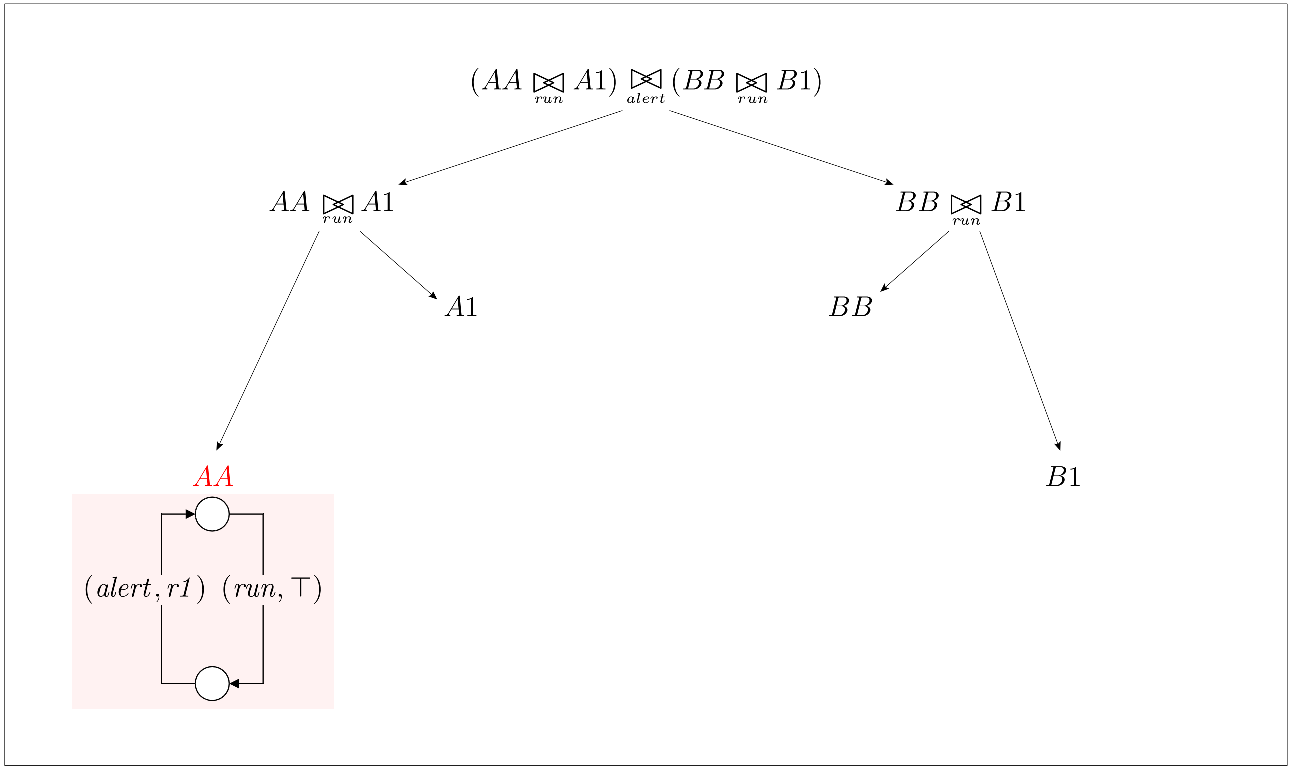
$$(AA \underset{run}{\bowtie} A1) \underset{alert}{\bowtie} (BB \underset{run}{\bowtie} B1)$$

$$\begin{array}{ccc} & (AA \underset{run}{\bowtie} A1) \underset{alert}{\bowtie} (BB \underset{run}{\bowtie} B1) & \\ \swarrow & & \searrow \\ AA \underset{run}{\bowtie} A1 & & BB \underset{run}{\bowtie} B1 \end{array}$$

