Czech Technical University in Prague
Faculty of Information Technology
Department of Digital Design

Automated Testing of Models of Cyber-Physical Systems

by

Tomáš Apeltauer

A Doctoral Study Report submitted to
the Faculty of Information Technology,
Czech Technical University in Prague

Doctoral degree study programme: Informatics

Prague, September 2018

Supervisor:
doc. Dipl.-Ing. Dr. techn. Stefan Ratschan
Department of Digital Design
Faculty of Information Technology
Czech Technical University in Prague
Thákurova 9
160 00 Prague 6
Czech Republic

Abstract

Area of verification...

Keywords:
keyword1, keyword2, keyword3, keyword4, keyword5.

Acknowledgement

This research has been partially supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS ..., and by the ...

Contents

List of Figures

List of Tables

Abbreviations

General[TODO delete if no other cathegory]

CPS	Cyber-Physical Systems
MBD	Model-based design
MTL	Metric Temporal Logic
GPS	Global Positioning System
EMBS	Electro-Mechanical Braking System
MBD	Model-Based Testing
FSM	Finite-State Machine
UML	Unified Modeling Language
SAT	Boolean satisfiability

Common Mathematical Functions and Operators

10_2	Numbers' radices are designated with a subscript
$\mathbf{b}$	Vector $\mathbf{b}$
b_i	the i^{th} element of vector $\mathbf{b}$
$\|\|\mathbf{b}\|\|$	Norm of vector $\mathbf{b}$
$\dim \mathbf{b}$	Dimension of vector $\mathbf{b}$
$\mathbf{A}$	Matrix $\mathbf{A}$
$a_{i,\,j}$	Element of matrix $\mathbf{A}$ at the i^{th} row, and the j^{th} column
$\mathbf{A}^{-1}$	Inverse matrix to matrix $\mathbf{A}$
$\mathbf{A}^T$	Transposed matrix to matrix $\mathbf{A}$
$\|\|\mathbf{A}\|\|$	Norm of matrix $\mathbf{A}$
$\operatorname{cond} \mathbf{A}$	Condition number of matrix $\mathbf{A}$
$\operatorname{rank} \mathbf{A}$	Rank of matrix $\mathbf{A}$ — how many independent rows/columns it has
$\max\{a,\, b\}$	Maximum of a and b, a when $a \geq b$, b when $a < b$
$\min\{a,\, b\}$	Minimum of a and b, a when $a \leq b$, b when $a > b$
$O(x)$	The big O notation
$\Theta(x)$	The big Θ notation

Mathematical Terminology

Q	Number of prime number modules
M	A product of individual modules $M = \prod_{i=1}^{Q} m_i$
...	...
...	...
...	...
...	...

Miscellaneous Abbreviations

FPU	Floating Point Unit
...	...
...	...
...	...
...	...

Chapter 1

Introduction

At the beginning of the 21.st century human race enters into a new era of industrial revolution generally called Industry 4.0[TODO citation]. So far humans used computers and automation to make industrial processes as efficient as possible. But now the technology allowes us to create Cyber-Physical Systems (CPS) and integrate them into the industrial process. All the work thus can be passed to fully autonomous devices that man will only oversee, giving us more space for something humans do the best, intellectual creativity. But if we are to put all the work on CPS, we must make sure that such devices will be as safe and secure as possible.

1.1 Motivation

CPS are specific in their strusture[5[Tariq,Florence]Design Specification of Cyber-Physical Systems]. They contain both a discrete unit and continuous unit. Most of such devices fit into cathegory of embedded systems, because they monitor variables of the physical world (temperature, pressure, chemical composition, speed, etc.) and also react based on the values of such variables. The manufacturing process of CPS is still very expensive. To address this issue, many companies all over the world use Model-based design (MBD) for prototyping and upgrading their products. MBD puts a lot of emphasis on the creation of digital model of CPS.

An important part of such process is model verification. Usually an engineer has a list of requirements that CPS must comply in order to be labeled as safe an secure. Manual process of verification of models of CPS is very time consuming and limited. That is why several verification tools have been developed to help companies by running automated tests simulations against a set of requirements. These tools use complex search algorithms to find a simulation trajectory that violates given requirement(s). It is not a trivial task, because of the conjuction of discrete and continuous worlds. For example continuous dynamic of rotating car wheel can be clearly described using set of differential equations, but when an Anti-lock braking system discrete controller locks the wheel, none of these equations holds.

