Model-based Testing for Safety-Critical Systems

Jan Peleska
University of Bremen
jp@informatik.uni-bremen.de
FoMSESS 2015

With contributions by
Anne E. Haxthausen, Wen-ling Huang, Felix Hübner and Linh Hong Vu

Overview

- Model-based testing
- Standards for safety-critical transportation systems
- What standards require about model-based testing
- Complete test strategies …
- … and their relevance for model-based testing of safety-critical systems
- Discussion

Overview

- **Model-based testing**
- Standards for safety-critical transportation systems
- What standards require about model-based testing
- Complete test strategies …
- … and their relevance for model-based testing of safety-critical systems
- Discussion

Model-based Testing

Instead of writing test procedures,

- develop a **test model** specifying expected behaviour of SUT (system under test)
- use **generator** to identify “relevant” test cases from the model and calculate concrete test data
- generate **test procedures** fully automatic
- perform **tracing** requirements ↔ test cases in a fully automatic way

Overview

- Model-based testing
- **Standards for safety-critical transportation systems**
- What standards require about model-based testing
- Complete test strategies …
- … and their relevance for model-based testing of safety-critical systems
- Discussion

Standards for Safety-critical Transportation Systems

- Railway domain: **CENELEC EN 50128:2011**
  Railway applications - Communication, signalling and processing systems - Software for railway control and protection systems

- Avionic domain: **RTCA DO-178C**
  Software considerations in airborne systems and equipment certification

- Automotive domain: **ISO 26262**
  Road vehicles – Functional safety

Overview

- Model-based testing
- Standards for safety-critical transportation systems
- **What standards require about model-based testing**
- Complete test strategies …
- … and their relevance for model-based testing of safety-critical systems
- Discussion

What Standards Require About Model-based Testing

- All standards acknowledge the model-driven development and V&V approach
- Only ISO26262 explicitly uses the term *model-based testing*
- RTCA DO-178B and its supplements addresses the model-driven approach in the most comprehensive way

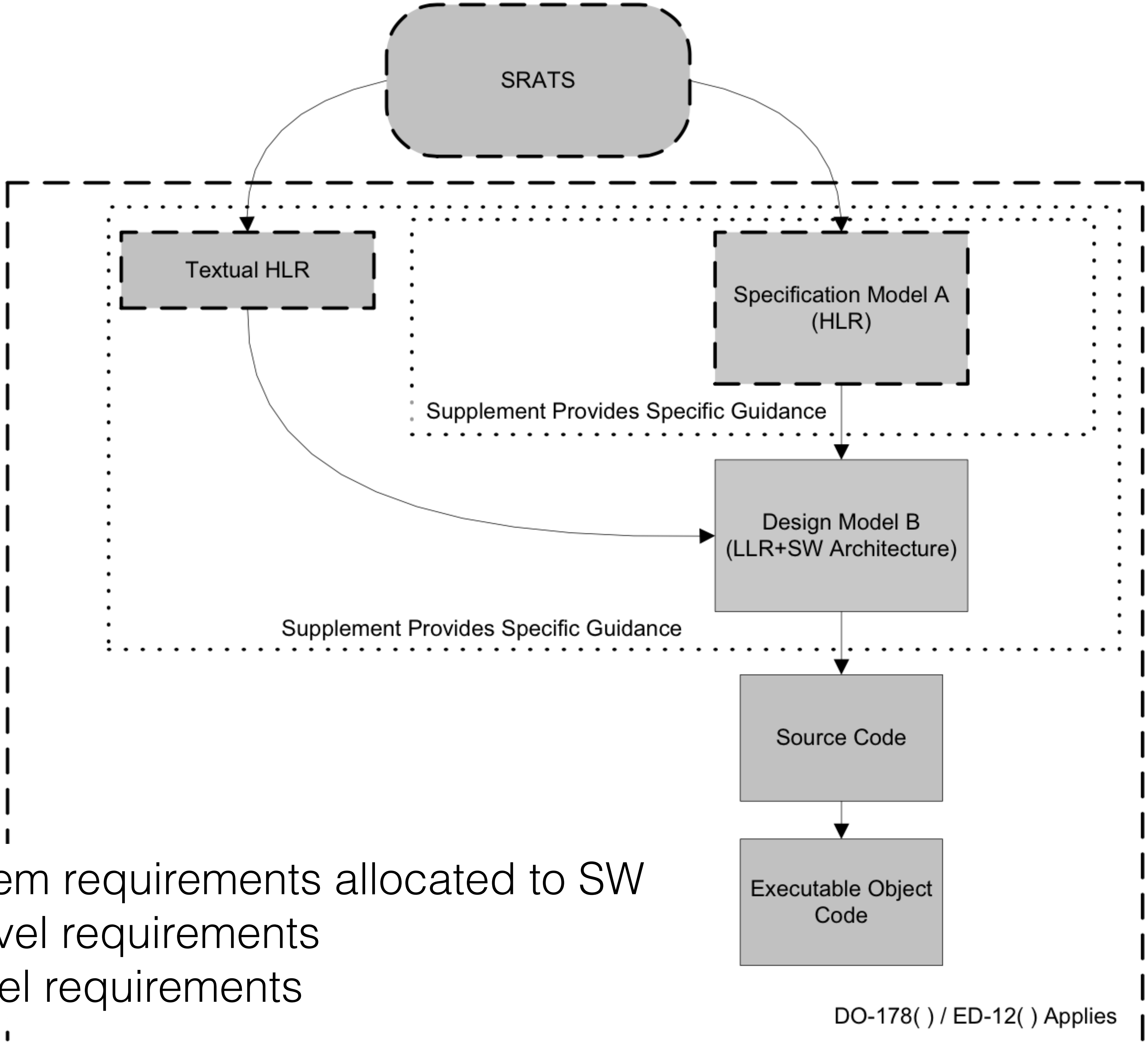
Automotive Domain

- ISO 26262 – Part 6: Product development: software level – Appendix B: Model-based development

  "[In model-based development,] Testing activities are also treated differently since models can be used as a useful source of information for the testing process (model-based testing)."

