Combining Model Inference and Passive Testing in the Same Framework to Test Industrial Systems

Sebastien Salva, University Clermont Auvergne, Aubriere, France
William Durand, Michelin/LIMOS/TailorDev, Clermont-Ferrand, France

ABSTRACT

Many software engineering approaches often rely on formal models to automate some steps of the software life cycle, particularly the testing phase. Even though automation sounds attractive, writing models is usually a tedious and error-prone task. In addition, with industrial software systems, models are often not up-to-date. Hence, testing these systems becomes problematic. In this context, this article proposes a framework called Autofunk to test production systems by combining two approaches: model generation and passive testing. Given a large set of events collected from a production system, Autofunk combines an expert system, formal models and machine learning to infer symbolic models while preventing over-generalisation. Afterwards, these models are considered to passively test whether another system is conforming to the models. As the generated models do not express all the possible behaviours that should happen, we define conformance with four specialised implementation relations.

KEYWORDS

Industrial Systems, Model Generation, Passive Testing, Symbolic Models

INTRODUCTION

This paper tackles the problem of testing production systems such as those of our industrial partner Michelin, one of the three largest tire manufacturers in the world. A production system is defined as a set of production machines controlled by software, in a factory. Such systems are composed of heterogeneous devices, interconnected with specialised. Testing them is often performed manually with simulations to replicate human operations and to not damage real devices. This testing phase usually requires a long period, from some weeks up to several months.

Passive testing is an approach that can partially automate this stage and shorten its deadline. Generally speaking, a passive tester (a.k.a. observer) collects observations from the system and aims at checking if its behaviour meets requirements expressed in a model. Testing can be performed in either online or offline mode. Online passive testing means that sequences of observations, called traces, are computed and analysed on-the-fly for the detection of defects; in offline mode, traces are collected.

DOI: 10.4018/IJISMD.2017010103

Copyright © 2017, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.
and analysed later. However, passive testing suffers from a common issue: we need a specification (models or properties). And writing a specification is known as a long and error-prone task.

Model inference is a research field, which brings appealing concepts to bypass this issue. It proposes a set of techniques that infer models describing how a system behaves by analysing system executions. A model, inferred from an initial production system, could help in the test of a new or updated one. Here comes the context proposed by our partner Michelin who wishes a way to automate the testing of new or updated systems, but without having models. To cope with this problematic, we have chosen to devise a framework, called Autofunk (for Automatic Functional model inference), which combines model inference and passive testing. This paper presents this framework, i.e., the theoretical background that we considered and preliminary results.

**Context**

Michelin is a worldwide tire manufacturer and designs most of its factories, production systems, and related software. Like many other industrial companies, Michelin follows the Computer Integrated Manufacturing (CIM, (Wu, YuShun, & Deyun, 2007)) approach, using computers and software to control the entire manufacturing process and acquire data. The CIM approach segments the manufacturing process and production strategies into several hierarchical levels: CIM1 is the device level, CIM2 includes all the applications that monitor and control devices. Levels 3 and 4 focus on the factory management.

A factory includes workshops, each devoted to a step of the tire building process, e.g., tire assembling (assembling the components onto a tire drum) or curing (applying pressure on assembled tires in molds to give their final shapes). A workshop gathers devices, branch points, conveyor belts and human operators that perform specific actions (removal of products to assess their qualities, etc.). A workshop is controlled by a set of CIM2 applications (except for the operators): every order (move, stop, change state, etc.), product modifications or alerts passing among industrial devices and software are materialised with messages that we call *production events*. These applications are continuously updated and sometimes replaced, for instance when the physical configuration of the workshop is modified, when new machines are added, when bugs are detected, or when it is decided that applications are too old and are no more maintainable.

As depicted in Figure 1, at the workshop level, we observe a continuous stream of products following assembly lines from specific *entry points* to *exit points*, i.e., where products go to reach the next step of the manufacturing process. Some factories produce over 30,000 tires a day, resulting in thousands of production events at the CIM2 level, which are collected and persisted in databases.

Production systems are tested when they are set up and every time they are updated with new applications, parameters, devices, etc. We do not focus on the device level here (CIM1), but on the CIM2 level (although physical devices are tested too). For readability, when we refer to production systems in the remainder of the paper, we actually focus on the software of the CIM2 level, which acquires and sends production events to the devices.

For testing a production system, Michelin engineers firstly build simulations by mocking most of the devices. Then, they run hundreds of scenarios composed of production events, collect all the observable production events and manually inspect them to detect abnormal behaviours. As simulations are not sufficient to run all the possible scenarios, production events are again collected when the system is running, and events are scrutinised every time an issue is detected. This manual testing process can be followed for a long period, depending on the modifications made on the system (up to 6 months). Michelin wished to partially automate this phase to:

- Detect more quickly potential regressions when CIM2 applications are modified or when devices are replaced to ensure that they are interoperable with the current application versions;
Related Content

Line-of-Sight Detection Using Center of Gravity with Pixel Number Variation

Resource Scheduling Techniques in Utility Computing: A Survey
[www.igi-global.com/article/resource-scheduling-techniques-in-utility-computing/114606?camid=4v1a](www.igi-global.com/article/resource-scheduling-techniques-in-utility-computing/114606?camid=4v1a)
A Semantic Web-Based Information Integration Approach for an Agent-Based Electronic Market
[www.igi-global.com/chapter/semantic-web-based-information-integration/29458?camid=4v1a](www.igi-global.com/chapter/semantic-web-based-information-integration/29458?camid=4v1a)

The Last Line of Defense: A Comparison of Windows and Linux Authentication and Authorization Features
[www.igi-global.com/chapter/last-line-defense/37944?camid=4v1a](www.igi-global.com/chapter/last-line-defense/37944?camid=4v1a)