Test and verification
approaches in conformance checking

KEVIN JAHNS
English Communication for Engineers
January 27, 2014

Introduction

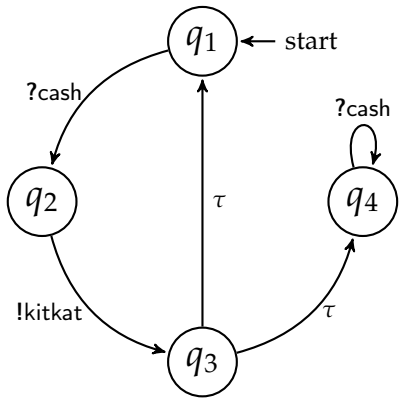
Testing is obligatory for good software development, and not just because it eases bug finding: Since we use technology everyday (e.g. airplanes and traffic circulation) it is evident that software bugs can cost lifes. Furthermore testing is important for the economy: In 2002 a study stated that software errors cost the U.S. economy $59.5 billion US-dollars annually, whereat most of this cost could be avoided by more exhaustive testing [1]. A recent study which was initiated by the Cambridge University states that software bugs cost the overall economy $312 billion US-dollars because debugging is inefficient [2]. Many software companies still use trivial testing approaches like *Monkey Testing*. More enhanced testing approaches could safe costs and in addition make the software better. Moreover there are approaches to *verify* that a software does not malfunction. But the crux of testing and verifying is expressing *conformance*.

Testing

Until now most software developers write test cases for software by hand or just execute the program by themselves. They write test in the form of executable code. Obviously this is not the state of the art. A more expressive definition of conformance is via an *Labeled Transition System* (LTS). In figure 1 we express what a candy machine can do and cannot do in the form of a specification.

1. A candy machine starts in state q_1

Figure 1: Candy machine specification



2. After the candy machine gets cash it must output a kitkat and go into state q_2
3. Then it may either go back into state q_1 or it goes to state q_4
4. In state q_4 it must not output a kitkat and only accept cash.

Conformance

There are various way to express conformance:

1. Statements like "Never do this" are called *persistent properties*
2. State

Conclusion

References

1. Department of 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.

2. 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.