Proofs by Resolution and Existential Variables

František Farka[1,2]

[1]University of Dundee, Dundee, Scotland

[2]University of St Andrews, St Andrews, Scotland

ff32@st-andrews.ac.uk

CoALP-Ty'16, Edinburgh
November 28, 2016

Motivation

- Horn-Clause Logic in program verification
  - HC as a general framework (Bjørner *et al.*; 2015)
  - type-theoretic interpretation (Fu *et al.*; 2016)
  - soundness of HC resolution for Type-Class resolution (Farka *et al.*; 2016)
- Limitation of the above - universal fragment
  - clauses without *existential variables*
  - resolution by *matching* / universally quantified goals
- Resolution with existential variables and unification / existentially qualified goals

Problem by an example

Example (Vytiniotis *et al.*; 2011)

```
data T a where
    T1 : Int → T Bool
    T2 : T a

test (T1 n) _ = false
test T2     r = r
```

In System F, there are two different most-general types of `test`:

$$\texttt{test} \quad : \quad \forall\alpha.\mathrm{T}\ \alpha \to \texttt{Bool} \to \texttt{Bool}$$

$$\texttt{test} \quad : \quad \forall\alpha.\mathrm{T}\ \alpha \to \alpha \to \alpha$$

Constrained-type solution:

$$\texttt{test} \quad : \quad \forall\alpha.\forall\beta.(\alpha \sim \texttt{Bool} \supset \beta \sim \texttt{Bool}) \ \Rightarrow \mathrm{T}\ \alpha \to \beta \to \beta$$

With existential quantification:

$$\texttt{test} \quad : \quad \forall\alpha.\exists\beta.\mathrm{T}\ \alpha \to \beta \to \beta$$

Calculus (matching)

Definition (Syntax)

$$
\begin{array}{rrl}
\text{Horn flae} & HC ::= & \mathcal{K} : At \leftarrow At, \ldots, At \\
\text{Proof terms} & PT ::= & \mathcal{K}(PT, \ldots, PT) \\
\text{Goals} & G ::= & \forall Var, \ldots, Var \exists Var, \ldots, Var . At
\end{array}
$$

Definition (LP-M)

$$
\kappa : A \leftarrow B_1, \ldots, B_n \in P \; \frac{P \vdash e_1 : \forall \overline{w_1} \exists \overline{z_1} . \sigma B_1 \qquad \ldots \qquad P \vdash e_n : \forall \overline{w_n} \exists \overline{z_n} . \sigma B_n}{P \vdash \kappa(e_1, \ldots, e_n) : \forall \overline{x} \exists \overline{y} . \sigma A}
$$

where $\overline{w_i} = \overline{x} \cap \text{var}(\sigma B_i)$ and $\overline{z_i} = \text{var}(\sigma B_i) \setminus w_i$.

Problem by an example revised

Example

constraint-based form (Simonet and Pottier; 2007)

```
data T a where
    T1 : ∀ a [a ∼ Bool] . Int → T a
    T2 : ∀ a . T a

test (T1 n) _ = false
test T2     r = r
```

Logic program:

$$
\begin{aligned}
\kappa_{refl} : \mathrm{eq}(x, x) &\Leftarrow \\
\kappa_{test} : \mathrm{test}(\mathrm{T}(x), y, z) &\Leftarrow (\mathrm{eq}(x, \mathrm{bool}) \Rightarrow \mathrm{eq}(z, \mathrm{bool})), \\
&\quad\ \mathrm{eq}(y, z)
\end{aligned}
$$

Outside Horn-clauses (Hereditary Harrop formulae)

Type checking by matching

The type test $: \forall a. Ta \to a \to a$ corresponds to a goal

$\forall a. test(\mathrm{T}(a), a, a)$

$$\dfrac{\dfrac{\dfrac{}{P, (\alpha : \mathrm{eq}(a,\ \mathrm{bool}) \Leftarrow\ ) \vdash\ \alpha : \forall a.\mathrm{eq}(a, \mathrm{bool})}}{P \vdash \lambda\alpha.\alpha : \forall a.\mathrm{eq}(a, \mathrm{bool}) \Rightarrow \mathrm{eq}(a, \mathrm{bool})} \qquad \dfrac{}{P \vdash\ \kappa_{refl} : \forall a.\mathrm{eq}(a, a)}}{P \vdash \kappa_{test}(\kappa_{refl}, \lambda\alpha.\alpha, \kappa_{refl}) : \forall a.\mathrm{test}(\mathrm{T}(a), a, a)}$$

Calculus (unification)