Avionic Domain

- The RTCA DO-178C standard is complemented by several supplements; relevant for this presentation are
  - **RTCA DO-331**. Model-Based Development and Verification Supplement to DO-178C and DO-278A
  - **RTCA DO-333**. Formal Methods Supplement to DO-178C and DO-278A



SRATS. System requirements allocated to SW
HLR. High-level requirements
LLR. Low-level requirements

Figure MB.DP1-4
Example C: Separate Models Used To Express HLR and LLR/SW Architecture

From RTCA DO-331

References of RTCA DO-331 to Model-based Testing

- The term “*model-base testing*” is never used, but (model-based) **simulation** is addressed [DO-331;MB.6.8]

For Design Models, simulation may be used in combination with testing and appropriate coverage analysis to satisfy objectives related to the verification of the Executable Object Code. As simulation may involve different object code and a different environment than the target application, simulation alone cannot be used to satisfy objectives related to verification of the Executable Object Code. … However, if simulation cases are run in the target computer environment using the Executable Object Code, then they are also considered test cases and DO-178C section 6.4 applies.

References of RTCA DO-331 to Model-based Testing

- (Design) models have to be verified with respect to the original (informal) requirements
- Model coverage analysis is performed to verify the completeness of the design model
- Model coverage analysis can also be used (as soon as the model has been verified) to check the completeness of testing activities

Model Coverage Criteria

- … according to RTCA DO-331
  - Requirements coverage
  - Transition coverage (of state machines)
  - Decision coverage (of guard conditions and decision tables etc.)
  - Coverage of all equivalence classes, boundary conditions and enumerable value ranges

Complete Test Strategies

- Recall: **complete = sound + exhaustive**
    - **sound =** every correct SUT (system under test) behaviour is accepted by each test suite generated according to the strategy
    - **exhaustive =** every erroneous SUT behaviour will be uncovered by at least one test case
- RTCA DO-331 acknowledges the existence of complete strategies (MB.12.3.1 Exhaustive Input Testing)

What is not Addressed by the Standards

- How are requirements reflected (traceable) in the models?
- Adequacy of test cases (suitability of strategies, coverage criteria) with respect to
    - **Selection of equivalence classes**
    - **Timing constraints**
    - **Concurrent behaviours**
- How can **security aspects** be modelled and verified?

Overview

- Model-based testing
- Standards for safety-critical transportation systems
- What standards require about model-based testing
- **Complete test strategies …**
- … and their relevance for model-based testing of safety-critical systems
- Discussion

Complete Test Strategies

- For black-box testing, completeness is specified with respect to a **fault model**
  - Reference model
  - Conformance relation
  - Fault domain

$$\mathcal{F} = (\mathcal{S}, \sim, \mathcal{D}(\mathcal{S}, m, \mathcal{I}_2))$$

Complete Test Strategies

- For black-box testing:
- **complete = sound + exhaustive**
  - **sound =** every correct SUT behaviour is accepted by each test suite generated according to the strategy
  - **exhaustive =** every erroneous SUT behaviour will be uncovered by at least one test case, **as long as the true SUT behaviour is reflected by a member of the fault domain**

Complete Test Strategies

- Why are black-box tests important for safety-critical systems ?
  - HW/SW integration testing and system integration testing **must be performed on the unaltered target system**
  - Typically, the target system does not provide sufficient monitoring means for white/grey-box testing
  - This is because standards do not allow for code to be present in the target, if it does not contribute to the specified functionality

Example

- Complete test strategy:
  - Novel equivalence class partition testing strategy
  - Applicable to
    - Nondeterministic models with Kripke Structure semantics
    - Infinite (or very large) input domains
    - Finite internal states
    - Finite control outputs

Example

- Typical applications:
  - Analogue input sensors and discrete control decisions
    - Airbag controller
    - Speed controllers (e.g. in train protection systems)
    - Route controller in interlocking systems

Example

- Typical applications:
  - Analogue input sensors and discrete control decisions
    - Airbag co
    - Spee … train protection system
    - **Route controller in interlocking systems**

Application example to be discussed below

Construction Principle for Equivalence Class Testing Strategies

- Associate test model with **Kripke Structure K** expressing behavioural semantics
  - States are valuation functions over input variables, internal model variables, and output variables
- Specify **conformance relation** between reference model behaviour K and SUT behaviour K’
  - **I/O-equivalence.** Every input sequence produces the same output sequence in K and K’ (suitable for deterministic applications)
  - **Reduction**. SUT behaviour K’ is a subset of reference model behaviours K (suitable for nondeterministic K)

Construction Principle for Equivalence Class Testing Strategies

- Factorise K-state space into **I/O-equivalence classes**
  - Classes are enumerated over internal state and output combinations
  - States of the same class produce the same responses to all input sequences

$$\mathcal{A} = \{A_1, \ldots, A_n\}$$
$$\forall i, s, s' \in A_i : s(\vec{m}, \vec{y}) = s'(\vec{m}, \vec{y}) \wedge L(s) = L(s')$$

Construction Principle for Equivalence Class Testing Strategies

- Factorise K-input space into **input equivalence class partitions**
  - All input vectors of the same input class, when applied to members of the same I/O-equivalence class,
    - produce the same outputs
    - have the same target I/O-equivalence classes

Construction Principle for Equivalence Class Testing Strategies

- Together, I/O-equivalence classes and input equivalence classes induce an FSM abstraction of the Kripke Structure

$$h : \mathcal{A} \times \mathcal{I} \to \mathcal{A} \times D_O$$

deterministic case

$$h : \mathcal{A} \times \mathcal{I} \to \mathbb{P}(\mathcal{A} \times D_O)$$

nondeterministic case

Construction Principle for Equivalence Class Testing Strategies

- For (deterministic or nondeterministic) FSM, complete strategies exist

- When generating a test suite for an FSM abstraction, choose any representative

$$\vec{c} \in X \in \mathcal{I}$$

when abstract input X is required

Complete Test Strategy

- **Fault model for black-box testing**
  - Reference model
  - Conformance relation
  - Fault domain

$$\mathcal{F} = (\mathcal{S}, \sim, \mathcal{D}(\mathcal{S}, m, \mathcal{I}_2))$$

Complete Test Strategy

CSM model as Kripke Structure -
semantic representation of SysML model

$$\mathcal{F} = (\mathcal{S}, \sim, \mathcal{D}(\mathcal{S}, m, \mathcal{I}_2))$$

Complete Test Strategy

I/O-equivalence as
conformance relation

$$\mathcal{F} = (\mathcal{S}, \sim, \mathcal{D}(\mathcal{S}, m, \mathcal{I}_2))$$

Complete Test Strategy

Maximal number of
I/O-equivalence classes for
each member of the fault domain

$$\mathcal{F} = (\mathcal{S}, \sim, \mathcal{D}(\mathcal{S}, m, \mathcal{I}_2))$$

