CyPhyAssure Spring School

Second Generation Model-based Testing

Provably Strong Testing Methods for the Certification of Autonomous Systems
Part III of III –
Complete Test Suites for CSP Refinement

Jan Peleska
University of Bremen and Verified Systems International GmbH
peleska@uni-bremen.de
2019-03-21

Universität Bremen

Verified

Finite Complete Test Suites for CSP Refinement Testing*

Jan Peleska, Wen-ling Huang, and Ana Cavalcanti
{peleska,huang}@uni-bremen.de
ana.cavalcanti@york.ac.uk

*The results presented here have been submitted to Science of Computer Programming

Completeness Result

Theorem. Let P be a non-terminating, divergence-free CSP process over alphabet Σ whose normalised transition graph $G(P)$ has p states. Define fault domain $\mathcal{D}$ as the set of all divergence-free CSP processes over alphabet Σ, whose transition graph has at most q states with $q \geq p$. Then the test suite

$$TS_F = \{U_F(j) \mid 0 \leq j < pq\}$$

is complete with respect to $\mathcal{F} = (P, \sqsubseteq_F, \mathcal{D})$.

Analogous theorem holds for trace refinement

Recall

- **Complete test suites**
  - are specified for a given conformance relation
  - accept every conforming implementation
  - reject every non-conforming implementation

Recall

- **Fault domain.** A collection of models that may or may not conform to a reference model

- **Finite complete black-box test suites**
  - are specified with respect to a fault domain
  - guarantee completeness provided that the true SUT behaviour is reflected by a member of the fault domain
  - provide a **conformance proof** with finitely many finite test cases

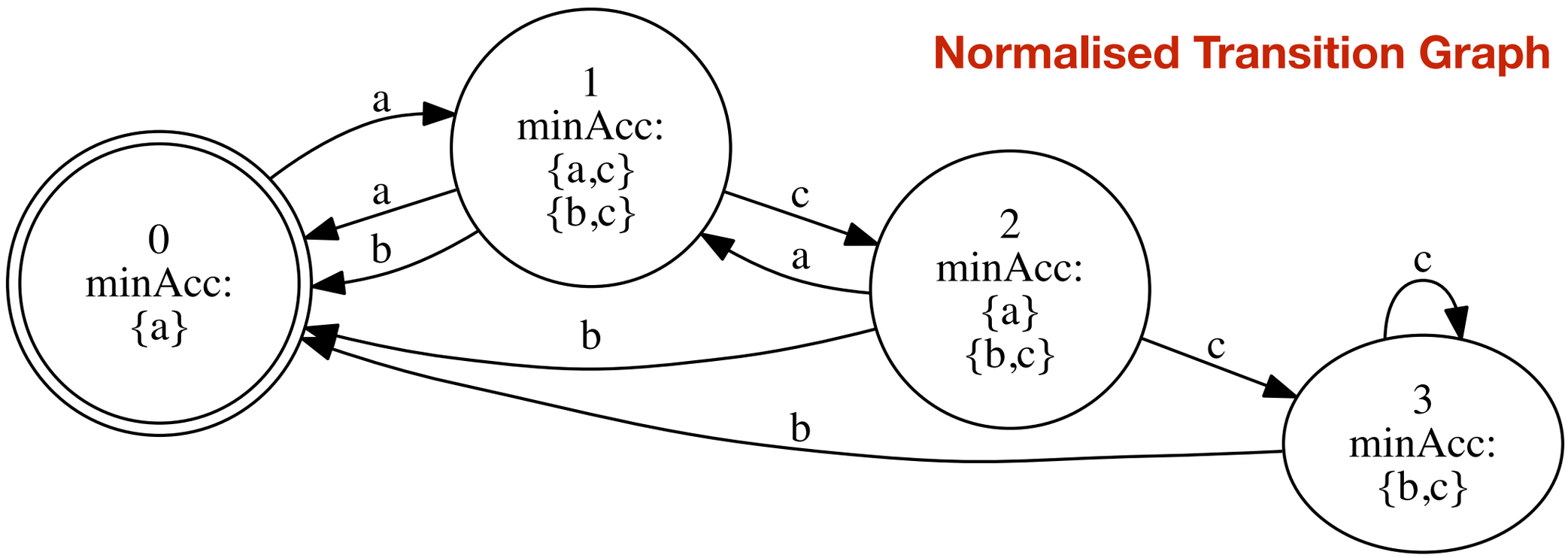
Motivation

- Finite complete test suites are of high interest, because they
  - can **establish full conformance**, provided that the SUT behaviour is captured by the fault domain
  - still possess **high test strength for SUTs outside the fault domain** (experimental evidence)
  - still have manageable size if **equivalence class partitioning methods** are applied