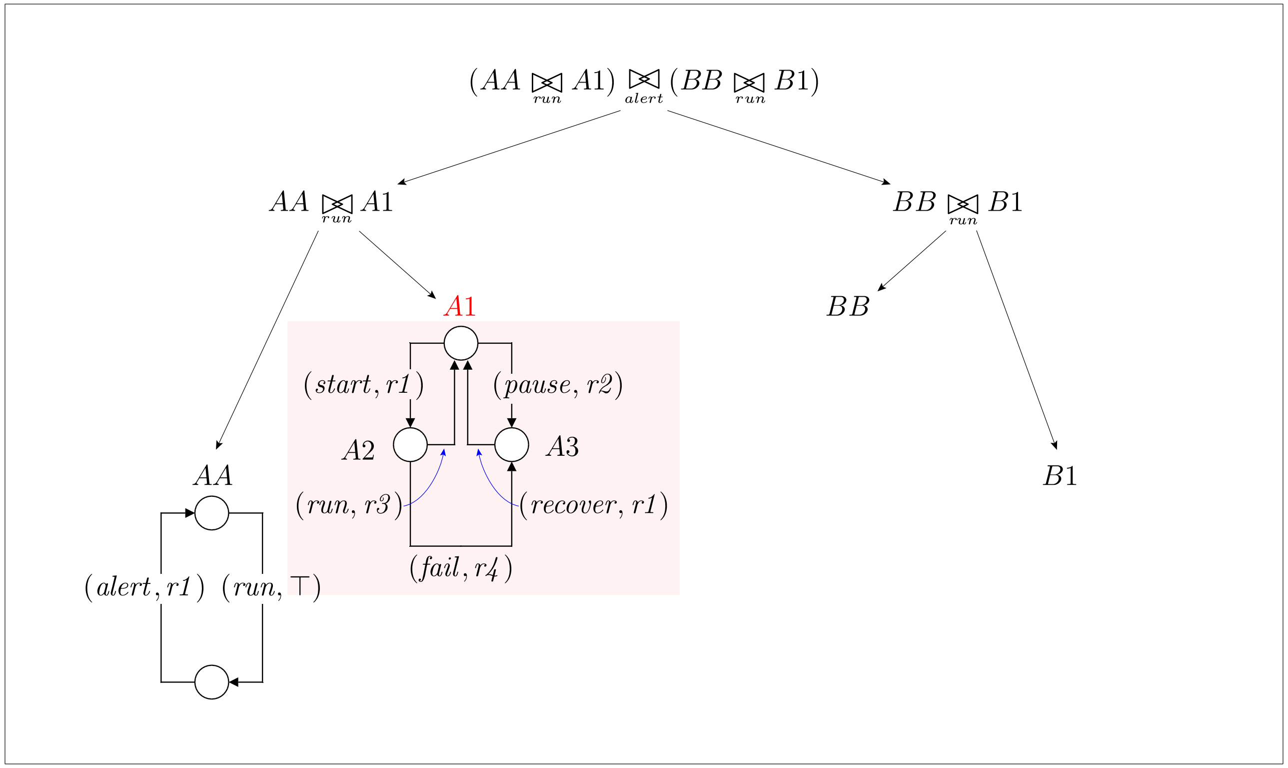
# Example ipc Compilation



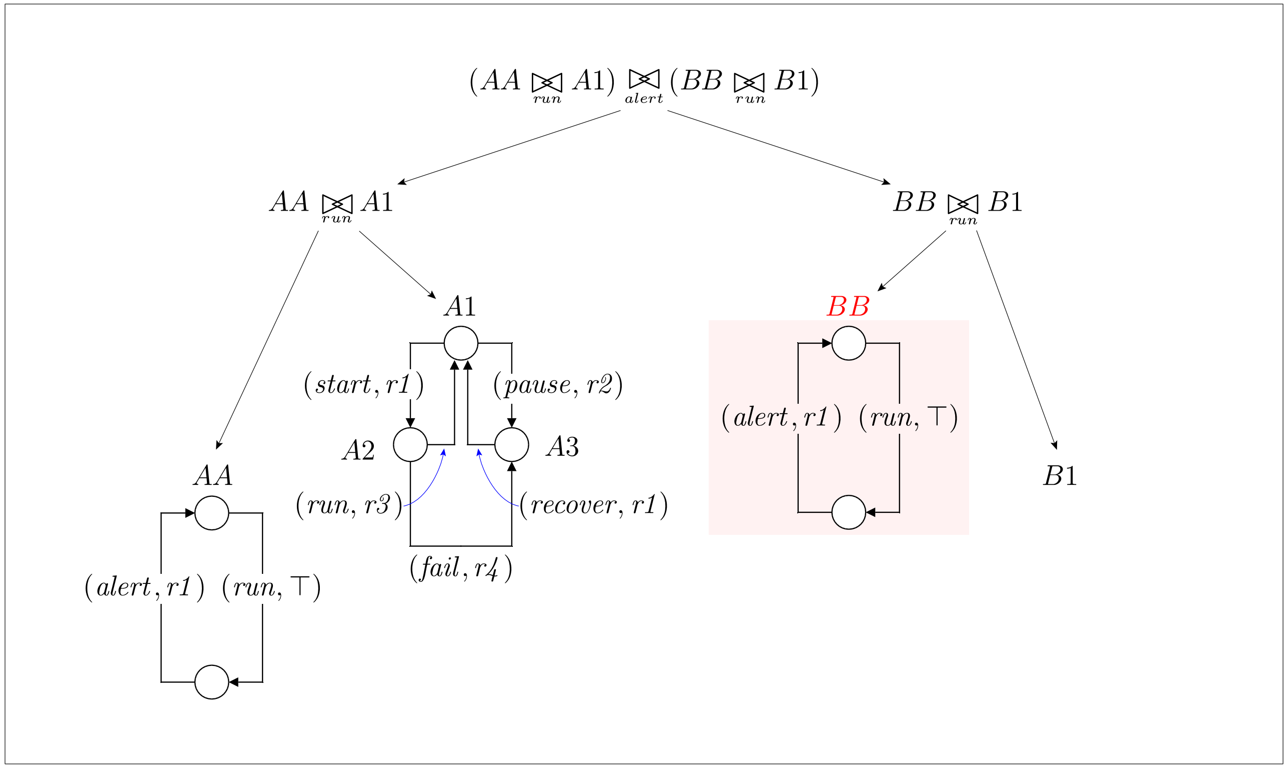
# Example ipc Compilation



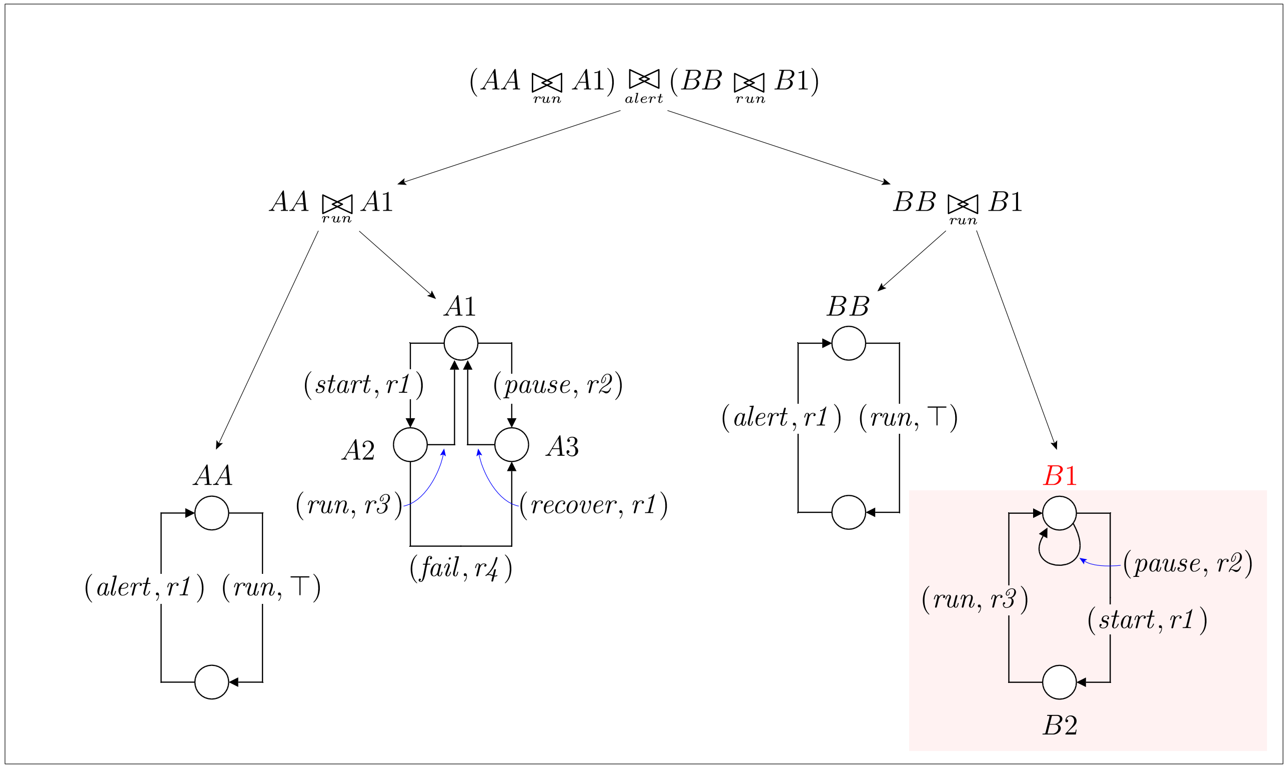
# Example ipc Compilation



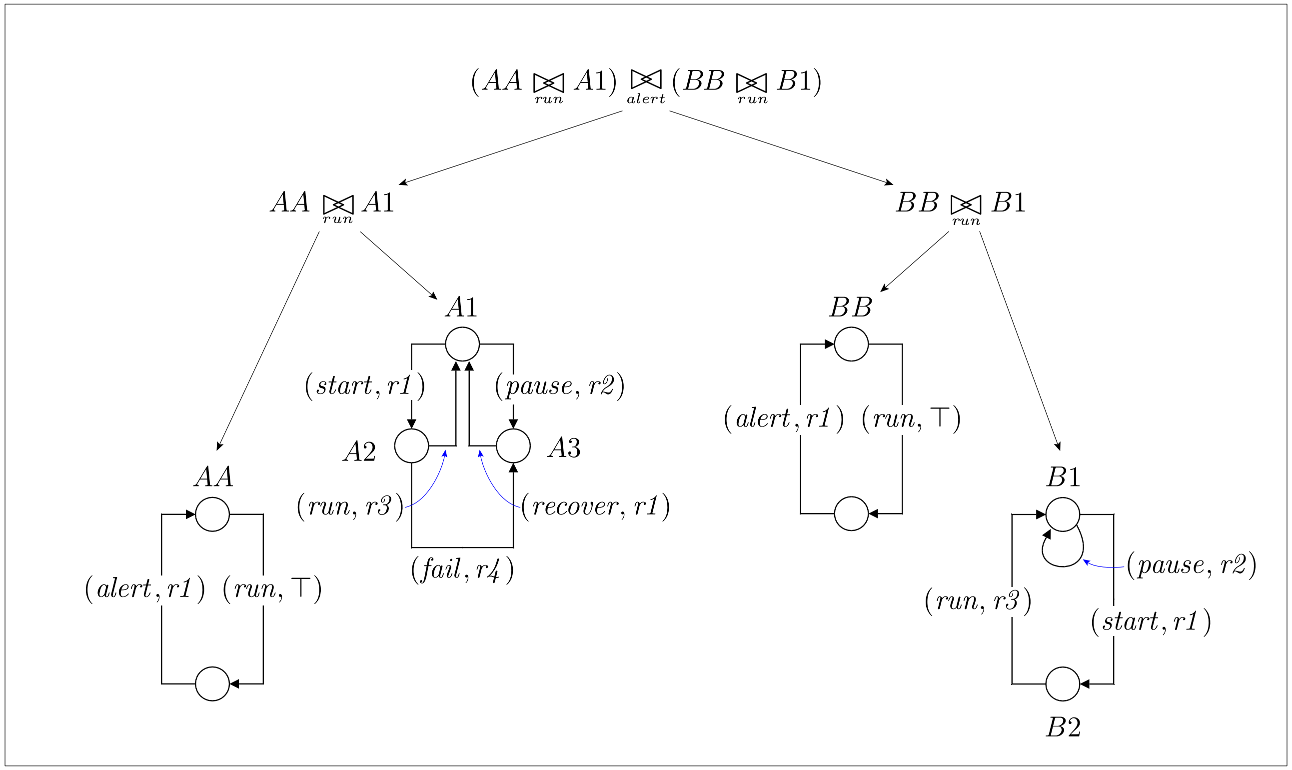
# Example ipc Compilation



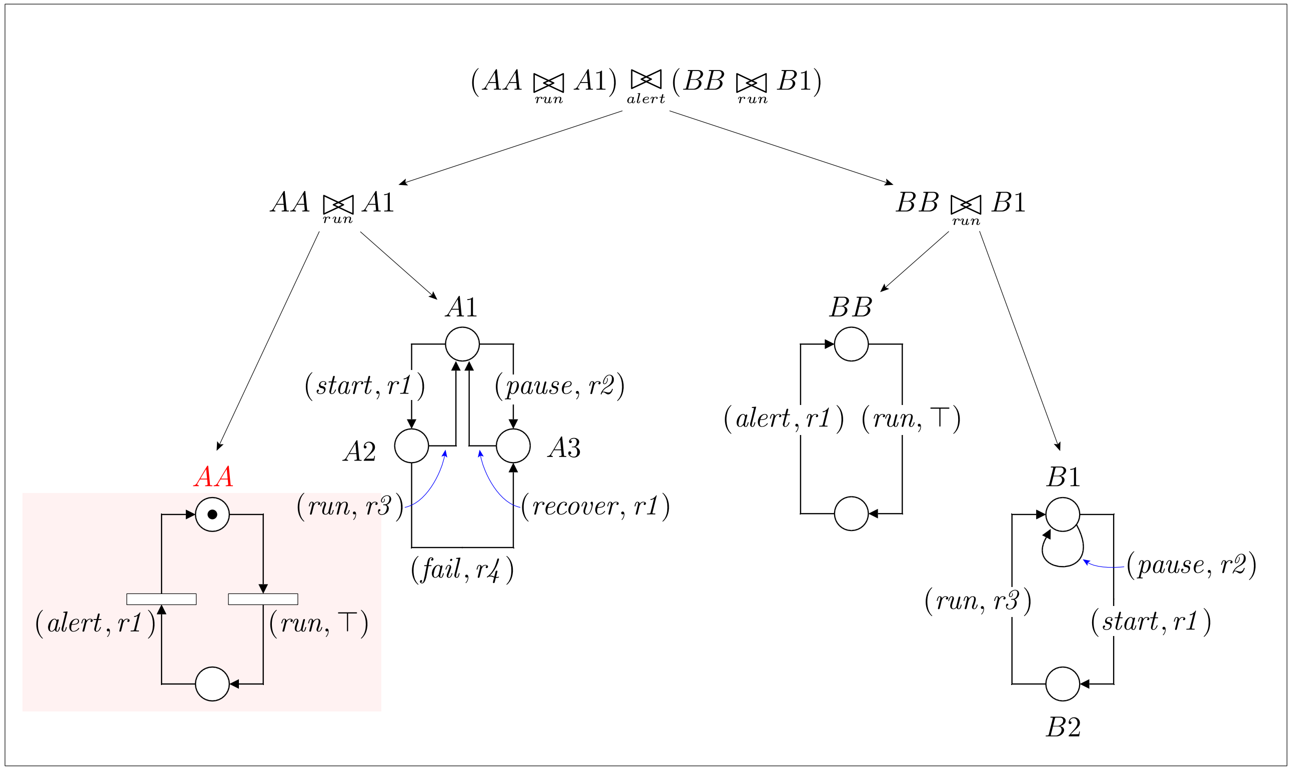
# Example ipc Compilation



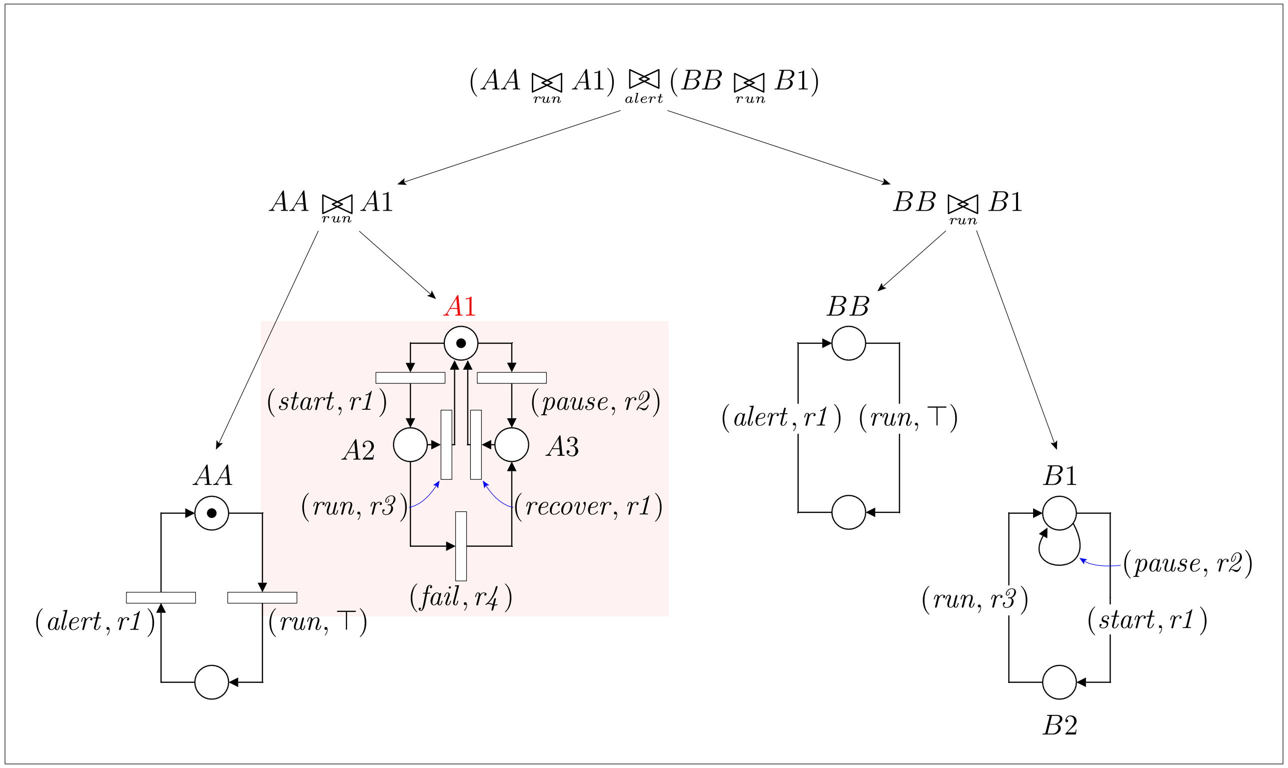
# Example ipc Compilation



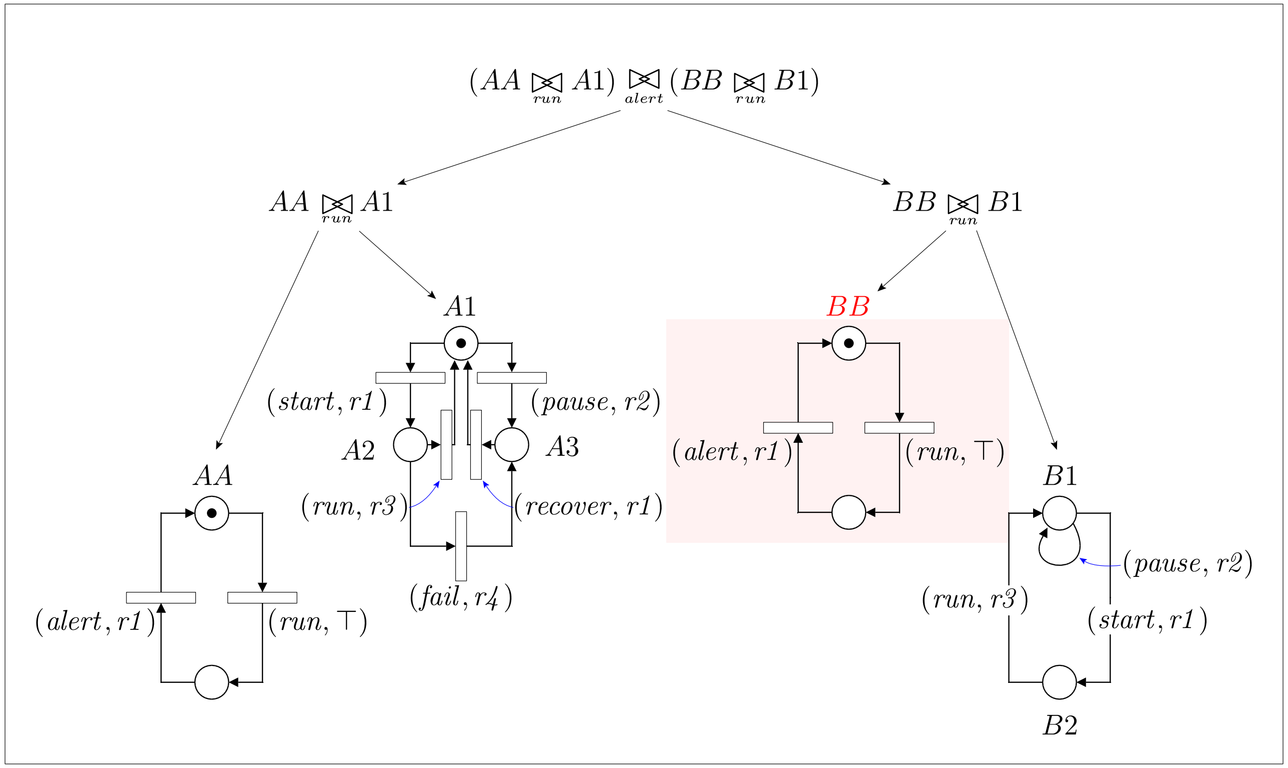
# Example ipc Compilation



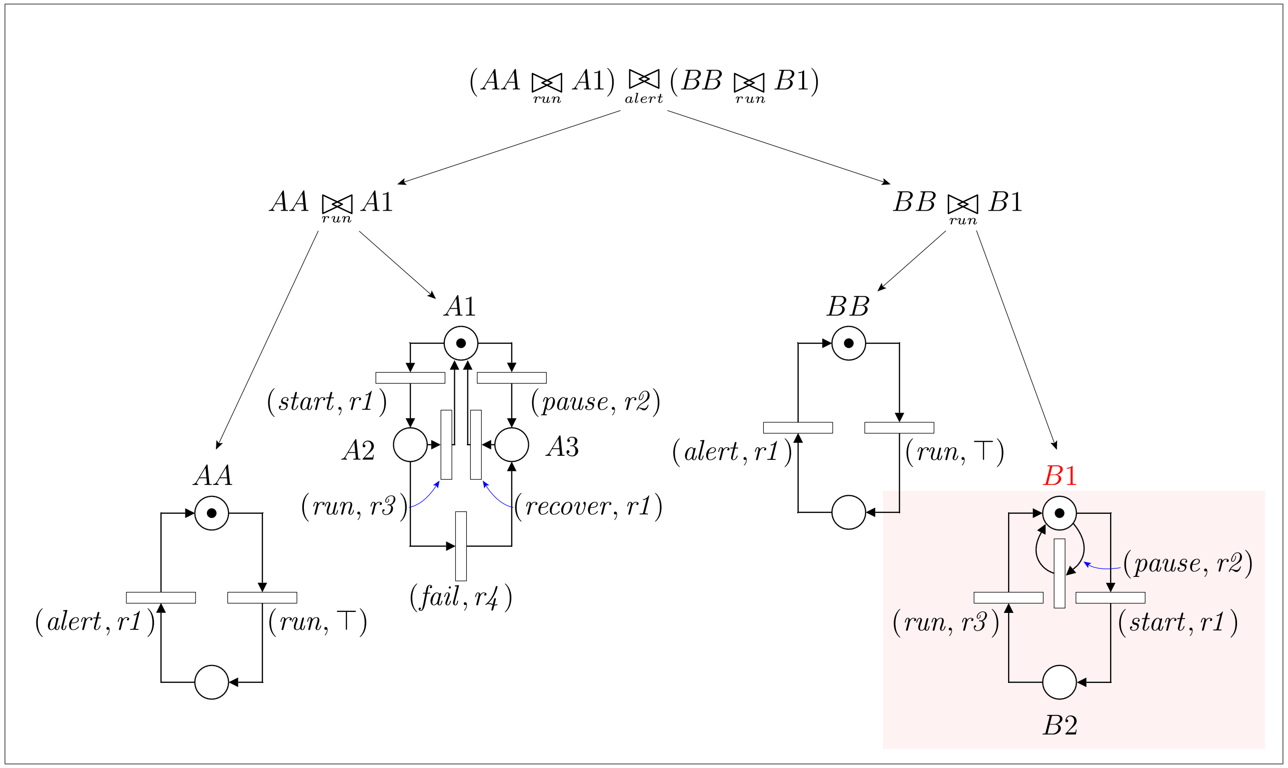
# Example ipc Compilation



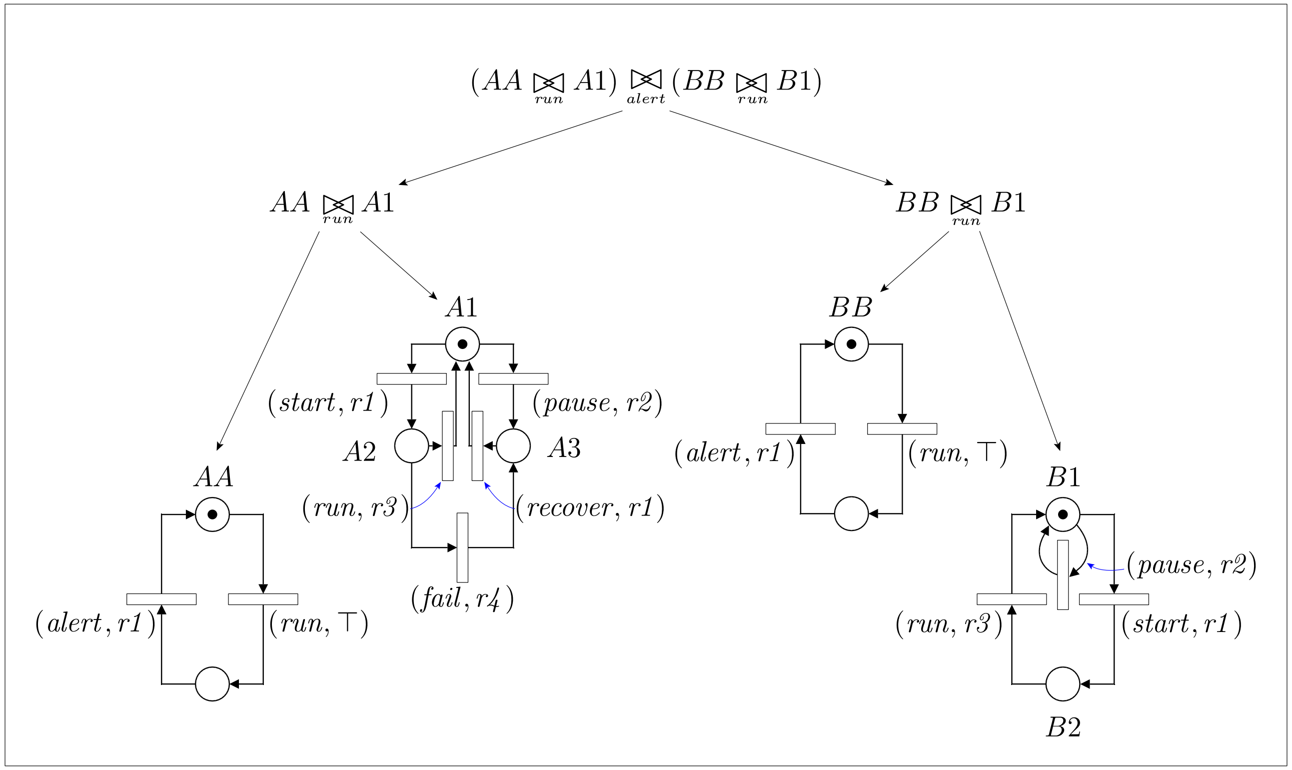
# Example ipc Compilation



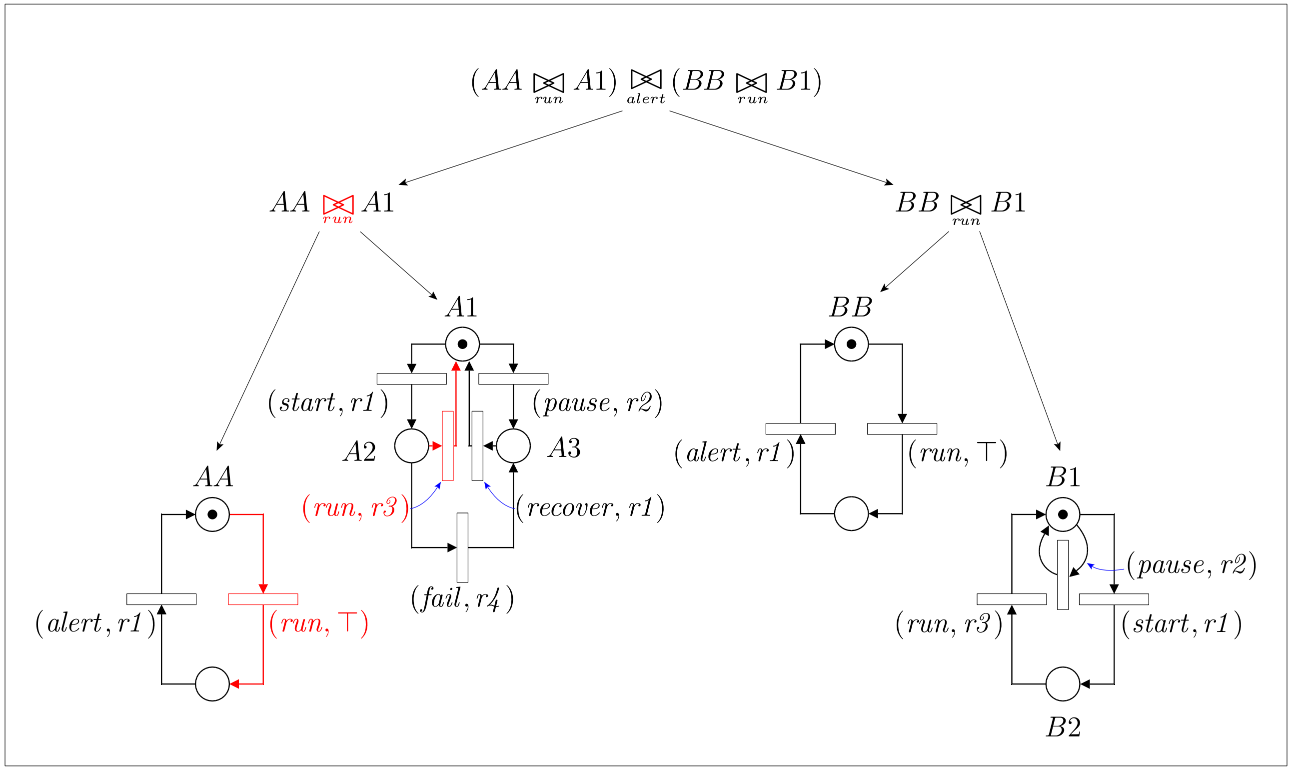
# Example ipc Compilation



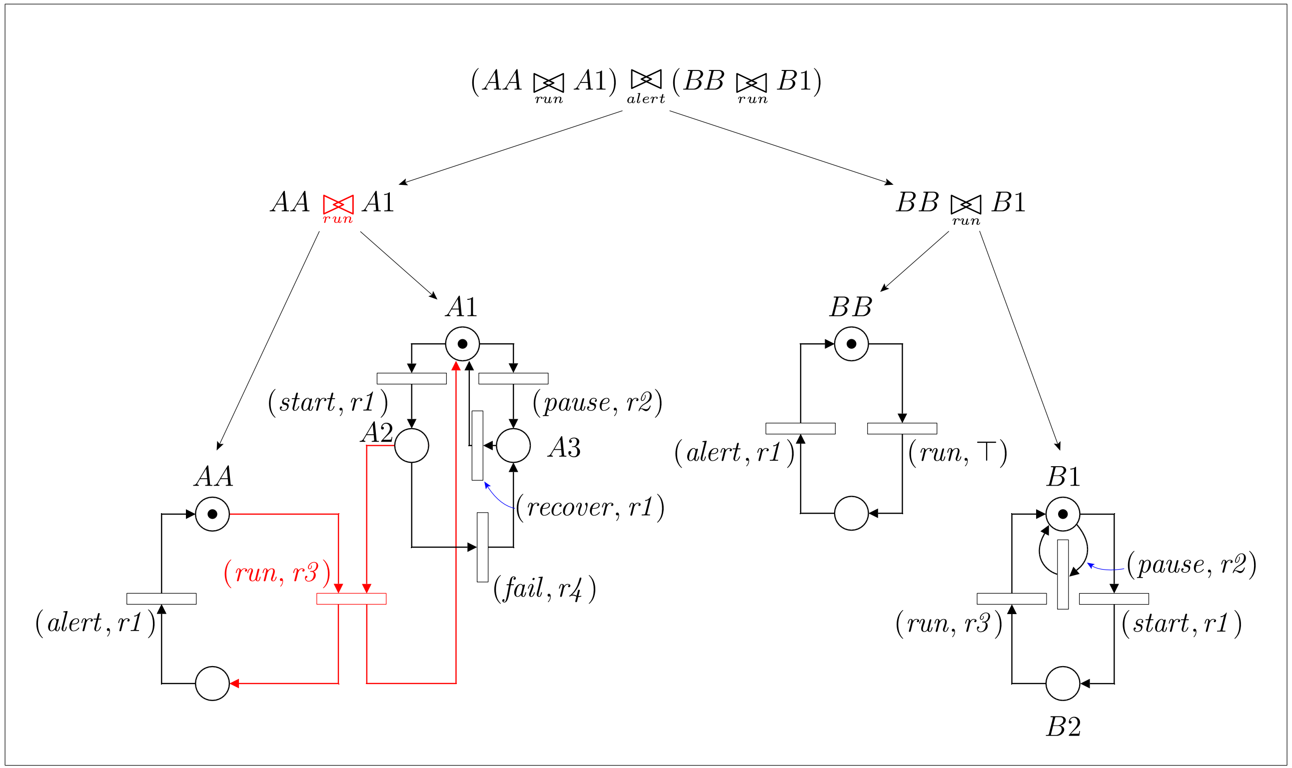
# Example ipc Compilation



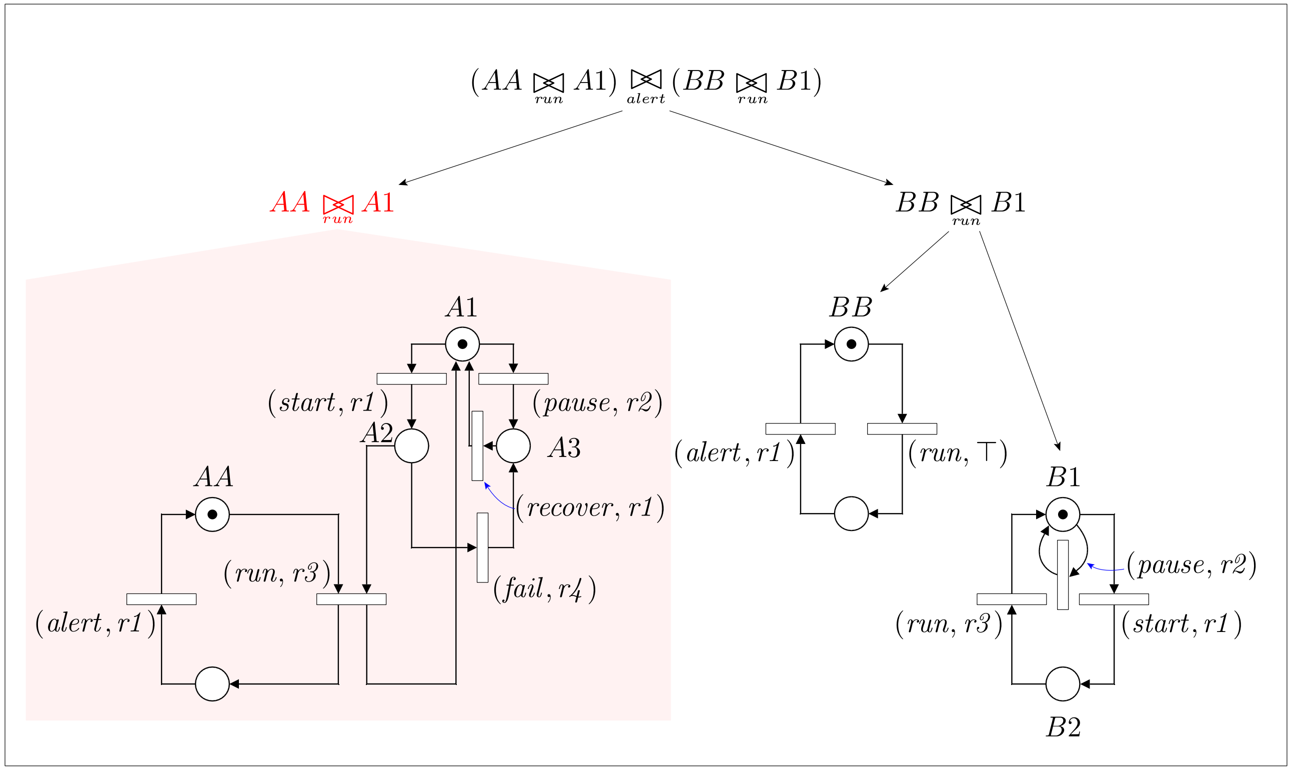
# Example ipc Compilation



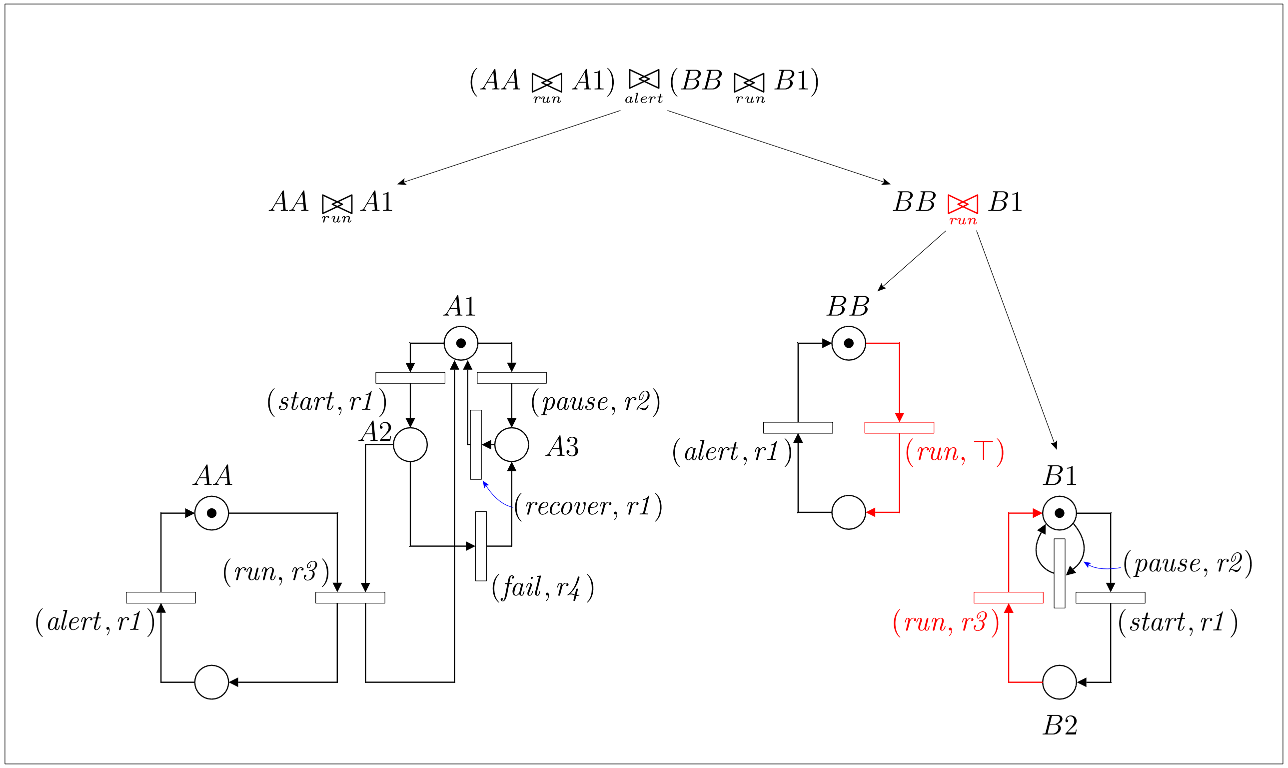
# Example ipc Compilation



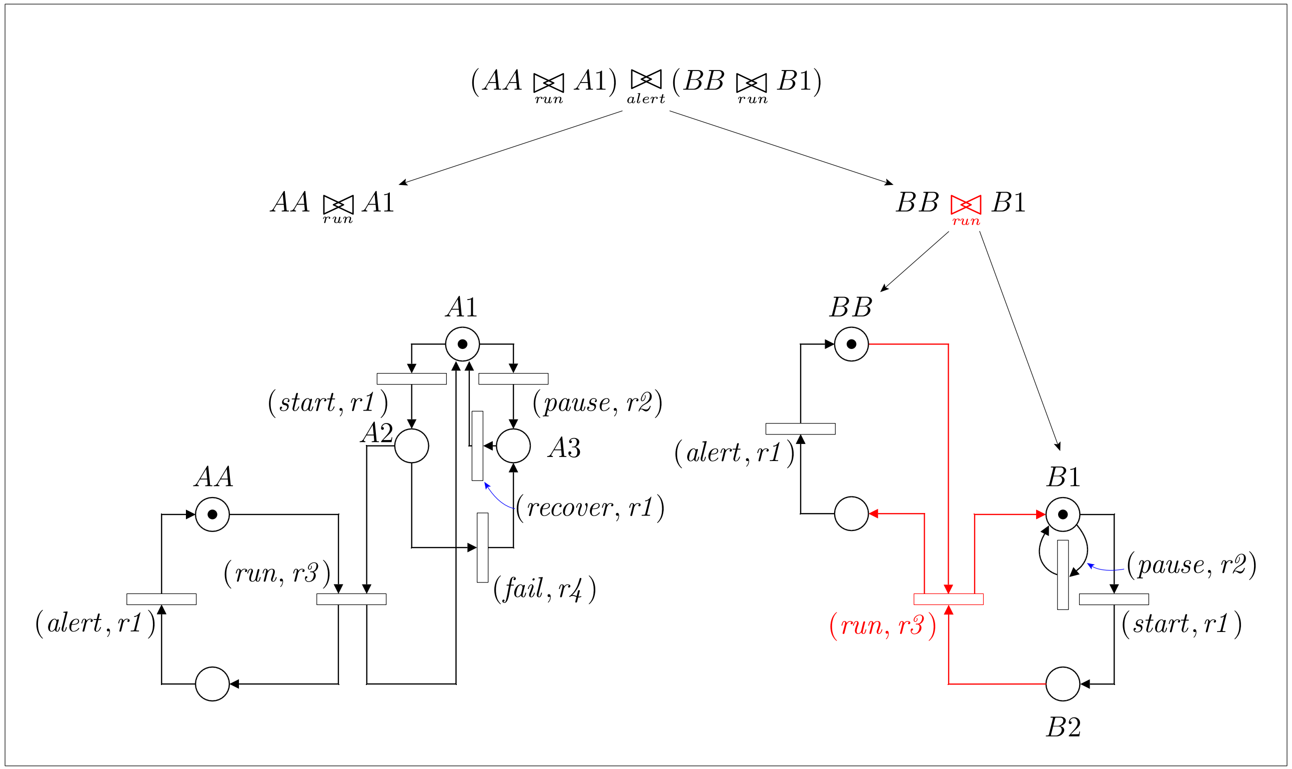
# Example ipc Compilation



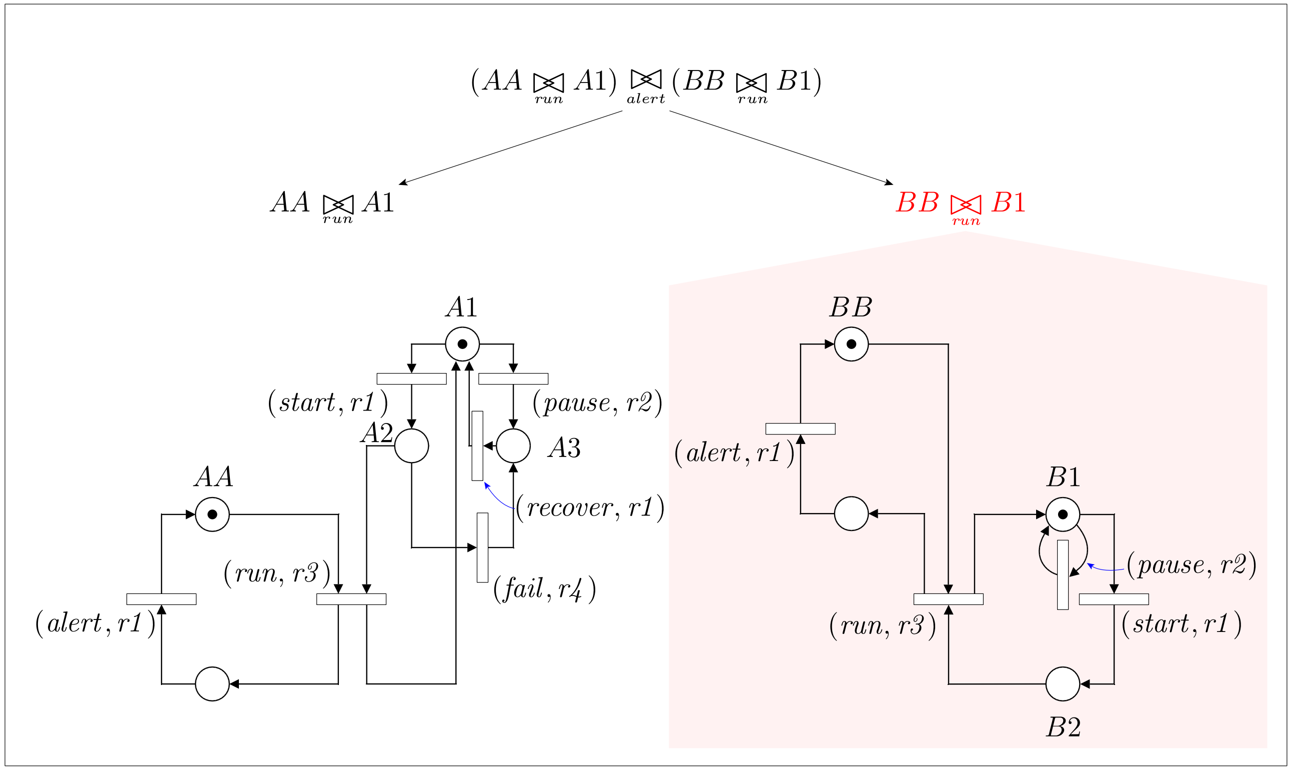