Complete Test Strategy

A refined IECP - satisfying

$$\forall X \in \mathcal{I}, X' \in \mathcal{I}' : \\ \big(X \cap X' \neq \varnothing \Rightarrow \\ \exists X_2 \in \mathcal{I}_2 : X_2 \subseteq X \cap X'\big)$$

$$\mathcal{F} = (\mathcal{S}, \sim, \mathcal{D}(\mathcal{S}, m, \mathcal{I}_2))$$

Complete Test Strategy

IECP of reference model

A refined IECP – satisfying

$$\forall X \in \mathcal{I}, X' \in \mathcal{I}' : \\ \big(X \cap X' \neq \varnothing \Rightarrow \\ \exists X_2 \in \mathcal{I}_2 : X_2 \subseteq X \cap X'\big)$$

$$\mathcal{F} = (\mathcal{S}, \sim, \mathcal{D}(\mathcal{S}, m, \mathcal{I}_2))$$

Complete Test Strategy

IECP of fault domain member

A refined IECP - satisfying

$$\forall X \in \mathcal{I}, X' \in \mathcal{I}' : \\ \big(X \cap X' \neq \varnothing \Rightarrow \\ \exists X_2 \in \mathcal{I}_2 : X_2 \subseteq X \cap X'\big)$$

$$\mathcal{F} = (\mathcal{S}, \sim, \mathcal{D}(\mathcal{S}, m, \mathcal{I}_2))$$

Complete Test Strategy

Refined IECP

A refined IECP - satisfying

$$\forall X \in \mathcal{I}, X' \in \mathcal{I}' : \\ \big(X \cap X' \neq \varnothing \Rightarrow \\ \exists X_2 \in \mathcal{I}_2 : X_2 \subseteq X \cap X'\big)$$

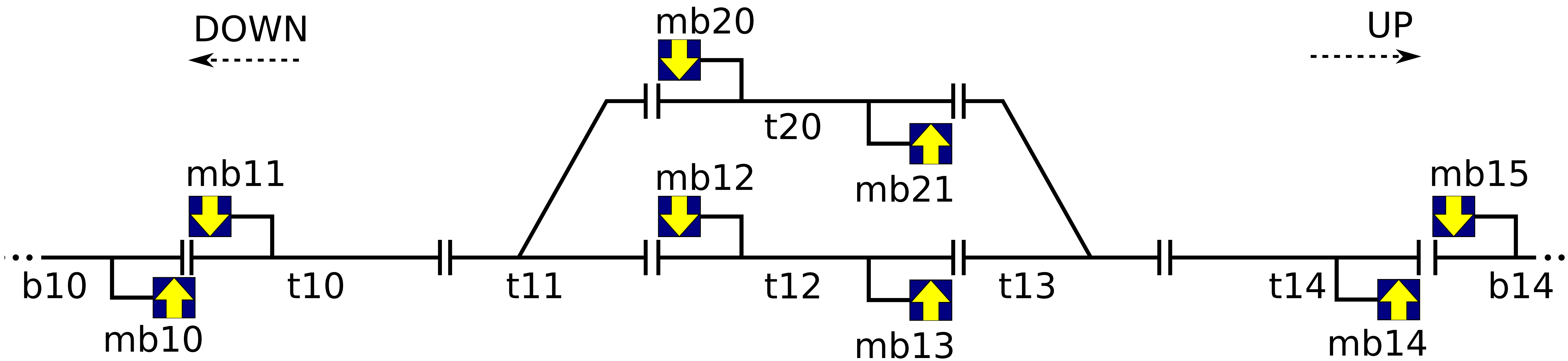
$$\mathcal{F} = (\mathcal{S}, \sim, \mathcal{D}(\mathcal{S}, m, \mathcal{I}_2))$$

If X triggers behaviour in reference model state s, and X' triggers non-conforming behaviour of model representing SUT behaviour, then there exists X2 in intersection of X, X', and a member of X2 will be used in the test

$$
\begin{aligned}
&\forall X \in \mathcal{I}, X' \in \mathcal{I}' : \\
&\quad \big(X \cap X' \neq \varnothing \Rightarrow \\
&\quad\quad \exists X_2 \in \mathcal{I}_2 : X_2 \subseteq X \cap X'\big)
\end{aligned}
$$

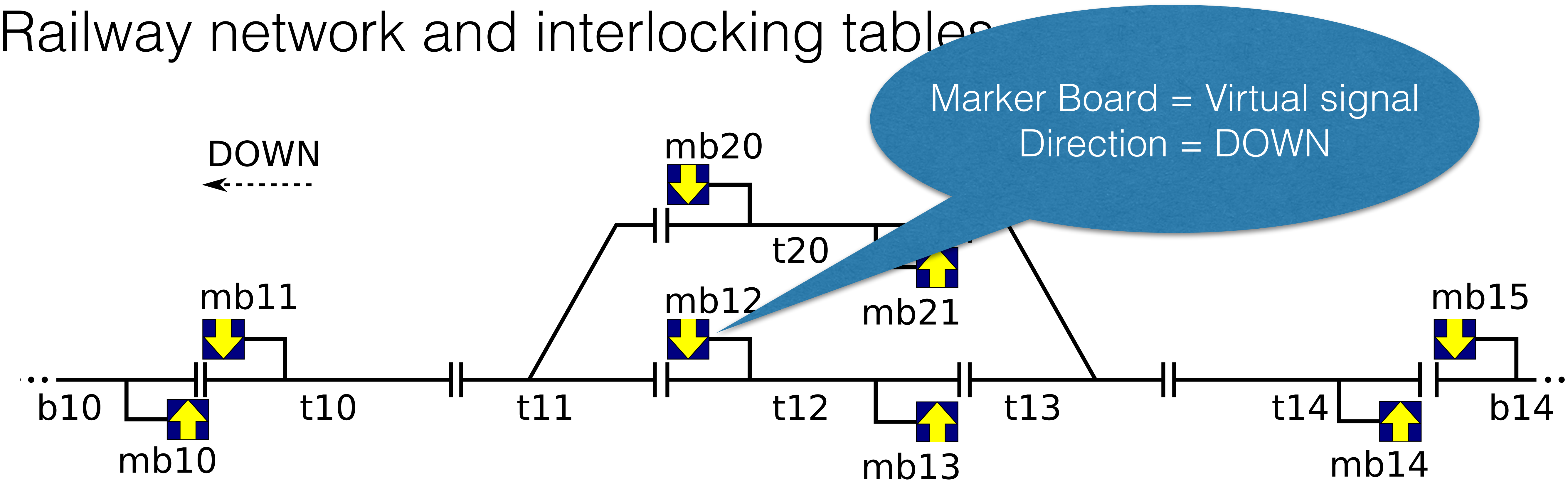
$$\mathcal{F} = (\mathcal{S}, \sim, \mathcal{D}(\mathcal{S}, m, \mathcal{I}_2))$$

