

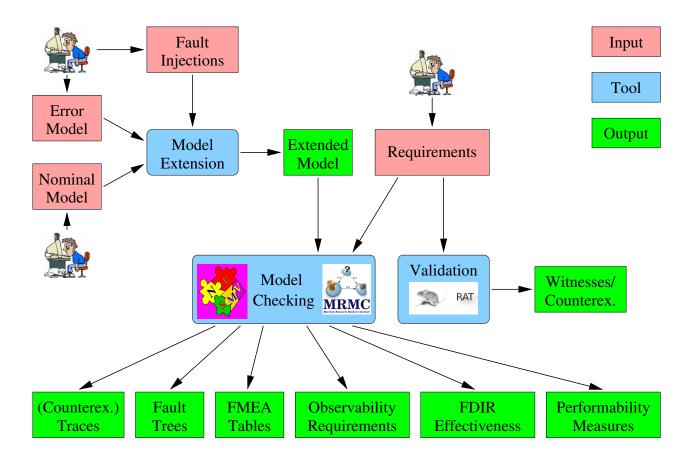
Statistical Approach for Timed Reachability in AADL Models

Harold Bruintjes, Joost-Pieter Katoen, David Lesens



Introduction

Methodology

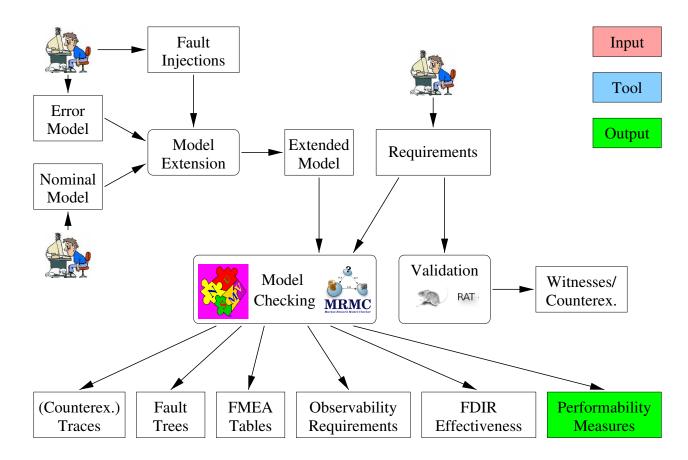






Introduction

Methodology







Introduction

Ever growing demand for probabilistic/dependability analysis.

HASDEL: Analysis of hybrid and probabilistic systems, used for space launchers and vehicles.





Ever growing demand for probabilistic/dependability analysis.

HASDEL: Analysis of hybrid and probabilistic systems, used for space launchers and vehicles.

What is in COMPASS:

- Analysis of timed/hybrid systems;
- Analysis of probabilistic systems;





Ever growing demand for probabilistic/dependability analysis.

HASDEL: Analysis of hybrid and probabilistic systems, used for space launchers and vehicles.

What is in COMPASS:

- Analysis of timed/hybrid systems;
- Analysis of probabilistic systems;
- No analysis of timed and probabilistic systems.





Ever growing demand for probabilistic/dependability analysis.

HASDEL: Analysis of hybrid and probabilistic systems, used for space launchers and vehicles.

What is in COMPASS:

- Analysis of timed/hybrid systems;
- Analysis of probabilistic systems;
- No analysis of timed and probabilistic systems.

Problem: No tools (or algorithms) that support probabilistic analysis of the systems that can be described in the toolset.

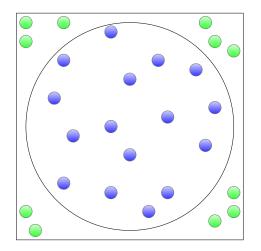
Our approach: Use (Monte Carlo) simulation to approximate the system behavior





Monte Carlo simulation

Generate samples for a process generating random events. When enough samples are generated, with a certain probability the likelihood of the events can be determined.



In our case: Event = Property true/false. Generating event = Generating path

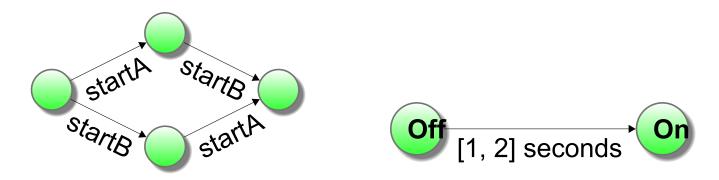




Statistical model checking for COMPASS/HASDEL

Non-determinism

Non-determinism: Underspecification in the model, to model e.g. freedom of implementation, or parallelism.



