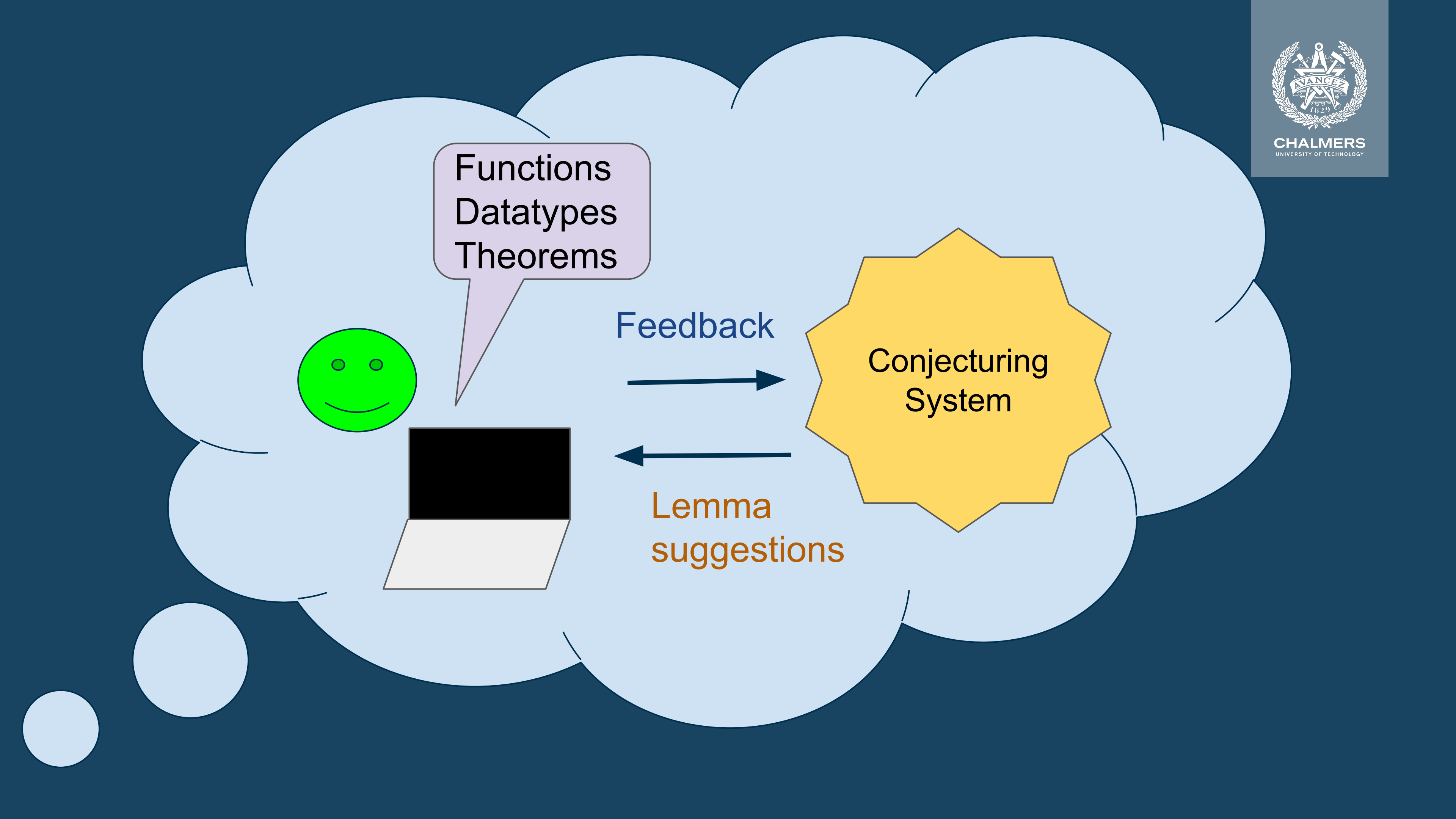
On Lemma Conjecturing using Neural, Symbolic and Neuro-symbolic approaches

Sólrún Halla Einarsdóttir, Yousef Alhessi,
Emily First, Moa Johansson

AITP 2024

CHALMERS
UNIVERSITY OF TECHNOLOGY


Functions
Datatypes
Theorems
Feedback
Conjecturing System
Lemma suggestions
CHALMERS
UNIVERSITY OF TECHNOLOGY

JUNE 8, 2024 | 12 MIN READ

AI Will Become Mathematicians' 'Co-Pilot'

Fields Medalist Terence Tao explains how proof checkers and AI programs are dramatically changing mathematics

BY CHRISTOPH DRÖSSER

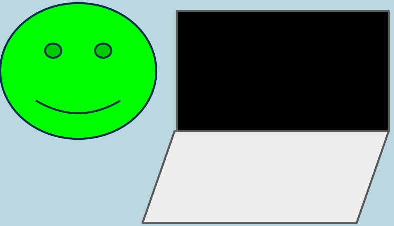
```
rev [] = []
rev (x : xs) = (rev xs) ++ (x :
[])

[] ++ xs = xs
(x : xs) ++ ys = x : (xs ++ ys)
```

QuickSpec

CHALMERS
UNIVERSITY OF TECHNOLOGY

```
1. rev [] = []
2. x ++ [] = x
3. [] ++ x = x
4. rev (rev x) = x
5. rev (x : []) = x : []
6. (x ++ y) ++ z =
   x ++ (y ++ z)
7. x : (y ++ z) = (x : y) ++ z
8. rev x ++ rev y =
   rev (y ++ x)
9. (xs ++ (y : (z : []))) =
   rev (z : (y : (rev xs)))
```



QuickSpec

- Leverages property-based testing (QuickCheck) and equational reasoning (Twee) to generate equational properties.

