Stochastic Probes

Ashok Argent-Katwala, Jeremy Bradley, Nicholas Dingle

a.argent-katwala@doc.ic.ac.uk

Imperial College London

WOSP 2004, Redwood City, CA

15th January 2004

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 *behaviourally 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 $\quad$ $P + P$: competitive choice

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

Stochastic Probe Syntax

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

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

Tagged action names start and stop the measurement over the model

regexp Probes

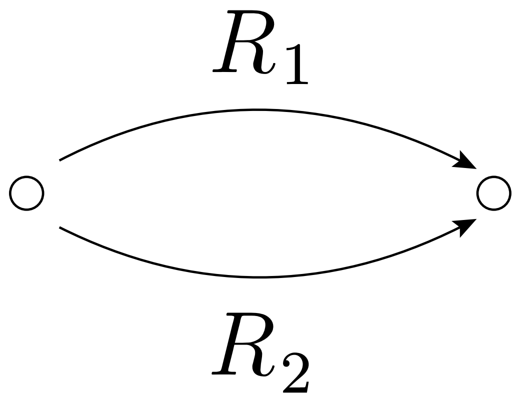
Sequence operator:

$$R_1, R_2 \qquad \circ \xrightarrow{R_1} \circ \xrightarrow{R_2} \circ$$

regexp Probes

Choice operator:

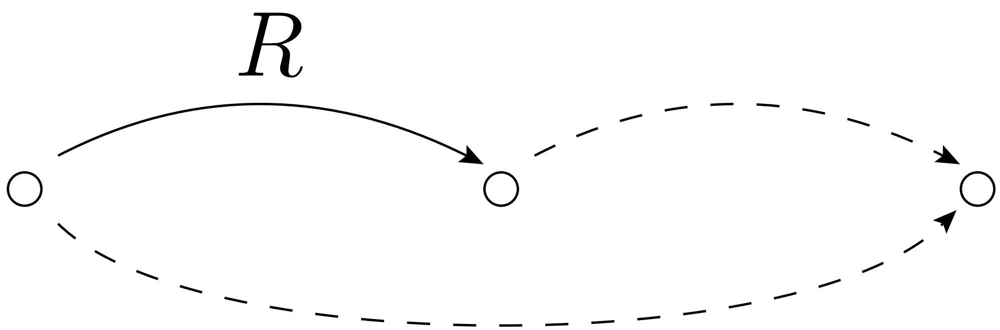
$$R_1 | R_2$$



regexp Probes

Zero-or-one operator:

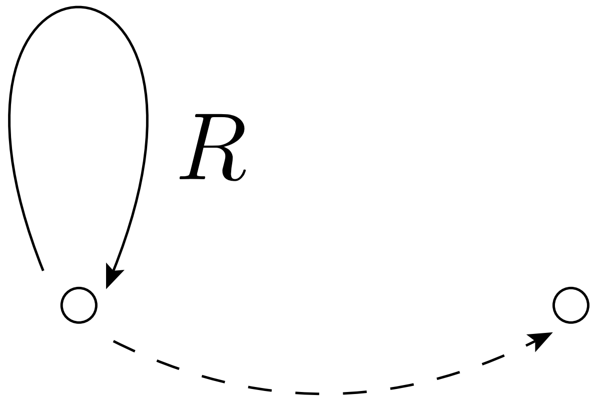
$R?$



regexp Probes

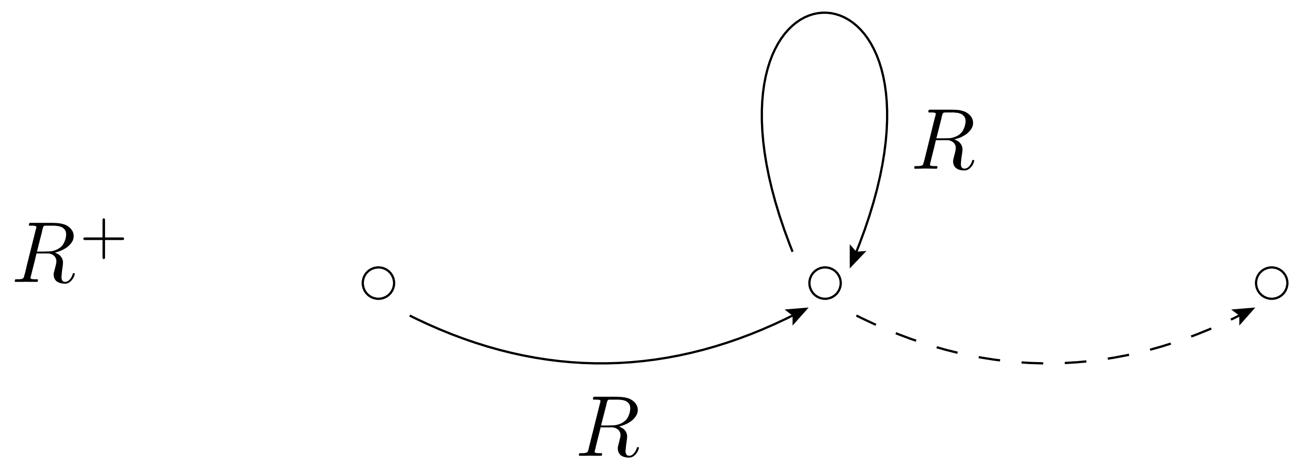
Closure operator:

R^*



regexp Probes

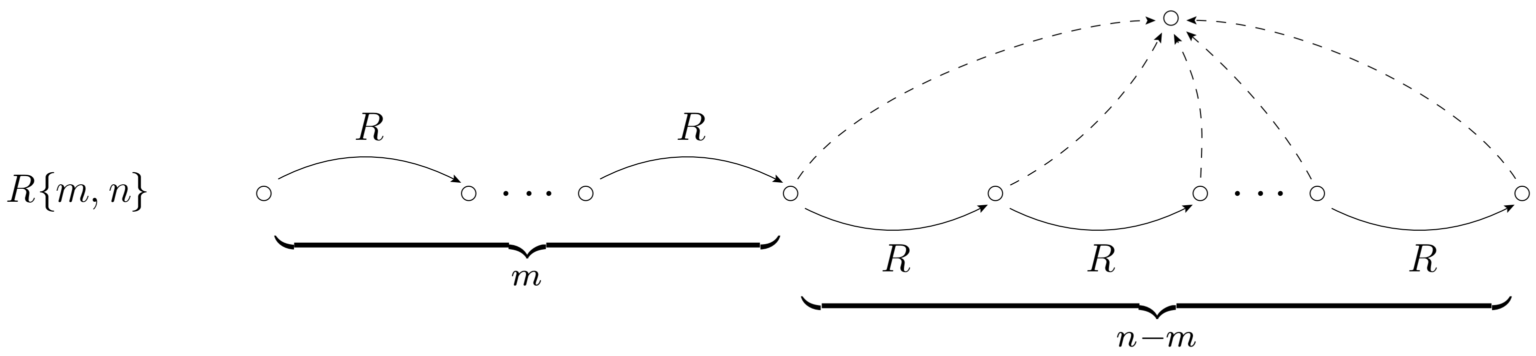
Positive closure operator:



Sugar for R, R^*

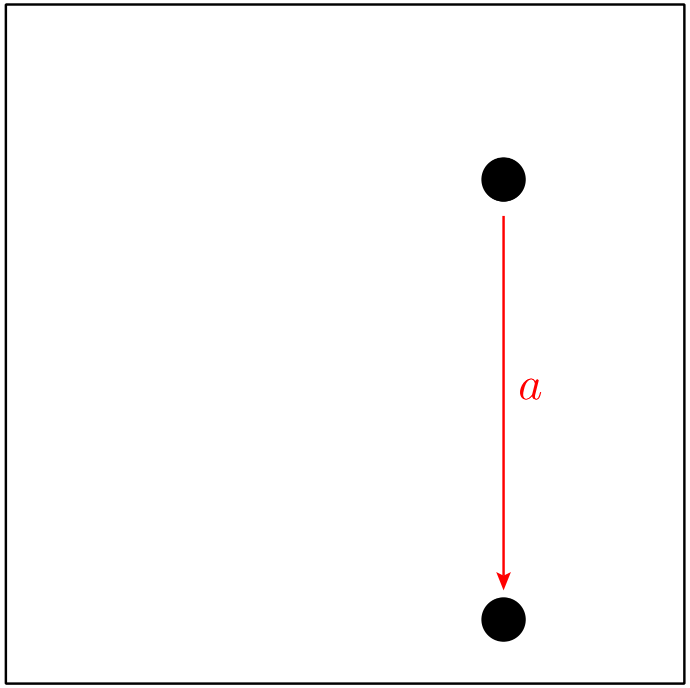
regexp Probes

Range operator:



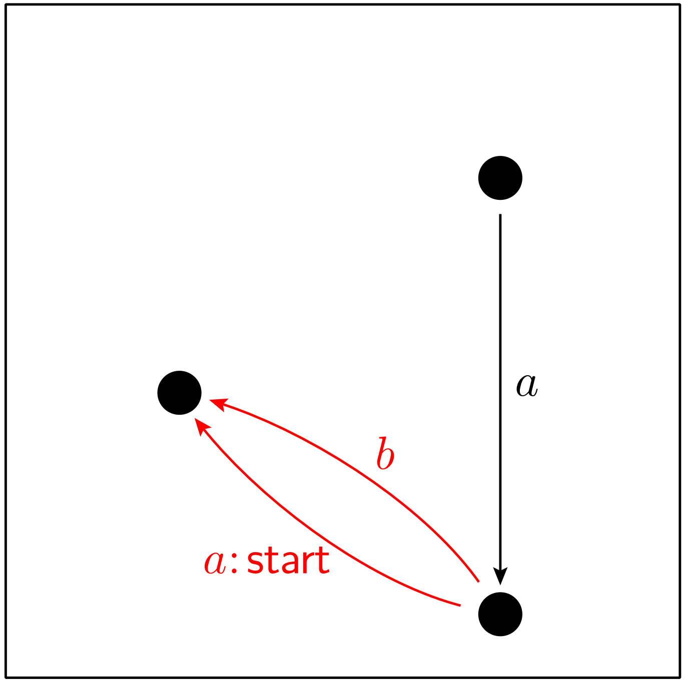
Probe Example

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



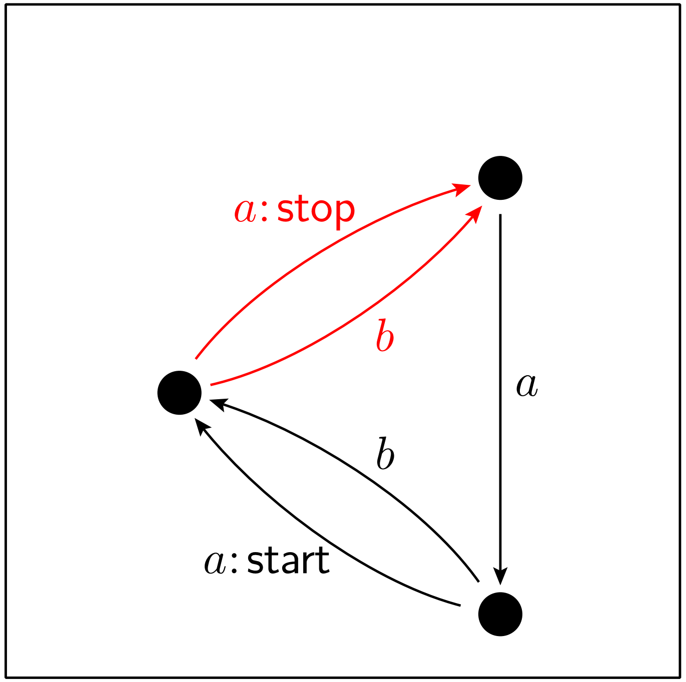
Probe Example

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



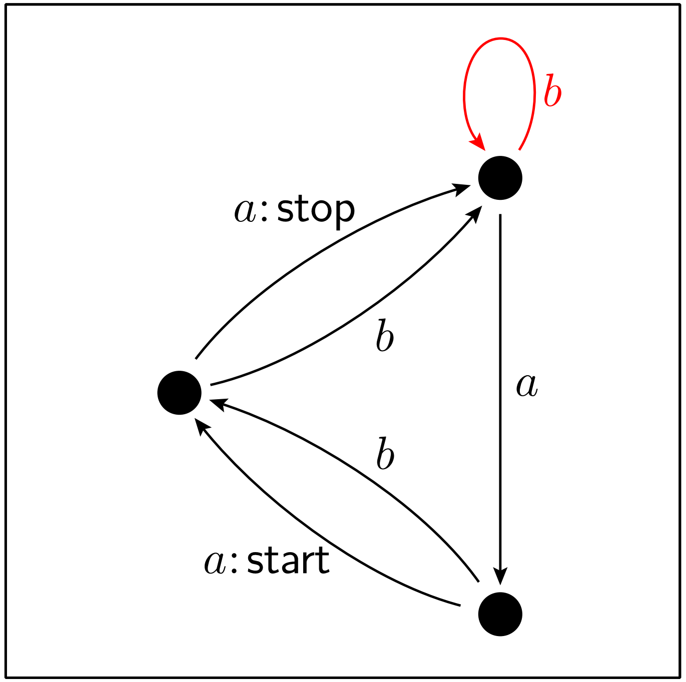
Probe Example

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



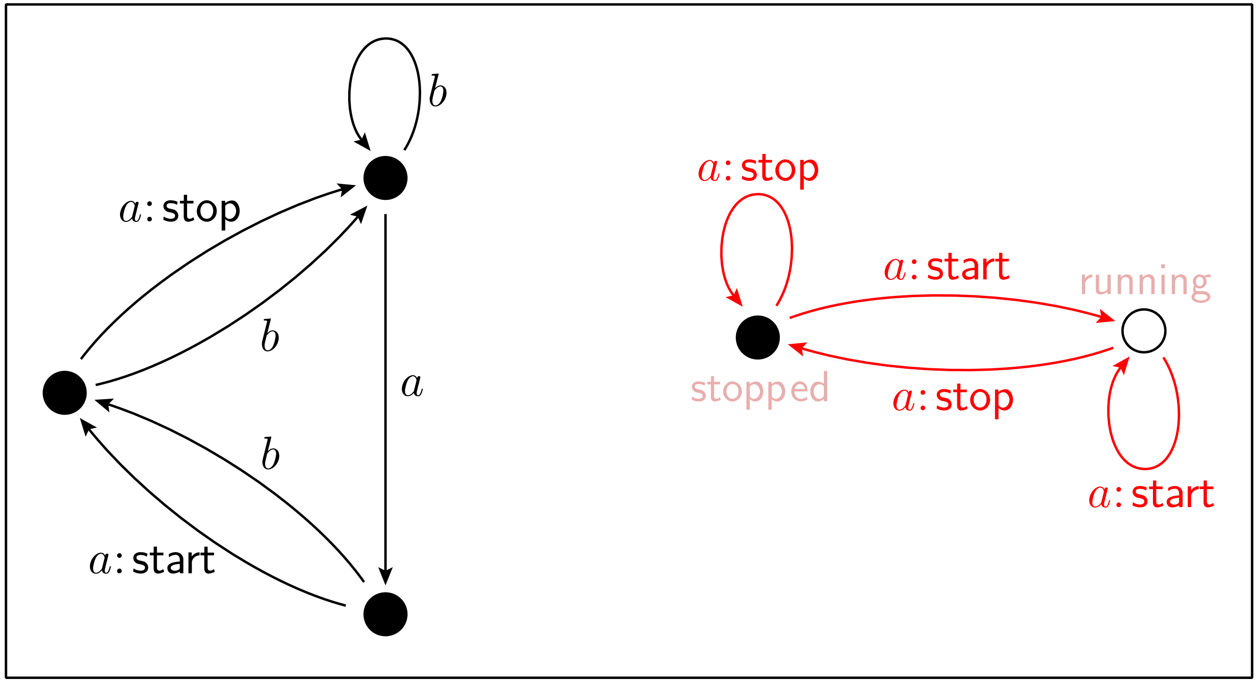
Probe Example

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



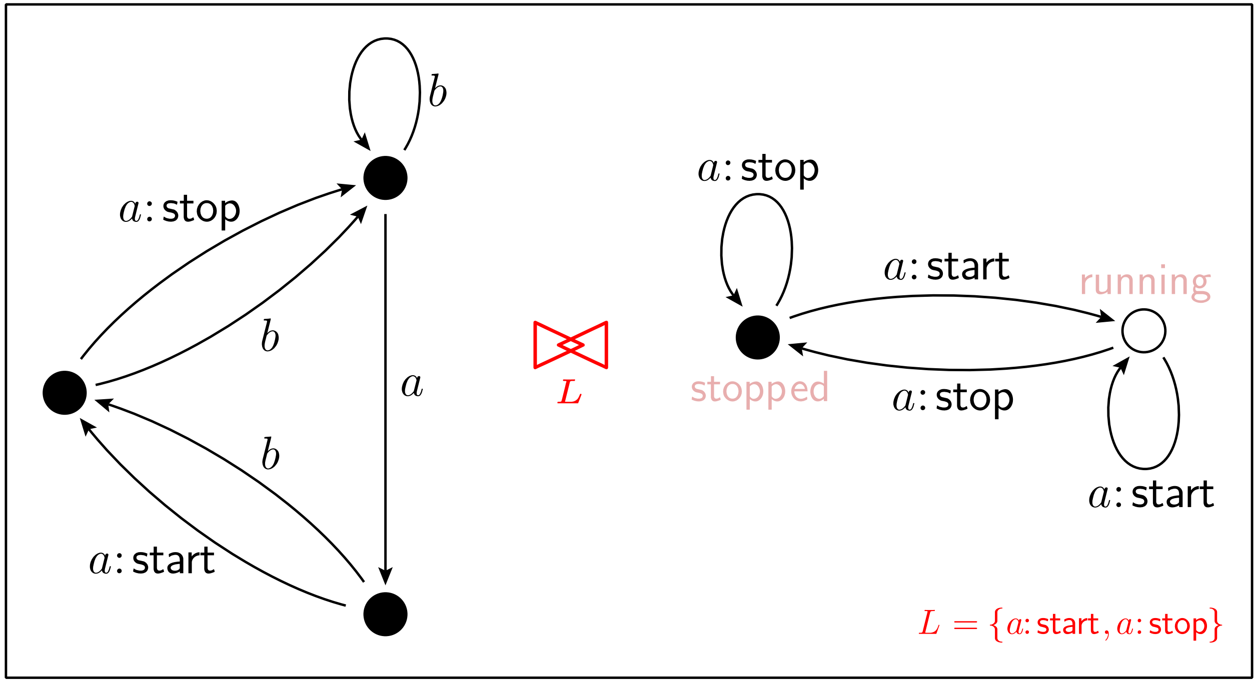
Probe Example

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



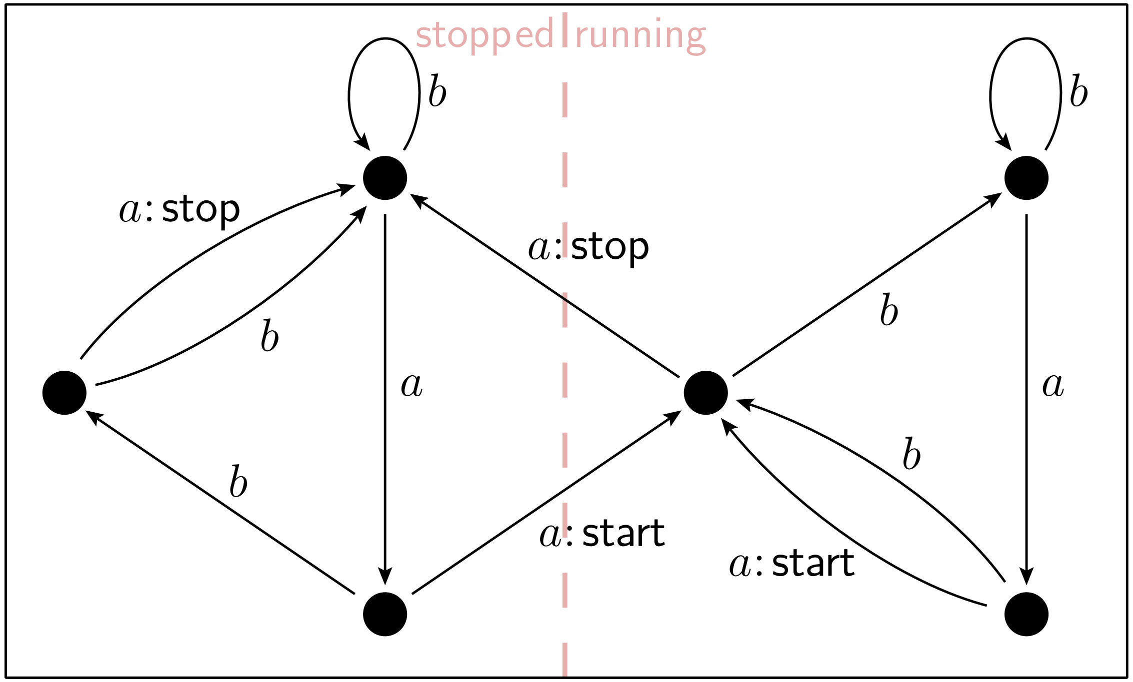
Probe Example

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



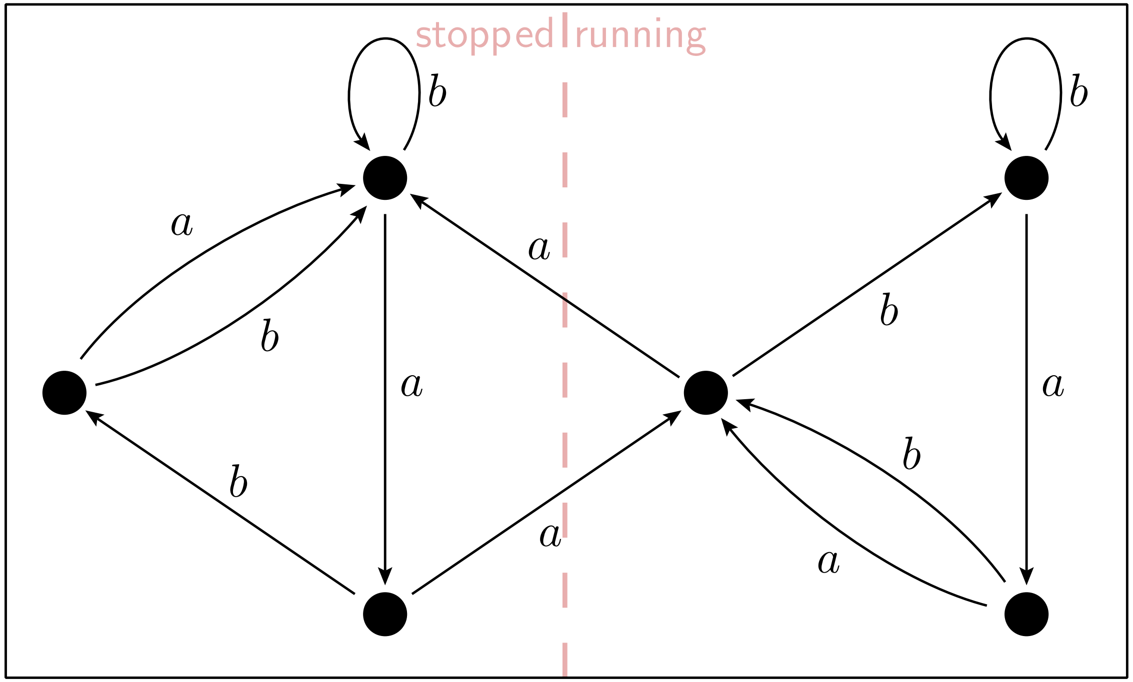
Probe Example

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

$$
\begin{array}{rcl}
\text{Server} & \stackrel{\mathit{def}}{=} & (\text{s\_read\_request}, \top).\text{Server\_read} \\
