Lwnn Typing Rules

CS 260, Fall 2013

1 Typing Rules

1.1 Helpers

We use a ClassTable object to embody the necessary global set of classes. Our ClassTable is adapted from the one used in the FetherweightJava paper and is a map from class names to class declarations. Providing a mechanism for looking up field and method types for a given class. A program then is a pair (CT, e), we also make the assumption that the ClassTable is fixed.

$$lookup : ClassTable \times ClassT \to Class$$
$$lookup(ct, C) = ct[C]$$

$$method : ClassT \times Var \to Type$$
$$method(C, m) = classTable.lookup(C).method(m)$$

$$field : ClassT \times Var \to Type$$
$$field(C, f) = classTable.lookup(C).field(f)$$

1.2 Subtyping

$$A <: A \qquad \text{(Reflexivity)}$$

$$\frac{C <: B \qquad B <: A}{C <: A} \qquad \text{(Transitivity)}$$

$$\frac{class\ C\ extends\ D\ \{...\}}{C <: D} \qquad \text{(Inheritance)}$$

1.3 Classes

$$\frac{\Gamma \vdash}{todo} \qquad \text{(T-Class)}$$

1.4 Methods

$$\frac{\Gamma \vdash}{todo} \qquad \text{(T-Method)}$$

1.5 Statements

$$\frac{\Gamma \vdash x : \tau \qquad \Gamma \vdash e : \tau}{\Gamma \vdash x := e : \texttt{null}} \qquad \text{(T-Assign)}$$

$$\frac{\Gamma \vdash e_1 : C \qquad field(C, x) = \tau_f \qquad \Gamma \vdash e_2 : \tau_f}{\Gamma \vdash e_1.x := e_2 : \texttt{null}} \qquad \text{(T-Update)}$$

$$\frac{\Gamma \vdash e : C \qquad method(C, m) = \vec{\tau_d} \to \tau_r \qquad \Gamma \vdash \vec{e_i} : \vec{\tau_c} \qquad \vec{\tau_c} <: \vec{\tau_d} \qquad \Gamma \vdash x : \tau_r}{\Gamma \vdash x := e.m(\vec{e_i}) : \texttt{null}}$$

(T-Method-Invocation)

$$\frac{\Gamma \vdash x : D \qquad \Gamma \vdash e : C \qquad method(C, \text{"}C\text{"}) = \vec{\tau_i} \to C \qquad \vec{e_i} : \vec{\tau_i} \qquad C <: D \qquad \textbf{new } C(\vec{e_i}) : D}{\Gamma \vdash x := \textbf{new } C(\vec{e_i}) : \texttt{null}}$$

(T-New)

$$\frac{\Gamma \vdash e : \texttt{bool}}{\Gamma \vdash \textbf{if } (e) \ \vec{s_1} \ \textbf{else } \vec{s_2} : \texttt{null}} \qquad \text{(T-If)}$$

$$\frac{\Gamma \vdash e : \texttt{bool}}{\Gamma \vdash \textbf{while } (e) \ \vec{s} : \texttt{null}} \qquad \text{(T-While)}$$

1.6 Expressions

$$\Gamma \vdash i : \texttt{int} \qquad \text{(T-Int)}$$

$$\Gamma \vdash str : \texttt{string} \qquad \text{(T-String)}$$

$$\Gamma \vdash \textbf{true} : \texttt{bool} \qquad \text{(T-True)}$$

$$\Gamma \vdash \textbf{false} : \texttt{bool} \qquad \text{(T-False)}$$

$$\Gamma \vdash \textbf{null} : \texttt{null} \qquad \text{(T-Nulls)}$$

$$\Gamma \vdash x : \Gamma(x) \qquad \text{(T-Var)}$$

$$\frac{\Gamma \vdash e : C \qquad field(C, f) = \tau}{\Gamma \vdash e.f : \tau} \qquad \text{(T-Access)}$$

$$\frac{\oplus \in \{+, -, *, \div, <, \leq\} \qquad \Gamma \vdash e_1 : \texttt{int} \qquad \Gamma \vdash e_2 : \texttt{int}}{\Gamma \vdash e_1 \oplus e_2 : \texttt{int}} \text{ (T-NumOps)}$$

$$\frac{\oplus \in \{\wedge, \vee\} \qquad \Gamma \vdash e_1 : \texttt{bool} \qquad \Gamma \vdash e_2 : \texttt{bool}}{\Gamma \vdash e_1 \oplus e_2 : \texttt{bool}} \text{ (T-BoolOps)}$$

$$\frac{\oplus \in \{=, \neq\} \qquad \Gamma \vdash e_1 : \tau_1 \qquad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash e_1 \oplus e_2 : \texttt{bool}} \text{ (T-EqOps)}$$