COMP3610/636
Principles of Programming Languages

Peter Höfner

Jul 20, 2023

Section 3

Types

Type systems

- describe when programs make sense
- prevent certain kinds of errors
- structure programs
- guide language design

Ideally, **well-typed programs do not get stuck**.

Run-time errors

Trapped errors

Cause execution to halt immediately.
Examples: jumping to an illegal address, raising a top-level exception.
Innocuous?

Untrapped errors

May go unnoticed for a while and later cause arbitrary behaviour.
Examples: accessing data past the end of an array, security loopholes in Java abstract machines.
Insidious!

Given a precise definition of what constitutes an untrapped run-time error, then a language is safe if all its syntactically legal programs cannot cause such errors. Usually, safety is desirable. Moreover, we'd like as few trapped errors as possible.

Formal type systems

We define a ternary relation $\Gamma \vdash E : T$

expression E has type T, under assumptions Γ on the types of locations that may occur in E.

For example (according to the definition coming up):

- $\{\} \vdash$ **if** `true` **then** 2 **else** $3 + 4$: int
- l_1 : intref $\vdash$ **if** $!l_1 \geq 3$ **then** $!l_1$ **else** 3 : int
- $\{\} \nvdash 3 + \texttt{true} : T$ for any type T
- $\{\} \nvdash$ **if** `true` **then** 3 **else** `true` : int

Types of IMP

Types of expressions

$$T \;::=\; \text{int} \mid \text{bool} \mid \text{unit}$$

Types of locations

$$T_{loc} \;::=\; \text{intref}$$

We write T and T_{loc} for the sets of all terms of these grammars.

- Γ ranges over TypeEnv, the finite partial function from $\mathbb{L} \rightharpoonup \mathbb{Z}$
- notation: write $l_1 : \text{intref}, \ldots, l_k : \text{intref}$ instead of $\{l_1 \mapsto \text{intref}, \ldots, l_k \mapsto \text{intref}\}$

Type Judgement (1 of 3)

(int) $\Gamma \vdash n : \textsf{int} \qquad \text{if } n \in \mathbb{Z}$

(bool) $\Gamma \vdash b : \textsf{bool} \qquad \text{if } b \in \mathbb{B} = \{\texttt{true}, \texttt{false}\}$

(op+) $$\frac{\Gamma \vdash E_1 : \textsf{int} \qquad \Gamma \vdash E_2 : \textsf{int}}{\Gamma \vdash E_1 + E_2 : \textsf{int}}$$

(op≥) $$\frac{\Gamma \vdash E_1 : \textsf{int} \qquad \Gamma \vdash E_2 : \textsf{int}}{\Gamma \vdash E_1 \geq E_2 : \textsf{bool}}$$

(if) $$\frac{\Gamma \vdash E_1 : \textsf{bool} \qquad \Gamma \vdash E_2 : T \qquad \Gamma \vdash E_3 : T}{\Gamma \vdash \textbf{if } E_1 \textbf{ then } E_2 \textbf{ else } E_3 : T}$$

Type Judgement – Example

Prove that $\{\} \vdash \textbf{if}\ \texttt{false}\ \textbf{then}\ 2\ \textbf{else}\ 3 + 4 : \mathsf{int}$.

$$
\dfrac{
\dfrac{}{\{\} \vdash \texttt{false} : \mathsf{bool}}\ (\text{BOOL})
\qquad
\dfrac{}{\{\} \vdash 2 : \mathsf{int}}\ (\text{INT})
\qquad
\dfrac{
(\text{INT})\ \dfrac{}{\{\} \vdash 3 : \mathsf{int}}
\quad
\dfrac{}{\{\} \vdash 4 : \mathsf{int}}\ (\text{INT})
}{\{\} \vdash 3 + 4 : \mathsf{int}}\ (\text{OP}+)
}{\{\} \vdash \textbf{if}\ \texttt{false}\ \textbf{then}\ 2\ \textbf{else}\ 3 + 4 : \mathsf{int}}\ (\text{IF})
$$

Type Judgement (2 of 3)

(assign) $$\frac{\Gamma(l) = \mathsf{intref} \qquad \Gamma \vdash E : \mathsf{int}}{\Gamma \vdash l := E : \mathsf{unit}}$$

(deref) $$\frac{\Gamma(l) = \mathsf{intref}}{\Gamma \vdash\ !l : \mathsf{int}}$$

Here, (for the moment) $\Gamma(l) = \mathsf{intref}$ means $l \in \mathsf{dom}(\Gamma)$

Type Judgement (3 of 3)

(skip) $\Gamma \vdash \textbf{skip} : \text{unit}$

(seq) $$\frac{\Gamma \vdash E_1 : \text{unit} \qquad \Gamma \vdash E_2 : T}{\Gamma \vdash E_1 \; ; \; E_2 : T}$$

(while) $$\frac{\Gamma \vdash E_1 : \text{bool} \qquad \Gamma \vdash E_2 : \text{unit}}{\Gamma \vdash \textbf{while } E_1 \textbf{ do } E_2 : \text{unit}}$$

Type Judgement – Properties

Theorem (Progress)

If $\Gamma \vdash E : T$ *and dom*$(\Gamma) \subseteq$ *dom*(s) *then either* E *is a value or there exist* E' *and* s' *such that* $\langle E\,,\, s\rangle \longrightarrow \langle E'\,,\, s'\rangle$*.*

Theorem (Type Preservation)

If $\Gamma \vdash E : T$*, dom*$(\Gamma) \subseteq$ *dom*(s) *and* $\langle E\,,\, s\rangle \longrightarrow \langle E'\,,\, s'\rangle$ *then* $\Gamma \vdash E' : T$ *and dom*$(\Gamma) \subseteq$ *dom*(s')*.*

Type Safety

Main result: Well-typed programs do not get stuck.

Theorem (Type Safety)

If $\Gamma \vdash E : T$, *dom*$(\Gamma) \subseteq$ *dom*(s), *and* $\langle E\,,\,s\rangle \longrightarrow^* \langle E'\,,\,s'\rangle$ *then either* E' *is a value with* $\Gamma \vdash E' : T$, *or there exist* E'', s'' *such that* $\langle E'\,,\,s'\rangle \longrightarrow \langle E''\,,\,s''\rangle$, $\Gamma \vdash E'' : T$ *and dom*$(\Gamma) \subseteq$ *dom*(s'').

Here, $\longrightarrow^*$ means arbitrary many steps in the transition system.

Type checking, typeability, and type inference

Type checking problem for a type system:
given Γ, E and T, is $\Gamma \vdash E : T$ derivable?

Type inference problem:
given Γ and E, find a type T such that $\Gamma \vdash E : T$ is derivable, or show there is none.

Type inference is usually harder than type checking, for a type T needs to be computed.

For our type system, though, both are easy.

Properties

Theorem (Type inference)

Given Γ and E , one can find T such that $\Gamma \vdash E : T$, or show that there is none.

Theorem (Decidability of type checking)

Given Γ, E and T, one can decide whether $\Gamma \vdash E : T$ holds.

Moreover

Theorem (Uniqueness of typing)

If $\Gamma \vdash E : T$ and $\Gamma \vdash E : T'$ then $T = T'$.