Lemma Discovery and Strategies for Automated Induction

Sólrún Halla Einarsdóttir, Márton Hajdu, Moa Johansson, Nicholas Smallbone, Martin Suda

CHALMERS
UNIVERSITY OF TECHNOLOGY

WAIT 2024

Automated Induction is a challenge!

- Why is it so challenging?
    - Often auxiliary lemmas are needed.
- Now induction is available in Vampire!
- In this work we augment Vampire's inductive proof capabilities and achieve SOTA performance on the TIP benchmarks.

Example: Prove `rev(rev x) = x`

Relevant Definitions:

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

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

Base case:

`rev (rev [])` = (rev def)

`rev []` = (rev def)

`[]`

Step case:

Assume `rev (rev as) = as` for some `as` **(i.h.)**

and show that `rev (rev (a : as)) = a : as`

`rev (rev (a : as))` = (rev def)

`rev ((rev as) ++ (a : []))` =

`??`

Lemmas conjectured by QuickSpec

```
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 : (x : (rev xs)))
```

```
(declare-sort sk 0)
(declare-datatype list ((nil) (cons (head sk) (tail list))))
(define-fun-rec
  ++
  ((x list) (y list)) list
  (match x
    ((nil y)
     ((cons z xs) (cons z (++ xs y))))))
(define-fun-rec
  rev
  ((x list)) list
  (match x
    ((nil nil)
     ((cons y xs) (++ (rev xs) (cons y nil))))))
(assert-not (forall ((x list)) (= (rev (rev x)) x)))
```

```
  ((x list) (y list)) list
  (match x
    ((nil y)
     ((cons z xs) (cons z (++ xs y))))))
(define-fun-rec
  rev
  ((x list)) list
  (match x
    ((nil nil)
     ((cons y xs) (++ (rev xs) (cons y nil))))))
(assert-not (forall ((x list)) (= (rev (rev x)) x)))
(assert-claim (= (rev nil) nil))
(assert-claim (forall ((y list)) (= (++ y nil) y)))
(assert-claim (forall ((y list)) (= (++ nil y) y)))
(assert-claim (forall ((y list)) (= (rev (rev y)) y)))
(assert-claim
  (forall ((y sk)) (= (rev (cons y nil)) (cons y nil))))
(assert-claim
  (forall ((y sk) (z list) (x2 list))
    (= (++ (cons y z) x2) (cons y (++ z x2)))))
(assert-claim
  (forall ((y list) (z list) (x2 list))
    (= (++ (++ y z) x2) (++ y (++ z x2)))))
(assert-claim
  (forall ((y list) (z list))
    (= (++ (rev z) (rev y)) (rev (++ y z)))))
(assert-claim
  (forall ((y sk) (z sk) (x2 list))
    (= (++ x2 (cons z (cons y nil)))
      (rev (cons y (cons z (rev x2)))))))
```

CHALMERS
UNIVERSITY OF TECHNOLOGY

Conjectured lemmas in Vampire

- QuickSpec may come up with many irrelevant conjectures.
- Speculative lemma use is straightforward in Vampire.
- Vampire only proves the lemmas that are used in the final proof.

```
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 : (x : (rev xs)))
```

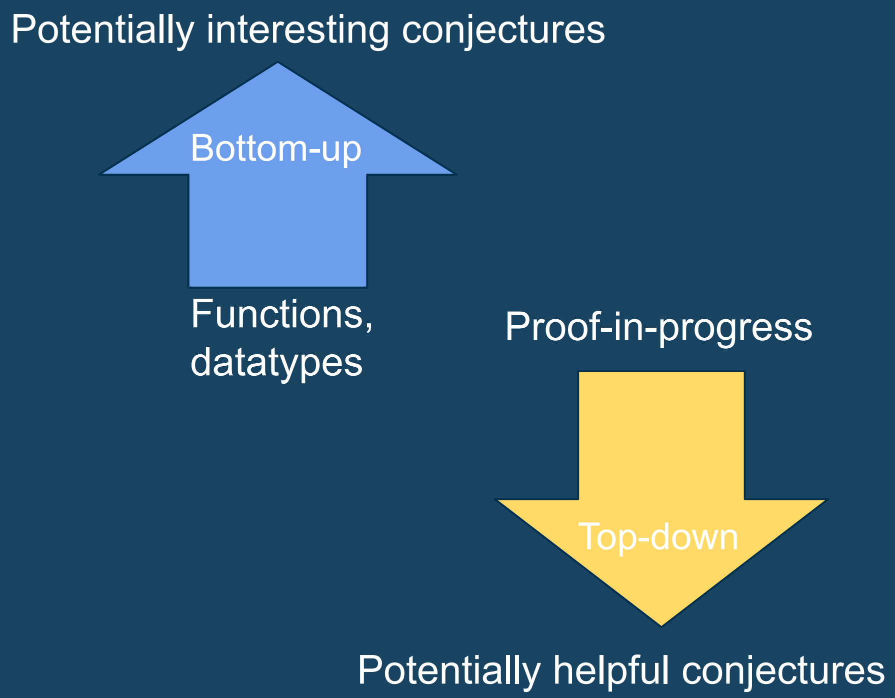
Lemma Discovery

Bottom-up:

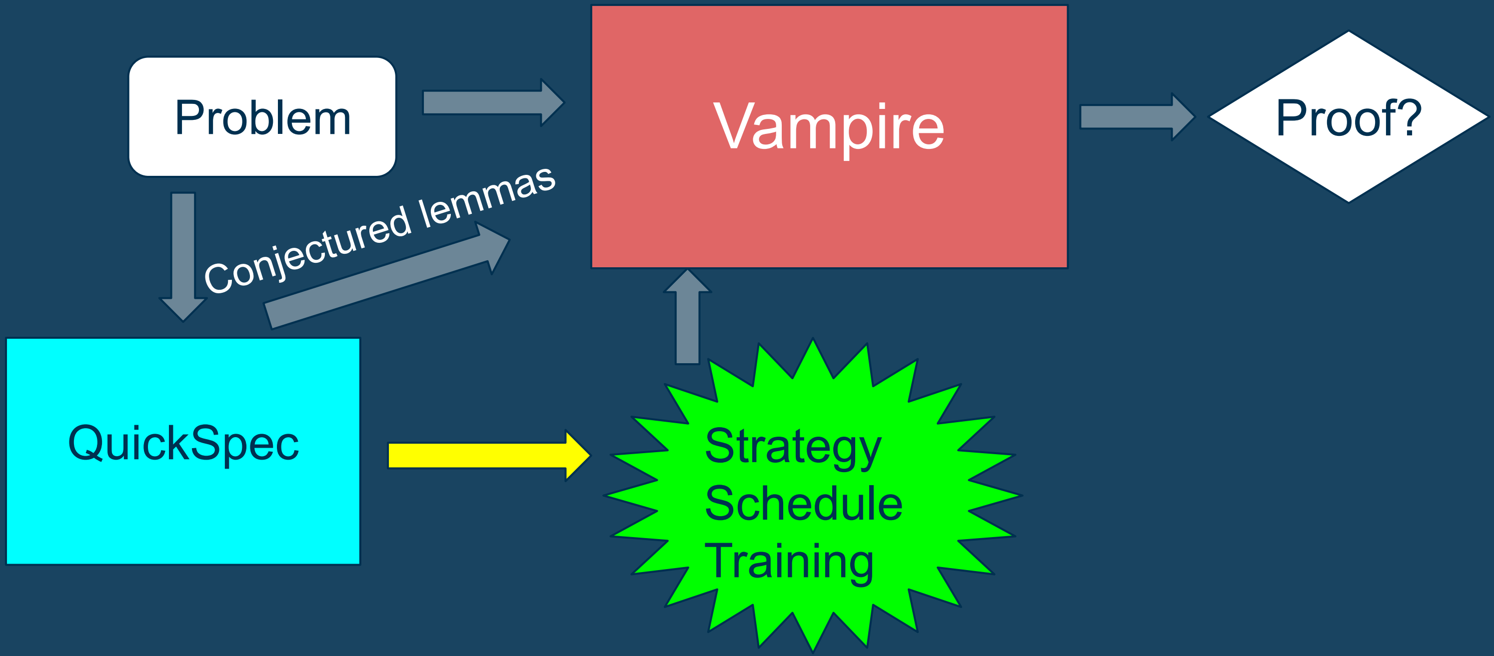
Potentially interesting lemmas
generated from definitions in scope

Top-down:

Lemmas are generated
while proving a goal theorem
by e.g. generalizing



Our experiments



Results

Proofs found by our methods on TIP benchmarks* (486 problems in total):

	Default Strategy	Trained Strategy Schedule	Schedule trained with lemmas
no lemmas	102	236	237
with lemmas	143	263	288
Total proofs found	153	269	289

* https://tip-org.github.io/

Conclusions

- Both lemma discovery and strategy schedule training can help make Vampire more effective as an inductive prover.
- A combination of lemma discovery and strategy schedule training was needed to beat previous SOTA.
- Lemma discovery and strategy schedule training are mostly complementary.

CHALMERS
UNIVERSITY OF TECHNOLOGY

OEIS benchmark experiments

- A Mathematical Benchmark for Inductive Theorem Provers (Gauthier et al. LPAR 2023).
- We trained a strategy schedule for these benchmarks.
- Conjectured lemmas with QuickSpec using various settings.
- Preliminary results were not so promising and adding conjectured lemmas made results worse:
  - From ~5% to ~1%
- Where do we go from here?

Unanswered questions

- What is needed to prove the remaining 197/486 TIP problems?
- What is needed to achieve better results on the OEIS benchmarks?
- How much can we get out of saturation-based ATP with optimal strategy selection?
- How much can we get out of external lemma generation?

Ongoing work - theory exploration

CHALMERS
UNIVERSITY OF TECHNOLOGY

- We want conditional lemmas!
- We want to restrict our search space and generate more targeted lemmas.
- RoughSpec generates conjectures of specific shapes.
    - Can machine learning help find useful shapes?
- What can we get out of LLMs?

Unanswered questions

- What is needed to prove the remaining 197/486 TIP problems?
- What is needed to achieve better results on the OEIS benchmarks?
- How much can we get out of saturation-based ATP with optimal strategy selection?
- How much can we get out of external lemma generation?

Find our paper here: