Analysis of Extended AADL Models: Perspectives

Performability, Safety & Testing

RWTH Aachen University
Software Modeling and Verification Group
Joost-Pieter Katoen

COMPASS Workshop, October 22, 2015

Outline

Performability analysis: Current status

Error models

AADL error models are finite automata enriched with probabilistic failures and repairs.

Two kinds of error models can be distinguished:

- Discrete-time
  - Failures and repairs are modelled by discrete probabilities
  - Instantaneous probabilistic decision to fail (or repair)
- Continuous-time
  - Failures and repairs are modelled by exponential distributions
  - Failures and repairs occur after a random duration

Limitation

All probabilities and distributions need to be known a priori.

New feature: Use parameter synthesis

Performability analysis: Future

Parameter synthesis

Inputs:

1. a (finite-state) parametric error model
2. a performability property
3. a (fixed) threshold on this property

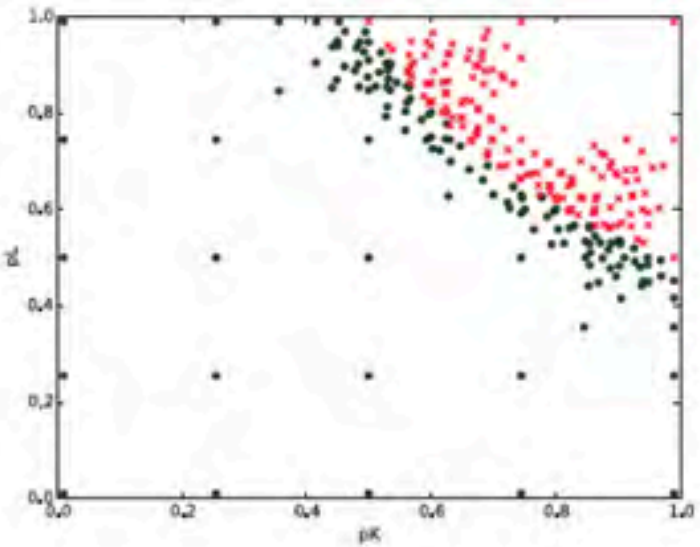
Desired output:

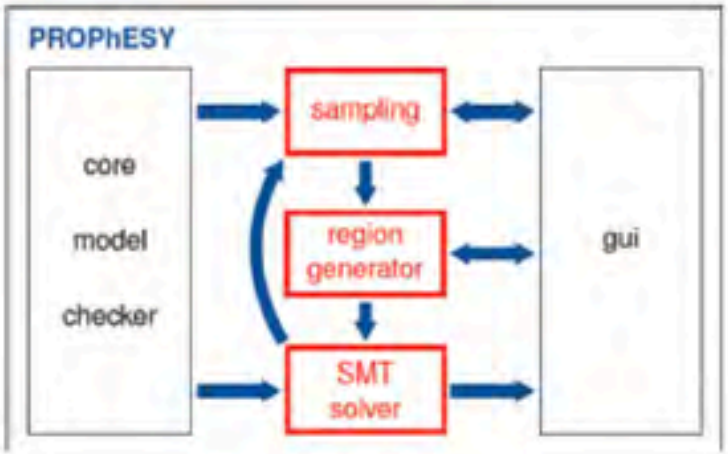
For which parameter values does the AADL model satisfy the property with the given threshold?

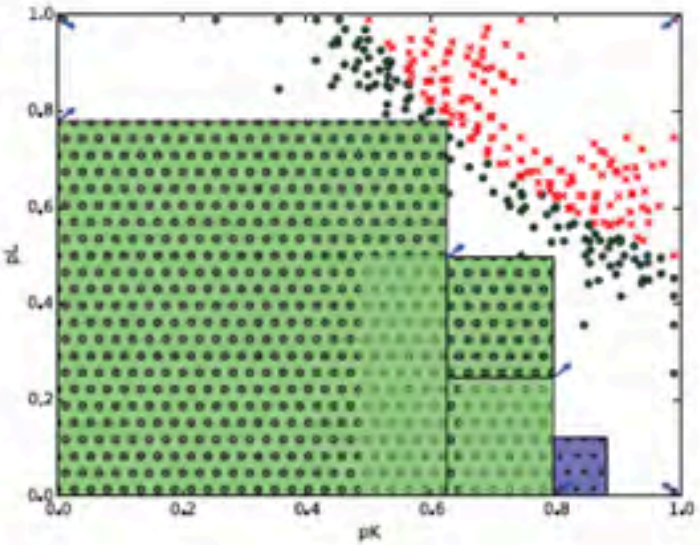
Problem instances:

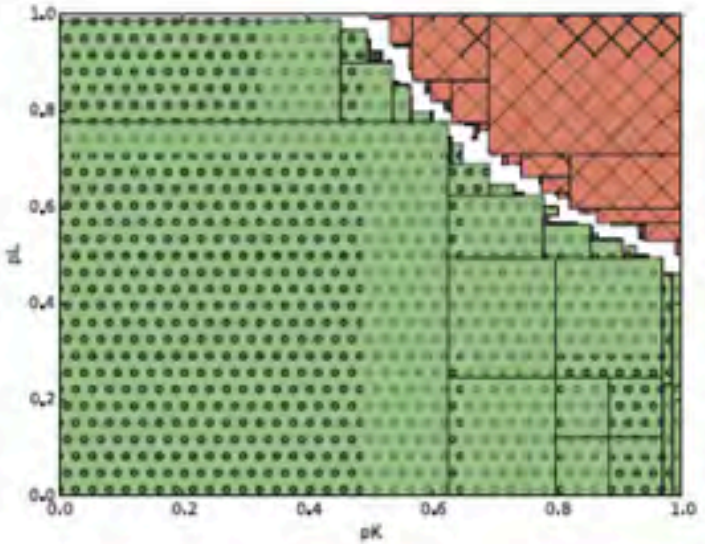
- What is the maximal tolerable failure rate for a given reliability?
- What is the maximal component's failure rate for a desired MTTF?
- .....

Performability analysis: parameter synthesis









Performability analysis: parameter synthesis

					PRISM		PARAM		PROPhESY	
		instance	#states	#trans	verif.	total	verif.	total	verif.	total
reachability	brp	(128, 5)	10376	13827	215	218	5	7	2	3
		(256, 5)	20744	27651	1237	1242	32	33	8	10
	crowds	(15, 5)	592060	1754860	TO	TO	18*	48*	1	46
		(20, 5)	2061951	7374951	TO	TO	75*	194*	4	165
	nand	(20, 2)	154942	239832	886	901	44	48	16	22
		(20, 5)	384772	594792	TO	TO	319	328	89	104
exp. reward	egl	(5, 4)	74750	75773	5	11	–	–	< 1	5
		(8, 4)	7536638	7602173	543	910	–	–	7	607
	nand	(20, 2)	154942	239832	TO	TO	264	2033	5	12
		(20, 5)	384772	594792	TO	TO	TO	TO	47	64
	zconf	(10000)	10004	20005	TO	TO	TO*	TO*	4	4
		(100000)	100004	200005	TO	TO	TO*	TO*	255	263
conditional	brp	(256, 2)	10757	13827	–	–	–	–	< 1	1
		(256, 5)	20744	27651	–	–	–	–	1	3
	crowds	(15, 5)	592060	1754860	–	–	–	–	5	50
		(20, 5)	2061951	7374951	–	–	–	–	14	174

http://moves.rwth-aachen.de/research/tools/prophesy/

Application: model repair

- Assume a reachability probability/MTTF threshold is not met.
- What to do? No problem! Counterexamples point to errors.
- ..... But they are often large and incomprehensible.
- Can we "fix" a model when a bad state is reached too often?
- What is the minimal change to be made?

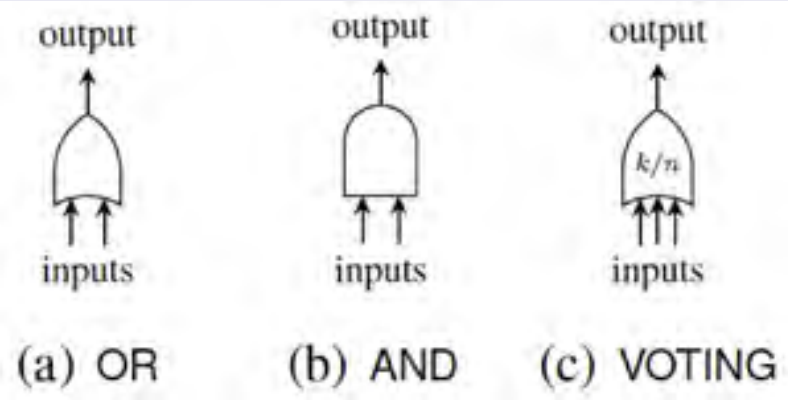
New feature: Model repair. (This is based on parametric models.)

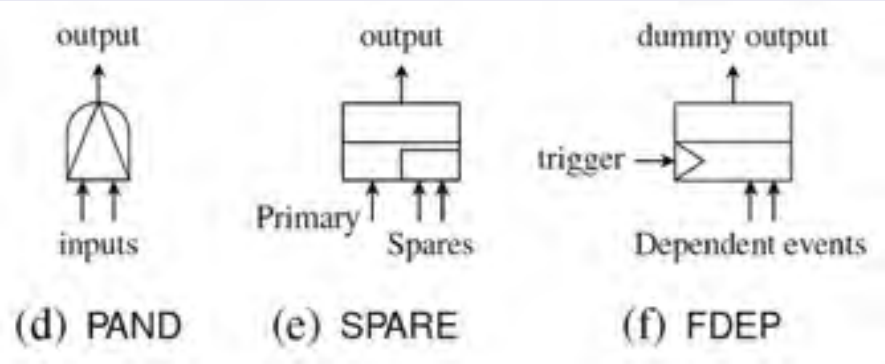
Outline

Safety analysis: current status

Dynamic fault trees

DFTs extend static fault trees with dynamic aspects such as ordering constraints, functional dependencies, and spare elements.





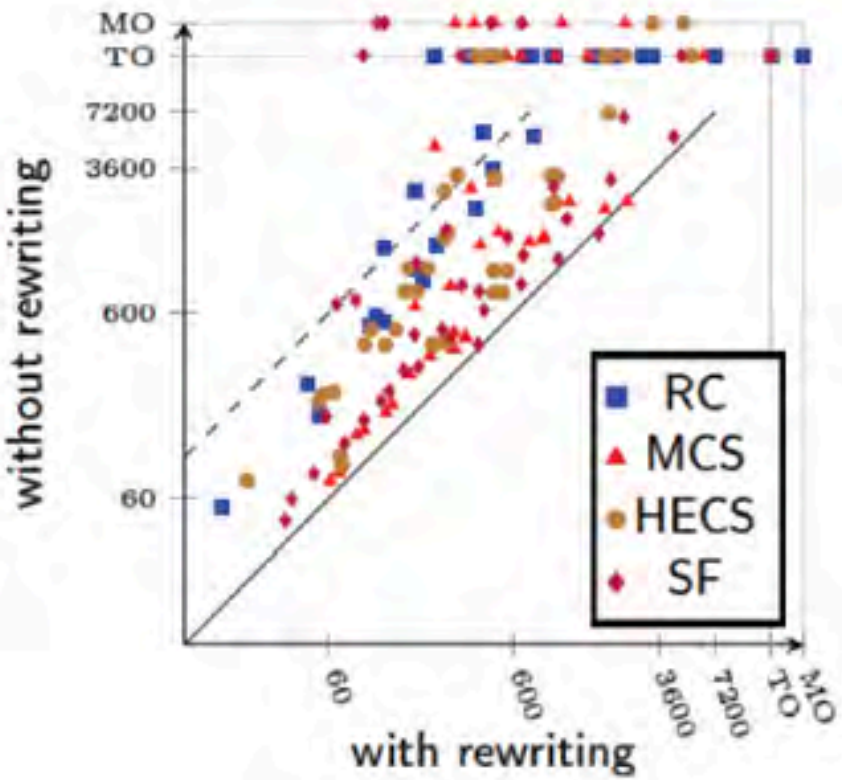
Limitation

Treatment of DFTs in COMPASS is rather restricted.