Definition (Extended syntax)

$$\text{Proof term} \qquad HC ::= \qquad \cdots \mid ind(Var.Term, PT)$$

Definition (LP-U)

$$\kappa : A \leftarrow B_1, \dots, B_n \in P \; \frac{P \vdash e_1 : \forall \overline{w_1} \exists \overline{z_1}.\sigma B_1 \qquad \dots \qquad P \vdash e_n : \forall \overline{w_n} \exists \overline{z_n}.\sigma B_n}{P \vdash ind(\overline{y}.\sigma \overline{y}, \kappa(e_1, \dots, e_n)) : \forall \overline{x} \exists \overline{y}.A'}$$

if $(\sigma \upharpoonright \overline{y})A' = (\sigma \upharpoonright \overline{y})A$. Moreover, $w_i = \overline{x} \cap \mathrm{var}(\sigma B_i)$ and $\overline{z_i} = \mathrm{var}(\sigma B_i) \setminus w_i$.

Type checking by unification

The type test : $\forall a.\exists b. Ta \to b \to b$ corresponds to a goal
$\forall a.\exists b.test(\mathrm{T}(a), b, b)$

$$\dfrac{\dfrac{\dfrac{}{P, (\alpha : \mathrm{eq}(a,\ \mathrm{bool}) \Leftarrow\ ) \vdash\ \kappa_{refl} : \mathrm{eq}(\mathrm{bool}, \mathrm{bool})}}{P \vdash \lambda\alpha.\kappa_{refl} : \mathrm{eq}(a, \mathrm{bool}) \Rightarrow \mathrm{eq}(\mathrm{bool}, \mathrm{bool})} \qquad \dfrac{}{P \vdash \kappa_{refl} : \mathrm{eq}(\mathrm{bool}, \mathrm{bool})}}{P \vdash \mathsf{ind}(b.\mathrm{bool}, \kappa_{test}(\kappa_{refl}, \lambda\alpha.\kappa_{refl}, \kappa_{refl})) : \forall a \exists b.\mathrm{test}(\mathrm{T}(a), b, b)}$$

Soundness and completeness

Soundness and completeness w.r.t. *first-order dependent type theory* (DTT) (Jacobs; 1999)

Theorem (Soundness)

If $P \vdash_{LP} e : \forall\overline{x}\exists\overline{y}.A$ *then* $P \vdash_{DTT} e : \Pi\overline{x}\Sigma\overline{y}.A$

Theorem (Completeness)

If $P \vdash_{DTT} e : G$ *where* e *is in ground normal form then* $P \vdash_{LP} G$*.*

Type inference

Utilising *Structural resolution* for type inference (Komendantskaya and Johann; 2015)

- exhaustive matching / LP-M steps
- one unification step / LP-U step

Definition (Structural resolution)

Given a logic program P, *structural resolution* is given by an abstract reduction system $(P, \rightarrow^*_{LP-M} \cdot \rightarrow^1_{LP-U})$

Quantifier assignment:

- LP-U binds substituted variables to $\exists$ quantifier
- LP-M binds eliminated variables to $\forall$ quantifier
- Normalise prefix to $\forall\overline{x}\exists\overline{y}$ by the appropriate substitutions

Type inference (cont'd)

Example

$\mathrm{test}(x, y, z)$

$\quad \rightarrow_{LP-U}$ $\qquad\qquad \{x/\mathrm{T}(w)\}; \exists x \forall w$

$\mathrm{eq}(w, \mathrm{bool}) \Rightarrow \mathrm{eq}(z, \mathrm{bool}), \mathrm{eq}(y, z)$

$\quad \rightarrow_{LAM}$

$\mathrm{eq}(z, \mathrm{bool}), \mathrm{eq}(y, z)$

$\quad \rightarrow_{LP-U}$ $\qquad\qquad \{z/\mathrm{bool}\}; \exists z$

$\mathrm{eq}(y, \mathrm{bool})$

$\quad \rightarrow_{LP-U}$ $\qquad\qquad \{y/\mathrm{bool}\}; \exists y$

$\emptyset$

Have: $\forall w. \exists z. \exists y. \mathsf{test}(\mathrm{T}(w), y, z)$

$\mathsf{ind}(z.\mathrm{bool}, \mathsf{ind}(y.\mathrm{bool}, \kappa_{test}(\kappa_{refl}, \lambda\alpha.\kappa_{refl}, \kappa_{refl})) :$

Related Work, Work in Progress & Future Work

Related Work

- Constraint-based type inference—HM(X), OutsideIn(X), ...
- Guarded ADT's (Simonet and Pottier; 2007)

Work in Progress

- Other language constructs than ADTs (Type Families, ...)
- Non-existential clauses

Future Work

- Semantics of the calculus
- Higher order formulae

Thank you