Exploratory Workshop Scheme

Scientific Review Group for Physical and Engineering Sciences
Scientific Review Group for the Humanities

ESF Exploratory Workshop on

Combining Learning and Symbolic Analysis for Software Documentation and Mastering Change

Darmstadt (Germany), September 10-11, 2014

Convened by:
Ulf Brefeld and Reiner Hähnle

SCIENTIFIC REPORT

1. Summary

Automatic software documentation and validation is becoming a major challenge in software engineering as programs and systems often distribute across millions of lines of code that cannot be analysed and verified manually. The objective of this workshop was to bring together researchers from software engineering, natural language processing, and machine learning, and discuss commonalities and differences in challenges faced, survey the different approaches, and provide a platform to present and learn about some of the most cutting edge research in this area. Moreover, the workshop was intended to serve as a forum for exchanging ideas to develop a new common understanding of the involved problems and to agree on future actions to pursue them.

The workshop on “Combining Learning and Symbolic Analysis for Software Documentation and Mastering Change” was held at TU Darmstadt, Germany, over 1.5 days in September 2014. Participation numbered 15 people from 9 countries. Even though 1.5 days are clearly not enough to establish a deep understanding of all involved scientific and practical challenges, the workshop has been considered a huge success by all participants. The group discussed ways to bridge the gap between the areas, elected organisers for a follow-up event, and will also share ideas and materials in the near future to further evaluate promising ways to cooperate. In the following, we briefly sketch the concrete outcomes and findings of the workshop.

2. Scientific content of the event

In both, static analysis of software and in automata learning, the last decade has brought major advances that most researchers outside the specific areas are unaware of: for example, there exist software modelling languages, such as ABS (http://www-hats-project.eu) that permit fully automatic resource analysis, behavioural verification, and test case generation for concurrent, industrial scale software. Functional verification of programs written in industrial languages such as Java or C is possible.

In automata learning, active approaches alternatively issue membership and equivalence queries to construct and validate hypotheses. Leveraging black-box inference to register automata has been a significant leap towards realistic applications. A recent active learning schema for Register Mealy Machines bridges the gap to data-independent processing which is relevant for protocol and interface programs. Other work studies active learning for test drivers.

From a machine learning point of view, kernel methods for learning automata have been proposed. The idea is to map accepted and rejected inputs to some high-dimensional space where a linear classifier is trained to separate the classes. Grammar and automata induction has also been studied in the context of software engineering, for instance to learn programming languages that are usually context-free, non-stochastic grammars. It is possible to infer context-free grammars from positive examples for programming dialects where the original programming language is used as prior knowledge. Genetic algorithms have been proposed to learn grammars for small domain specific languages, such as one that validates UML class diagrams from use cases.

In summary, one can observe that static analysis techniques, specifically deductive approaches, scale to industrial problems and are largely automatic (i.e., work without supervision). Recently, such techniques tend to be used in connection with modelling languages. At the same time the target formalisms of automata learning become more expressive and active learning is essential for performance. This leads to a historically unprecedented narrowing of the methodological and language gap between symbolic and learning-based approaches. The central role of the workshop will be to identify opportunities, where joint research can lead to synergies that, ultimately, form the basis for breakthroughs. We illustrate the vast possibilities by two examples:

- Active learning algorithms typically rely on two kinds of queries to improve/ assure convergence: membership queries and equivalence queries. The former are simple tests, but the latter are much harder and often left out for performance reasons. But this is exactly, where a suitable static analysis can help, even if it is only approximative.

- Formal software models, which are the basis of symbolic approaches, must usually be expensively built manually and even then, are often incomplete and inaccurate. This process could be greatly speeded up and made more reliable by modern learning techniques.

It became evident during the workshop that there is already interesting work under way that connects software analysis with automata learning and machine translation with machine learning. From an abstract point of view programs and automata are both state transition systems, however, it became very clear that the gap between modern programming languages and the kind of automata considered in the learning community is considerable. It is not impossible to bridge this gap (and first promising work exists in the form of learning "interface programs"), but it will be a challenge. All participants agreed that there are vast possibilities in this: static software analysis methods are symbolic and rely heavily on deductive reasoning, while learning methods typically are statistical/combinatorial. So we have a set of very complementary, highly developed approaches. Can we employ learning techniques to incorporate runtime information into static software analysis or to make it rely less on external specifications? Can we use deduction and symbolic reasoning to make learning more efficient and the learned objects more expressive? Can ideas and concepts, such as, for example, Transfer Learning, be usefully applied in the world of software?

As to natural language, the most promising starting point to bridge the gap between disciplines is to better understand the differences between natural and artificial languages. This not only allows for an easier transfer of results developed in NLP to the software domain but may also lead to new problem settings, as many prior assumptions will be very different when dealing with software. E.g., protocols and code are supposed to be noise-free and finite languages with a very limited set of atomic words compared to natural languages. Utilising these restrictions may be the key to developing new kinds of automata learning algorithms. In this way, machine learning could provide the tool bridge between programming languages and documentation in natural language.