 & + & (\text{s\_fail}, r_{s_f}).\text{Server\_fail} \\
 & + & (\text{s\_write}, r_{s_w}).\text{Server} \\
\text{Server\_read} & \stackrel{\mathit{def}}{=} & (\text{s\_read\_lookup}, r_{s_{rl}}).\text{Server} \\
\text{Server\_fail} & \stackrel{\mathit{def}}{=} & (\text{s\_fail\_recover}, r_{s_{fr}}).\text{Server} \\
 & + & (\text{s\_fail\_recover\_all}, \top).\text{Server} \\
 & + & (\text{s\_write}, r_{s_{fsw}}).\text{Server\_fail}
\end{array}
$$

Server group definition

$$
\begin{aligned}
\mathsf{Server\_group}_0 &\stackrel{\mathit{def}}{=} (\mathsf{s\_fail}, \top).\mathsf{Server\_group}_1 \\
\mathsf{Server\_group}_i &\stackrel{\mathit{def}}{=} (\mathsf{s\_fail}, \top).\mathsf{Server\_group}_{i+1} \\
&\quad + (\mathsf{s\_fail\_recover}, \top) \\
&\quad\quad .\mathsf{Server\_group}_{i-1} : 1 \leq i < S \\
\mathsf{Server\_group}_S &\stackrel{\mathit{def}}{=} (\mathsf{s\_fail\_recover\_all}, r_{s_{gsfra}}) \\
&\quad\quad .\mathsf{Server\_group}_0
\end{aligned}
$$

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

$$
\begin{aligned}
\mathsf{Write\_buffer}_0 &\stackrel{\mathit{def}}{=} (\mathsf{b\_write}, \top).\mathsf{Write\_buffer}_1 \\
\mathsf{Write\_buffer}_{\mathsf{i}} &\stackrel{\mathit{def}}{=} (\mathsf{b\_write}, \top).\mathsf{Write\_buffer}_{\mathsf{i}+1} \\
&\quad + \ (\mathsf{s\_write}, \top).\mathsf{Write\_buffer}_0 \\
&\qquad\qquad : 1 \leq i < B \\
\mathsf{Write\_buffer}_{\mathsf{B}} &\stackrel{\mathit{def}}{=} (\mathsf{s\_write}, \top).\mathsf{Write\_buffer}_0
\end{aligned}
$$

Writer definition

$$\begin{aligned} \mathsf{Writer} &\stackrel{\mathit{def}}{=} (\mathsf{b\_write}, r_{w_{bw}}).\mathsf{Writer\_writ} \\ \mathsf{Writer\_writ} &\stackrel{\mathit{def}}{=} (\mathsf{w\_reset}, r_{w_r}).\mathsf{Writer} \end{aligned}$$

and an arbitrary group writers:

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

Reader defintion

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

and an arbitrary number or readers:

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

Overall System

$$
\begin{aligned}
\mathsf{Environment} &\stackrel{\mathit{def}}{=} \mathsf{Writers} \underset{\emptyset}{\bowtie} \mathsf{Readers} \\
\mathsf{Web\_cluster} &\stackrel{\mathit{def}}{=} \mathsf{Servers} \underset{\{\mathsf{s\_write}\}}{\bowtie} \mathsf{Write\_buffer}_0 \\
\mathsf{System} &\stackrel{\mathit{def}}{=} \mathsf{Environment} \underset{\mathcal{N}}{\bowtie} \mathsf{Web\_cluster}
\end{aligned}
$$

where $\mathcal{N} = \{\mathsf{b\_write}, \mathsf{s\_read\_request}, \mathsf{s\_read\_lookup}\}$.