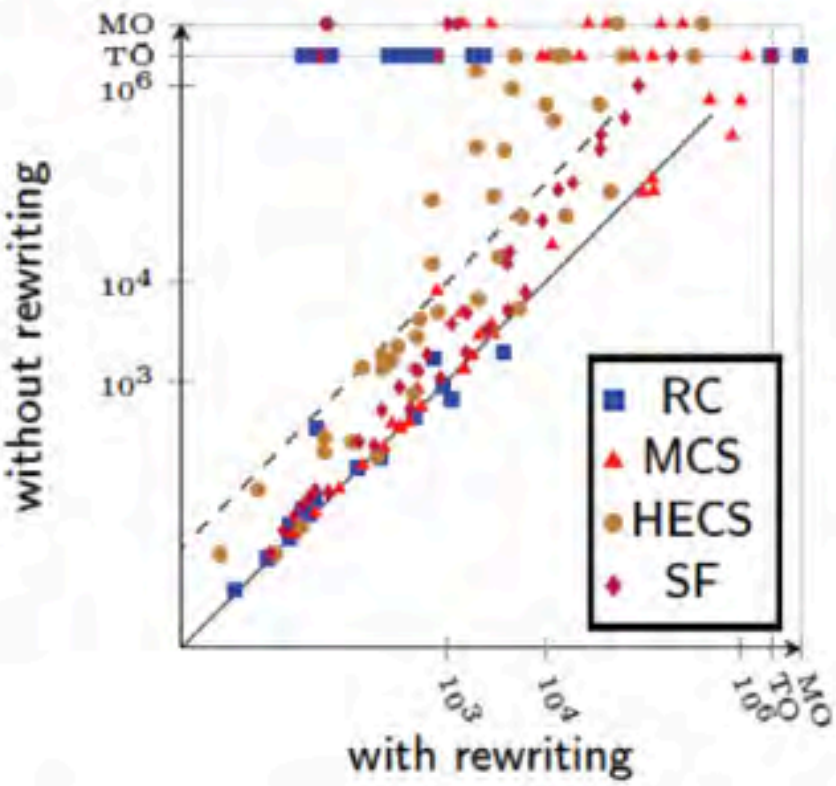
- Analysis of ordering constraints between basic events
- Priority AND gates are used to describe the ordering

New feature: Efficient DFT analysis. For all DFTs.

Safety analysis: DFT rewriting



total verification and minimisation time



state space size of resulting CTMDP

49 out of 179 case studies could be treated now that could not be treated before

Outline

AADL2Simulink

Positive experiences so far

+ Abstraction level of models is appropriate
  - mode transition systems, *not* source code
+ Support of incremental approach to system design
  - hierarchical modelling
  - separation of component interface and implementation
+ Valuable feedback from analysis to system designer
  - design inconsistencies, treating (multiple) failures.
  - improved safety modelling due to automated FMEA/FT generation.

Limitation

No automated link AADL to engineering models (UML, Simulink) hampering the integration into engineering tool suites.

New feature: AADL2Simulink. Enables Simulink tools + code generation.

Outline

Testing of extended AADL models

Current status

All COMPASS analysis features are based on models.

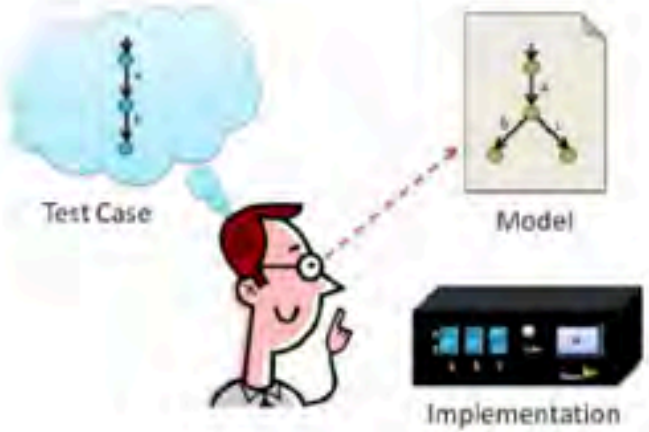
Limitation

No code analysis. No connection of analysis results to the real software.

New feature: Model-based testing of extended AADL models.

Model-based testing in a nutshell

1. Construct a model:
   - 1.1 separate concerns,
   - 1.2 abstract from irrelevant concerns,
   - 1.3 specify the relevant behavior of the system and its environment
2. Define a test selection criteria based on test goals
3. Generate a test suite
4. Build the test infrastructure (adaptor, scripts, coverage tools)
5. Run the tests and analyze the results



Conclusion

New potentially interesting COMPASS features

1. Parametric analysis of performability characteristics
2. Model repair if performability is insufficient
3. Dynamic fault tree (generation and) analysis
4. AADL2Simulink
5. Model-based testing of extended AADL models