Failures Semantics – Representation of Finite-State Processes

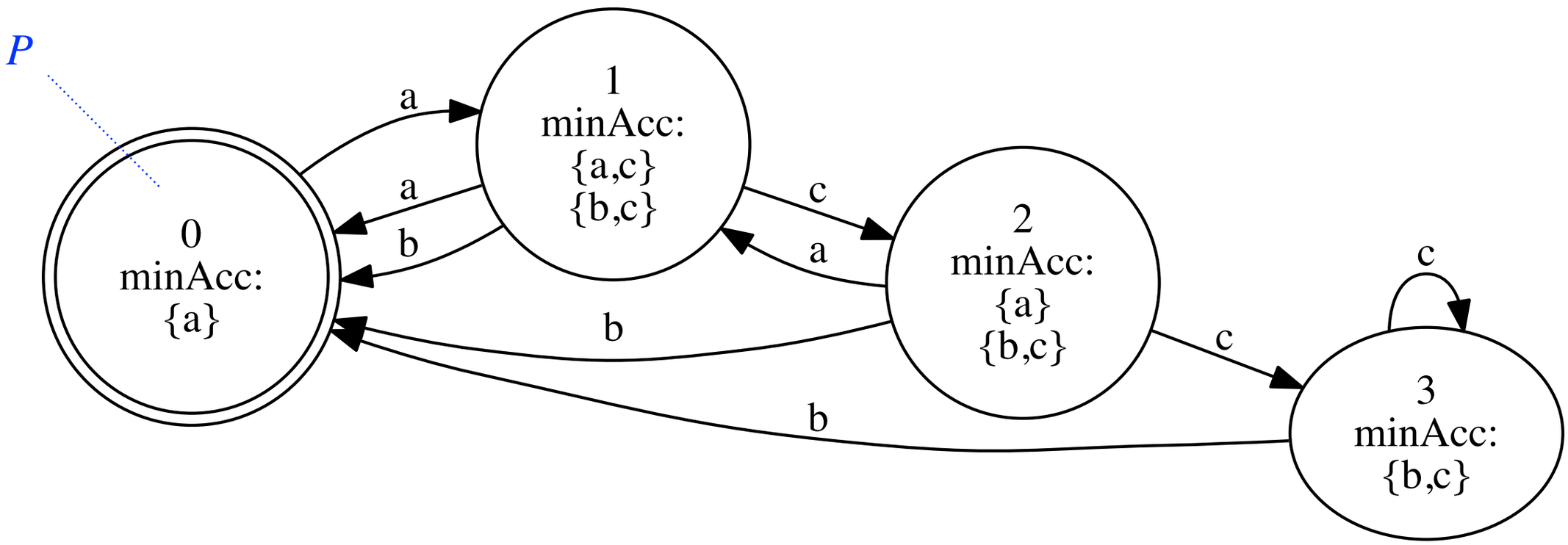
$$
\begin{aligned}
P &= a \to (Q \sqcap R) \\
Q &= a \to P \;\Box\; c \to P \\
R &= b \to P \;\Box\; c \to R
\end{aligned}
$$

A. W. Roscoe, Model-checking CSP, in: A. W. Roscoe (Ed.), A Classical Mind: Essays in Honour of C. A. R. Hoare, Prentice Hall International (UK) Ltd., Hertfordshire, UK, UK, 1994, Ch. 21, pp. 353–378.



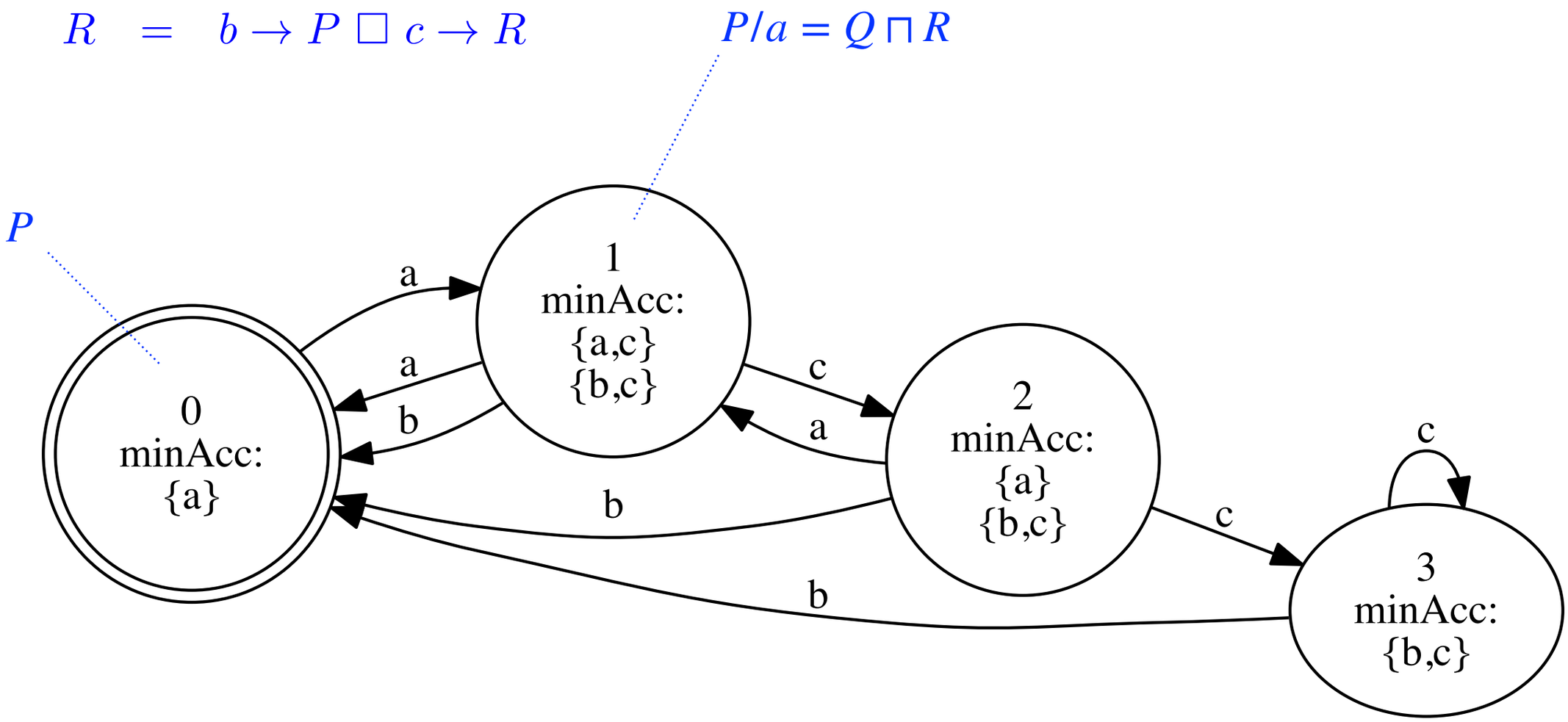
Failures Semantics – Representation of Finite-State Processes

$$
\begin{aligned}
P &= a \to (Q \sqcap R) \\
Q &= a \to P \,\Box\, c \to P \\
R &= b \to P \,\Box\, c \to R
\end{aligned}
$$