Probe Query A

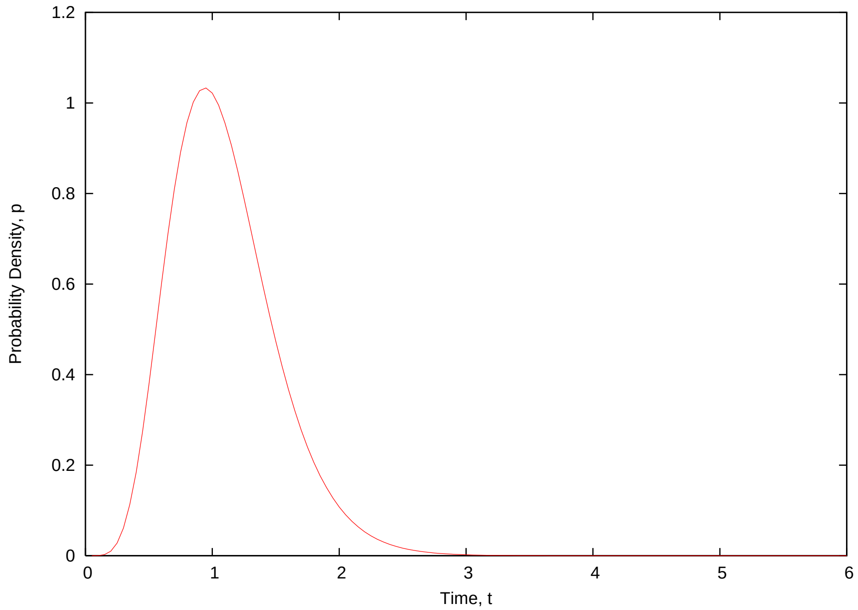
Probe A: measures the length of time to process at least 4 read events, then reset:

$$\Pi_A = \text{s\_read\_request}:\mathsf{start}, (\text{s\_read\_lookup}\{4\}), \text{r\_reset}:\mathsf{stop}$$

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

Probe Query A

Probability density function for probe A:



Probe Query B

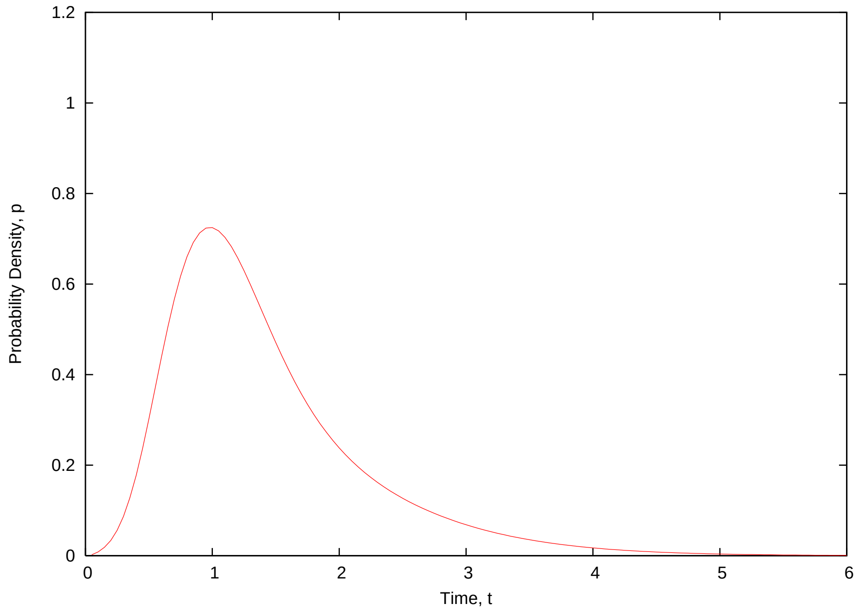
Probe B: observes the length of time it takes to observe either 4 completed reads or 2 completed writes, then reset:

$$\Pi_B \;=\; (\mathsf{s\_read\_request}{:}\,\mathsf{start}, (\mathsf{s\_read\_lookup}\{4\}), \mathsf{r\_reset}{:}\,\mathsf{stop}) \\ \mid (\mathsf{b\_write}{:}\,\mathsf{start}, \mathsf{b\_write}\{2\}, \mathsf{w\_reset}{:}\,\mathsf{stop})$$