As Monte-Carlo simulation is purely stochastic, this needs to be dealt with. Simulation cannot execute all choices at the same time.





Strategies

Non-determinism of choice is resolved by uniform distribution: Each possibility is equally likely to occur.

Four strategies have been implemented to resolving non-determinism of time:

- ASAP: Execute a step as soon as possible;
- Progressive: Randomly select a time delay where any transition is possible;
- Local: Randomly select any valid time delay;
- MaxTime: Delay as much as possible.





Strategies

Non-determinism of choice is resolved by uniform distribution: Each possibility is equally likely to occur.

Four strategies have been implemented to resolving non-determinism of time:

- ASAP: Execute a step as soon as possible;
- Progressive: Randomly select a time delay where any transition is possible;
- Local: Randomly select any valid time delay;
- MaxTime: Delay as much as possible.

(For testing there is also the option of manually inputting the next step)





Monte-Carlo simulator for COMPASS implemented in slimsim Inputs:

- SLIM models: nominal and error;
- Fault injections;
- Property;
- Strategy;
- *Error bound*: Width of resulting probability range;
- Confidence: Probability of actual value being within returned range.

Output: Probability range that the property holds true in the given model.





Toolset integration * 3 COMPASS Toolset \odot \odot \otimes <u>File Edit View Activities</u> <u>H</u>elp Model Properties Mission Validation Correctness Performability Safety FDIR Loaded Files Fault Injections Filename Use Error Implementation Error State Effect sensorfilter.slim __default__::SensorFailures.Impl Dead sensors.sensor1.output := 50 ~ sensorfilterErr.slim ~ __default__::SensorFailures.Impl Glitched sensors.sensor1.output := output + 10 ~ __default__::SensorFailures.lmpl Drifted1 sensors.sensor1.output := output + 1 __default__::SensorFailures.Impl Drifted2 sensors.sensor1.output := output + 2 ~ Reload All Remove Add ~ __default__::SensorFailures.Impl Dead sensors.sensor2.output := 50 default :::SensorFailures.Impl Glitched ~ sensors.sensor2.output := output + 10 FDIR Components ~ __default__::SensorFailures.lmpl Drifted1 sensors.sensor2.output := output + 1 Implementation Filename __default__::SensorFailures.lmpl Drifted2 sensors.sensor2.output := output + 2 \checkmark __default__::Monitor.Impl sensorfilter.slim Root Implementation Filename __default__::Acquisition.lmpl sensorfilter.slim **□<>** Add Output Console: Compiling 'sensorfilter.slim'... OK Compiling 'sensorfilterErr.slim'... OK Loading fault injections 'sensorfilter.fixml'... OK > Loaded 8 of 8 fault injections. Extended Model Metrics Logging Compiler





Toolset integration

* ⊗		COMPASS Toolset	\odot \odot		
<u>F</u> ile (* 🛞	COMPASS Toolset	$\otimes \otimes \otimes$		
Mod	<u>F</u> ile <u>E</u> dit ⊻iew <u>A</u> ctivities <u>H</u> elp				
	Model Properties Mission TFPG Validation Correctness Pr	erformability Safety FDIR			
F	Properties				
1	- The probability that after proposition sensors.sensor1.error = error:Dead becomes true, this is responded by proposition sensors.sensor2.error = error:OK within timebound [0, 10].				
1	- The probability that proposition sensors.sensor2.error = error:OK precedes proposition sensors.sensor2.error = error:Gitched within timebound [0, 10].				
	The probability that sensors.sensor2.error = error:OK holds continuously within timebound [0,1]. The probability that proposition sensors.sensor2.error = error:OK held continuously.				
		error = error:OK heid continuously.			
I L	The probability that proposition sensors.sensor2.error = error:Dead will eventually hold within timebound [1, 5]. The probability that proposition sensors.sensor1.error = error:Dead will eventually hold within timebound [1, 50].				
	The probability that proposition sensors sensor i error e error. Dead will eventually nota within timebound [1, 50].				
ſ	+ ⊗ COMPASS 1	oolset <2>			
(Categories	Patterns provapilisucinvariance			
	All	probabilisticExistence			
	Propositional	probabilisticUntil			
	Functional	probabilisticPrecedence			
	Timed Probabilistic	probabilisticResponse			
I I		absence			
		existence			
		universality			
		precedence			
	Pattern Story				
ſ	The probability that proposition sensors.sensor1.error = error:Dead].	will eventually hold within timebound [<mark>1</mark> , <mark>50</mark>			
	Description: Property 2				
	Description, Frépériy 2				
		🥝 Discard 🛛 🖋 Confirm	Delete 🖉 Edit 🕂 Add		





