Safety Proofs of Simple Type System

AAA551: Programming Language Theory

Korea University

1. Simply Typed Lambda Calculus

Syntax We consider lambda calculus with boolean types and conditional expressions:

$$
\begin{array}{rcll}
t & ::= & x & \text{variable} \\
 & | & \lambda x : T.t & \text{abstraction} \\
 & | & t\ t & \text{application} \\
 & | & \texttt{true} \mid \texttt{false} & \text{boolean values} \\
 & | & \texttt{if}\ t\ t\ t & \text{conditional expression}
\end{array}
$$

The values in this language are terms defined by the following grammar:

$$v ::= \texttt{true} \mid \texttt{false} \mid \lambda x : T.t$$

Types include primitive boolean types and function types:

$$T ::= \mathit{Bool} \mid T \to T$$

Evaluation Rules

$$\frac{t_1 \to t_1'}{t_1\ t_2 \to t_1'\ t_2}\ \text{E-APP1}$$

$$\frac{t_2 \to t_2'}{v_1\ t_2 \to v_1\ t_2'}\ \text{E-APP2}$$

$$\frac{}{(\lambda x : T.t_{12})\ v_2 \to [x \mapsto v_2]t_{12}}\ \text{E-APPABS}$$

$$\frac{}{\texttt{if true}\ t_2\ t_3 \to t_2}\ \text{E-IFTRUE}$$

$$\frac{}{\texttt{if false}\ t_2\ t_3 \to t_3}\ \text{E-IFFALSE}$$

$$\frac{t_1 \to t_1'}{\texttt{if}\ t_1\ t_2\ t_3 \to \texttt{if}\ t_1'\ t_2\ t_3}\ \text{E-IF}$$

Typing Rules

$$\frac{\Gamma(x) = T}{\Gamma \vdash x : T}\ \text{T-VAR}$$

$$\frac{\Gamma[x \mapsto T_1] \vdash t_2 : T_2}{\Gamma \vdash \lambda x : T_1.t_2 : T_1 \to T_2}\ \text{T-ABS}$$

$$\frac{\Gamma \vdash t_1 : T_{11} \to T_{12} \quad \Gamma \vdash t_2 : T_{11}}{\Gamma \vdash t_1\ t_2 : T_{12}}\ \text{T-APP}$$

$$\frac{}{\Gamma \vdash \texttt{true} : \mathit{Bool}}\ \text{T-TRUE}$$

$$\frac{}{\Gamma \vdash \texttt{false} : \mathit{Bool}}\ \text{T-FALSE}$$

$$\frac{\Gamma \vdash t_1 : \mathit{Bool} \quad \Gamma \vdash t_2 : T \quad \Gamma \vdash t_3 : T}{\Gamma \vdash \texttt{if}\ t_1\ t_2\ t_3 : T}\ \text{T-IF}$$

2. Safety Proofs

Theorem 1 (Type Safety)**.** *Suppose t is a closed term. If $\vdash t : T$, then t does not get stuck during evaluation. Furthermore, if t reaches a value v, then v is of the T type.*

Proof. Immediate from Lemma 1 and Lemma 4. □

Lemma 1 (Progress)**.** *Suppose t is a closed term. If t is well-typed (i.e., $\vdash t : T$ for some T), then either t is a value or there is some t' with $t \to t'$:*

$$\vdash t : T \implies t \text{ is a value or } \exists t'.\ t \to t'$$

Proof. By structural induction on t.

- $t \in \{\texttt{true}, \texttt{false}\}$: Immediate, since t is a value.
- $t = \lambda x : T.t_1$: Immediate, since t is a value.
- $t = x$: Cannot occur (because t is closed).
- $t = t_1\ t_2$: What we have to show in this case is as follows:

  $$\vdash t_1\ t_2 : T \implies \exists t'.\ (t_1\ t_2) \to t'$$

  First, by typing rule T-APP, we know that t_1 and t_2 are well-typed:

  $$\frac{\Gamma \vdash t_1 : T_{11} \to T_{12} \quad \Gamma \vdash t_2 : T_{11}}{\Gamma \vdash t_1\ t_2 : T_{12}}$$

  where $T = T_{12}$. By the induction hypothesis (IH), either t_1 is a value or else it can make a step of evaluation, and likewise t_2:

  $$t_1 \text{ is a value or } \exists t_1'.\ t_1 \to t_1' \quad \cdots \text{IH1}$$

  $$t_2 \text{ is a value or } \exists t_2'.\ t_2 \to t_2' \quad \cdots \text{IH2}$$

  There are three cases to consider.

  - t_1 is not a value: by IH1, there exists t_1' such that

    $$t_1 \to t_1'$$

    and E-APP1 applies to t:

    $$t_1\ t_2 \to t_1'\ t_2$$

  - t_1 is a value and t_2 is not a value: by IH2, there exists t_2' such that

    $$t_2 \to t_2'$$

    and E-APP2 applies to t:

    $$t_1\ t_2 \to t_1\ t_2'$$

  - Both t_1 and t_2 are values: because t_1 is well-typed as function abstraction($\vdash t_1 : T_{11} \to T_{12}$), t_1 has the form $\lambda x : T_{11}.t_{12}$ and so rule E-APPABS applies to t.
- $t = \text{if}\ t_1\ t_2\ t_3$: By typing rule T-IF

  $$\frac{\Gamma \vdash t_1 : \mathit{Bool} \quad \Gamma \vdash t_2 : T \quad \Gamma \vdash t_3 : T}{\Gamma \vdash \texttt{if}\ t_1\ t_2\ t_3 : T}$$

  and induction hypothesis, either t_1 is a value or else there is some t_1' such that $t_1 \to t_1'$.

