Synthesizing Adaptive Test Strategies from Temporal Logic Specifications

Roderick Bloem, Robert Könighofer, Ingo Pill, **Franz Röck**
Institute of Applied Information Processing and Communications
Graz University of Technology, Austria

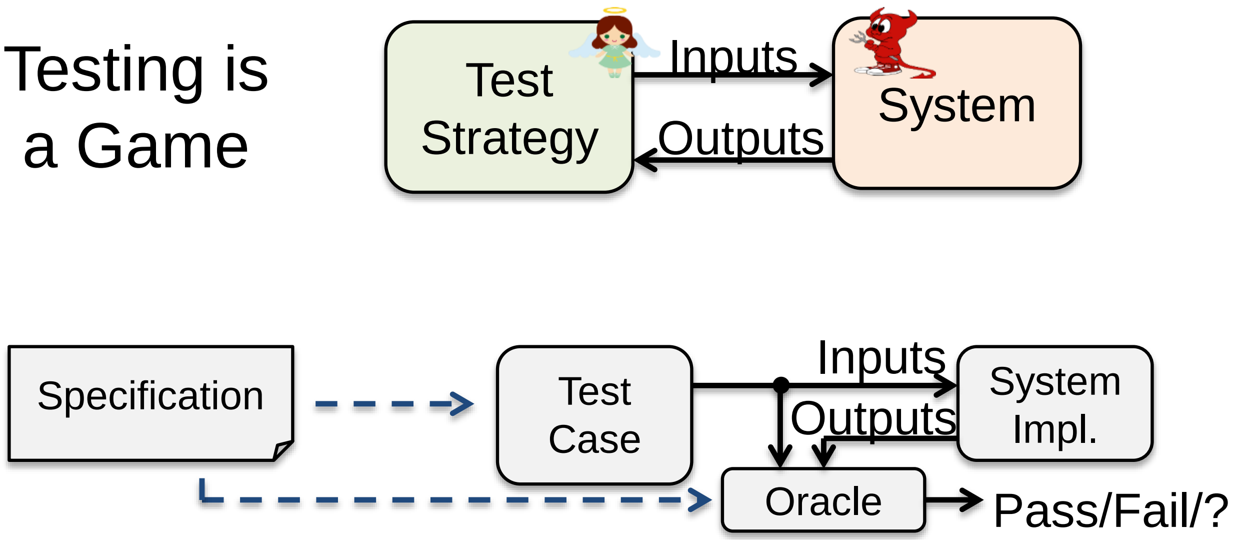
2016-10-04

Outline

- Motivation
- Our Approach
- Fault Models
- Experimental Results
- Conclusion

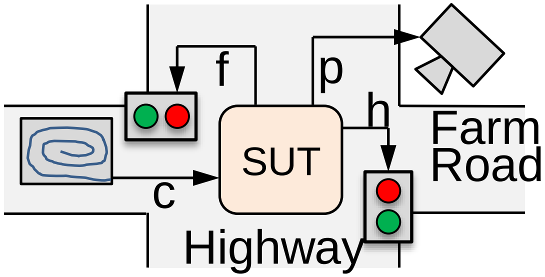
Motivation

Testing is a Game



Motivating Example

1. The lines must never be green simultaneously.
2. If a car is waiting, f eventually turns true.
3. If no car is waiting, h eventually becomes true.
4. A picture is taken if a car does a head start.



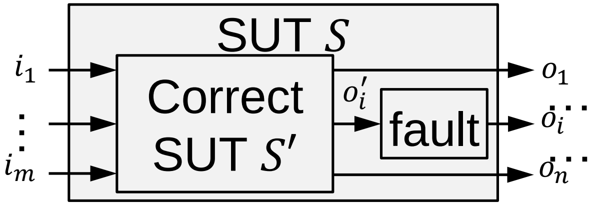
(1) (2) (3) (4)

$$\Phi = G(\neg f \vee \neg h) \wedge G(c \rightarrow Ff) \wedge G(\neg c \rightarrow Fh) \wedge G[(\neg f \wedge X(c \wedge f \wedge X\neg c)) \leftrightarrow XXp]$$