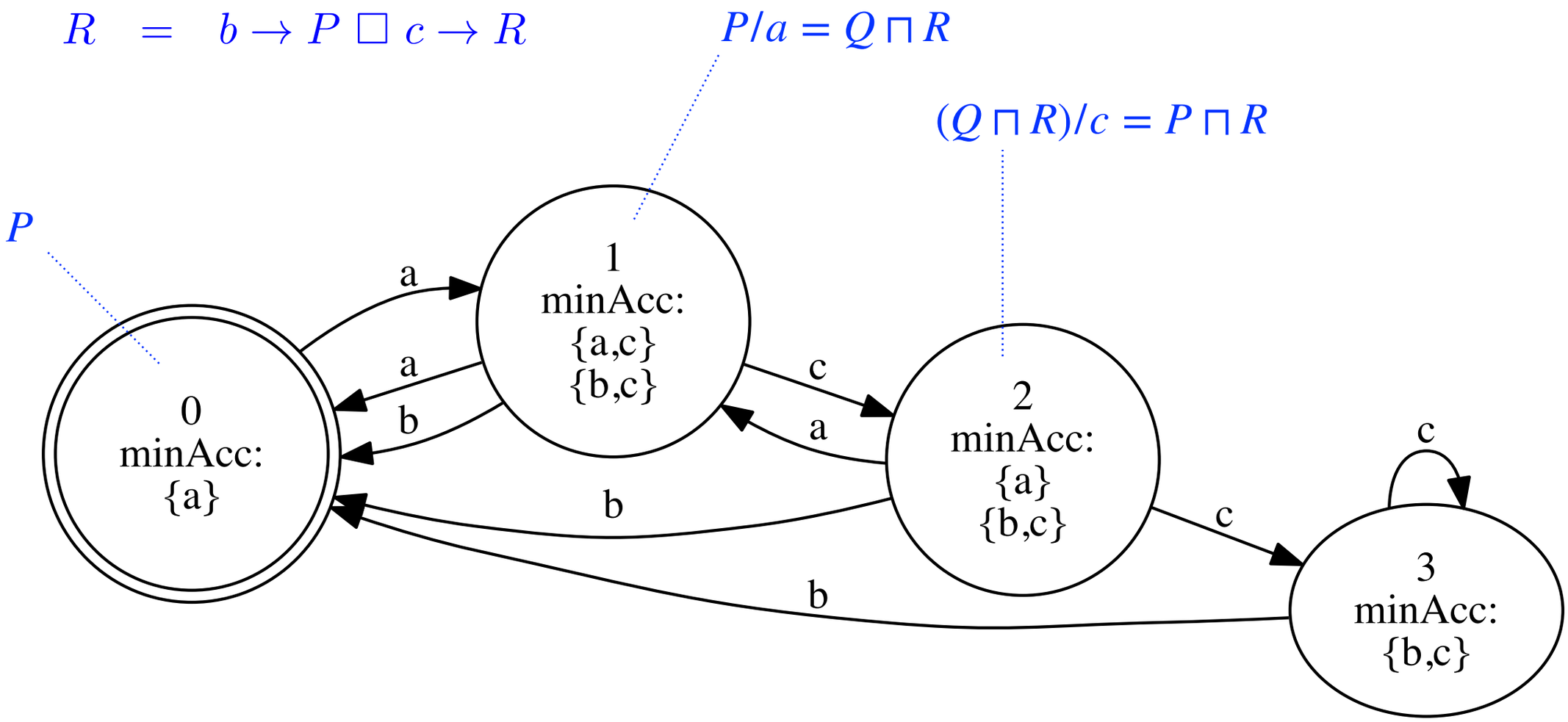
Failures Semantics – Representation of Finite-State Processes

$$
\begin{aligned}
P &= a \rightarrow (Q \sqcap R) \\
Q &= a \rightarrow P \;\Box\; c \rightarrow P \\
R &= b \rightarrow P \;\Box\; c \rightarrow R
\end{aligned}
$$



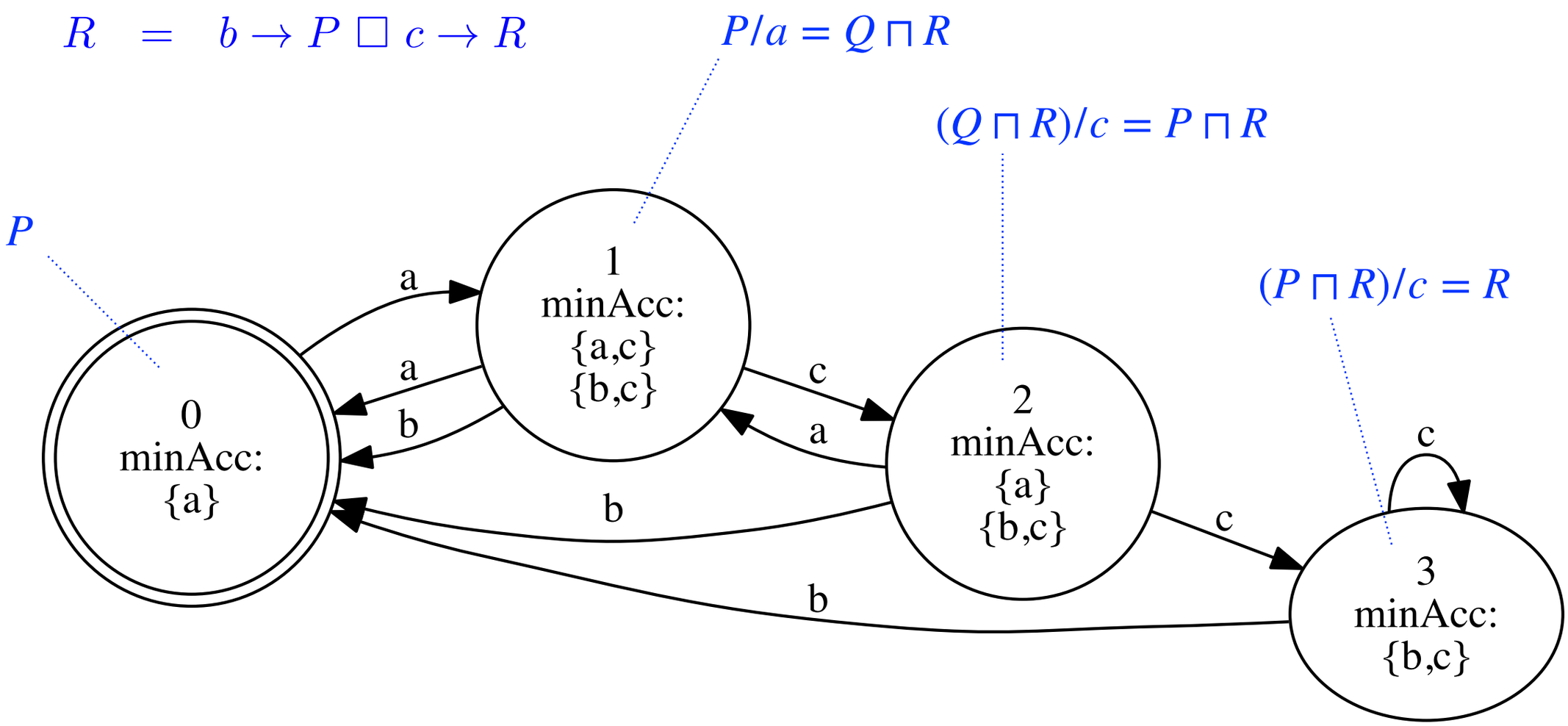
Failures Semantics – Representation of Finite-State Processes

