A Hierarchy of Tractable Subclasses for SAT and Counting SAT Problems

Stefan Andrei
Lamar University
Department of Computer Science
Beaumont, USA
Stefan.Andrei@lamar.edu

Gheorghe Grigoras
Cuza University of Iasi
Department of Computer Science
Romania
grigoras@infoiasi.ro

Martin Rinard
Massachusetts Institute of Technology
Cambridge, USA
rinard@lcs.mit.edu

Roland Hock Chuan Yap
National University of Singapore
Singapore
ryap@comp.nus.edu.sg

Abstract - Finding subclasses of formulæ for which the SAT problem can be solved in polynomial time has been an important problem in computer science. We present a new hierarchy of propositional formulæ subclasses for which the SAT and counting SAT problems can be solved in polynomial time. Our tractable subclasses are those propositional formulæ in conjunctive normal form where any set of k + 1 clauses are related, i.e., there exists at least one literal in one clause that appears negated in another clause of the considered set of k + 1 clauses. We say this subclass of formulæ is of rank k and it is different from previously known subclasses that are solvable in polynomial time. This is an improvement over the SAT Dichotomy Theorem and the counting SAT Dichotomy Theorem, since our subclass can be moved out from the NP-complete class to the P class. The membership problem for this new subclass can be solved in $O(n \cdot l^{k+1})$, where n, l and k are the number of variables, clauses and the rank ($1 \leq k \leq l - 1$), respectively. We give an efficient algorithm to approximate the number of assignments for any arbitrary conjunctive normal form propositional formula by an upper bound.

A Calculus for Imperative Programs: Formalization and Implementation

Madalina Erascu and Tudor Jebelean
Research Institute for Symbolic Computation
Johannes Kepler University, Linz, Austria
f merascu, tjebeleag @risc.uni-linz.ac.at

Abstract—As an extension of our previous work on imperative program verification, we present a formalism for handling the total correctness of While loops in imperative programs, consisting in functional based definitions of the verification conditions for both partial correctness and for termination. A specific feature of our approach is the generation of verification conditions as first order formulae, including the termination condition which is expressed as an induction principle. The method is implemented in the frame of the Theorema system and is illustrated on several examples.

On Herbrand-like Theorems for Cut-free Modal Sequent Logics

Alexander Lyaletski
Faculty of Cybernetics,
Kyiv National Taras Shevchenko University,
Kyiv, Ukraine
Email: forlav@bigmir.net

Abstract—The purpose of this research is to extend the author's results from [1] concerning Herbrand theorems for classical and intuitionistic logics to the classical and intuitionistic modal sequent logics investigated in [2], where their validity and completeness were proved. It was found that the technique developed in [1] and [2] can be applied with satisfactory for proving Herbrand theorems for the logics under consideration.

Formal proof of theorems on genetic regulatory networks

Maxime Denes and Benjamin Lesage
Universite de Rennes

Yves Bertot
INRIA Sophia Antipolis Mediterranee

Adrien Richard
CNRS I3S (UMR 6070), Sophia Antipolis

Abstract—We describe the formal verification of two theorems of theoretical biology. These theorems concern genetic regulatory networks: they gives, in a discrete modeling framework, relations between the topology and the dynamics of these biological networks. In the considered discrete modeling framework, the dynamics is described by a transition graph, where vertices are vectors indicating the expression level of each gene, and where edges represent the evolution of these expression levels. The topology is also described by a graph, called interaction graph, where vertices are genes and where edges correspond to influences between genes. The two results we formalize show that circuits of some kind must be present in the interaction graph if some behaviors are possible in the transition graph. This work was performed using the `ssreflect` extension of the Coq system.

Using a fUML Action Language to construct UML models

C.-L. Lazar, I. Lazar, B. Parv, S. Motogna and I.-G. Czibula
Department of Computer Science
Babes-Bolyai University
Cluj-Napoca, Romania
Email: { clazar, ilazar, bparv, smotogna, istvanc} @cs.ubbcluj.ro

Abstract—In this paper we introduce a fUML based action language and describe its concrete syntax. The action language uses only elements allowed by the fUML standard for its abstract syntax. The concrete syntax resembles the syntax of existing structured programming languages, with some elements inspired by OCL. The action language is part of the ComDeValCo framework and we are using it to model the functionality of the components, to simulate the execution of the models and to test the models.

Simplification and Generalization in CIRC

Eugen-Ioan Goriac, Georgiana Caltais, Dorel Lucanu
{egoriac, gcaltais, dlucanu}@info.uaic.ro
Faculty of Computer Science
Alexandru Ioan Cuza University
Iasi, Romania

Abstract—CIRC is an automated theorem prover based on the circular conduction principle. The tool is used for the verification of programs, behavioral equivalence checking, and proving properties over infinite data structures. In this paper we present two extensions of CIRC that handle the case when the prover indicates an infinite execution for a certain goal. The first extension involves goal simplification rules and a procedure for checking that the new execution is indeed a proof, while the second one refers to finding and proving a generalization of the goal. Each of the extensions is presented based on a case study: Binary Process Algebra (BPA) for checking the proof correctness and Streams for using generalization.

Physarum Spatial Logic

Andrew Schumann
Department of Philosophy and Science Methodology,
Belarusian State University
Minsk, Belarus
e-mail: Andrew.Schumann@gmail.com

Andy Adamatzky
Department of Computer Science,
UWE
Bristol, UK
e-mail: andrew.adamatzky@uwe.ac.uk

Abstract— Plasmodium of Physarum polycephalum is a large single cell capable for distributed sensing, information processing, decentralized decision-making and collective action. In the paper we interpret basic features of the plasmodium foraging behavior in terms of process calculus and spatial logic and show that this behavior could be regarded as one of the natural implementations of spatial logic without modal operators.

Sparrow: Towards a New Multi-Paradigm Language

Lucian Radu Teodorescu, Alin Suciu and Rodica Potolea

Abstract - We present here the design of a new computer language, named Sparrow, that aims to integrate flexibility, efficiency and naturalness. The language is based on C++ as this has good support for efficiency and flexibility. The central feature of the language is static metaprogramming. In Sparrow, programs can be written with high-level concepts that are translated through static metaprogramming into efficient code. The compiler can also be extended with metaprograms and plugins. Moreover, we show how static metaprogramming can be used to implement new programming paradigms in the language.