Descriptive Control Theory

Sicun Gao

(student & postdoc, 2007 to next Friday)

Thank you, Ed, for not throwing me out seven years ago when I first came to you with a polynomial-time algorithm for SAT, and for still giving me even harder problems with even more patience through these years.

There exists a theory **X** that provides the key design methodology of a broad class of software, which

- runs on 99% of the CPUs around us
- has produced most of the notorious bugs commonly used to motivate formal verification.

Should **X** be studied by formal methods?

There exists a theory **X** that

- has matured before "computer science"
- has experienced much difficulty in solving (what we now understand as*) NP-hard problems for long.

What's your suggestions to practitioners of **X**?

* [Gao et al. LICS'12, CADE'12, FMCAD'13, arXiv'14]

We need a

logical and computational “overhaul” of control theory

that sets the foundation for building

highly complex, reliable, and secure software controllers

in

nonlinear, hybrid, and safety-critical systems.

Descriptive control theory aims to study

- Computational complexity
- Automated reasoning
- Logical foundation

for all topics/results in control theory, with a focus on nonlinear and hybrid systems.

Approach:

First (or second) order logic over the reals with Type 2 computable functions [Gao et al. LICS'12]

- Encode control problems in $\mathcal{L}_{\mathbb{R}_{\mathcal{F}}}$
- Infer complexity from logical descriptions
- Automatically solve using decision procedures
- Mine formal proofs after solving

Descriptive Complexity

Complexity of bounded Lyapunov stability* is in $(\Pi_3^P)^C$

$$\forall^{[0,e]}\varepsilon \exists^{[0,\varepsilon]}\delta \forall^{[0,T]} t \forall^X x_0 \forall^X x_t$$

$$(||x_0|| < \delta \wedge x_t = \int_0^t f(s)ds + x_0) \rightarrow ||x_t|| < \varepsilon.$$

* We always talk about the delta-variation [Gao et al. arXiv'14]

Automated Solving

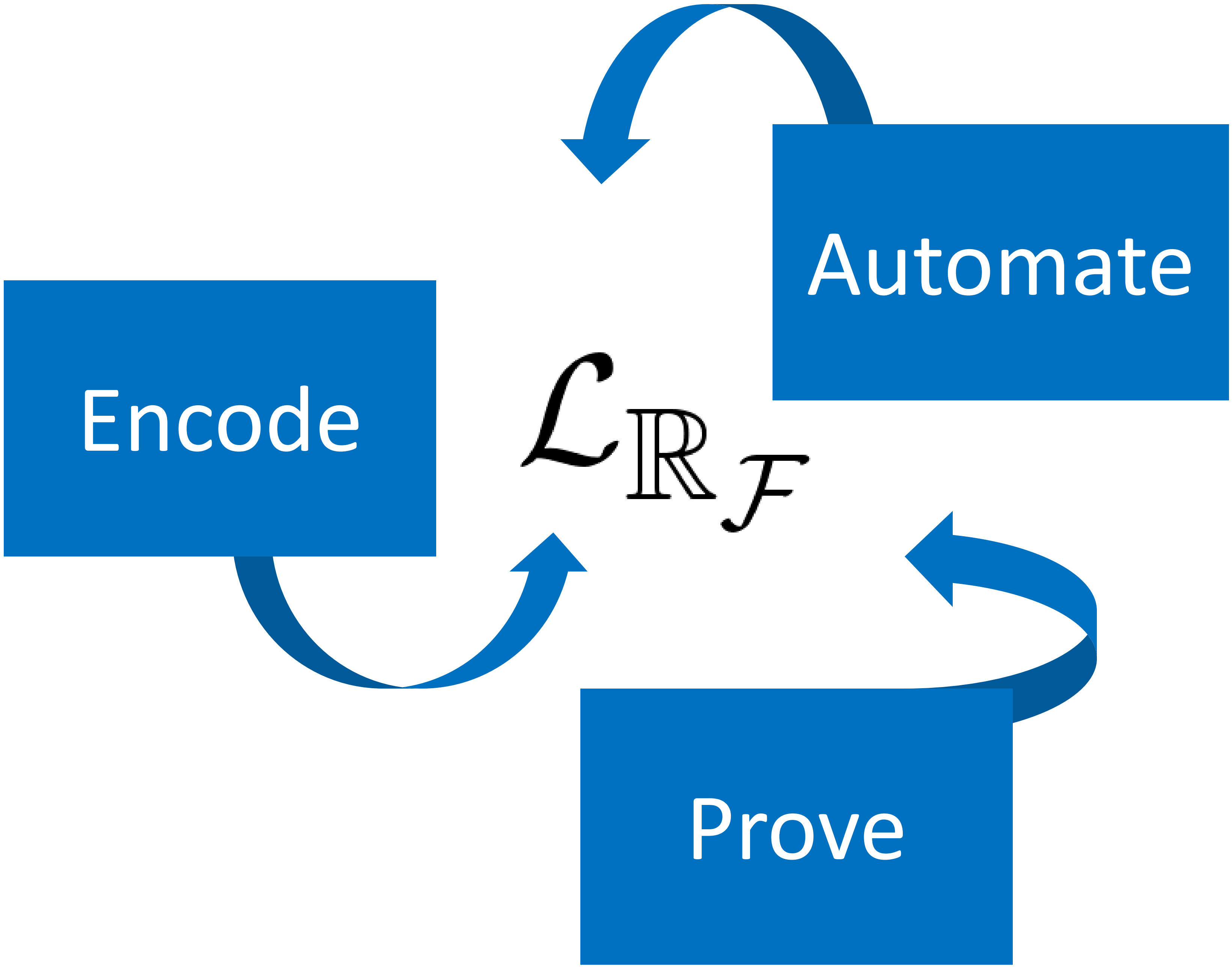
- No harder than SAT/QBF [Gao et al. CADE’12]
- Must combine symbolic and numerical algorithms
- Alternations of quantifiers

github.com/dreal

Formal Proofs

- Automatic proof generation from solvers
- Formalize proofs of classical theorems in control theory
- Combine them to produce full proofs of correctness