Example. Railway network and interlocking tables



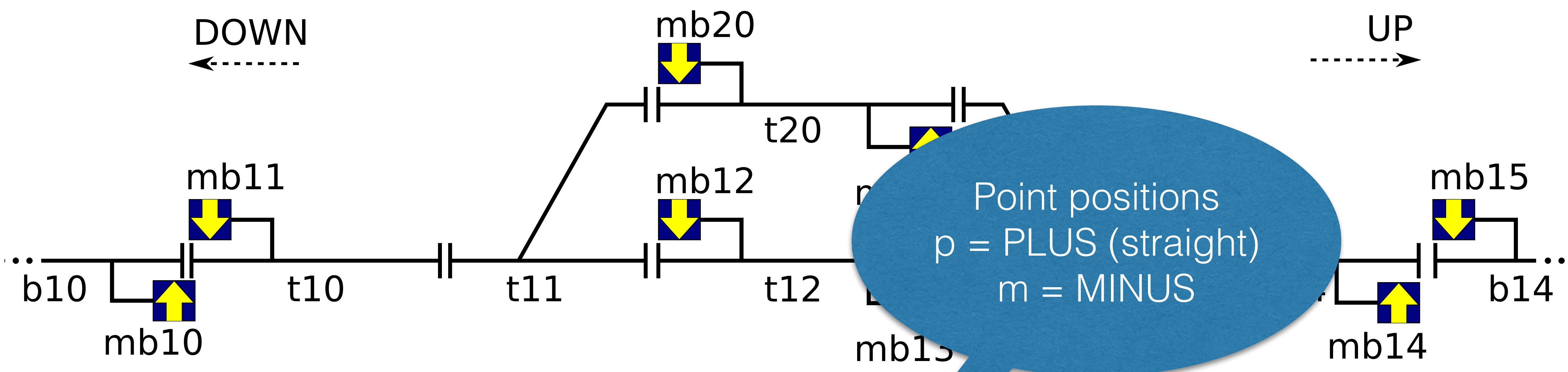
id	Src	Dest	Points	Signals	Path	Conflicts
1a	mb10	mb13	t11:p; t13:m	mb11; mb12; mb20	t10; t11; t12	1b;2a;2b; 3;4;5a;5b;6b; 7
...	...	...	...	...	...	...
7	mb20	mb11	t11:m	mb10;mb12	t11;t10	1a;1b;2a;2b; 3;5b;6a

Railway network and interlocking tables



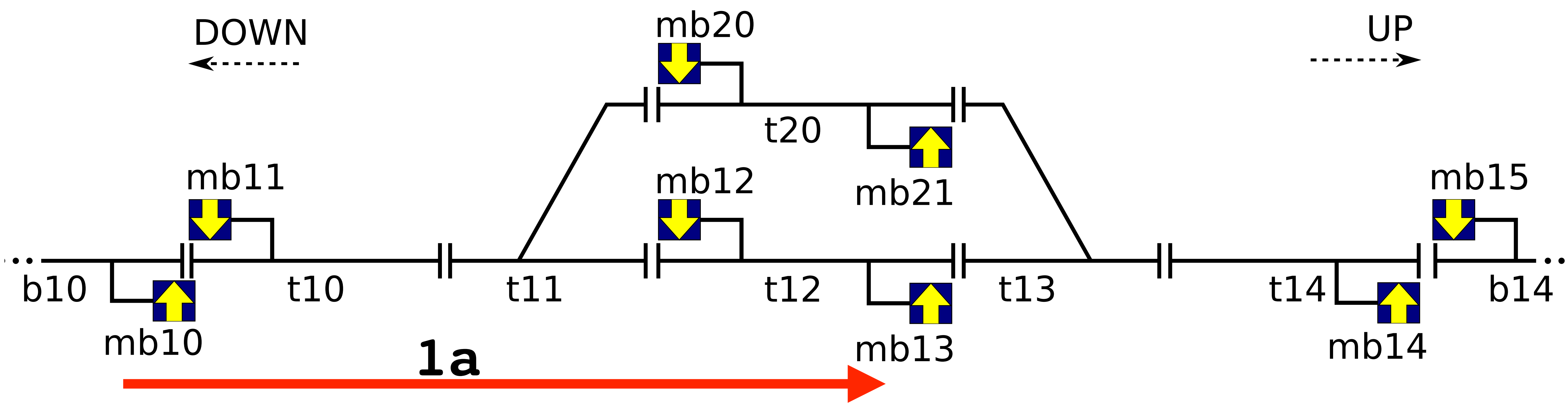
id	Src	Dest	Points	Signals	Path	Conflicts
1a	mb10	mb13	t11:p; t13:m	mb11; mb12; mb20	t10; t11; t12	1b;2a;2b; 3;4;5a;5b;6b; 7
…	…	…	…	…	…	…
7	mb20	mb11	t11:m	mb10;mb12	t11;t10	1a;1b;2a;2b; 3;5b;6a