$$
\begin{aligned}
P &= a \to (Q \sqcap R) \\
Q &= a \to P \;\Box\; c \to P \\
R &= b \to P \;\Box\; c \to R
\end{aligned}
$$



Failures Semantics – Representation of Finite-State Processes

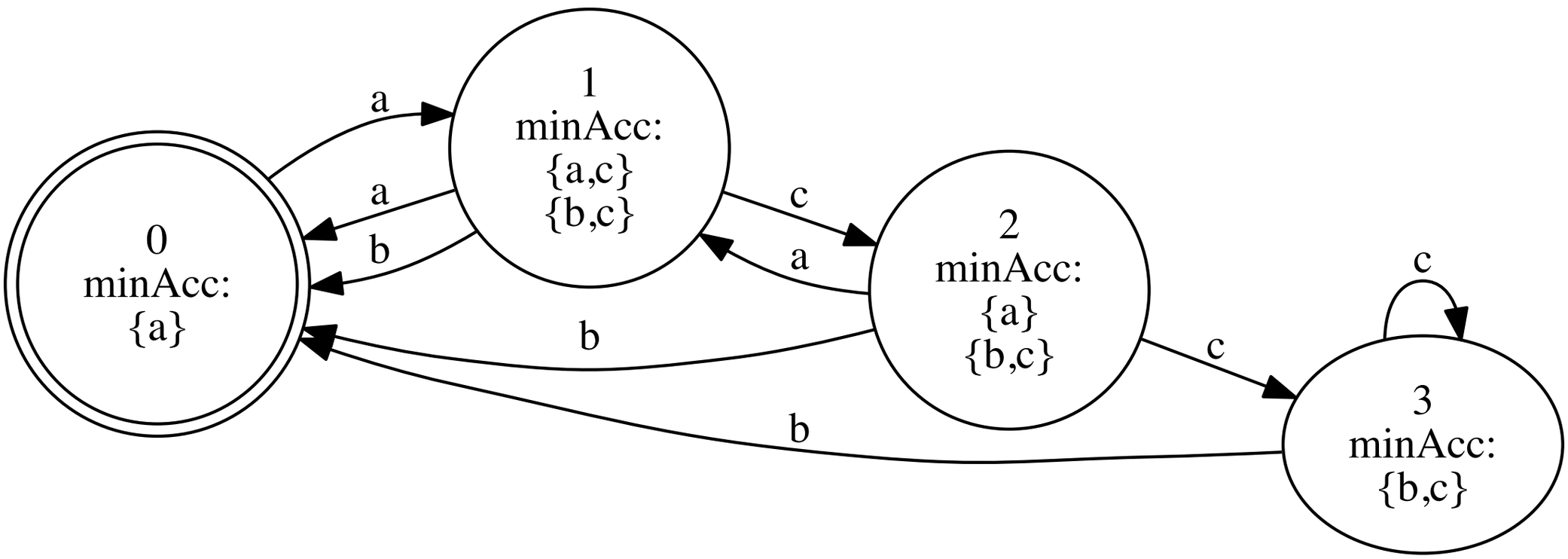
$$
\begin{aligned}
P &= a \to (Q \sqcap R) \\
Q &= a \to P \;\Box\; c \to P \\
R &= b \to P \;\Box\; c \to R
\end{aligned}
$$