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

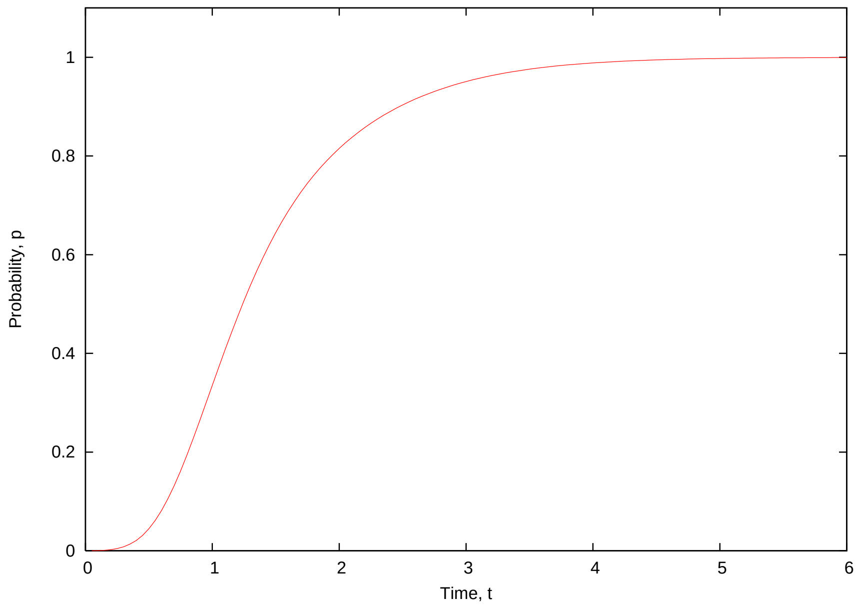
Probe Query B

Probability density function for probe B:



Probe Query B

Cumulative distribution function for probe B:



Probe Queries

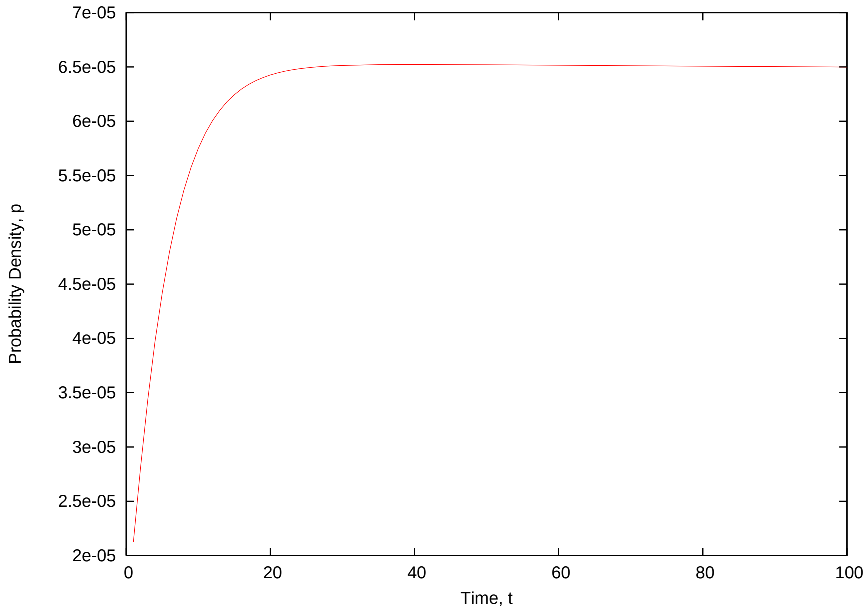
Probe C: extracts the passage-time to first full system failure and recovery:

$$\Pi_C = (\text{s\_read\_request}{:}\,\mathsf{start} \mid \text{b\_write}{:}\,\mathsf{start}), \text{s\_fail\_recover\_all}{:}\,\mathsf{stop}$$

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

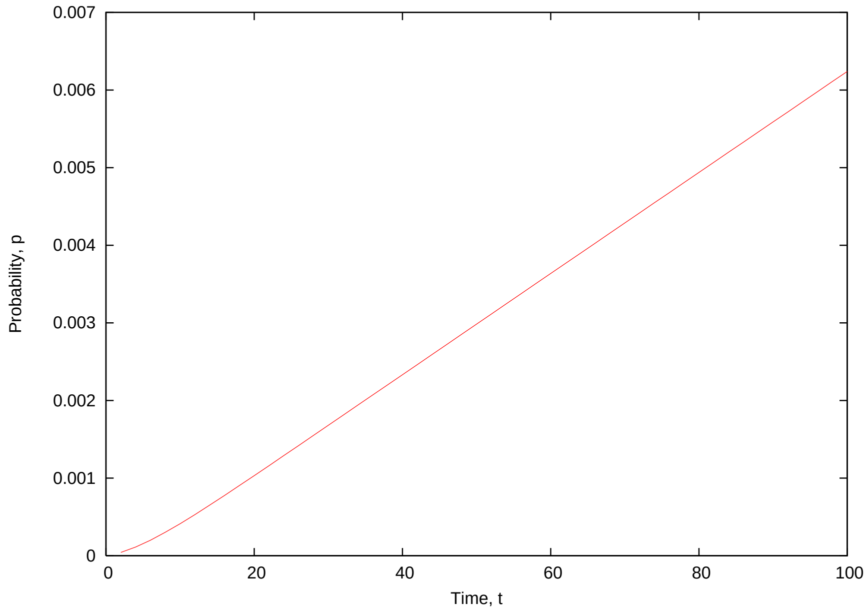
Probe Query C

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?