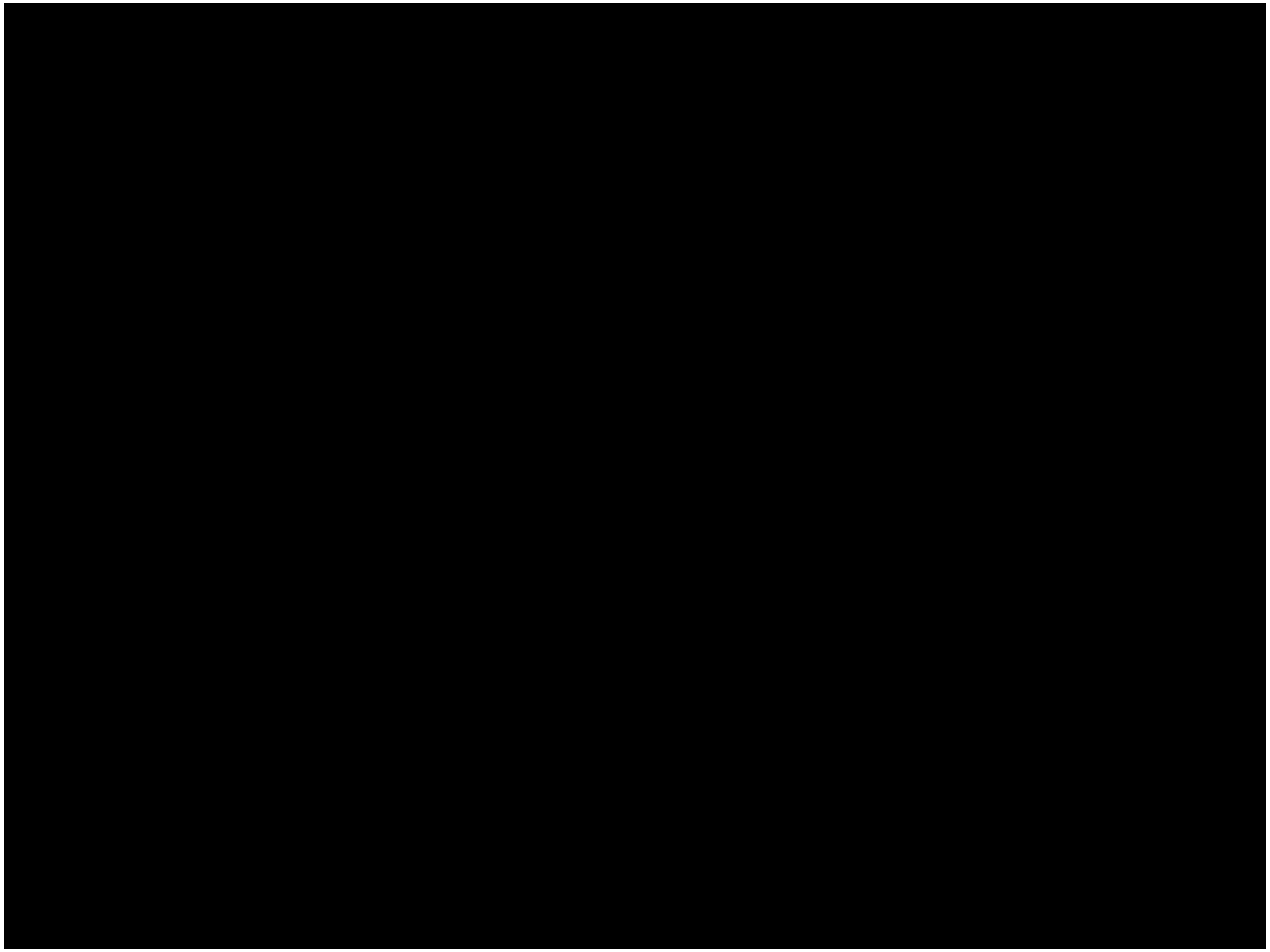
github.com/leanprover


Automate
Encode
$\mathcal{L}_{\mathbb{R}_{\mathcal{F}}}$
Prove

Goal of DCT:

Develop automated and proof-producing methods for solving nonlinear and hybrid control problems, whose complexity typically ranges from NP-hard to PSPACE-complete.

The core for a correctness-by-construction framework for building complex cyber-physical systems.



http://www.youtube.com/watch?v=3u9ZeCaDeic

In theory, this is (almost) not harder than SAT.

[Gao et al. LICS'12, CADE'12, FMCAD'13]

But can we really have control systems like this someday?

I don't know. Tons of things to do. But I'm always naively optimistic.

Is there any other way of doing it before good progress in DCT?

Unlikely.

Let’s see how things stand at Ed’s 80th birthday.