Bachelor and Master Theses

Specialization: All Bachelor and Master Specializations

Remarks:

1. All theses must be written in English.
2. Usage of Latex is mandatory.
3. In order to work with me on the following topics, you have to show disponibility on tackling research problems. Typically, students working with me publish their theses at workshops, conferences or students' symposia.

Nr	Topic	Observations
1.	Symbolic Automata: Theory and Applications (1 thesis)	Classic automata theory builds on the assumption that the alphabet is finite. For practical applications (e.g. XML processing, program trace analysis) this is inconvenient because they use values for individual symbols that are typically drawn from an infinite domain. Even when the alphabet is finite, classic automata may sometimes be a bad choice: for example, a deterministic finite automaton modelling a language over the UTF16 alphabet requires 2^16 transitions out of each state! *Symbolic Finite Automata (SFA)* are finite state automata in which the alphabet is given by a Boolean algebra that may have an infinite domain, and transitions are labeled with first-order predicates over such algebra. *SFA* are more expressive than deterministic finite automata, however, are closed under Boolean operations and admit decidable equivalence. Moreover, for large alphabets SFA outperforms their classic counterpart. The aim of these theses is to: Present theory of SFA: definitions, examples, comparisons to classical finite automata Implement certain algorithms related to *Decision Problems and Closure Properties*, *Learning*, *Applications*. **Difficulty**: medium **Requirements**: *Theory*: Formal Languages and Automata Theory (notions from 1st year lecture); *Programming*: Python; **Resources**: http://pages.cs.wisc.edu/~loris/symbolicautomata.html
2.	Benchmark problems for the constraints satisfaction problems (CSP) repository (1 thesis)	The project involves preparing and submitting existing constraints satisfaction problems to the constraints satisfaction problems repository. **Difficulty**: medium **Requirements**: *Programming*: Python; *Math*: computational logic, in particular the notions taught in the lecture Logic for Computer Science and/or Formal Methods in Software Development. **Resources**: http://www.csplib.org
3.	Predicting the fastest method for constrained satisfaction/ optimization problems (2 theses)	Constrained optimization/satisfaction problems can be encoded in different logical theories (propositional logic, integers, reals, or combinations). The encoding influences the running time of the algorithms/tools solving the problem. We propose two theses: • One investigates the best algorithm/tool, from the computational time point of view, for solving the problem. • The other studies, implements, and performs experiments with incremental techniques for SAT/SMT solving for speeding up the existing algorithms/tools. **Difficulty**: high **Requirements**: *Programming*: Python; *Math*: computational logic, in particular the notions taught in the lecture Logic for Computer Science and/or Formal Methods in Software Development. **Resources**: (1) Influence of Variables Encoding and Symmetry Breaking on the

		Performance of Optimization Modulo Theories Tools Applied to Cloud Resource Selection - Madalina Erascu, Flavia Micota, Daniela Zaharie
4.	Binarized Neural Networks. Training and Verification (2 theses)	Deep learning is everywhere. It has been shown its practical application in a variety of fields, image recognition, natural language processing, recommendation systems, autonomous driving, just to name a few. Deep learning algorithms are mainly used as a black-box and hence difficult to debug. In fact, the main criticisms to deep learning algorithms are *uncertainty* and unexpected behavior on *adversarial examples*. When we talk about safety-critical systems, it is important that correctness guarantees exist. This leads to the application of *formal verification* to deep neural networks (DNNs), that is, given a DNN and a specification, is there a proof that the DNN satisfies the specification for all inputs? Not surprisingly, the main challenge of applying formal methods to the verification of DNNs is *scalability*. This is because verification is a non-trivial problem: DNNs are large (high number of neurons and layers) and involve activation functions which are non-linear and non-convex. These make the problem NP-complete. We offer three theses for studying three different verification approaches. The theses should contain a comprehensive state-of-the-art as well demo with at least one of the tools from the state-of-the-art. The demo will ensure reproducibility of the results obtained by state-of-the-art. **Difficulty**: high **Requirements**: *Programming*: Python; *Math*: Logic, linear algebra and statistics **Resources**: (1) Verifying Properties of Binarized Deep Neural Networks – N. Narodytska et all, AAAI-18 (2) Formal Analysis of Deep Binarized Neural Networks – N. Narodytska, IJCAI-18
5.	Invariant generation (1 thesis)	Program analysis requires the generation of program properties expressing conditions to hold at intermediate program locations. When it comes to programs with loops, these properties are typically expressed as loop invariants. **Difficulty**: high **Requirements**: *Programming*: Python; *Computer Science*: Algorithms and Data Structures **Resources**: (1) Dafny (https://github.com/dafny-lang/dafny). (2) *Assigning meaning to programs* by Robert Floyd.
6.	Synthesis of optimal numerical algorithms (1 thesis)	Program synthesis is the automatic construction of software that provably satisfies a given specification (input and output condition). Given a specification of what a program should do, the synthesizer generates an implementation that satisfies this specification. The aim of the thesis is to study the possibility of the synthesis of algorithms (e.g. reciprocal, square root, reciprocal square root of numbers) suitable for hardware implementations. The main characteristic of these algorithms is that they do not contain the division operation, which is expensive. The experiments will be conducted in Mathematica. **Difficulty**: high **Requirements**: *Programming*: Mathematica; *Math*: computational logic **Resources**: (1) Madalina Erascu, Hoon Hong: Real quantifier elimination for the synthesis of optimal numerical algorithms (Case study: Square root computation). J. Symb. Comput. 75: 110-126 (2016)

7.	*Comparative study of formal analysis methods for biological networks involved in the development of resistance of microorganisms to antibiotics.* (1 thesis).	Formal analysis of biological networks has the potential of developing reliable and efficient methods and tools for patterns (motifs) identification which could help in *understanding the mechanisms behind complex phenomena* (e.g. antimicrobial resistance). **Difficulty**: high **Requirements**: *Programming*: Python; *Math*: basic abstract algebra, computational logic, in particular the notions taught in the lecture Formal Methods in Software Development. *Interest* in bioinformatics. **Resources**: (1) Formal Analysis of Network Motifs, Hillel Kugler, Sara-Jane Dunn, Boyan Yordano, https://www.biorxiv.org/content/10.1101/347500v1.full.pdf
8.	*Investigation of* symmetry breaking methods for *formal analysis methods for biological networks involved in the development of resistance of microorganisms to antibiotics.* (1 thesis).	Formal analysis of biological networks has the potential of developing reliable and efficient methods and tools for patterns (motifs) identification which could help in *understanding the mechanisms behind complex phenomena* (e.g. antimicrobial resistance). As it is an intractable task, we aim to study the usability of symmetry breaking methods for speeding it up. **Difficulty**: high **Requirements**: *Programming*: Python; *Math*: basic abstract algebra, computational logic, in particular the notions taught in the lecture Formal Methods in Software Development. *Interest* in bioinformatics. **Resources**: (1) Formal Analysis of Network Motifs, Hillel Kugler, Sara-Jane Dunn, Boyan Yordano, https://www.biorxiv.org/content/10.1101/347500v1.full.pdf