Failures Semantics – Representation of Finite-State Processes

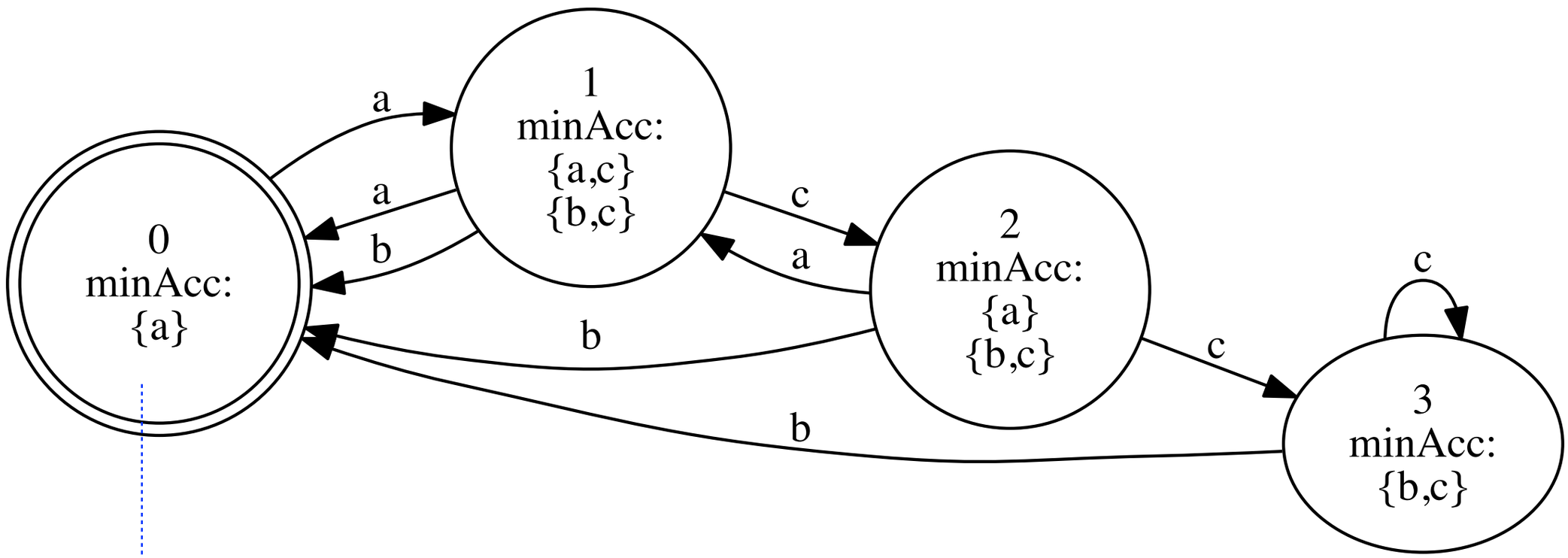
$$
\begin{aligned}
P &= a \rightarrow (Q \sqcap R) \\
Q &= a \rightarrow P \,\Box\, c \rightarrow P \\
R &= b \rightarrow P \,\Box\, c \rightarrow R
\end{aligned}
$$

Assume that implementation process Z has transition graph with 4 states – just like P



Failures Semantics – Representation of Finite-State Processes

$$
\begin{array}{lcl}
P & = & a \to (Q \sqcap R) \\
Q & = & a \to P \;\Box\; c \to P \\
R & = & b \to P \;\Box\; c \to R
\end{array}
$$

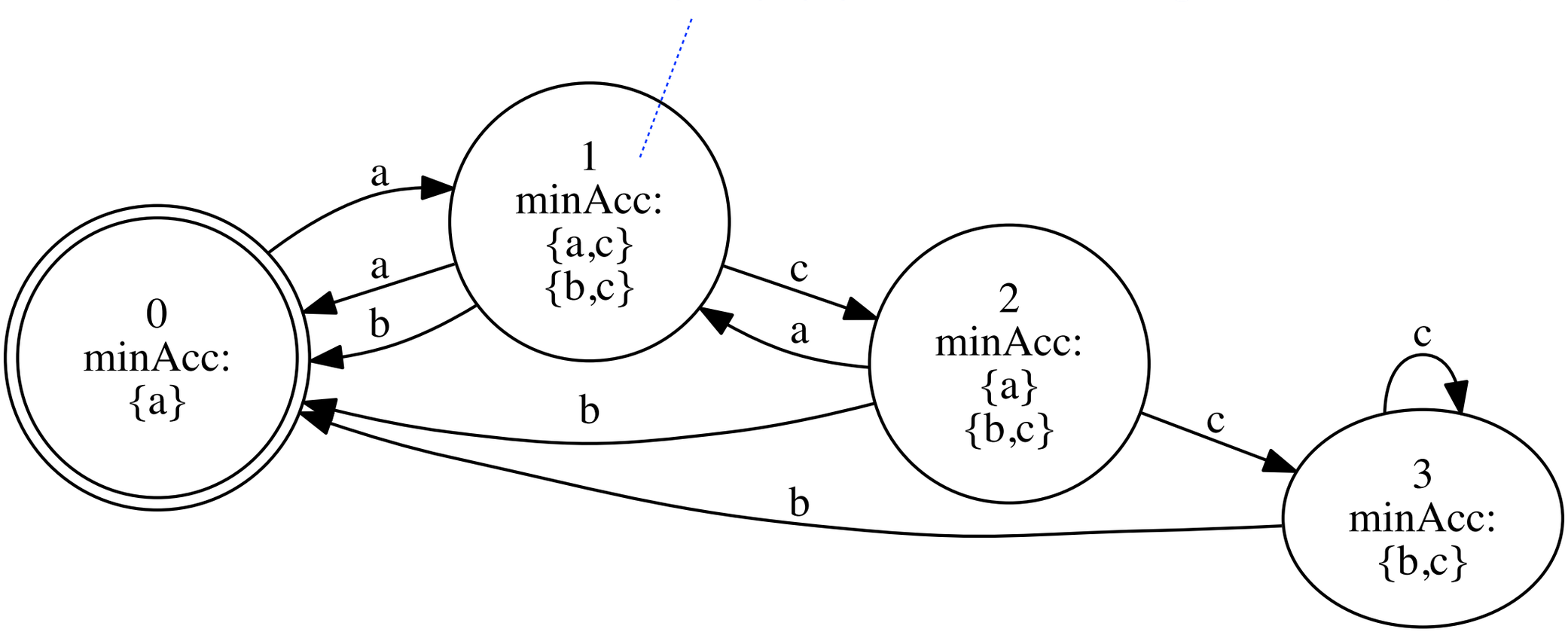


$\{a\}$ **must be a hitting set of** $acc(Z)$

Failures Semantics – Representation of Finite-State Processes

$$
\begin{aligned}
P &= a \to (Q \sqcap R) \\
Q &= a \to P \;\Box\; c \to P \\
R &= b \to P \;\Box\; c \to R
\end{aligned}
$$

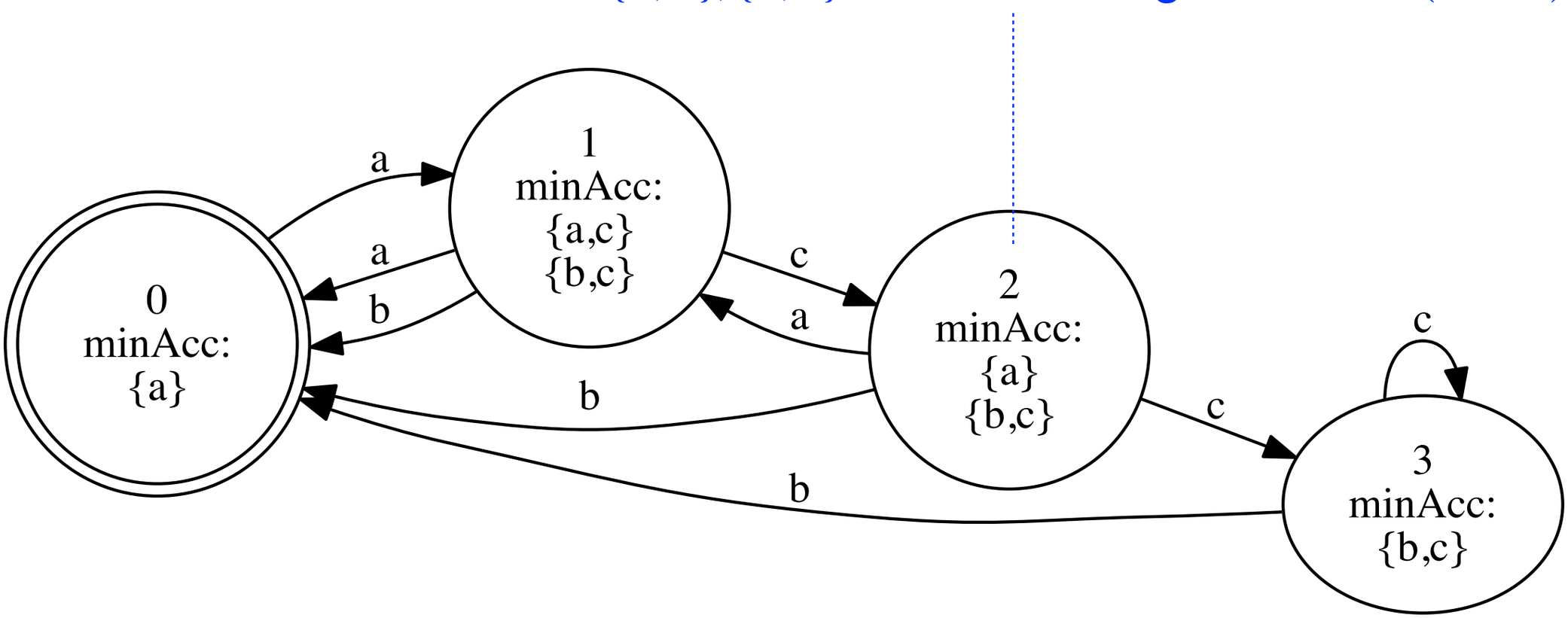
$\{a, b\}, \{c\}$ **must be hitting sets of** $acc(Z/a)$



Failures Semantics – Representation of Finite-State Processes

$$
\begin{array}{lcl}
P & = & a \to (Q \sqcap R) \\
Q & = & a \to P \;\Box\; c \to P \\
R & = & b \to P \;\Box\; c \to R
\end{array}
$$

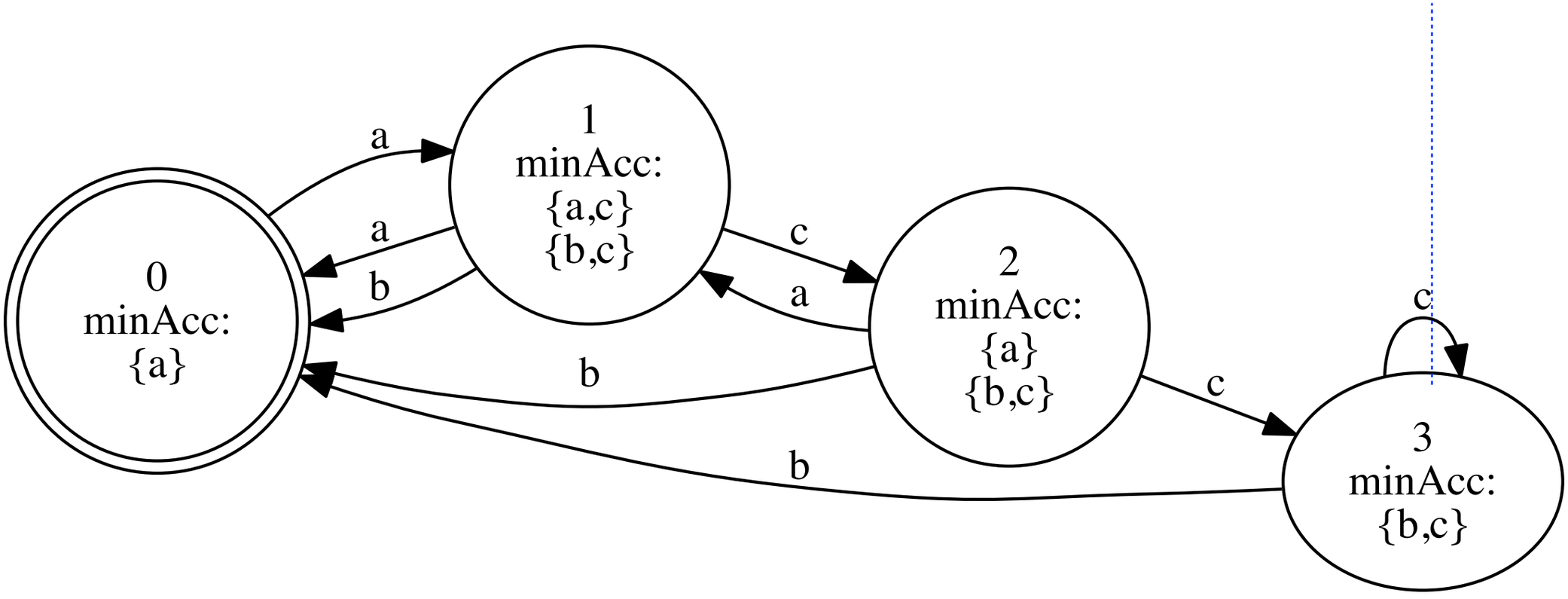
$\{a,b\},\{a,c\}$ **must be hitting sets of** $acc(Z/a\,.\,c)$