3. Assessment of the results & contribution to the future direction of the field

A key observation during the workshop was that the automata learning community historically has been much concerned with theoretical questions centring on learnability, completeness, and complexity. The software analysis and verification community, however, is now mainly driven by empirical challenges arising from attempts to scale the technology towards industrial applicability. This led to scalable and robust software engineering tools in the last decade and helped to get formal verification out of its theoretical corner. Several workshop participants from the learning community agreed that the demand for high performance coming from applications in software analysis could be a very positive stimulus for applied research in machine learning (as this is the case already with applications coming from the life sciences).

As a next step, the workshop participants plan a follow-up event at Dagstuhl that will serve as a platform to evaluate collaborations and efforts that have been initiated today and to start preparing more ambitious proposals based on the ideas sketched above. Concretely, Karl Meinke (KTH Stockholm), Anssi Yli-Jyra (U Helsinki), Amel Bennaceur (The Open University), plus a researcher from North America (to be confirmed) will submit a proposal for a Dagstuhl perspectives workshop to be held in 2015. The theme of the workshop will be in the direction of combining statistical and symbolic (learning) approaches with a clear focus on applications in software documentation.

The feedback we obtained from the participants during and after the workshop was invariably positive and we believe the workshop was a big success. We really enjoyed organising the event and we would like to warmly thank ESF for supporting this event as it allowed us to have so many exceptional invited speakers.

4. Final programme

The 1.5 days workshop was structured in three parts. On the first day, we reserved the morning for tutorial introductions by leading researchers in each area to familiarise everyone with the terminology, research methodologies, etc., of the different communities. The afternoon of the first day and the morning of the second day is spent with a selection of shorter presentations that report about the state of art in each area, the capabilities and limitations of tools, etc. The remaining time on the second day was used to collect topics for joint research projects and discussions of follow-up activities. To facilitate collaborations, we had a joint social event with room for informal discussions on the evening of the first day. Participants arrived on the evening before the workshop, that we used for an informal welcome dinner and get together.

<u>Tuesday, September 9</u>
19:00 Welcome reception and get together

<u>Wednesday, September 10</u>
09:00 Welcome
09:15 Reiner Hähnle, Deductive Verification
10:00 Karl Meinke, Learning-based Testing
10:45 Break
11:15 Bernhard Steffen, Active Automata Learning
12:00 Colin de La Higuera, Probabilistic finite automata: using them and learning them
12:45 Lunch
14:30 Stefan Jähnichen, ESF Rapporteur
14:45 Session 1
    Andreas Maletti, Rule Extraction for Multi Bottom-up Tree Transducers
    Anssi Yli Jyra, Decision Tree Learning and Transducer Induction
    Amaury Habrard, Spectral learning for (probabilistic) grammatical inference
16:00 Break
16:30 Session 2
    Paola Inverardi, Producing Correct Software by Integration
    Einar Broch Johnsen, Model-Based Analysis of Resource-Aware Virtualized Services

    Ina Schaefer, Deductive Verification of Software Product Lines
17:45 End of day 1

Thursday, September 11

09:00 Session 3
    Andrzej Wąsowski, Logic-based Synthesis of Feature Models
    Ricard Gavaldà, Spectral learning of probabilistic automata and related models
    Amel Bennaceur, The Role of Machine Learning in Achieving Interoperability
10:15 Break
10:45 Session 4
    Falk Howar, Interface Generation through Static, Dynamic, and Symbolic Analysis
    Ulf Brefeld, Kernels for Automata
11:35 Break
11:50 Time for discussions

13:00 End of day 2

5. Participants

In total, there have been 15 participants at the workshop (3 female, 12 male), six of which are considered young researchers. By the time of the workshop, participants came from Germany (6), France (2), Norway (1), Spain (1), Italy (1), Denmark (1), and Finland (1), United Kingdom (1), and Sweden (1). The following colleagues participated in the workshop.

- Amel Bennaceur, The Open University [slides]
- Ulf Brefeld, Technische Universität Darmstadt [slides]
- Einar Broch Johnsen, University of Oslo [slides]
- Colin de La Higuera, Université de Nantes [slides]
- Ricard Gavaldà, Universitat Politècnica de Catalunya [slides]
- Amaury Habrard, University Jean Monnet of Saint-Etienne [slides]
- Reiner Hähnle, Technische Universität Darmstadt [slides]
- Falk Howar, TU Clausthal [slides]
- Paola Inverdardi, Università degli Studi dell'Aquila [slides]
- Andreas Maletti, Universität Stuttgart/Universität Leipzig [slides]
- Karl Meinke, KTH Stockholm [slides]
- Ina Schaefer, Technische Universität Braunschweig [slides]
- Bernhard Steffen, Universität Dortmund [slides]
- Andrzej Wasowski, IT University [slides]
- Anssi Yli Jyra, University of Helsinki [slides]