1.2 Problem Statement

In addition to a vast complexity of behaviour of CPS, verification tools treat models of CPS only as black boxes, not considering its inner structure. This approach have its limitations. Testing a model without the knowledge of its inner structure will never be as effective as if we would test it with structure and contextual analysis. That is why we aim to propose new algorithms for automated testing of models of CPS with the consideration of their inner structure.

The first objective of our research is to gather useful models of CPS, preferably the ones used in the industry area. Then we focus on the verification process itself and the tools generally used in practise. We try to find use cases when the performance of conventional or academic tools is insufficient and enhance them by providing deep model analysis information.

1.3 Related Work/Previous Results

This research is based upon the work of the research group from Cyber-Physical Systems Laboratory at Arizona State University. [TODO publication citations] They created a verification tool named S-TaLiRo [TODO citation of S-TaLiRo paper] and also presented their own metric for effective searching for simulation trajectories when verifying a model against given specification [citation of robustness metric paper].

When working with specification and requirements we use Metric Temporal Logic (MTL) developed by Ron Koymans [citation]. This way of specifying demands on CPS is suitable because MTL allows us to formulate restrictions as: "There is a maximum number of time units so that each occurrence of an event E is responded to within this bound".

Our efforst were presented on the student seminar PAD 2017 [citation] where we gathered a lot of valuable feedback. This helped us to concreticise our goals and form reasonable milestones.

1.4 Structure of the Report

The report is organized into ... chapters as follows:

1. *Introduction*: Describes the motivation behind our efforts together with our goals. There is also a list of contributions of this report.

2. *Background and State-of-the-Art*: Introduces the reader to the necessary theoretical background and surveys the current state-of-the-art.

3. *Overview of Our Approach*: ...

4. *Prelimitary Results*: ...

5. *Conclusions*: Summarizes the results of our research, suggests possible topics of your doctoral thesis and further research, and concludes the report.

Chapter 2

Background and State-of-the-Art

In the last decade we have seen a dramatic decrease in the cost of certain computation technologies and such phenomenon gave a birth of a new family of embedded control systems that are much better prepared for fluent, realistic interaction with the continuous physical world around them. For systems that combine physical world around us with the world of cybernetics, we use a term cyber-physical system. Although certain forms of CPS have been in industril use since 1980s, only recently has the technology for processors, wireless communication, and sensors matured to allow the production of components with impressive capabilities at a low cost.[Rajeev Alur Principles of CPS]

Advance in the field of Cyber-physical systems will bring us closer to usage of high-speed, low-cost, and real-time embedded computers in technologies like electric networks that employ advanced monitoring [Smart Grids: A Cyber–Physical Systems Perspective, By Xinghuo Yu and Yusheng Xue], networked autonomous vehicles [E. A. Lee, "Cyber Physical Systems: Design Challenges," 978-0-7695-3132-8] or prosthesis like neural controlled artificial leg [On Design and Implementation of Neural-Machine Interface for Artificial Legs, Xiaorong Zhang, Yuhong Liu]. CPS are a research priority for both, government agencies (National Science foundation) and industry (automotive, avionics, medical devices).

2.1 Cyber-Physical Systems

The concept of a cyber-physical system is a generalization of embedded systems.[Rajeev Alur Principles of CPS] An embedded system consists of hardware and software integrated within a mechanical or and electrical system designed for a specific purpose. CPS consist of a computational unit, sensors, actuators and a physical world which it must observe and react on it. In a CPS the controller consists of discrete software concurrent components, operating in multiple modes of operation, interacting with the continuosly evolving physical environment. Examples of on-board sensors include a global positioning system (GPS) receiver, a camera or an infrared thermal sensor.[Rajeev Alur Principles of CPS] CPS are reactive systems which interact with its environment in an ongoing manner. There is an

endless loop of data collection and input evaluation throughout the time.