Failures Semantics – Representation of Finite-State Processes

$$
\begin{aligned}
P &= a \to (Q \sqcap R) \\
Q &= a \to P \,\Box\, c \to P \\
R &= b \to P \,\Box\, c \to R
\end{aligned}
$$

$\{b\}, \{c\}$ **must be hitting sets of** $acc(Z/a\,.\,c\,.\,c)$



Adaptive Test Cases for Checking Failures Refinement

$$
\begin{aligned}
U_F(j) &= U_F(j, 0, \underline{n}) \\
U_F(j,k,n) &= \big(e : (\Sigma - [n]^0) \to fail \to \mathbf{STOP}\big) \\
&\quad \Box \\
&\quad (\mathrm{minHit}(n) = \varnothing) \& \big(pass \to \mathbf{STOP}\big) \\
&\quad \Box \\
&\quad (k < j) \& \big(e : [n]^0 \to U_F(j, k+1, t(n,e))\big) \\
&\quad \Box \\
&\quad (k = j \wedge \mathrm{minHit}(n) \neq \varnothing) \& \big(\sqcap_{H \in \mathrm{minHit}(n)} \, (e : H \to pass \to \mathbf{STOP})\big)
\end{aligned}
$$

Check all traces up to length *j+1*

$$
\begin{aligned}
U_F(j) &= U_F(j, 0, \underline{n}) \\
U_F(j,k,n) &= \big(e : (\Sigma - [n]^0) \to fail \to \mathbf{STOP}\big) \\
&\quad \Box \\
&\quad (\mathrm{minHit}(n) = \varnothing) \& \big(pass \to \mathbf{STOP}\big) \\
&\quad \Box \\
&\quad (k < j) \& \big(e : [n]^0 \to U_F(j, k+1, t(n,e))\big) \\
&\quad \Box \\
&\quad (k = j \wedge \mathrm{minHit}(n) \neq \varnothing) \& \big(\sqcap_{H \in \mathrm{minHit}(n)} \, (e : H \to pass \to \mathbf{STOP})\big)
\end{aligned}
$$

When residing in node *n* of *P*'s transition graph, …

$$
\begin{aligned}
U_F(j) &= U_F(j,0,\underline{n}) \\
U_F(j,k,n) &= \big(e:(\Sigma-[n]^0)\to fail\to \mathbf{STOP}\big) \\
&\quad \Box \\
&\quad (\mathrm{minHit}(n)=\varnothing)\&\big(pass\to \mathbf{STOP}\big) \\
&\quad \Box \\
&\quad (k<j)\&\big(e:[n]^0\to U_F(j,k+1,t(n,e))\big) \\
&\quad \Box \\
&\quad (k=j\wedge \mathrm{minHit}(n)\neq\varnothing)\&\big(\sqcap_{H\in\mathrm{minHit}(n)}\,(e:H\to pass\to \mathbf{STOP})\big)
\end{aligned}
$$

… offer any illegal event to *Q* which should not be accepted

Initials of node *n*

$$
\begin{aligned}
U_F(j) &= U_F(j,0,\underline{n}) \\
U_F(j,k,n) &= \big(e:(\Sigma-[n]^0)\to fail\to \mathbf{STOP}\big) \\
&\quad \Box \\
&\quad (\mathrm{minHit}(n)=\varnothing)\&\big(pass\to \mathbf{STOP}\big) \\
&\quad \Box \\
&\quad (k<j)\&\big(e:[n]^0\to U_F(j,k+1,t(n,e))\big) \\
&\quad \Box \\
&\quad (k=j\wedge \mathrm{minHit}(n)\neq\varnothing)\&\big(\sqcap_{H\in\mathrm{minHit}(n)}\,(e:H\to pass\to \mathbf{STOP})\big)
\end{aligned}
$$

… allow to stop with verdict PASS if *P* allows to refuse everything at node *n*

$$
\begin{aligned}
U_F(j) &= U_F(j,0,\underline{n}) \\
U_F(j,k,n) &= \big(e : (\Sigma - [n]^0) \to fail \to \mathbf{STOP}\big) \\
&\quad \Box \\
&\quad (\mathrm{minHit}(n) = \varnothing)\&\big(pass \to \mathbf{STOP}\big) \\
&\quad \Box \\
&\quad (k < j)\&\big(e : [n]^0 \to U_F(j, k+1, t(n,e))\big) \\
&\quad \Box \\
&\quad (k = j \wedge \mathrm{minHit}(n) \neq \varnothing)\&\big(\sqcap_{H \in \mathrm{minHit}(n)} (e : H \to pass \to \mathbf{STOP})\big)
\end{aligned}
$$