# Example ipc Compilation



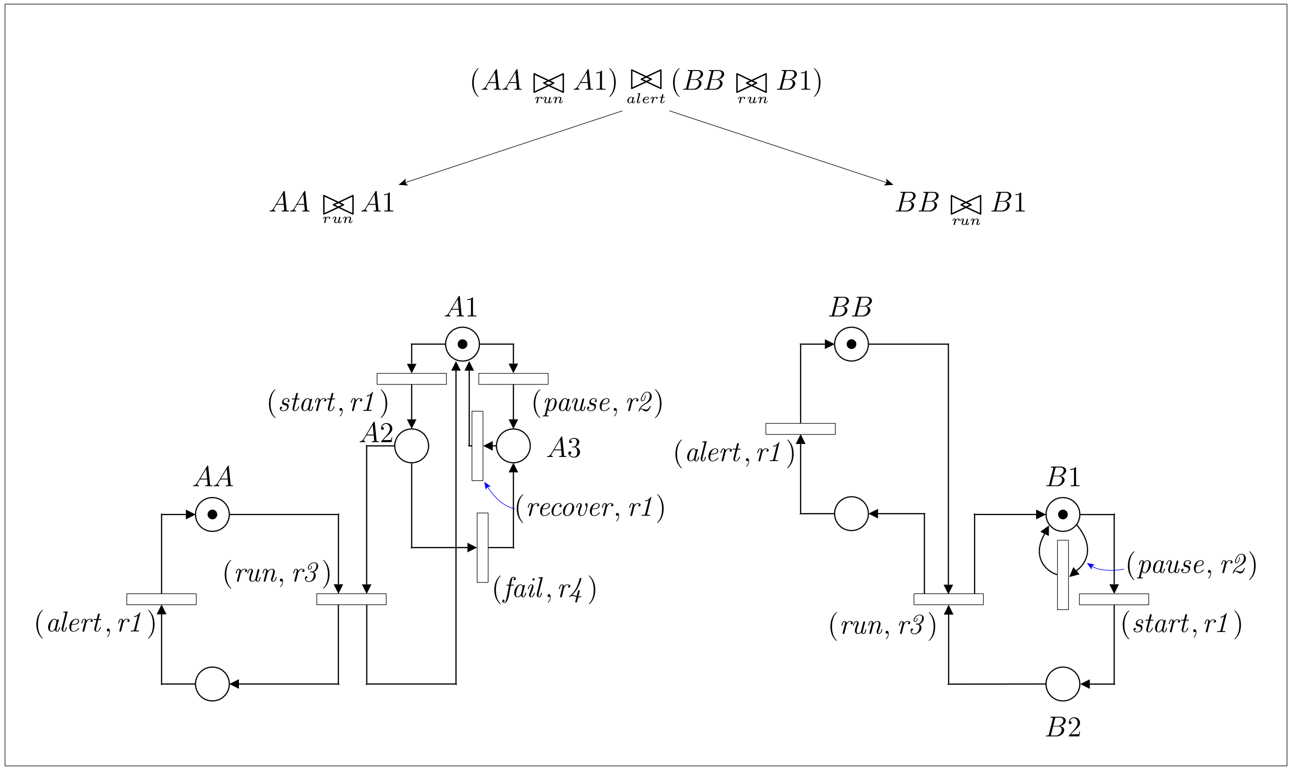
# Example ipc Compilation



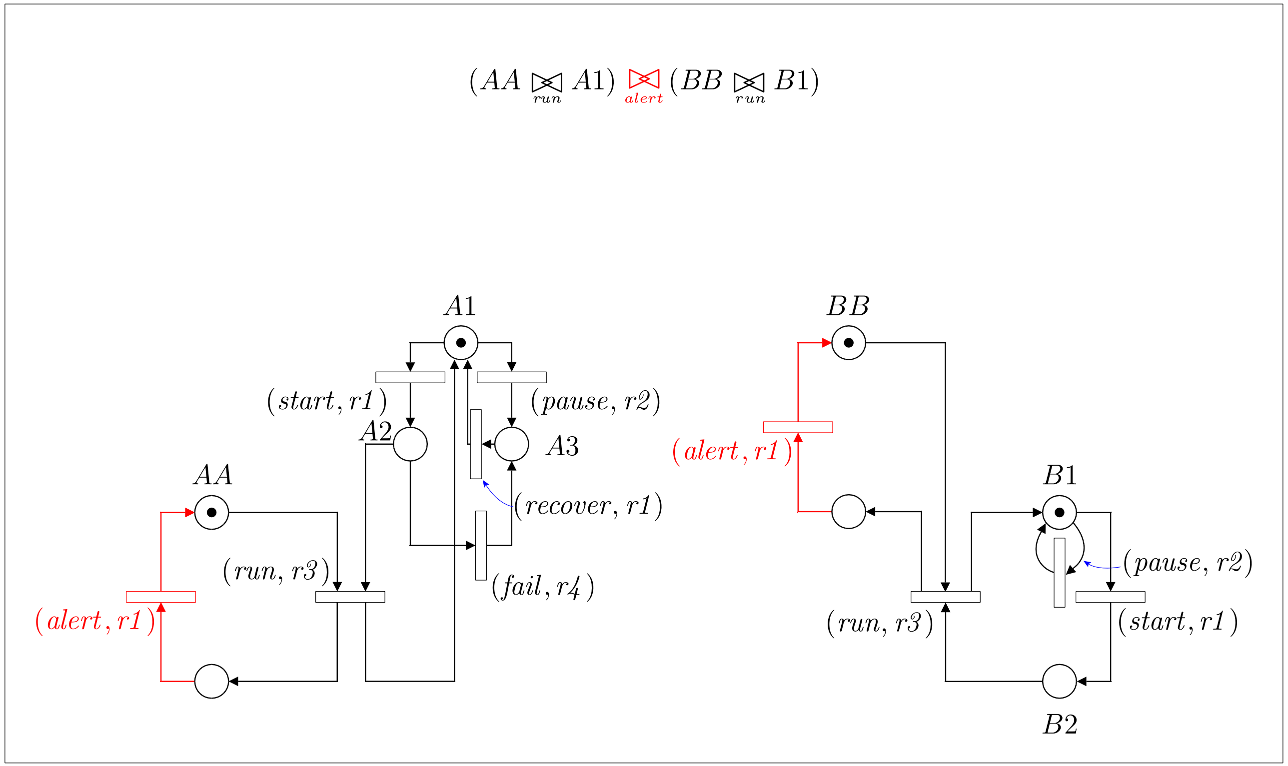
# Example ipc Compilation



# Example ipc Compilation

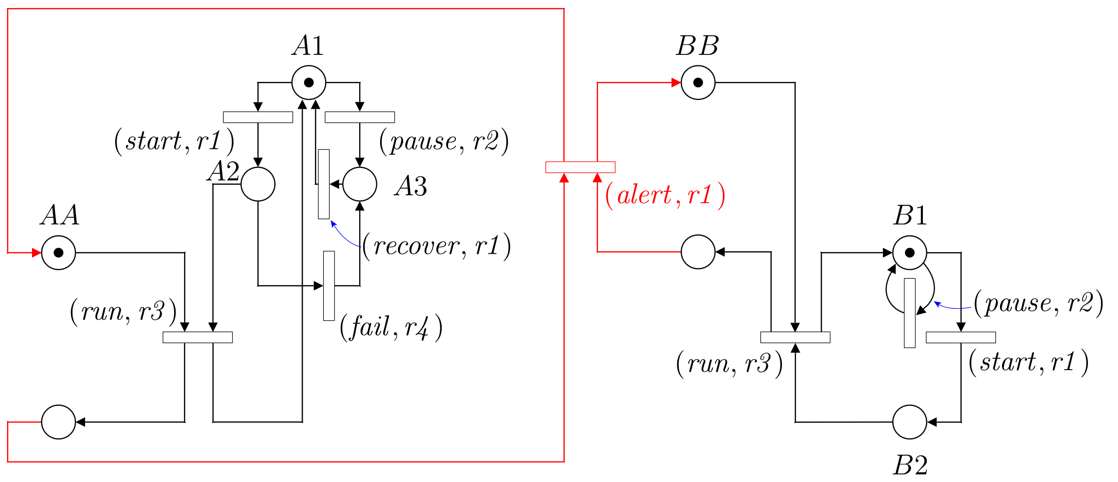


# Example ipc Compilation

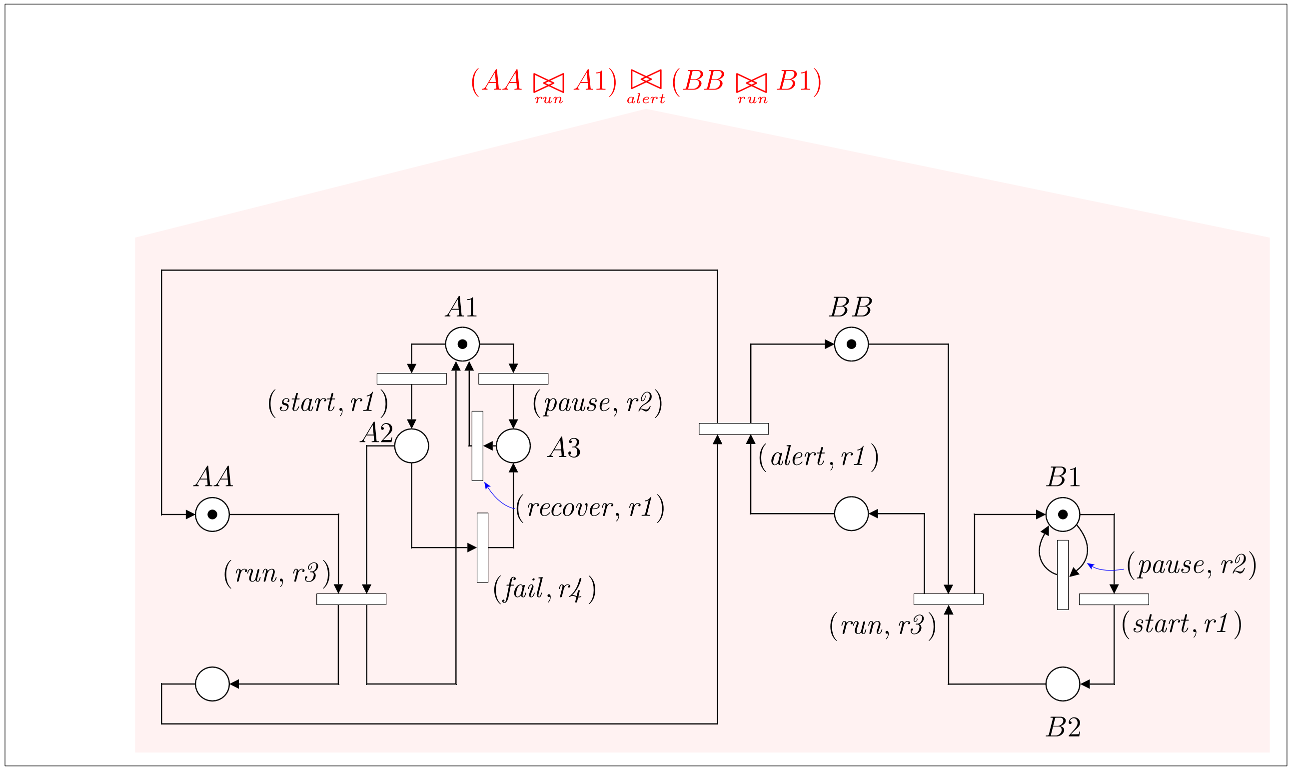


# Example ipc Compilation

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

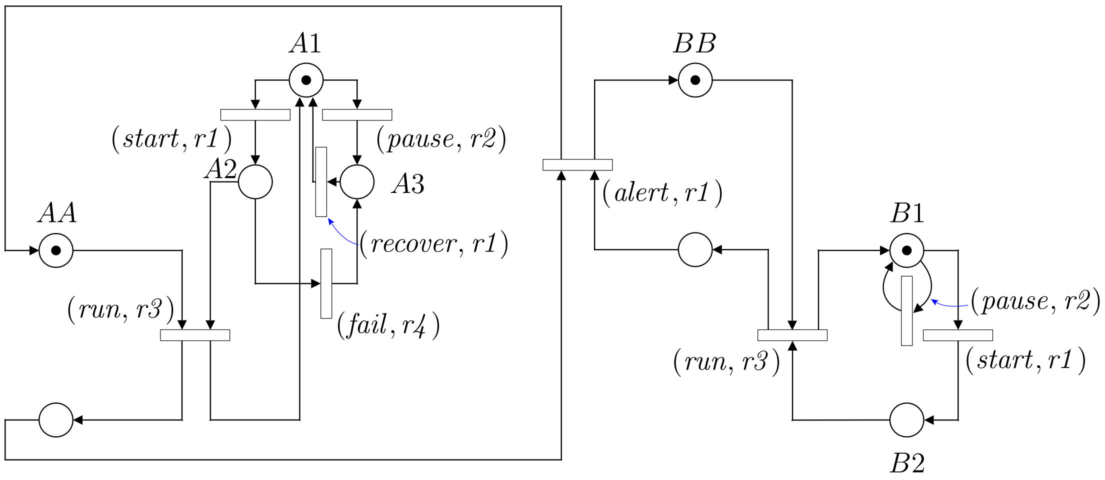


# Example ipc Compilation



# Example ipc Compilation

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



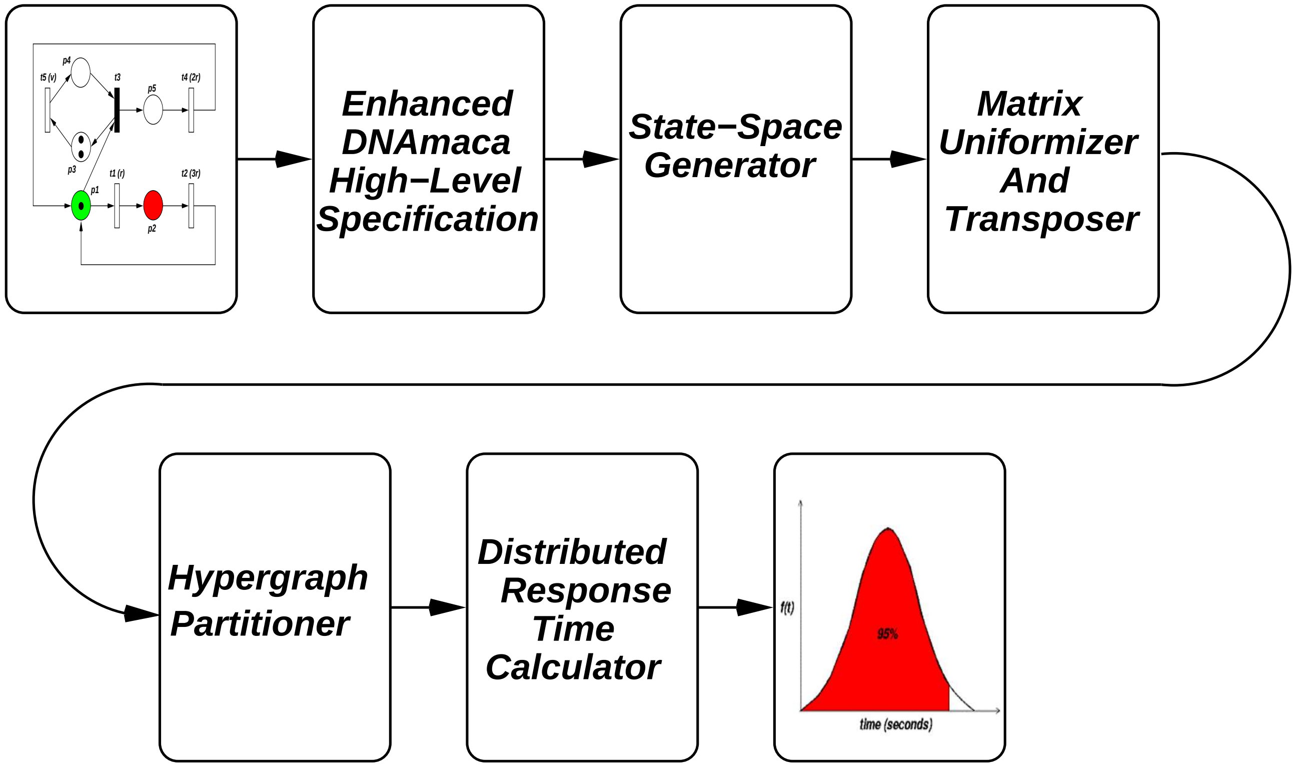
# DNAmaca: Example

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 }
}
```

# HYDRA :: DNAmaca $\longrightarrow$ Passage

# HYDRA Tool Pipeline



# Uniformization (I)

- Used for transient and passage-time analysis of continuous-time Markov chains (CTMCs)

- A CTMC generator matrix , $Q$, is "uniformized" with:

$$P = Q/q + I$$

  where $q > \max_i |q_{ii}|$

- Transforms a CTMC into one in which all states have the same mean holding time $1/q$