TODO Chart of CPS

In comparison to the traditional software development architecture, the creation of CPS differs in the emphasis on the security, confidence, reliability and performance of the system. CPS are often used in areas with many safety requirements (medicine [cite S-TaLiRo insulin pump], automotive [cite EMB paper], civil engineering [cite some smarthome papers], avionics, etc.). Apart from embedded systems, CPS will not be operating in a controlled environments and must be robust to unexpected conditions and adaptable to subsystem failures. [E. A. Lee, Cyber Physical Systems: Design Challenges, 2008,978-0-7695-3132-8]. An example of such systems is an autopilot system used on Airbus aircraft. It is a device used to guide an aircraft without direct assistance from the pilot. Modern autopilots are capable of controlling every part of the flight from just after take-off to landing and are normally integrated with the flight management system.

2.1.1 Reactive Computation

CPS are intended to seamlessly interact with the physical world around in an infinite feedback loop. Such real-time computing can be very challenging, because it usually consist of processing huge amount of inputs and delivering immediate reactions. The traditional computing device process an input and produces an output. An example is a program that process an unsorted list of numbers and returns a sorted list (based on given criteria, e.g. in an ascending order).

A reactive system, in contrast, interacts with its environment in an ongoing manner via inputs and outputs. As a typical example of reactive computation consider a program for a cruise controller in a car. CPS are reactive systems.[Rajeev Alur Principles of CPS]

The design of a complex cyber-physical system — especially one with heterogeneous subsystems distributed across networks — is a demanding task. Commonly employed design techniques are sophisticated and include mathematical modeling of physical systems, formal models of computation, simulation of heterogeneous systems, software synthesis, verification, validation, and testing.[J. C. Jensen, D. H. Chang and E. A. Lee, A model-based design methodology for cyber-physical systems,978-1-4244-9538-2].

Embedded system is usually constructed from the physical plant and the controller module. The controller contains specific algorithm, designed for capabilities and resources of given embedded system. In industry production area a Model-driven development paradigm has been deployed and successfully tested for development of embedded systems. Unfortunately when we move from simple programs to more complex software systems and particularly to cyber-physical systems, former design techniques and tools are no longer applicable.

During the process of creation and implementation of an autopilot system, we expect a high level of assurance in the correct behavior of the system. If it would be the other way around, any error can lead to unacceptable consequences such as losses of lifes. Systems where safety requirements have higher priority than other design objectives such as performance and developement cost are called safety-critical. CPS generally fit into this

cathegory. That is why ensurance of a system's corectness during design is of utmost importance and sometimes even mandatory because of goverment regulations.

Former approach of system developement is divided into several phases: design, implementation, testing and validation. More suitable and practial approach is to write mathematically precise requirements of the desired system, design models od system components and using analysis tools check if the system meets the requirements. Usage of formal models and verification is fitting for the area of safety-critical applications.

And that is why a different paradigm for developing CPS was created. It is called Model-Based Design and it is increasingly adopted by industry.[J. C. Jensen, D. H. Chang and E. A. Lee, A model-based design methodology for cyber-physical systems,978-1-4244-9538-2]

2.2 Model-Based Design

The goal of modeling in system design is to provide mathematical abstractions to manage the complexity of design. In the context of reactive systems, the basic unit of modeling is a component that interacts with its environment via inputs and outputs.[Rajeev Alur Principles of CPS] What exactly is a model? A model is defined as a small object, usually built to scale, that represents in detail another, often larger object. A schematic description of a system, theory, or phenomenon that accounts for its known or inferrend properties and may be used for further study of its characteristics. [American Heritage Dictionary]

In their work Jensen et al. 2011 [J. C. Jensen, D. H. Chang and E. A. Lee, A model-based design methodology for cyber-physical systems,978-1-4244-9538-2] propose a 10-step methodology for developing cyber-physical systems:

1. State the problem
2. Model physical processes
3. Characterize the problem
4. Derive a control algorithm
5. Select models of computation
6. Specify hardware
7. Simulate
8. Construct
9. Synthesize software
10. Verify, validate and test

