A brief introduction to Autosubst

Elliot Bobrow

Advised by Stephanie Weirich and Yiyun Liu

What is Autosubst 2?

- Tool for proofs about programming languages in Coq
- Provides tactics to automatically simplify terms
    - Prove equality by simplifying two terms to the same form

Parallel substitutions

Untyped Lambda Calculus

$$
\begin{aligned}
e ::= {} & x \\
\mid {} & \lambda x.e \\
\mid {} & e_1\, e_2
\end{aligned}
$$

Untyped Lambda Calculus

$$
\begin{aligned}
e ::= {} & n \\
\mid {} & \lambda x.e \\
\mid {} & e_1\ e_2
\end{aligned}
$$

De Bruijn Indices

$$\lambda x.(\lambda y.y\,(\lambda z.z))(\lambda y.x\,y) \Longrightarrow \lambda(\lambda 1\,(\lambda 1))(\lambda 2\,1)$$

Single Substitutions

$$(\lambda a)\, b \longrightarrow a[b/1]$$

Single Substitutions

$$(\lambda a)\, b \longrightarrow a[b/1][2/2][3/3]\cdots$$

Parallel Substitutions

$$(\lambda a)\, b \longrightarrow a[\sigma]$$

where

$$\sigma = (b, 2, 3, \dots)$$

Primitives

$$\mathrm{id} = (1, 2, 3, \dots) \qquad\qquad \uparrow = (2, 3, 4, \dots)$$

$$e \cdot \sigma = (e, \sigma(1), \sigma(2), \dots)$$

Instantiation

$$\begin{aligned} x[\sigma] &= \sigma(x) \\ (a\,b)[\sigma] &= (a[\sigma])\,(b[\sigma]) \\ (\lambda a)[\sigma] &= \lambda(a[1 \cdot (\sigma \circ \uparrow)]) \end{aligned}$$

Instantiation Example

$$1\,(\lambda 2\,1)$$

Instantiation Example

$$\color{red}{1}\,(\lambda \color{red}{2}\; 1)$$

Instantiation Example

$$\color{red}{1}\,(\lambda \color{red}{2}\, 1)[(e, 2, 3, \ldots)]$$

Instantiation Example

$$(1[(e, 2, 3, \dots)])\,((\lambda 2\, 1)[(e, 2, 3, \dots)])$$

Instantiation Example

$$e\,((\lambda 2\;1)[(e, 2, 3, \ldots)])$$

Instantiation Example

$$e\,(\lambda(\textcolor{red}{2}\;1)[1 \cdot ((e, 2, 3, \dots) \circ \uparrow)])$$

Instantiation Example

$$e\,(\lambda(2\;1)[(1, e[\uparrow], 3, 4, \ldots)])$$

Instantiation Example

$$e\,(\lambda e[\uparrow]\,1)$$

Autosubst in action

Syntax specification

```
tm : Type
ty : Type

app : tm -> tm -> tm
abs : ty -> (tm -> tm) -> tm

fun : ty -> ty -> ty
I : ty
```

Figure 1: stlc.sig

Generated code

```
Inductive ty : Type :=
  | fun : ty -> ty -> ty
  | I : ty.

Inductive tm : Type :=
  | var_tm : nat -> tm
  | app : tm -> tm -> tm
  | abs : ty -> tm -> tm.
```

Figure 2: stlc.v

Example

$$\frac{\begin{array}{l}\sigma\ \texttt{: nat -> tm}\\ \texttt{e v : tm}\end{array}}{\texttt{P (app (abs e)[}\sigma\texttt{] v)}}$$

Example

$$\frac{\sigma \texttt{ : nat -> tm} \qquad \texttt{e v : tm}}{\texttt{P e[1} \cdot (\sigma \circ \uparrow)\texttt{][v} \cdot \texttt{id]}}$$

Example

σ `: nat -> tm`
`e v : tm`

`P e[v` $\cdot$ σ`]`

So what have I been working on?

Termination of STLC

Theorem
For all terms e, if $\vdash e : \tau$, then there exists a value v such that $e \to^* v$.

Proof using named syntax

Lines of code 866

Lemma count 58

Admit count 14

Proof using Autosubst 2

Lines of code 313

Lemma count 21

Admit count 1

Summary

- Single substitutions $\rightarrow$ parallel substitutions
- Code generation
- Autosubst 2 makes proofs easier

Further reading

Autosubst paper: `https://www.ps.uni-saarland.de/Publications/documents/SchaeferEtAl_2015_Autosubst_-Reasoning.pdf`

Parallel substitutions: `http://www.lucacardelli.name/Papers/SRC-054.pdf`

Autosubst 2 paper: `https://www.ps.uni-saarland.de/Publications/documents/StarkEtAl_2018_Autosubst-2_.pdf`

Scoped terms: `https://link.springer.com/chapter/10.1007/11617990_1`

Logical relations:
`https://www.cs.uoregon.edu/research/summerschool/summer23/topics.php`