- Has been used to generate lemmas to automate inductive proofs.
  (most recently with Vampire, see IJCAR ‘24 paper)

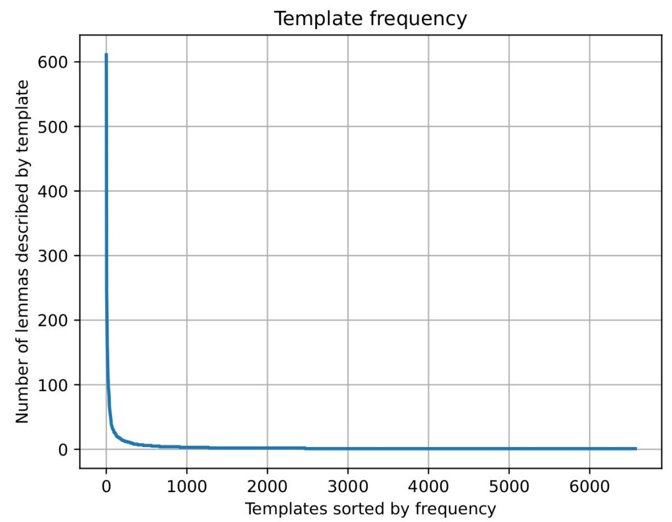
- Hipster: Isabelle tool for automated inductive proof using QuickSpec.

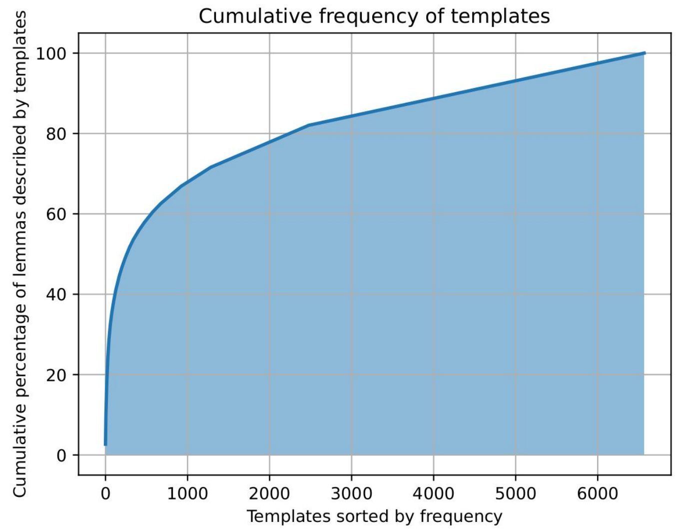
- Downsides: Becomes less efficient as the number of functions in scope grows.

RoughSpec

- Searches for conjectures that match a given template, e.g.
  - `?F(?F(x,y),z) = ?F(x,?F(y,z))`
  - Given `++`,`rev` and this template we'd find

    `(x ++ y) ++ z = x ++ (y ++ z)`
- A small number of templates can provide many lemmas
- How can we automatically come up with good templates to use for a new theory?





Symbolic vs. Neural conjecturing

Neural conjecturing:

Strengths

- Unrestricted in shape and size of conjectures.
- Can use information from names.

Weaknesses

- Prone to repetition.
- Generate false conjectures.
- Needs computational resources.

Symbolic conjecturing:

Strengths

- (More-or-less) true conjectures.
- Avoid repetition.
- Runs locally on a laptop.

Weaknesses

- Restricted in size and shape.
- Can't use name information.

Neuro-Symbolic Conjecturing: using RoughSpec!

- What if we get an LLM to generate templates for conjectures, which can then be filled in by RoughSpec?
- LLMs are good at capturing patterns/intuition.
- Compare to Neural-only approach: train a model to generate conjectures.
- Compare to symbolic-only approach:
  - QuickSpec
  - RoughSpec using simple heuristics/statistical analysis to choose templates.

Ongoing experiments: Neural Only Conjecturing

Fine-tuning Facebook OPT 1.3B parameter model

Data: Isabelle-HOL Library (around 30k examples)

Input: Symbols, Definitions	Target: Lemma Statement

(Near) Future: Bigger model, more data (AFP)

Evaluation

- Syntax-checking
- Counterexample-checking

(Near) Future:

- Provable?
- Trivial?
- Coverage?
- Usefulness in proof automation?

# Generated conjectures	3062
Pass syntax check	375
Pass counterexample check	163

```
"x \<and> y => x \<and> y"
```

```
"sq x * sq y = sq (x + y)"
```

Challenges/ Future Work

- Workflow: How do we interact with the LLM?
  - Sample once or many times, interleave/iterate back and forth?
  - Predict lemmas independently or many at a time?
  - Conditioned prediction?
  - What context to provide?
- How to evaluate lemma quality/interestingness?
- Training data leakage?
- Could this be extended to support other languages (Lean, Coq)?