This approach helps designers break enormous task of creation of CPS into manageable iterations, which can be repeated if needed. Main goal is to identify any bugs or errors as soon as possible and preferably before the construction phase. For example we would like to design new experimental electro-mechanical braking system (EMBS) . Such system consists of an electric engine, a brake caliper, a brake disc and a wheel. The brake disc is connected to the wheel, so that contact between caliper and disc will result in vehicle deceleration. [Strathmann and Oehlerking in Verifying properties of an electro-mechanical braking system]. According to traditional design process, we would propose a system design, implement it in real life, test it manually or randomly and then launch pilot project. In case of such complicated device as EMBS conventional approach is unsatisfactory. How can we be sure whether the manually or randomly generated scenarios capture the worst-case conditions for the system under test? An alternative approach is to utilizea Model Based Development (MBD) framework and use the models to simulate the system and intelligently search for corner cases. [Tuncali, Fainekos, Functional Gradient Descent optimization for automatic test case generation for vehicle controllers] This way we use search-based methods to detect corner cases that violate the safety requirements. We refer to such process as falsification because these methods strive to generate counterexamples that disprove or falsify safety requirements. In case of our EMBS, an example of a safety reqirement could be formulated as: As soon as braking is requested, the contact between caliper and disc should occur within 23 ms.[Strathmann and Oehlerking in Verifying properties of an electro-mechanical braking system]

2.3 Model-Based Testing

Is it really so important to test and verify CPS? We can present a few examples when perfunctory testing led to a catastrophic failure:

1. The Ariane 5 rocket exploded 36 seconds after the lift-of; the amount lost was of half a billion dollars. The failure was caused by an uncaught exception [https://esamultimedia.esa.int/do x-1819eng.pdf]

2. An error in the software of the baggage handling system delayed of 9 months the opening of the Denver airport, with a loss of 1.1 million dollars per day [http://www.eis.mdx.ac.uk/research/SFC/Reports/TR2002-01.pdf]

3. Intel lost 475 million dollars for replacing Pentium II processors that had a faulty floting-point division unit [Khan, M. A., Ansari, A. Q. (2012). Handbook of Research on Industrial Informatics and Manufacturing Intelligence: Innovations and Solutions (pp. 1-662). Hershey, PA: IGI Global. doi:10.4018/978-1-4666-0294-6]

4. Toyota recalled some vehicles in 2010 for a bug in the anti-lock brake software [https://instituteforpr.org/wp-content/uploads/JFGRA-InfoTrend-case-study-ver-2.pdf]

5. Because of an error in the radiation therapy machine Therac-25, some patients were exposed to an overdose of radiation and six of them died [http://sunnyday.mit.edu/papers/therac.p

Generally software testing requires up to 50% of software development costs and in case of safety-critical applications these costs are even higher. This can be reduced by automatization of test execution or test generation. An exhaustive testing is not feasible in practice and the input domain may be infinite (e.g. avionic system fed with input sequences). Since exhaustive testing is not feasible, we have to select a subset of inputs and that is why a test generation process for CPS is an active topic in an academic environment. [TODO citation]

White box testing considers the inner structure of a testing subject. The internal structure/design/implementation of the item being tested is known to the tester. This tecnique is widely used in software testing where the tester chooses inputs to exercise paths through the code and determines the appropriate outputs. Programming know-how and the implementation knowledge is essential. This method is named so because the software program, in the eyes of the tester, is like a white/transparent box; inside which one clearly sees.

Black box testing does not use information of the internal structure. It observes the testing item only through its interface and considers only the requirements of the system. Tests are only derived from the requirements. Model-Based testing (MBT) is a kind of black box testing. Inputs are applied to the testing item and the output is observed. The corectness of the output is checked with respect to the given expected output. When we are speaking about the safety requirements that describe the expected output, we refer to it as falsification. As stated above these methods attemp to generate counterexamples that disprove or falsify safety requirements, thus disprove the expected output.

We can identify the following families of modeling notations:

2.3.0.1 Transition-based notations

They describe transitions between states of the system. E.g. Finite-State Machine (FSM), Unified Modeling Language (UML) state machines, Statecharts, Labeled Transition Systems, etc.

2.3.0.2 Input-domain notations

They describe the inputs and their constraints. E.g. combinatorial testing, feature modeling.