Railway network and interlocking tables



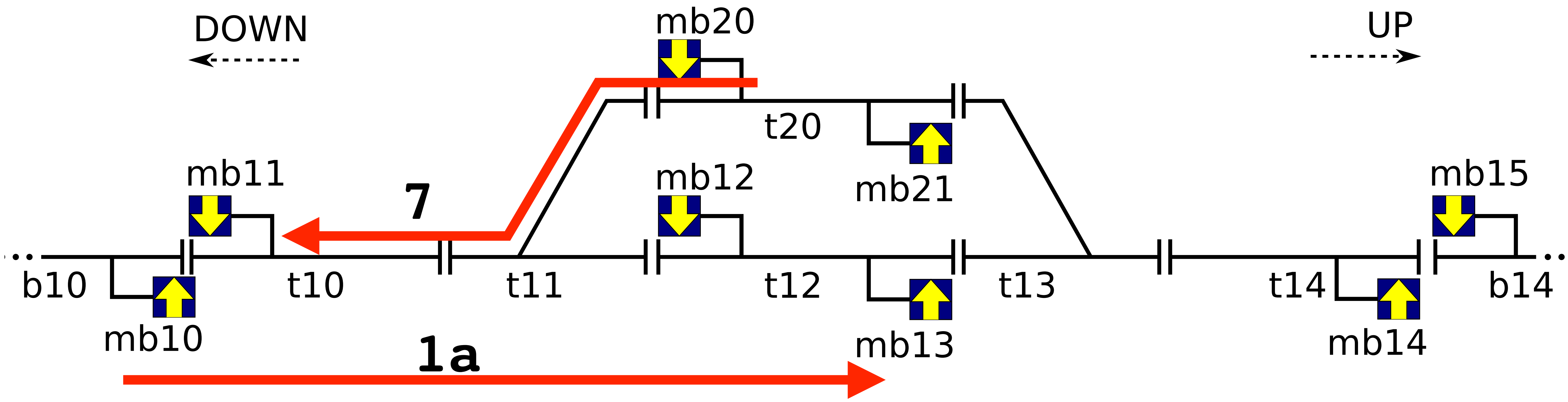
id	Src	Dest	Points	Signals	Path	Conflicts
1a	mb10	mb13	t11:p; t13:m	mb11; mb12; mb20	t10; t11; t12	1b;2a;2b; 3;4;5a;5b;6b; 7
…	…	…	…	…	…	…
7	mb20	mb11	t11:m	mb10;mb12	t11;t10	1a;1b;2a;2b; 3;5b;6a

Railway network and interlocking tables



id	Src	Dest	Points	Signals	Path	Conflicts
1a	mb10	mb13	t11:p; t13:m	mb11; mb12; mb20	t10; t11; t12	1b;2a;2b; 3;4;5a;5b;6b; 7
…	…	…	…	…	…	…
7	mb20	mb11	t11:m	mb10;mb12	t11;t10	1a;1b;2a;2b; 3;5b;6a

Railway network and interlocking tables

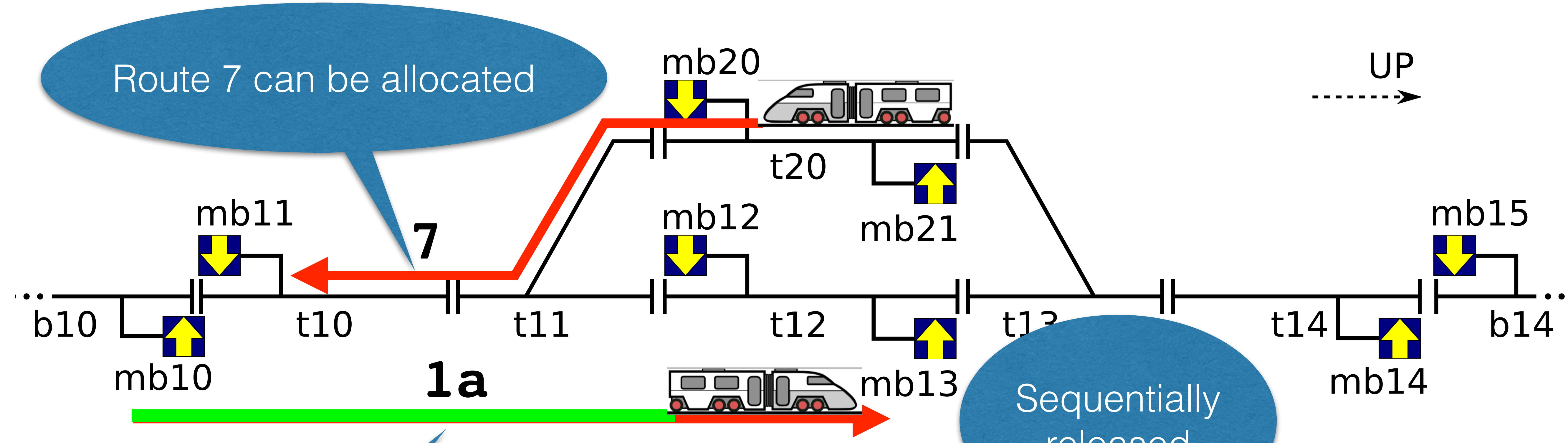


id	Src	Dest	Points	Signals	Path	Conflicts
1a	mb10	mb13	t11:p; t13:m	mb11; mb12; mb20	t10; t11; t12	1b;2a;2b; 3;4;5a;5b;6b; 7
…	…	…	…	…	…	…
7	mb20	mb11	t11:m	mb10;mb12	t11;t10	1a;1b;2a;2b; 3;5b;6a

Route Controller Requirements

- All routes can be allocated
- Conflicting routes are never allocated at the same time, as long as conflicting elements are still in use (**sequential release method**)
- An element along a route is sequentially released, if
  - all previous elements have been sequentially released
  - the train has left the element under consideration
- Routes are marked as free, as soon as all elements have been sequentially released

Sequential release method



Sequentially released

Route 1 is allocated

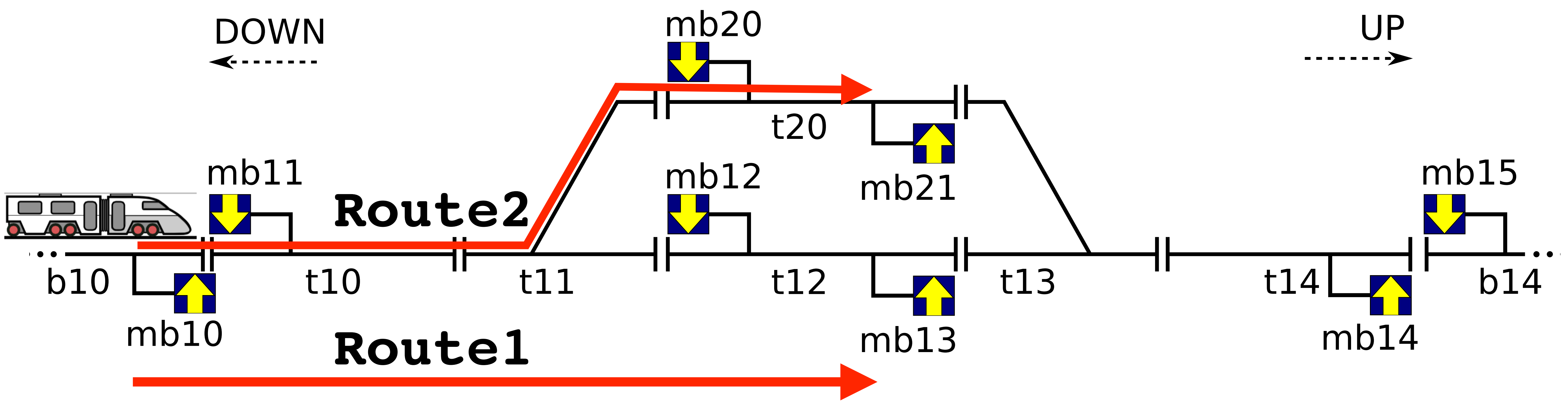
			Points	Signals	Path	Conflicts
1	mb10	mb13	t11:p; t13:...	mb11; mb12; ...20	**t10; t11;** t12	2;3;4;5;6;7
…	…	…			…	…
7	mb20	mb11	t11:m	mb10;mb12	t11;t10	1;2;3;5;6

No conflicts anymore with Route 1

Example – Nondeterministic Reference Models

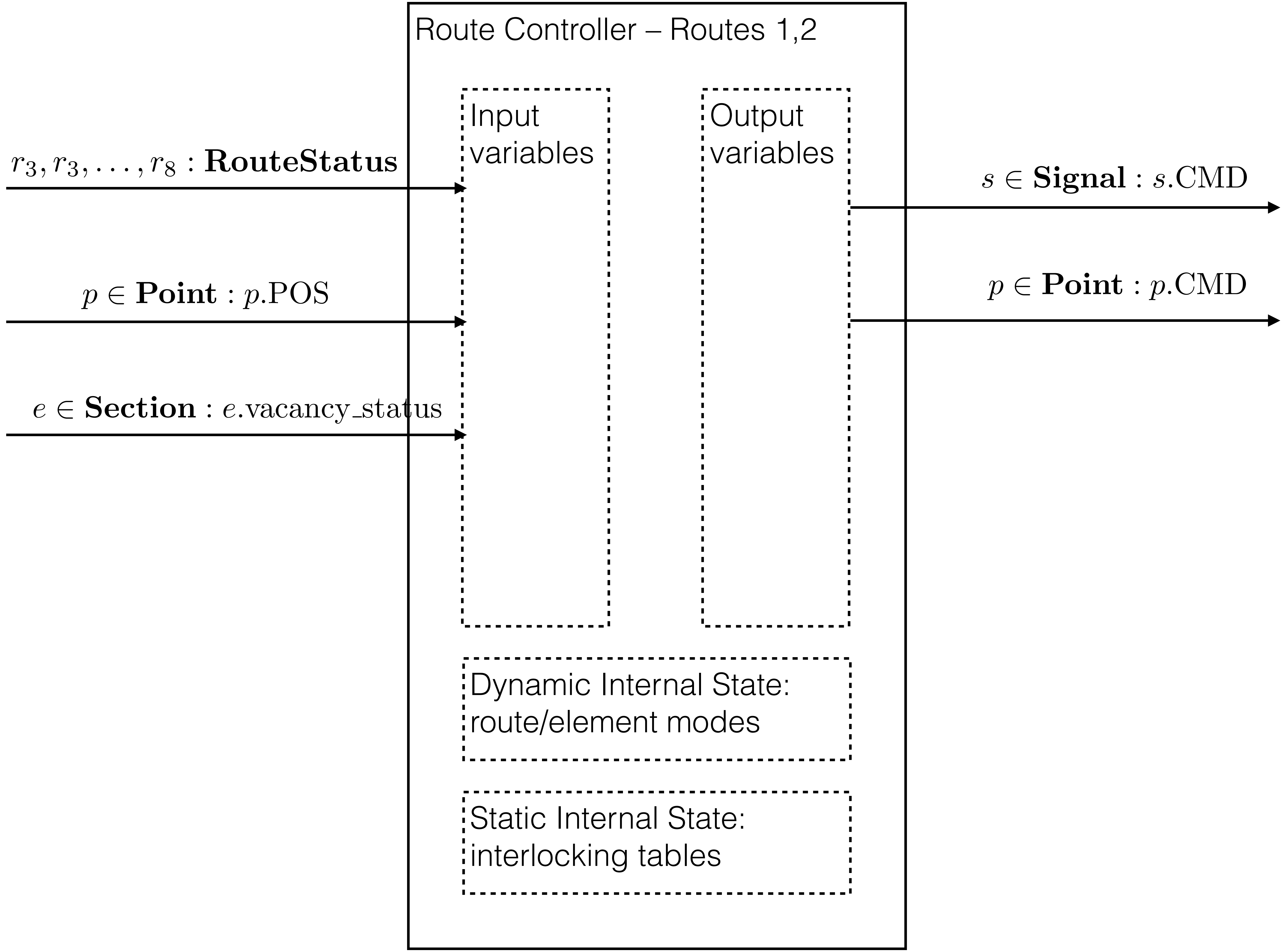
- Even in a safety-critical context, nondeterministic models may be used …
  - … mostly to **reduce model checking complexity** by means of **over-approximation**
- As a consequence, it is useful to have complete model-based testing strategies at hand that can cope with nondeterministic models …
- … though the SUT will usually be deterministic

Railway network and interlocking tables



If no conflicting routes are allocated, route controller model specifies **nondeterministic decision** whether to allocate Route 1 or Route 2

Id	Source	Dest	Conflicts	Id	Source	Dest	Conflicts
1	mb10	mb14	3,4,5,7	5	mb15	mb12	. . .
2	mb10	mb21	3,6,7,8	6	mb15	mb20	. . .
3	mb12	mb11	. . .	7	mb20	mb11	. . .
4	mb13	mb14	. . .	8	mb21	mb14	. . .


