Test and verification approaches in conformance checking

Kevin Jahns

RWTH Aachen University

kevin.jahns@rwth-aachen.de

January 28, 2014

Overview

1. Why testing

2. Conformance

3. Test and verification approaches
   - Monkey testing
   - Model based testing
   - Model checking

4. Conclusion

Why software testing and verifying is important

National Institute of Standards and Technology (2002)

Software errors cost the U.S. economy $59.5 billion US dollars annually [2]

Cambridge University (2013)

Software errors cost the whole economy $312 billion US dollars annually [3]

Why software testing and verifying is important

What is *conformance*?

Conformance is ..

What is *conformance*?

Conformance is ..

(1) when it does not explode ;)

What is *conformance*?

Conformance is ..

(1) when it does not explode ;)

(2) when it does not throw errors?

What is *conformance*?

Conformance is ..

(1) when it does not explode ;)

(2) when it does not throw errors?

(3) when it works for the developer (everything else is a user error)?

What is *conformance*?

Conformance is ..

(1) when it does not explode ;)

(2) when it does not throw errors?

(3) when it works for the developer (everything else is a user error)?

(4) when it works for the user?

What is *conformance*?

Conformance is ..

(1) when it does not explode ;)

(2) when it does not throw errors?

(3) when it works for the developer (everything else is a user error)?

(4) when it works for the user?

(5) when it conforms to some sort of specification?

What is *conformance*?

Conformance is ..

(1) when it does not explode ;)

(2) when it does not throw errors?

(3) when it works for the developer (everything else is a user error)?

(4) when it works for the user?

(5) when it conforms to some sort of specification?

→ Conformance is hard to express

How to check conformance

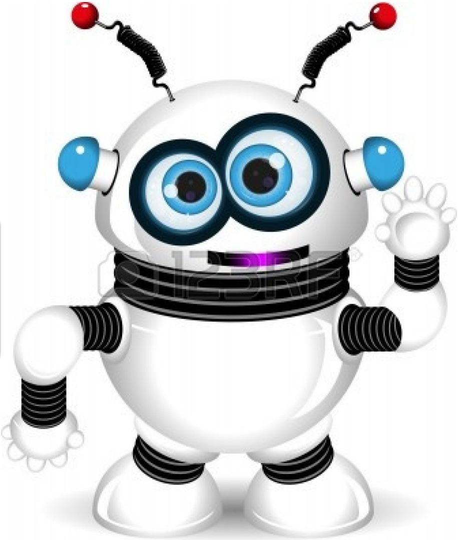
Expressing conformance → checking conformance

Test vs. verification

Testing a robot

Test “Don’t kill me”

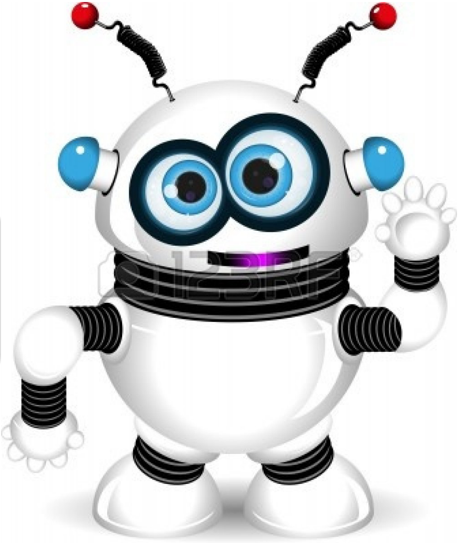
- If the robot kills you, you can be sure that the property is not fulfilled.



Verifying a robot

Verify "Don't kill me"

- After verifying that a robot won't kill you, he won't kill you ;)



Monkey testing

Infinite monkey theorem

The **infinite monkey theorem** states that a monkey hitting keys at random on a typewriter keyboard for an infinite amount of time will almost surely type a given text, such as the complete works of William Shakespeare.[1]

Model based testing

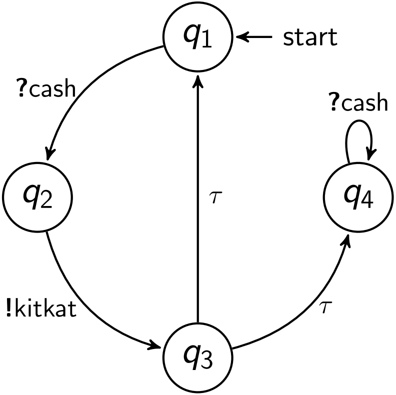
Idea

1. Create Specification
2. Derive test cases
3. Test against software
4. If all tests succeed: Unit under test conforms

Pros and cons

- + Minimizes human error
- + Test cases are derived automatically
- - Evolving topic
- - Complicated

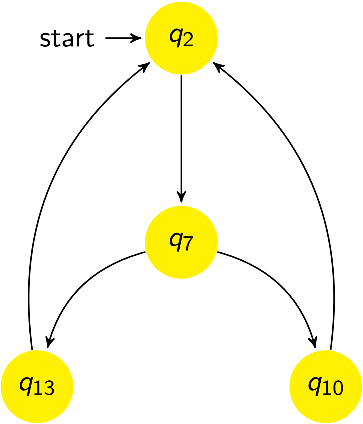
Figure: Candy machine specification



Model checking

```
1  main = do
2    putStrLn $
3        "What is the the"
4      ++ "answer to life"
5      ++ "the universe"
6      ++ "and everything?"
7    answer <- getLine
8    case answer of
9      "42" ->
10       putStrLn
11         "You're right"
12     _ ->
13       putStrLn
14         "Nope"
15   main
```

Figure: Simple transition system



Complexety of model checking

Verifying average software	Real world TS	*thousands* of states

Complexety of model checking

Verifying average software	Real world TS	*thousands* of states
Each state depends on the variables of the Programm	Real world programs have thousands of variables	dimension of new TS $\approx 1000^{1000}$

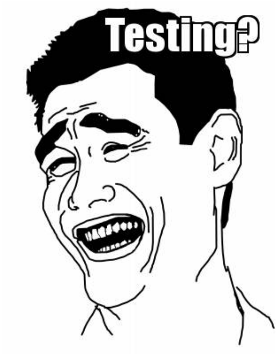
Complexety of model checking

Verifying average software	Real world TS	*thousands* of states
Each state depends on the variables of the Programm	Real world programs have thousands of variables	dimension of new TS $\approx 1000^{1000}$
Time complexety of model checking algorithm is NP-hard	$O(2^{TS})$ computation steps	$\approx 2^{1000^{1000}} \approx 10^{10^{3000}} cumputationsteps$

Complexety of model checking

Verifying average software	Real world TS	*thousands* of states
Each state depends on the variables of the Programm	Real world programs have thousands of variables	dimension of new TS $\approx 1000^{1000}$
Time complexety of model checking algorithm is NP-hard	$O(2^{TS})$ computation steps	$\approx 2^{1000^{1000}} \approx 10^{10^{3000}} cumputationsteps$
Number of atoms in the entire observable universe		$\approx 10^{80}$

Which approaches do software companies use to test software



References

- Infinite monkey effect.

- Department ofg Commerce's National Institute of Standards and Technology (NIST).
  Software errors cost u.s. economy $59.5 billion annually, June 2002.
  URL: http://www.abeacha.com/NIST_press_release_bugs_cost.htm.

- Cambridge University.
  Cambridge university study states software bugs cost economy $312 billion per year, 2013.
  URL: http://markets.financialcontent.com/stocks/news/read/23147130/Cambridge_University_Study_States_Software_Bugs_Cost_Economy_$312_Billion_Per_Year.

That's it: Questions?