2.3.0.3 Pre/post notations

They describe the system by means of some variables and operations. The models specify pre-conditions that must be satisfied before an operation and post-conditions that must be guaranteed after the operation execution. E.g. Java Modeling Language, Spec#, etc.

2.3.0.4 History-based notations

They describe the allowable traces of the system behavior over time. They are good for describing the interactions among components. E.g. message-sequence charts, UML sequence diagrams, temporal logics.

2.3.0.5 Functional notations

They describe the model as a set of mathematical functions.

2.3.0.6 Operational notations

They describe system as a set of executing processes. E.g. process algebras, Petri nets, ASMs.

2.3.0.7 Stochastic notations

They describe a probabilistic model of the inputs of the system.

2.3.0.8 Data-flow notations

They model the flow of data (rather than control flow)

The model itself could contain errors. Test generation depends on the notation used. Some approaches are based on:

1. exploration (simulation) of the model
2. logical solvers (Boolean satisfiability (SAT) problem solvers)
3. model checkers (SPIN, NuSMV, etc.) which check more complicated temporal properties

2.3.1 Model checking

Model checking is an automated formal verification technique. It aims to discover whether an abstract description $\mathcal{M}$ of a system satisfies a property φ, i.e.,

$$\mathcal{M} \models \varphi \tag{2.1}$$

2.4 Safety requirements

Test requirements can be generated from formal models. TODO collect all the examples of the requirements plus from the course at Matfyz the requirements collection and specification and all engineering process around the ccreation of a new model of a system.

2.5 Metric Temporal Logic

2.6 Verification process

The evaluation of whether or not a product, service or system complies with a regulation, requirement, specification, or imposed condition. [Barry W. Boehm (Ed.). 1989. Software Risk Management. IEEE Press, Piscataway, NJ, USA.]

2.7 Previous Results and Related Work

Chapter 3

Overview of Our Approach

The sample Fig. 3.1 shows ...

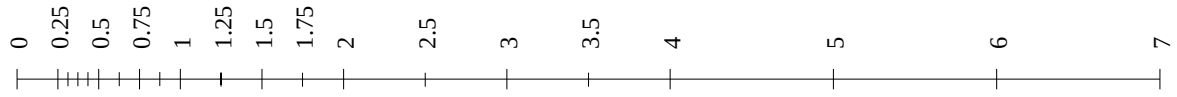

Figure 3.1: Distribution of the floating point numbers. This figure shows a distribution of a sample floating point number set with a precision $t = 3$, and $e_{min} = -1$ and $e_{max} = 3$.

There are two basic floating point data types , as defined by the IEEE 754-2008 [1] standard, are shown in Tab. 3.1.

	Sign [b]	**Exponent** [b]	**Mantissa** [b]	**Prec.** [dig]	**Total** [b]
binary32	1	8	24	8	32
binary64	1	11	53	16	64

Table 3.1: Basic floating point data types.

Chapter 4

Prelimitary Results

4.1 Prelimitary Result 1

4.2 Prelimitary Result 2

4.3 Prelimitary Result 3

4.4 Discussion

4.5 Summary

Chapter 5

Conclusions

5.1 Proposed Doctoral Thesis

Title of the thesis:
TITLE
The author of the report suggests to present the following:

5.1.1 Topic 1

5.1.2 Topic 2

5.1.3 Topic 3

Bibliography

[1] IEEE Computer Society Standards Committee. *IEEE Standard for Floating-Point Arithmetic*. ANSI/IEEE STD 754-2008. The Institute of Electrical and Electronics Engineers, Inc., 2008.

Publications of the Author

[A.1] R. Gortz, F. Tölökö. *On the Carpathian Castle*. Transylvanian Journal of ..., Werst, Romania, 2010.

The paper has been cited in:

- Š. Nováků. *Carpathian Castle Revealed*, International Symposium on Carpathian Legends, 1:319–323, 2010.

[A.2] R. Gortz *Another publication*. 36$^{\text{th}}$ International Conference on ..., pp. 19-24, Štrbské pleso, Slovak Republic, 2010.

Appendix A

...

A.1 ...

Section not in the Table of Contents