Toolset integration

* 🛞	COMPASS Toolset	\odot \odot
Eile 💠 🛞	COMPASS Toolset	$\otimes \odot \otimes$
Mod <u>File Edit View Activitie</u>		
Hedal Dranartian Mission	TEDO Volidation Correctness Berformability Sofety EDIP COMPASS Toolset	$\odot \odot $
I P Eile Edit ⊻iew Activ t Model Properties Missi		
Properties Name Fo	mula IMC Model Simulation	
Property 7 Th Property 6 Th	e probability Results	
Property 3 Th Property 3 Th Property 3 Th Property 3 Th Property 2 Th	e probability e probability e probability Probability	[55.10638298%, 65.10638298%]
	Settings Error bound 0.05 Confidence 0.99 Strategy asap v	Run
	Warning The root component or Performability results a	ontains ports. are undefined.





Avionics case study

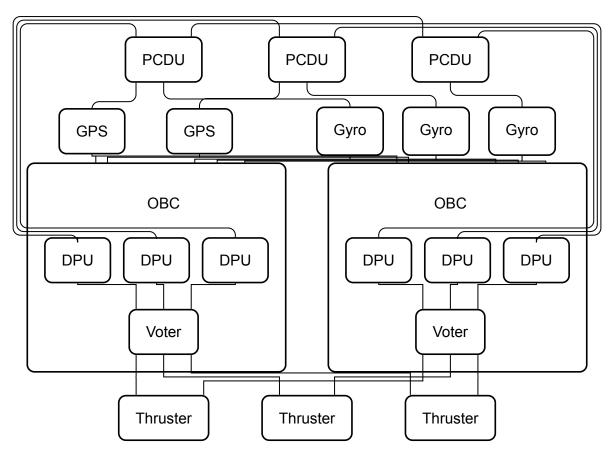
As part of the evaluation of the HASDEL toolset, together with Airbus Defense and Space a case study has been performed. It defines some systems for a hypothetical launcher:

- Triple redundant Power Conditioning and Distribution Units (PCDU)
- Two redundant GPS devices
- Triple redundant Gyroscopes (Gyro)
- Two redundant On-Board Computers (OBC), consisting of:
 - Triple redundant Data Processing Units (DPU)
 - A Voter for the DPU outputs
- Three thrusters





Avionics case study



The architecture of the industrial case study. The connections between the GPS, Gyro and DPU units have been hidden for clarity. Rounded connections are for power, the others for signals.





Avionics case study

For each component, an error model is associated, defining transient, hot and permanent faults.

- 20 nominal and error component definitions
- 37 component instances
- 20 fault injections

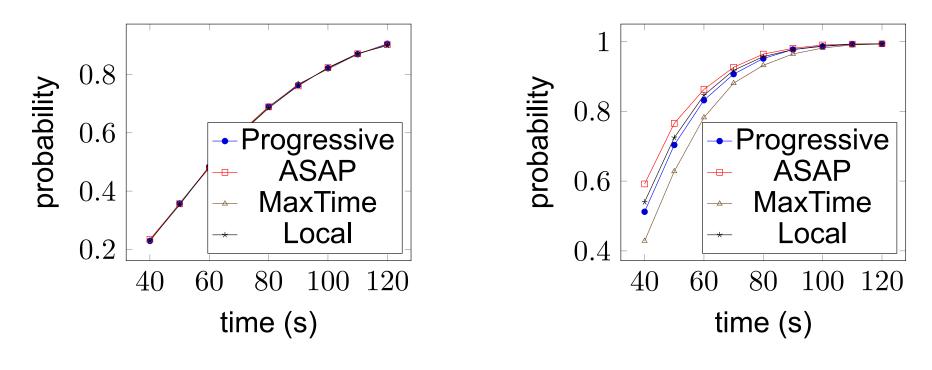
Failure analysis was performed using Monte-Carlo simulation. System failure defined as both OBCs having failed.





Experimental results

Probabilities of system failure containing DPUs without (left) and with (right) repair







Project websites

http://compass.informatik.rwth-aachen.de
https://es-static.fbk.eu/projects/hasdel