Route Controller – Routes 1,2
Input variables
Output variables
$r_3, r_3, \ldots, r_8$: $\mathbf{RouteStatus}$
$p \in \mathbf{Point}$: $p.\mathrm{POS}$
$e \in \mathbf{Section}$: $e.\mathrm{vacancy\_status}$
$s \in \mathbf{Signal}$: $s.\mathrm{CMD}$
$p \in \mathbf{Point}$: $p.\mathrm{CMD}$
Dynamic Internal State: route/element modes
Static Internal State: interlocking tables

Examples for I/O-Equivalence classes

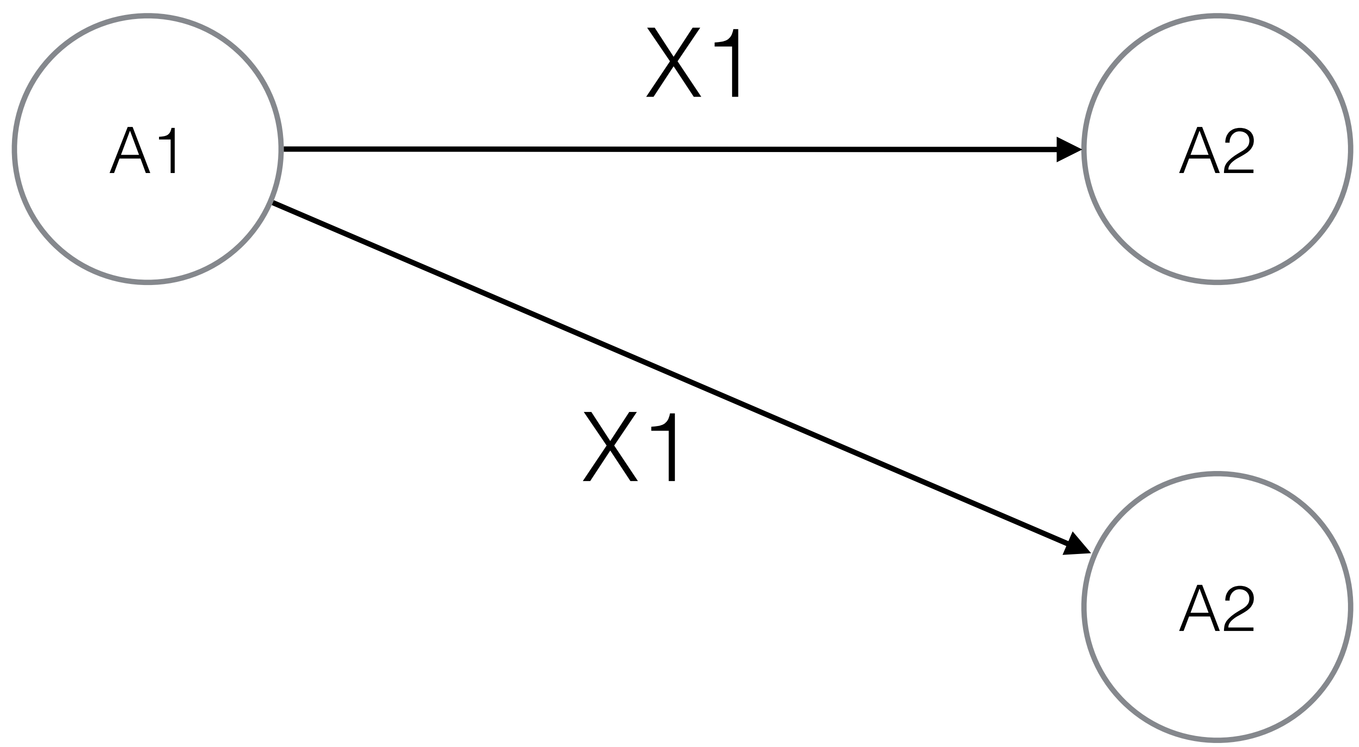
- A1: No train at b10 and Routes 1,2 free and Routes 3,5,6,7 free and t12,t13,t20 empty

- A2: Train in direction UP at b10 and Route 1 allocated and Routes 2,3,5,6,7 free and t12,t13,t20 empty

- A3: Train in direction UP at b10 and Route 2 allocated and Routes 1,3,5,6,7 free and t12,t13,t20 empty

Examples for Input Equivalence Classes and FSM Transitions

- X1: Train at b10 in direction up and Routes 3,5,6,7 free and t12,t13,t20 empty



Overview

- Model-based testing
- Standards for safety-critical transportation systems
- What standards require about model-based testing
- Complete test strategies …
- **… and their relevance for model-based testing of safety-critical systems**
- Discussion

Relevance of Complete Strategies

- Relevance from the certification viewpoint
  - Justified strategy:
    - e.g. selection of equivalence classes
    - new test cases are guaranteed to increase the test strength
- General relevance
  - Superior test strength – also for SUT behaviours outside the fault domain

Table 2. Results for the Ceiling Speed Monitor

		IECP-Tests (B) / (C)		(A) (Random Testing)		
Suite B,C	No. TC	Mutation Score	Line Cov.	No. TC	Mutation Score	Line Cov.
(B,$\mathcal{D}_1$)	21	62 %	86 %	21	34 %	75 %
(C,$\mathcal{D}_1$,1)	21	76 %	97 %	21	34 %	75 %
(C,$\mathcal{D}_1$,10)	183	82 %	97 %	183	54 %	87 %
(C,$\mathcal{D}_1$,25)	453	82 %	97 %	453	72 %	97 %
(B,$\mathcal{D}_2$)	186	87 %	100 %	186	63 %	92 %
(C,$\mathcal{D}_2$,1)	186	88 %	100 %	186	63 %	92 %
(C,$\mathcal{D}_2$,10)	882	94 %	100 %	882	84 %	97 %
(B,$\mathcal{D}_3$)	610	93 %	100 %	610	80 %	97 %
(C,$\mathcal{D}_3$,1)	610	100 %	100 %	610	80 %	97 %
(C,$\mathcal{D}_3$,10)	3002	100 %	100 %	3002	92 %	97 %

Column No. TC records the number of test cases applied. (B,$\mathcal{D}_i$) denotes application of strategy (B) with fault domain $\mathcal{D}_i, i = 1, 2, 3$, (C,$\mathcal{D}_i$,q) denotes application of strategy (C) with fault domain $\mathcal{D}_i, i = 1, 2, 3$ and min $= q$. Columns 'Line Cov.' record the line coverage achieved with the execution of the respective test suite.