… continue with any event which is admissible according to *P*'s initials, as long as the trace is still shorter than *j*. Continue with the successor node of *n* according to *P*'s transition function *t* ***(Back-to-Back Test Q against P)***

$$
\begin{aligned}
U_F(j) &= U_F(j, 0, \underline{n}) \\
U_F(j,k,n) &= \big(e : (\Sigma - [n]^0) \to fail \to \mathbf{STOP}\big) \\
&\quad \Box \\
&\quad (\mathrm{minHit}(n) = \varnothing)\&\big(pass \to \mathbf{STOP}\big) \\
&\quad \Box \\
&\quad (k < j)\&\big(e : [n]^0 \to U_F(j, k+1, t(n,e))\big) \\
&\quad \Box \\
&\quad (k = j \wedge \mathrm{minHit}(n) \neq \varnothing)\&\big(\sqcap_{H \in \mathrm{minHit}(n)} (e : H \to pass \to \mathbf{STOP})\big)
\end{aligned}
$$

… and check whether Q accepts an event from every minimal hitting set of node *n* without blocking, if the length of the trace is *j*

$$
\begin{aligned}
U_F(j) &= U_F(j,0,\underline{n}) \\
U_F(j,k,n) &= \big(e : (\Sigma - [n]^0) \to fail \to \mathbf{STOP}\big) \\
&\quad \Box \\
&\quad (\mathrm{minHit}(n) = \varnothing)\&\big(pass \to \mathbf{STOP}\big) \\
&\quad \Box \\
&\quad (k < j)\&\big(e : [n]^0 \to U_F(j, k+1, t(n,e))\big) \\
&\quad \Box \\
&\quad (k = j \wedge \mathrm{minHit}(n) \neq \varnothing)\&\big(\sqcap_{H \in \mathrm{minHit}(n)} (e : H \to pass \to \mathbf{STOP})\big)
\end{aligned}
$$

PASS Criterion

- *Q* passes test $U_F(j)$, if and only if for every possible execution of
- no FAIL event is ever produced,
- the test always terminates with PASS

$$Q \underline{pass}\ U_F(j) \equiv (pass \rightarrow \mathbf{STOP}) \sqsubseteq_F (Q \parallel_\Sigma U_F(j)) \backslash \Sigma$$

Complete Testing Assumption

- There exists a constant c ≥ 1, such that every possible behaviour of the SUT, when running in parallel to test case *U,* is exhibited within *c* executions of *U*

Completeness Result

Theorem. Let P be a non-terminating, divergence-free CSP process over alphabet Σ whose normalised transition graph $G(P)$ has p states. Define fault domain $\mathcal{D}$ as the set of all divergence-free CSP processes over alphabet Σ, whose transition graph has at most q states with $q \geq p$. Then the test suite

$$TS_F = \{U_F(j) \mid 0 \leq j < pq\}$$

is complete with respect to $\mathcal{F} = (P, \sqsubseteq_F, \mathcal{D})$.

Complexity Considerations

Maximal Number of Test Executions Required

Theorem. The maximal number of test executions to be performed using the complete test suite $TS_F = \{U_F(j) \mid 0 \leq j < pq\}$ created from P is of order

$$O(\binom{n}{\lfloor \frac{n}{2} \rfloor} \cdot n^{pq-1}) \quad \text{with } n = |\Sigma|.$$

For processes P satisfying $(s, \Sigma) \notin \text{failures}(P)$ for all traces s, the reachable precise upper bound is given by

$$\binom{n}{\lfloor \frac{n}{2} \rfloor} \cdot \frac{1 - n^{pq}}{1 - n} \quad \text{with } n = |\Sigma|.$$

Maximal Number of Test Executions Required

Theorem. The maximal number of test executions to be performed using the complete test suite $TS_F = \{U_F(j) \mid 0 \leq j < pq\}$ created from P is of order

$$O(\binom{n}{\lfloor \frac{n}{2} \rfloor} \cdot n^{pq-1}) \quad \text{with } n = |\Sigma|.$$

For processes P satisfying $(s, \Sigma) \notin \text{failures}(P)$ for all traces s, the reachable precise upper bound is given by

$$\binom{n}{\lfloor \frac{n}{2} \rfloor} \cdot \frac{1 - n^{pq}}{1 - n} \quad \text{with } n = |\Sigma|.$$

Maximal number of hitting sets to be tested at the end of each test execution

Why we Cannot use Shorter Traces

Theorem. Let $2 \leq p, q \in \mathbb{N}$. Then there exists a reference process P and an implementation process Q with the following properties.

1. $G(P)$ has p states.
2. $G(Q)$ has q states.
3. $P \not\sqsubseteq_T Q$, and therefore, also $P \not\sqsubseteq_F Q$.
4. $\forall s \in \text{traces}(Q) : \#s < pq \implies s \in \text{traces}(P)$.
5. $Q \; conf \; P$.

Discussion

Discussion

- **Could the test effort be further reduced?**
  - We know that maximal length of traces and number of probes *H* cannot be reduced without losing completeness
  - Translate results about adaptive state counting methods from FSMs to CSP

R. M. Hierons, Testing from a nondeterministic finite state machine using adaptive state counting, IEEE Trans. Computers 53 (10) (2004) 1330–1342. doi:10.1109/TC.2004.85

Discussion

- **Is it such a good idea to check for *refinement*?**
  - From complete methods for FSMs we know that **complete equivalence checking** can be performed with much shorter traces**: maximal length *p+q–1*** (instead of *pq - 1* as required for refinement checking)
  - Alternative approach to be preferred:
    - refine original model and check correctness by model checker FDR4
    - Stop refinement as soon as implementation can be required to be equivalent to last refinement
    - Then **test for failures equivalence**

Acknowledgements

The authors would like to thank Bill Roscoe and Thomas Gibson-Robinson for their advice on using the FDR4 model checker and for very helpful discussions concerning the potential implications of this paper in the field of model checking. We are also grateful to Li-Da Tong from National Sun Yat-sen University, Taiwan, for suggesting the applicability of Sperner's Theorem in the context of the work presented here. Moreover, we thank Adenilso Simao for several helpful suggestions. The work of Ana Cavalcanti is funded by the Royal Academy of Engineering and UK EPSRC Grant EP/R025134/1.


PUSH
ARRET D'URGENCE
EMERGENCY STOP