# Uniformization (II)

- Passage/transient calculation has two main components:
  - The time to complete $n$ hops ($n = 1, 2, 3, \ldots$), which is an Erlang distribution with parameters $n$ and $q$
  - The probability that the transition between source and target states occurs in exactly $n$ hops
  - In equation form:

$$f_{\vec{i}\vec{j}}(t) = \sum_{n=1}^{\infty} \frac{q^n t^{n-1} e^{-qt}}{(n-1)!} \sum_{k \in \vec{j}} \tilde{\pi}_k^{(n)}$$

# Example: Web server

- Servers receive read requests from surfers' web-browsers and write requests from content authors out of a buffer

- Successful write request requires that none of the servers are in the process of processing a read

- Servers may fail and recover individually or collectively

# Example: Web server

- A server is described as:

$$
\begin{aligned}
\mathsf{Server} &\stackrel{\text{def}}{=} (\mathsf{sReadRequest}, \top).\mathsf{ServerRead} \\
&+ (\mathsf{sFail}, rsf).\mathsf{ServerFail} \\
&+ (\mathsf{sWrite}, rsw).\mathsf{Server} \\
\mathsf{ServerRead} &\stackrel{\text{def}}{=} (\mathsf{sReadLookup}, rsrl).\mathsf{Server} \\
\mathsf{ServerFail} &\stackrel{\text{def}}{=} (\mathsf{sFailRecover}, rsfr).\mathsf{Server} \\
&+ (\mathsf{sFailRecoverAll}, \top).\mathsf{Server} \\
&+ (\mathsf{sWrite}, rsfsw).\mathsf{ServerFail}
\end{aligned}
$$

# Example: Web server

- A component tracks the number of failed servers:

$$
\begin{array}{rcl}
\mathsf{ServerGroup}_0 & \stackrel{\text{def}}{=} & (\mathsf{sFail}, \top).\mathsf{ServerGroup}_1 \\
\mathsf{ServerGroup}_\mathsf{i} & \stackrel{\text{def}}{=} & (\mathsf{sFail}, \top).\mathsf{ServerGroup}_{\mathsf{i}+1} \\
 & + & (\mathsf{sFailRecover}, \top).\mathsf{ServerGroup}_{\mathsf{i}-1} \quad : 1 \leq \mathsf{i} < \mathsf{S} \\
\mathsf{ServerGroup}_\mathsf{S} & \stackrel{\text{def}}{=} & (\mathsf{sFailRecoverAll}, rsgsfra).\mathsf{ServerGroup}_0
\end{array}
$$

- When $S = 4$:

$$
\mathsf{Servers} \stackrel{\text{def}}{=} (\mathsf{Server} \bowtie_{\mathcal{L}} \mathsf{Server} \bowtie_{\mathcal{L}} \mathsf{Server} \bowtie_{\mathcal{L}} \mathsf{Server}) \bowtie_{\mathcal{L}'} \mathsf{ServerGroup}_0
