Beagle as a HOL4 external ATP method

Thibault Gauthier

May 21, 2014

Two types of provers

	HOL4	Beagle
Type	Interactive	Automated
Expressivity	Higher-order	First-order
Soundness	Small kernel (LCF)	Long optimized code
Family	HOL Light, ProofPower, Isabelle/HOL	Spass + T

Problem statement

Problem Here are two HOL4 internal provers.

- Metis: first-order
- Cooper: arithmetic

Problem statement

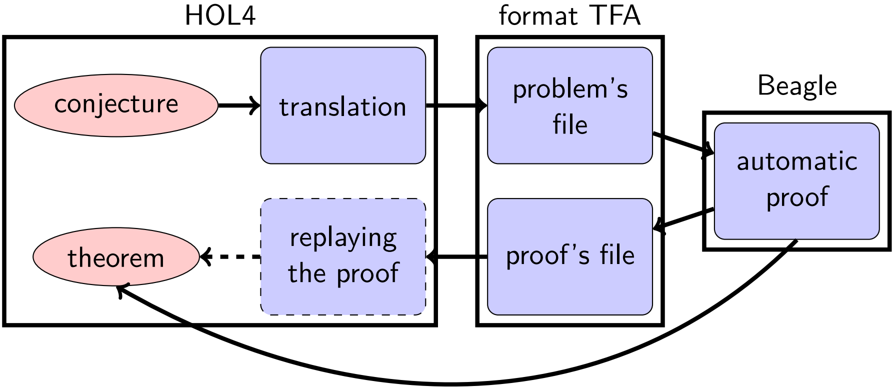
Problem Here are two HOL4 internal provers.

- Metis: first-order
- Cooper: arithmetic

Solution An external prover.

- Beagle: first-order and arithmetic

Interaction



Translation's order

1. Monomorphization
2. Negation of the conclusion
3. Rewriting to conjunctive normal form
4. λ-lifting
5. Boolean argument elimination:
   $P(f) \rightarrowtail f \Rightarrow P(T) \wedge \neg f \Rightarrow P(F)$
6. Rewriting to a clause set
7. Defunctionalization
8. Mapping numeral to integers

Monomorphization

Instantiation of polymorphic types (a,...).

Problem

Thm 1: $\forall x : a.\ D\ x\ 0$ Thm 2: $C = \lambda x : a.\ D\ x\ 0$

Conjecture : $C\ 2$

Monomorphization

Instantiation of polymorphic types (a,...).

Problem

Thm 1: $\forall x : a.\ D\ x\ 0$ Thm 2: $C = \lambda x : a.\ D\ x\ 0$
Conjecture : $C\ 2$

Matching type of $C : a \rightarrow bool$ and $C : int \rightarrow bool$

Thm 1: $\forall x : a.\ D\ x\ 0$ Thm 2: $C = \lambda x : int.\ D\ x\ 0$
Conjecture: $C\ 2$

Monomorphization

Instantiation of polymorphic types (a,...).

Problem

Thm 1: $\forall x : a.\ D\ x\ 0$ Thm 2: $C = \lambda x : a.\ D\ x\ 0$

Conjecture : $C\ 2$

Matching type of $C : a \rightarrow bool$ and $C : int \rightarrow bool$

Thm 1: $\forall x : a.\ D\ x\ 0$ Thm 2: $C = \lambda x : int.\ D\ x\ 0$

Conjecture: $C\ 2$

Matching type of $D : a \rightarrow int \rightarrow bool$ and $D : int \rightarrow int \rightarrow bool$

Thm 1: $\forall x : int.\ D\ x\ 0$ Thm 2: $C = \lambda x : int.\ D\ x\ 0$

Conjecture: $C\ 2$

λ-lifting

Problem

Thm 1: $\forall x.\ D\ x\ 0$ Thm 2: $C = \lambda x.\ D\ x\ 0$

Conjecture: $C\ 2$

λ-lifting

Problem

Thm 1: $\forall x.\ D\ x\ 0$ Thm 2: $C = \lambda x.\ D\ x\ 0$
Conjecture: $C\ 2$

Negation of the conclusion

$$\{\forall x.\ D\ x\ 0\ ,\ C = \lambda x.\ D\ x\ 0\ ,\ \neg(C\ 2)\}$$

λ-lifting

Problem

Thm 1: $\forall x.\ D\ x\ 0$ Thm 2: $C = \lambda x.\ D\ x\ 0$
Conjecture: $C\ 2$

Negation of the conclusion

$$\{\forall x.\ D\ x\ 0\ ,\ C = \lambda x.\ D\ x\ 0\ ,\ \neg(C\ 2)\}$$

λ-lifting: $\exists Gen.\ (\forall x.\ Gen\ x = D\ x\ 0) \wedge C = Gen$

λ-lifting

Problem

Thm 1: $\forall x.\ D\ x\ 0$ Thm 2: $C = \lambda x.\ D\ x\ 0$
Conjecture: $C\ 2$

Negation of the conclusion

$$\{\forall x.\ D\ x\ 0\ ,\ C = \lambda x.\ D\ x\ 0\ ,\ \neg(C\ 2)\}$$

λ-lifting: $\exists Gen.\ (\forall x.\ Gen\ x = D\ x\ 0) \wedge C = Gen$
Combinators: $C = S\ (S\ (K\ D)\ I)\ (K\ 0)$

