Softwaretechnik

Vorlesung 03: Types and Type Soundness

Peter Thiemann

Universität Freiburg, Germany

SS 2008

Inhalt

Typen und Typkorrektheit
JAUS: Java-Ausdrücke
Auswertung von Ausdrücken
Typkorrektheit
Ergebnis

Typen und Typkorrektheit

- Große Softwaresysteme : viele Beteiligte
  - Projektmanager, Designer, Programmierer, ...
- Essentiell: Aufteilung in Komponenten mit klar definierter Schnittstelle und Spezifikation
  - Problemaufteilung
  - Arbeitsaufteilung
  - Testaufteilung
- Probleme
  - Gibt es geeignete Bibliothekskomponenten?
  - Passen die erstellten Komponenten zusammen?
  - Erfüllen sie ihre Spezifikation?

Forderungen

- Programmiersprache bzw -umgebung muss sicherstellen
  - Komponente implementiert alle Schnittstellen
  - Implementierung erfüllt die Spezifikation
  - Korrekte Verwendung der Komponente
- Grundproblem: Einhalten von Schnittstellen und Spezifikationen
  - Einfachste Schnittstelle: *Namensmanagement*
    Welche Operationen stellt die Komponente bereit?
  - Einfachste Spezifikation: *Typen*
    Welchen Typ haben die Argumente und Ergebnisse der Operationen?
  - Vgl. Interfaces in Java

Fragen

- Welche Sicherheit bieten Typen?
- Welche Fehler können prinzipiell mit Typen erkannt werden?
- Wie lässt sich Typsicherheit garantieren?
- Wie lässt sich Typsicherheit formalisieren?

JAUS: Java-Ausdrücke

Die Sprache JAUS beschreibt eine Teilmenge der Java-Ausdrücke

$$
\begin{array}{lcll}
x & ::= & \ldots & \text{Variable} \\
n & ::= & 0 \mid 1 \mid \ldots & \text{Zahlen} \\
b & ::= & \texttt{true} \mid \texttt{false} & \text{Wahrheitswerte} \\
e & ::= & x \mid n \mid b \mid e{+}e \mid !e & \text{Ausdrücke}
\end{array}
$$

Korrekte und inkorrekte Ausdrücke

- Typkorrekte Ausdrücke

```
boolean flag;
...
        0
        true
        17+4
        !flag
```

Korrekte und inkorrekte Ausdrücke

- Typkorrekte Ausdrücke

```
boolean flag;
...
        0
        true
        17+4
        !flag
```

- Ausdrücke mit Typfehler

```
int rain_since_April20;
boolean flag;
...
        !rain_since_April20
        flag+1
        17+(!false)
        !(2+3)
```

Typregeln

- Für jede Art von Ausdruck gibt es eine Typregel, die besagt,
  - wann ein Ausdruck typkorrekt ist und
  - wie sich der Ergebnistyp des Ausdrucks aus den Typen seiner Teilausdrücke bestimmen lässt.
- Fünf Arten von Ausdrücken (zunächst verbal)
  - Zahlkonstanten n haben den Typ `int`.
  - Wahrheitswerte b haben den Typ `boolean`.
  - Der Ausdruck e_1+e_2 hat den Typ `int`, aber nur falls e_1 und e_2 ebenfalls den Typ `int` haben.
  - Der Ausdruck $!e$ hat den Typ `boolean`, aber nur falls e auch den Typ `boolean` hat.
  - Eine Variable x hat den Typ, mit dem sie deklariert ist.

Formalisierung "Typkorrekte Ausdrücke"

Die Typsprache

$$t \quad ::= \quad \texttt{int} \mid \texttt{boolean} \qquad \text{Typen}$$

Typurteil: Ausdruck *e* hat Typ *t*

$$\vdash e : t$$

Formalisierung von Typregeln

- Ein Typurteil ist *gültig*, falls es mit Hilfe von *Typregeln* herleitbar ist.
- Zur Herleitung eines gültigen Typurteils J dient ein *Deduktionssystem*.
- Ein Deduktionssystem besteht aus einer Menge von Typurteilen und einer Menge von Typregeln.
- Eine Typregel (*Inferenzregel*) ist ein Paar $(J_1 \dots J_n, J_0)$ aus einer Liste von Urteilen (den *Voraussetzungen*) und einem Urteil (der *Folgerung*), geschrieben als
$$\frac{J_1 \dots J_n}{J_0}$$
- Falls $n = 0$, heißt die Regel (ε, J_0) *Axiom*.

Beispiel: Typregeln für JAUS

- Zahlkonstanten n haben den Typ `int`.

$$(\mathrm{INT})\ \frac{}{\vdash n : \mathtt{int}}$$

- Wahrheitswerte b haben den Typ `boolean`.

$$(\mathrm{BOOL})\ \frac{}{\vdash b : \mathtt{boolean}}$$

- Der Ausdruck e_1+e_2 hat den Typ `int`, aber nur falls e_1 und e_2 ebenfalls den Typ `int` haben.

$$(\mathrm{ADD})\ \frac{\vdash e_1 : \mathtt{int} \quad \vdash e_2 : \mathtt{int}}{\vdash e_1+e_2 : \mathtt{int}}$$

- Der Ausdruck $!e$ hat den Typ `boolean`, aber nur falls e auch den Typ `boolean` hat.

$$(\mathrm{NOT})\ \frac{\vdash e : \mathtt{boolean}}{\vdash !e : \mathtt{boolean}}$$

Herleitungsbäume und Gültigkeit

- Ein Urteil ist *gültig*, falls ein Herleitungsbaum dazu existiert.
- Ein *Herleitungsbaum mit Urteil J* ist definiert durch

  1. $\frac{}{J}$, falls $\frac{}{J}$ ein Axiom ist
  2. $\frac{\mathcal{J}_1 \dots \mathcal{J}_n}{J}$, falls $\frac{J_1 \dots J_n}{J}$ eine Regel ist und jedes der $\mathcal{J}_k$ ein Herleitungsbaum mit Urteil J_k ist.

Beispiel: Herleitungsbäume

- $(\mathrm{INT})\ \dfrac{}{\vdash 0 : \mathtt{int}}$ ist Herleitungsbaum mit Urteil $\vdash 0 : \mathtt{int}$.
- $(\mathrm{BOOL})\ \dfrac{}{\vdash \mathtt{true} : \mathtt{boolean}}$ ist Herleitungsbaum für $\vdash \mathtt{true} : \mathtt{boolean}$.
- Das Urteil $\vdash 17 + 4 : \mathtt{int}$ gilt aufgrund des Herleitungsbaums

$$(\mathrm{ADD})\ \dfrac{(\mathrm{INT})\ \dfrac{}{\vdash 17 : \mathtt{int}} \qquad (\mathrm{INT})\ \dfrac{}{\vdash 4 : \mathtt{int}}}{\vdash 17 + 4 : \mathtt{int}}$$

Variable

- Variable werden deklariert
- Sie müssen ihrer Deklaration gemäß verwendet werden.
- Diese Deklaration wird in einer *Typumgebung* oder *Typannahme* abgelegt.

$$A \quad ::= \quad \emptyset \mid A, x : t \qquad \text{Typumgebung}$$

- Ein (erweitertes) Typurteil enthält auch eine Typumgebung:
  In der Typumgebung A hat Ausdruck e den Typ t

$$A \vdash e : t$$

- Typregel für Variable:
  Eine Variable hat den Typ, mit dem sie deklariert ist

$$(\text{VAR})\ \frac{x : t \in A}{A \vdash x : t}$$

Erweiterung der restlichen Typregeln

- Die Typumgebung A wird nur durchgereicht.

$$\text{(INT)}\ \frac{}{A \vdash n : \texttt{int}}$$

$$\text{(BOOL)}\ \frac{}{A \vdash b : \texttt{int}}$$

$$\text{(ADD)}\ \frac{A \vdash e_1 : \texttt{int} \quad A \vdash e_2 : \texttt{int}}{A \vdash e_1{+}e_2 : \texttt{int}}$$

$$\text{(NOT)}\ \frac{A \vdash !e : \texttt{boolean}}{A \vdash e : \texttt{boolean}}$$