- t_1 is a value: t_1 is either `true` or `false`, in which either E-IfTrue or E-IfFalse applies to t.
- $t_1 \to t_1'$: E-If applies to t and therefore $t \to$ if $t_1'\ t_2\ t_3$.

□

Lemma 2 (Weakening). *If $\Gamma \vdash t : T$ and $x \notin dom(\Gamma)$, then $\Gamma[x \mapsto S] \vdash t : T$ for any S.*

Proof. **(exercise 1)** Straightforward induction on t. □

Lemma 3 (Preservation under Substitution). *If $\Gamma[x \mapsto S] \vdash t : T$ and $\Gamma \vdash s : S$, then $\Gamma \vdash [x \mapsto s]t : T$.*

Proof. By induction on a derivation of the statement $\Gamma[x \mapsto S] \vdash t : T$.

- $t = z$: In this case, by typing rule T-Var, we have

$$\Gamma[x \mapsto S](z) = T$$

  There are two cases to consider:
  - $z = x$: We have

$$\Gamma[x \mapsto S] \vdash x : S \qquad [x \mapsto s]x = s$$

    and to show is $\Gamma \vdash s : S$, which is among the assumptions of the lemma.
  - $z \neq x$: In this case, we have

$$\Gamma[x \mapsto S] \vdash z : T \qquad [x \mapsto s]z = z$$

    and to show is $\Gamma \vdash z : T$, which is immediate.
- $t = \lambda y : T_2.t_1$: In this case, we have

$$\Gamma[x \mapsto S][y \mapsto T_2] \vdash t_1 : T_1 \qquad T = T_2 \to T_1$$

  where we assume that y is fresh (i.e., $y \notin \{x\} \cup dom(\Gamma)$). Because typing holds for all permutation of the type environment, we also have

$$\Gamma[y \mapsto T_2][x \mapsto S] \vdash t_1 : T_1$$

  By weakening the assumption ($\Gamma \vdash s : S$) of this lemma, we have

$$\Gamma[y \mapsto T_2] \vdash s : S$$

  Now, we apply the induction hypothesis and get

$$\Gamma[y \mapsto T_2] \vdash [x \mapsto s]t_1 : T_1$$

  We apply T-Abs and have

$$\Gamma \vdash \lambda y : T_2.[x \mapsto s]t_1 : T_2 \to T_1$$

  which, by the definition of the substitution, implies

$$\Gamma \vdash [x \mapsto s](\lambda y : T_2.t_1) : T_2 \to T_1$$

  as desired.
- $t = t_1\ t_2$: In this case, we have

$$\Gamma[x \mapsto S] \vdash t_1 : T_2 \to T_1, \quad \Gamma[x \mapsto S] \vdash t_2 : T_2, \quad T = T_1$$

  By the induction hypothesis,

$$\Gamma[x \mapsto S] \vdash [x \mapsto s]t_1 : T_2 \to T_1, \quad \Gamma[x \mapsto S] \vdash [x \mapsto s]t_2 : T_2,$$

  By T-App,

$$\Gamma \vdash [x \mapsto s]t_1\ [x \mapsto s]t_2 : T$$

  which, by the definition of substitution, implies

$$\Gamma \vdash [x \mapsto s](t_1\ t_2) : T$$

  as desired.
- Other cases: **(exercise 2)**

□

Lemma 4 (Preservation). *If $\Gamma \vdash t : T$ and $t \to t'$, then $\Gamma \vdash t' : T$.*

Proof. By structural induction on t.

- $t = x$ or $t = \lambda x : T.t_1$: Vacuously satisfied.
- $t = t_1\ t_2$: In this case, we have

$$\Gamma \vdash t_1 : T_{11} \to T_{12} \qquad \Gamma \vdash t_2 : T_{11} \qquad T = T_{12}$$

  Looking at the evaluation rules, we find that there are three possible cases for $t \to t'$:
  - E-App1: In this case $t' = t_1'\ t_2$ where $t_1 \to t_1'$ and the induction hypothesis is

$$\Gamma \vdash t_1' : T_{11} \to T_{12}$$

    Combining this with $\Gamma \vdash t_2 : T_{11}$, we can apply T-App to conclude that $\Gamma \vdash t' : T$
  - E-App2: Similar.
  - E-AppAbs: In this case we have

$$t_1 = \lambda x : T_{11}.t_{12} \qquad t_2 = v_2 \qquad t' = [x \mapsto v_2]t_{12}$$

    We also have

$$\Gamma[x \mapsto T_{11}] \vdash t_{12} : T_{12}$$

    and, by $\Gamma \vdash v_2 : T_{11}$ and the substitution lemma, we obtain

$$\Gamma \vdash t' : T_{12}$$

- Other cases: **(exercise 3)**

□