Experimental results submitted to TAP 2015

Table 2. Results for the Ceiling Speed Monitor

		IECP-Tests (B) / (C)		(A) (Random Testing)		
Suite B,C	No. TC	Mutation Score	Line Cov.	No. TC	Mutation Score	Line Cov.
(B,$\mathcal{D}_1$)	[illegible]	[illegible]	[illegible]	21	34 %	75 %
(C,$\mathcal{D}_1$,1)	[illegible]	[illegible]	[illegible]	[illegible]	34 %	75 %
(C,$\mathcal{D}_1$,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	54 %	87 %
(C,$\mathcal{D}_1$,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	72 %	97 %
(B,$\mathcal{D}_2$)	[illegible]	[illegible]	[illegible]	[illegible]	63 %	92 %
(C,$\mathcal{D}_2$,1)	[illegible]	[illegible]	[illegible]	186	63 %	92 %
(C,$\mathcal{D}_2$,10)	[illegible]	[illegible]	100 %	882	84 %	97 %
(B,$\mathcal{D}_3$)	610	93 %	100 %	610	80 %	97 %
(C,$\mathcal{D}_3$,1)	610	100 %	100 %	610	80 %	97 %
(C,$\mathcal{D}_3$,10)	3002	100 %	100 %	3002	92 %	97 %

> **Complete IECP strategy — full requirements coverage — boundary tests included — <u>random</u> selection from each input partition**

Column No. TC records the number of test cases applied. (B,$\mathcal{D}_i$) denotes application of strategy (B) with fault domain $\mathcal{D}_i, i = 1, 2, 3$, (C,$\mathcal{D}_i$,q) denotes application of strategy (C) with fault domain $\mathcal{D}_i, i = 1, 2, 3$ and min $= q$. Columns 'Line Cov.' record the line coverage achieved with the execution of the test suite.

Experimental results submitted to TAP 2015

Table 3. Results for the Airbag Controller

		IECP-Tests (B) / (C)		(A) Random Testing		
Suite	No. TC	Mutation Score	Line Cov.	No. TC	Mutation Score	Line Cov.
(B,$\mathcal{D}_1$)	368	89 %	97 %	368	66 %	94 %
(C,$\mathcal{D}_1$,1)	368	96 %	100 %	368	66 %	94 %
(C,$\mathcal{D}_1$,10)	3816	97 %	100 %	3816	68 %	97 %
(B,$\mathcal{D}_2$)	3248	99 %	100 %	3248	68 %	94 %
(C,$\mathcal{D}_2$,1)	3248	100 %	100 %	3248	68 %	94 %

Experimental results submitted to TAP 2015

Table 3. Results for the Airbag Controller

				(A) Random Testing		
Suite	[illegible]	[illegible]	[illegible]	[illegible]	Mutation Score	Line Cov.
$(\mathrm{B},\mathcal{D}_1)$	[illegible]	[illegible]	[illegible]	[illegible]8	66 %	94 %
$(\mathrm{C},\mathcal{D}_1,1)$	[illegible]	[illegible]	[illegible]	368	66 %	94 %
$(\mathrm{C},\mathcal{D}_1,10)$	[illegible]	[illegible] %	100 %	3816	68 %	97 %
$(\mathrm{B},\mathcal{D}_2)$	3248	99 %	100 %	3248	68 %	94 %
$(\mathrm{C},\mathcal{D}_2,1)$	3248	100 %	100 %	3248	68 %	94 %

Complete IECP strategy —
full requirements coverage —
random selection from each input partition

Experimental results submitted to TAP 2015

Overview

- Model-based testing
- Standards for safety-critical transportation systems
- What standards require about model-based testing
- Complete test strategies …
- … and their relevance for model-based testing of safety-critical systems
- **Discussion**

Discussion

- Conclusion
  - We have motivated that complete test suites are of practical value, because they exhibit considerable test strength also outside the specified fault domain
  - We advocate the inclusion of complete test suites into the catalogues of recommended methods in standards for safety-critical standards

Discussion

- Open questions
  - Is the strength of „conceptually complete“ test suites still superior to other approaches, if **only a part of the test suite** is executed?
  - Is the strength of complete test suites still superior to other approaches, if the true SUT performs **security attacks** whose behaviour is outside the fault domain ?

Further Reading

Jan Peleska and Wen-ling Huang: Complete model-based equivalence class testing. Int J Softw Tools Technol Transfer. Published online: 21 November 2014. DOI 10.1007/s10009-014-0356-8.

Linh Hong Vu, Anne Elisabeth Haxthausen, and Jan Peleska: A Domain-Specific Language for Railway Interlocking Systems. In: Eckehard Schnieder and Géza Tarnai (eds.): FORMS/FORMAT 2014 - Formal Methods for Automation and Safety in Railway and Automotive Systems [10th Symposium on Formal Methods for Automation and Safety in Railway and Automotive Systems, Braunschweig, Germany, Sep. 30 - Oct. 2, 2014.] Institute for Traffic Safety and Automation Engineering, Technische Universität Braunschweig, ISBN 978-3-9816886-6-5, pp. 200-209, 2014. (Best paper award)

Cécile Braunstein, Anne E. Haxthausen, Wen-ling Huang, Felix Hübner, Jan Peleska, Uwe Schulze, and Linh Vu Hong: Complete Model-Based Equivalence Class Testing for the ETCS Ceiling Speed Monitor. In S. Merz and J. Pang (eds.): Proceedings of the ICFEM 2014. Springer, LNCS 8829, pp. 380-395, 2014. DOI 10.1007/978-3-319-11737-9_25

Jörg Brauer, Jan Peleska and Uwe Schulze: Efficient and Trustworthy Tool Qualification for Model-based Testing Tools. In Brian Nielsen and Carsten Weise (eds.): Testing Software and Systems. Proceedings of the 24th IFIP WG 6.1 International Conference, ICTSS 2012, Aalborg Denmark, November 2012, Springer LNCS 7641, pp. 8-23 (2012).

Jan Peleska: Industrial-Strength Model-Based Testing - State of the Art and Current Challenges. In Petrenko, Alexander K. and Schlingloff, Holger (eds.): Proceedings Eighth Workshop on Model-Based Testing, Rome, Italy, 17th March 2013, Electronic Proceedings in Theoretical Computer Science 111, pp. 3-28 (2013). DOI:10.4204/EPTCS.111.1

Further Reading

Rob M. Hierons. Testing from a nondeterministic finite state machine using adaptive state counting. IEEE Transactions on Computers, 53(10):1330–1342 (2004).

A. Petrenko, N. Yevtushenko, and G. V. Bochmann. Testing deterministic imple- mentations from nondeterministic FSM specifications. In In Testing of Communicating Systems, IFIP TC6 9th International Workshop on Testing of Communicating Systems, pages 125–141. Chapman and Hall.

Chow, T.S.: Testing software design modeled by finite-state machines. IEEE Trans. Softw. Eng. 4(3), 178–187 (1978)

Vasilevskii, M.P.: Failure diagnosis of automata. Kibernetika (Transl.) 4, 98–108 (1973)

Many thanks for listening!