FORMAL METHODS AND ALGORITHMS

Presenters: Patrick Baillot / Simon Bliudze / Raphaël Monat / Charles Paperman

Formal methods and algorithms

Description

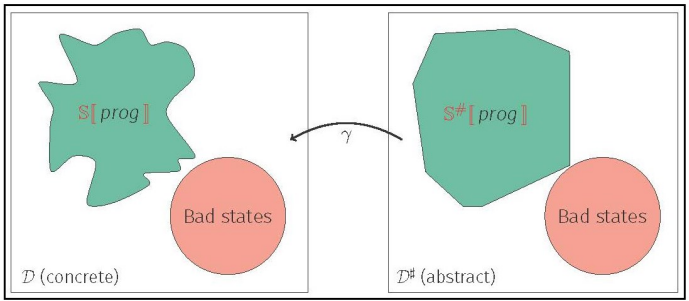
This area focuses on the design and application of formal methods and mathematical methods in computer science and modelling. We are interested in the design and analysis of algorithms and software systems for which we prove properties such as correctness, complexity, safety, responsiveness, etc. This methodological approach is applied to various fields in the unit.

'Emblematic' projects

- ANR: PRCI Symbiont, PRCE Corteva, JCJC Sywext, SATAS, Headwork, Delta, CQFD, PERSICON, SmartCloud, ADAPT, franco-allemand TinyPART
- Exploratory action Inria, AVoCat
- CPER: Data Alloy@Scale, Data Commode
- I-Site ULNE BRiCoS
- European COST action CA20111 EuroProofNet
- ICPEI Next generation cloud infrastructure and service
- PEPR Cloud TARANIS

Teams concerned

- ♠ ***GT CO2 (Commande et Calcul Scientifique):*** CFHP
- ♠ ***GT DatInG (Data Intelligence Group):*** MAGNET
- ♠ ***GT GL (Génie logiciel):*** Spirals
- ♠ ***GT I2C (Interaction et Intelligence Collective):*** Algomus
- ♠ ***GT MSV (Modélisation pour les Sciences du Vivant):*** BioComputing, Bonsai
- ♠ ***GT SISE (Systèmes Informatiques Sûrs et Efficaces):*** LINKS, SyCoMoRES, 2XS



Formal methods and algorithms

This area focuses on the design and application of formal methods and mathematical methods in computer science and modelling. We are interested in the design and analysis of algorithms and software systems for which we can prove properties such as correctness, complexity, safety and responsiveness, etc.

This involves the following steps:

(i) defining a formal representation of the object of study

(ii) developing analyses or algorithms based on these representations

(iii) demonstrating mathematical theorems on these analyses and algorithms

The formalisation in (i) can be based on notions such as automata, Petri nets, graphs, formal languages, computations, semantics, etc.

The analyses and algorithms in (ii) will draw on the fields of algorithms and data structures, complexity, static analysis, learning, etc.

Demonstrations in (iii) can be based on discrete mathematics, logic, mathematical analysis, algebra, etc. and can take the form of 'paper' proofs or computer-assisted proofs such as those developed with the Coq tool.

These methods are applied to a wide range of fields in the unit, such as critical systems, software that has to satisfy safety or performance constraints, modelling and simulation of biological systems, analysis of biological sequences, formal calculation, databases, learning or computer, music, etc. Certain approaches are applied in various contexts by several teams, for example static analysis applied to embedded software or biological systems, or the use of the Coq proof assistant for validated numerical computaion or for the certification of operating systems. One of the aims of this area will therefore be to highlight the similarities in methods between different teams, so that new collaborations and synergies can emerge.

Keywords :

- formal calculation
- certified calculation
- algorithmic
- data structures
- formal languages
- discrete mathematics
- semantics
- formal proof
- Boolean networks
- logic
- correct by construction
- static analysis
- abstract interpretation
- symbolic analysis
- complexity
- graphs
- automata