“Good” Tests

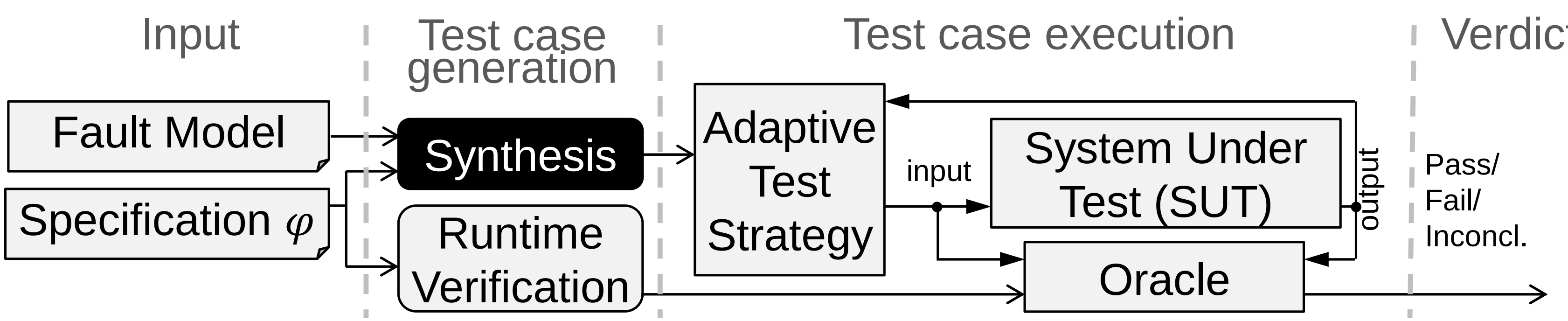
- **Challenge**: what are *good* test cases?
  - Many coverage metrics have been proposed
- Fault based: Tests should reveal certain faults
  - Assume “almost”-correct system under test (SUT)
  - Simple faults (flip, stuck-at-0, …) at single outputs
  - Faults can be permanent or transient
  - Tests must cause a specification violation for these faults

    → Tests will also reveal other faults



Goal

- From temporal logic specifications
- Test goals: certain faults must result in specification violation
- **Enforces** test goals for **every** implementation using **adaptive** test strategies

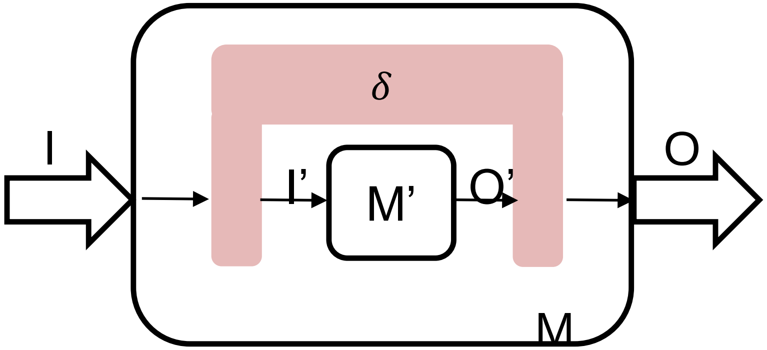


Test Case Generation Approach

Input: I

Output: O

Output' (not observable): I', O'



δ (I,I',O,O') ... fault model

Φ_{corr} *(I',O') ... specification of correct system behavior*

Φ_{obs} *(I,O) ... observable behavior w. r. t. the specification*

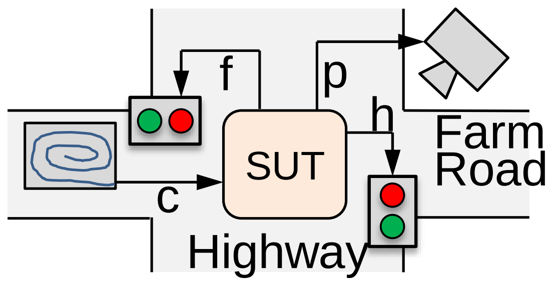
$$(\delta \wedge \Phi_{corr}) \rightarrow \neg\Phi_{obs}$$