λ-lifting

Problem

Thm 1: $\forall x.\ D\ x\ 0$ Thm 2: $C = \lambda x.\ D\ x\ 0$
Conjecture: $C\ 2$

Negation of the conclusion

$$\{\forall x.\ D\ x\ 0\ ,\ C = \lambda x.\ D\ x\ 0\ ,\ \neg(C\ 2)\}$$

λ-lifting: $\exists Gen.\ (\forall x.\ Gen\ x = D\ x\ 0) \wedge C = Gen$
Combinators: $C = S\ (S\ (K\ D)\ I)\ (K\ 0)$

Clause set

$$\{\forall x.\ D\ x\ 0\ ,\ \forall x.\ Gen\ x = D\ x\ 0\ ,\ C = Gen\ ,\ \neg(C\ 2)\}$$

Defunctionalization

Let App be the apply functor verifying $App\ f\ x = f\ x$ $(f\ x \rightarrowtail App\ f\ x)$. We defunctionalize a function only when:

- it is not a mapped function

Defunctionalization

Let App be the apply functor verifying $App\ f\ x = f\ x$ ($f\ x \rightarrowtail App\ f\ x$). We defunctionalize a function only when:

- it is not a mapped function
- it is quantified universally $!h.\ h\ x\ y \ \rightarrowtail\ !h.\ App\ (App\ h\ x)\ y$

Defunctionalization

Let App be the apply functor verifying $App\ f\ x = f\ x$ $(f\ x \rightarrowtail App\ f\ x)$. We defunctionalize a function only when:

- it is not a mapped function
- it is quantified universally $!h.\ h\ x\ y\ \rightarrowtail\ !h.\ App\ (App\ h\ x)\ y$
- it is used with different number of arguments
  $\{h\ x\ y\ z\ ,\ h\ x = j\}\ \rightarrowtail\ \{App\ (App\ (h\ x)\ y)\ z\ ,\ h\ x = j\}$
- it has the same type as an universally quantified function

Defunctionalization

Let *App* be the apply functor verifying $App\ f\ x = f\ x$ ($f\ x \rightarrowtail App\ f\ x$). We defunctionalize a function only when:

- it is not a mapped function
- it is quantified universally $!h.\ h\ x\ y \ \rightarrowtail\ !h.\ App\ (App\ h\ x)\ y$
- it is used with different number of arguments $\{h\ x\ y\ z\ ,\ h\ x = j\} \ \rightarrowtail\ \{App\ (App\ (h\ x)\ y)\ z\ ,\ h\ x = j\}$
- it has the same type as an universally quantified function

Defunctionalization

Let *App* be the apply functor verifying $App\ f\ x = f\ x$ $(f\ x \rightarrowtail App\ f\ x)$. We defunctionalize a function only when:

- it is not a mapped function
- it is quantified universally $!h.\ h\ x\ y \ \rightarrowtail\ !h.\ App\ (App\ h\ x)\ y$
- it is used with different number of arguments
  $\{h\ x\ y\ z\ ,\ \ h\ x = j\}\ \rightarrowtail\ \{App\ (App\ (h\ x)\ y)\ z\ ,\ \ h\ x = j\}$
- it has the same type as an universally quantified function

Defunctionalization

$$\{\forall x.\ D\ x\ 0,\ \forall x.\ Gen\ x = D\ x\ 0,\ C = Gen,\ \neg(C\ 2)\}$$

$$\{\forall x.\ D\ x\ 0,\ \forall x.\ App\ Gen\ x = D\ x\ 0,\ C = Gen,\ \neg(C\ 2)\}$$

Translation's order

1. Monomorphization
2. Negation of the conclusion
3. Rewriting to conjunctive normal form
4. λ-lifting
5. Boolean argument elimination
6. Rewriting to a clause set
7. Defunctionalization
8. Mapping numeral to integers

Demonstration

Problem

Thm 1: $\forall x : a.\ D\ x\ 0$ Thm 2: $C = \lambda x : a.\ D\ x\ 0$

Conjecture : $C\ 2$

Translated problem

$$\{\forall x : a.\ D\ x\ 0,\ \forall x : a.\ App\ Gen_0\ x = D\ x\ 0,\ C = Gen_0,$$

$$\forall x : int.\ D\ x\ 0,\ \forall x : int.\ App\ Gen_1\ x = D\ x\ 0,\ C = Gen_1,$$

$$\neg(C\ 2)\}$$

Results

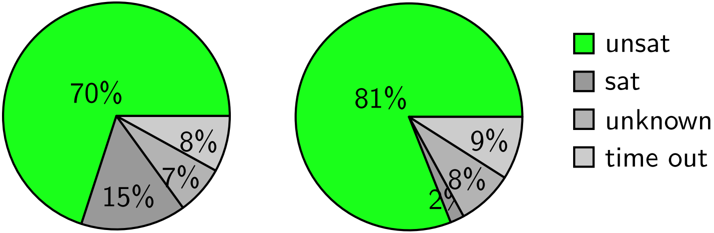


Table: With and without monomorphization

Only 56% of higher-order problems are solved.
And 79% of arithmetic problems are solved. (containing at least one mapped arithmetic constant)

Summary on the current HOL4-Beagle interaction

Qualities:

- Its translation is correct (preserve satisfiability).
- It proves 81% of conjectures solved by Metis without arithmetic lemmas.
- It uses a well-known format, TFA (TPTP).

Limitations:

- it is incomplete (doesn't preserve unsatisfiability).
- it doesn't reconstruct the proof.
- it doesn't support real and rational arithmetic.
- it was not tested extensively (for example, as part of a lemma mining method).