$$

where $\mathcal{L} = \{\mathsf{sWrite}, \mathsf{sFailRecoverAll}\}$ and $\mathcal{L}' = \mathcal{L} \cup \{\mathsf{sFailRecover}\}$.

# Example: Web server

- Buffered writes are tracked by a dedicated component:

$$
\begin{aligned}
\mathsf{WriteBuffer}_0 &\stackrel{\text{def}}{=} (\mathsf{bWrite}, \top).\mathsf{WriteBuffer}_1 \\
\mathsf{WriteBuffer}_i &\stackrel{\text{def}}{=} (\mathsf{bWrite}, \top).\mathsf{WriteBuffer}_{i+1} \\
&\quad + (\mathsf{sWrite}, \top).\mathsf{WriteBuffer}_0 \quad : 1 \leq i < B \\
\mathsf{WriteBuffer}_B &\stackrel{\text{def}}{=} (\mathsf{sWrite}, \top).\mathsf{WriteBuffer}_0
\end{aligned}
$$

# Example: Web server

- An individual writer is described as:

$$\begin{aligned} \mathsf{Writer} &\stackrel{\text{def}}{=} (\mathsf{bWrite}, rwbw).\mathsf{WriterWrite} \\ \mathsf{WriterWrite} &\stackrel{\text{def}}{=} (\mathsf{rwResetAll}, \top).\mathsf{Writer} \end{aligned}$$

- When $W = 3$:

$$\mathsf{Writers} \stackrel{\text{def}}{=} \mathsf{Writer} \bowtie_{\mathcal{K}} \mathsf{Writer} \bowtie_{\mathcal{K}} \mathsf{Writer}$$

where $\mathcal{K} = \{\mathsf{rwResetAll}\}$.

# Example: Web server

- Readers send read requests and then await the response from the server:

$$\begin{aligned} \mathsf{Reader} &\stackrel{\mathrm{def}}{=} (\mathsf{sReadRequest}, rrrsrr).(\mathsf{sReadLookup}, \top).\mathsf{ReaderRead} \\ \mathsf{ReaderRead} &\stackrel{\mathrm{def}}{=} (\mathsf{rwResetAll}, \top).\mathsf{Reader} \end{aligned}$$

- For $R = 3$:

$$\mathsf{Readers} \stackrel{\mathrm{def}}{=} \mathsf{Reader} \underset{\mathcal{M}}{\bowtie} \mathsf{Reader} \underset{\mathcal{M}}{\bowtie} \mathsf{Reader}$$

where $\mathcal{M} = \{\mathsf{rwResetAll}\}$.

- Readers and writers are reset by a dedicated component:

$$\mathsf{RWReset} \stackrel{\mathrm{def}}{=} (\mathsf{rwResetAll}, rwra).\mathsf{RWReset}$$

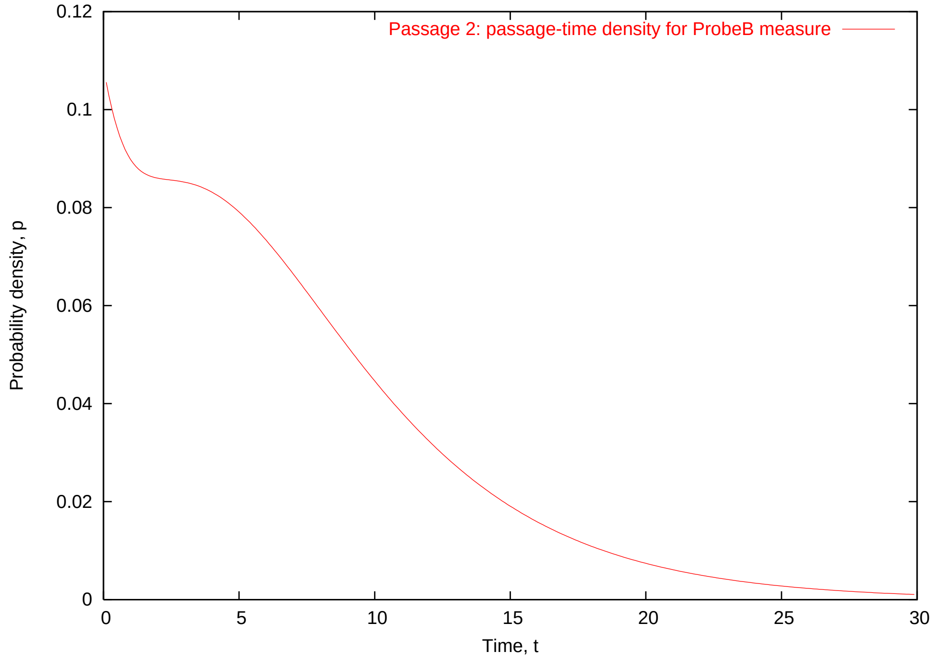
# Example: Web server

- The full model:

$$\begin{aligned}
\mathsf{Environment} &\stackrel{\mathrm{def}}{=} \mathsf{Writers} \underset{\{\mathsf{rwResetAll}\}}{\bowtie} \mathsf{Readers} \underset{\{\mathsf{rwResetAll}\}}{\bowtie} \mathsf{RWReset} \\
\mathsf{WebSystem} &\stackrel{\mathrm{def}}{=} \mathsf{Servers} \underset{\{\mathsf{sWrite}\}}{\bowtie} \mathsf{WriteBuffer}_0 \\
\mathsf{System} &\stackrel{\mathrm{def}}{=} \mathsf{Environment} \underset{\mathcal{N}}{\bowtie} \mathsf{WebSystem}
\end{aligned}$$

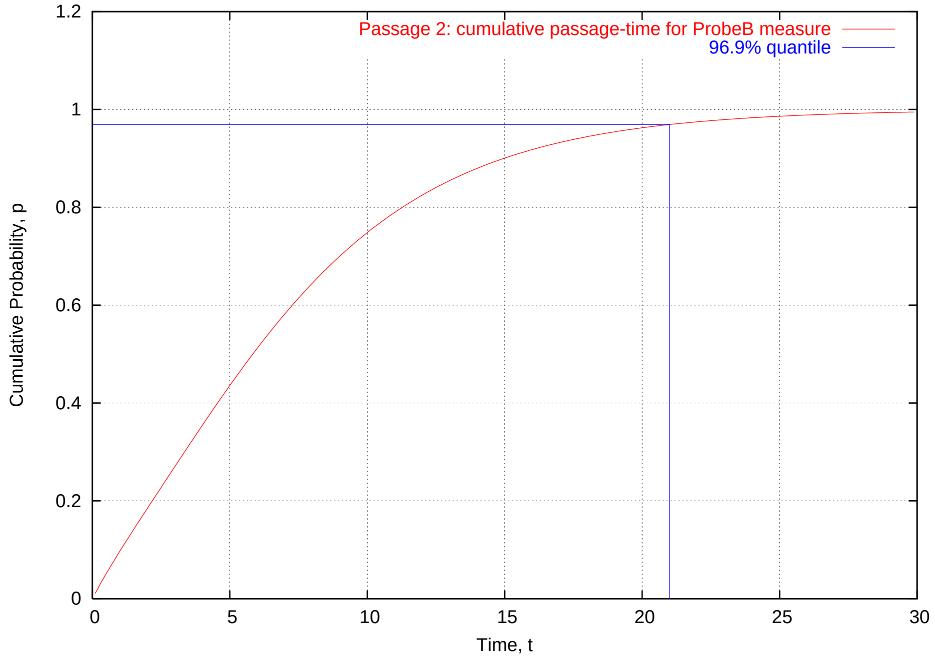
where $\mathcal{N} = \{\mathsf{bWrite}, \mathsf{sReadRequest}, \mathsf{sReadLookup}\}$.

# Output – passage time




Passage-time density for $R$ reads and $W$ writes. ($S$ = 5, $B$ = 4, $W$ = 3, $R$ = 3: 69,440 states)

# Output – passage time cdf




Cumulative passage-time distribution for $R$ reads and $W$ writes and quantile measure. ($S = 5$, $B = 4$, $W = 3$, $R = 3$: 69,440 states)

# Conclusions

- Purpose: to extract passage times from PEPA models
  - Introduced new tools ipc and DNAmaca/HYDRA
  - ipc: PEPA $\rightarrow$ DNAmaca
  - HYDRA: DNAmaca $\rightarrow$ passage time densities/quantiles