Fault models

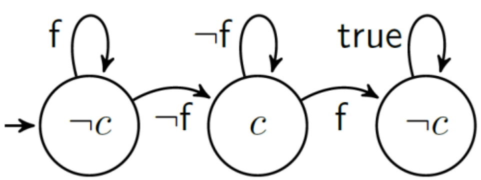
- Frequency
  - Permanent fault (globally)
  - From some point on permanent (eventually globally)
  - …
  - Occurs only once (eventually)

- Fault description
  - Bit flip ($o_i \leftrightarrow \neg o_i'$)
  - Stuck at zero/one ($o_i = 0/1$)
  - Delayed signal ($X(o_i) \leftrightarrow o_i'$)
  - …

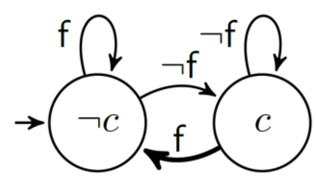
Motivating Example – Test Strategy



Permanent stuck-at-0 fault of p



Stuck-at-0 fault of p that occurs from some point in time onwards



AMBA

TABLE I

RESULTS FOR THE AMBA BUS ARBITER. THE SUFFIX "K" MULTIPLIES BY 10^3.

Fault	o_i	Decide Next frq	$\lvert\mathcal{T}\rvert$	sec	MB	Start Access frq	$\lvert\mathcal{T}\rvert$	sec	MB	Grant Bus frq	$\lvert\mathcal{T}\rvert$	sec	MB	Full Spec frq	$\lvert\mathcal{T}\rvert$	sec	MB
Stuck at 0 ($\kappa = \neg o_i$)	hmaster0	FG	2	359	peak: 574 MB	-	-	147	peak: 138 MB	-	-	146	peak: 131 MB	GF	2	4,848	peak: 2,207 MB
	hgrant0	F	2	18						G	2	150		F	2	2,082	
	hgrant1	-	-	856						-	-	172		GF	2	4,991	
	hmastlock	-	-	803		-	-	133		-	-	133		GF	2	5,808	
	start					G	2	126		G	2	230		FG	2	9,367	
	locked	-	-	736						-	-	170		GF	2	5,236	
	decide	G	2	689										FG	2	9,934	
Stuck at 1 ($\kappa = o_i$)	hmaster0	FG	2	1,237	peak: 783 MB	G	2	133	peak: 130 MB	G	2	153	peak: 131 MB	F	2	2,388	peak: 1,917 MB
	hgrant0	-	-	6,775						-	-	171		GF	2	5,681	
	hgrant1	F	2	19						G	2	151		F	2	1,970	
	hmastlock	G	2	9,64		G	2	115		G	2	186		F	2	1,473	
	start					GF	3	53		-	-	129		GF	2	5,934	
	locked	GF	2	800						-	-	202		GF	2	5,423	
	decide	-	-	1,011										GF	2	4,169	
Flip ($\kappa = o_i \leftrightarrow \neg o_i'$)	hmaster0	G	2	22k	peak: 6,176 MB	G	2	54k	peak: 472 MB	GF	2	1,828	peak: 1,476 MB	Timeout (> 6 days for first output)			
	hgrant0	F	2	29						F	2	10					
	hgrant1	F	2	38						F	2	10					
	hmastlock	G	2	3,385		G	2	53k		GF	2	1,057					
	start					FG	2	43k		G	2	163					
	locked	GF	2	1,525						GF	2	86					
	decide	F	3	61													

Door locked with a PIN

TABLE II
RESULTS FOR THE DOOR SPECIFICATION.

Fault	o_i	frq	$\|\mathcal{T}\|$	sec	MB
stuck-at-0					
	doorclosed	GF	25	22,341	347
	doorlocked	FG	29	2,425	285
stuck-at-1					
	doorclosed	GF	45	23,290	1,000
	doorlocked	FG	52	3.100	148

Conclusion

- Automatic generation of adaptive test strategies from temporal logic specifications
- Independent from implementation details
- No complete information necessary
- Discovers faults that are described in the fault model

Thank you for your attention ☺