Beispiel: Herleitungsbaum mit Variable

Die Deklaration `boolean flag;` entspricht der Typannahme

$$A = \emptyset, \texttt{flag} : \texttt{boolean}$$

Damit

$$\cfrac{\cfrac{\texttt{flag} : \texttt{boolean} \in A}{A \vdash \texttt{flag} : \texttt{boolean}}}{A \vdash \texttt{!}\,\texttt{flag} : \texttt{boolean}}$$

Zwischenstand

- Formales System für
  - Syntax von Ausdrücken und Typen (KFG, BNF)
  - Typurteile
  - Gültigkeit von Typurteilen
- Ausstehende Fragen
  - Auswertung von Ausdrücken
  - Zusammenhang zwischen Auswertung und Typurteil

Auswertung von Ausdrücken

(Ein möglicher) Ansatz

- *Berechnungsrelation* $e \longrightarrow e'$ auf Ausdrücken
- Zwei Ausdrücke stehen in Relation, falls zwischen ihnen ein Unterschied von einem Berechnungsschritt besteht.
- Beispiel:
  - $5+2 \longrightarrow 7$
  - $(5+2)+14 \longrightarrow 7+14$

Ergebnisse von Berechnungen

- Ein *Wert* v ist entweder eine Zahl oder ein Wahrheitswert.
- Ein Ausdruck kann in mehreren Schritten einen Wert erreichen:
  - 0 Schritte: `0`
  - 1 Schritt: `5+2` $\longrightarrow$ `7`
  - 2 Schritte: `(5+2)+14` $\longrightarrow$ `7+14` $\longrightarrow$ `21`
- Oder eben nicht:
  - `!4711`
  - `1+false`
  - `(1+2)+false` $\longrightarrow$ `3+false`
- Diese Ausdrücke können keinen Berechnungsschritt ausführen und entsprechen daher Laufzeitfehlern.
- Beobachtung: Dies sind genau die Ausdrücke mit Typfehlern!

Formalisierung: Ergebnisse und Berechnungsschritte

- Ein *Wert* v ist entweder eine Zahl oder ein Wahrheitswert.

$$v ::= n \mid b \qquad \text{Werte}$$

- Einzelne Berechnungsschritte
  - Falls bei einer Addition beide Operanden Zahlen sind, so kann die Addition ausgeführt werden.

$$\text{(B-ADD)}\ \frac{}{\lceil n_1 \rceil + \lceil n_2 \rceil \longrightarrow \lceil n_1 + n_2 \rceil}$$

    $\lceil n \rceil$ ist die syntaktische Repräsentation der Zahl n.
  - Falls bei einer Negation der Operand ein Wahrheitswert ist, so kann die Negation ausgeführt werden.

$$\text{(B-TRUE)}\ \frac{}{!\texttt{true} \longrightarrow \texttt{false}} \qquad \text{(B-FALSE)}\ \frac{}{!\texttt{false} \longrightarrow \texttt{true}}$$

Formalisierung: geschachtelte Berechnungsschritte

Was geschieht, wenn die Operanden einer Operation noch nicht Werte sind? Es werden zunächst die Teilausdrücke ausgewertet.

- Die Negation

$$\text{(B-NEG)}\ \frac{e \longrightarrow e'}{!e \longrightarrow\ !e'}$$

- Addition, erster Operand

$$\text{(B-ADD-L)}\ \frac{e_1 \longrightarrow e_1'}{e_1 + e_2 \longrightarrow e_1' + e_2}$$

- Addition, zweiter Operand (erst wenn erster Operand schon Wert ist)

$$\text{(B-ADD-R)}\ \frac{e \longrightarrow e'}{v + e \longrightarrow v + e'}$$

Variable

- Ein Ausdruck, der Variable enthält, kann nicht durch Berechnungsschritte zu einem Wert werden.
- Stattdessen eliminiere Variable durch *Substitution* von Werten.
- Eine Substitution $[v_1/x_1, \dots v_n/x_n]$ angewandt auf einen Ausdruck e geschrieben als
$$e[v_1/x_1, \dots v_n/x_n]$$
ersetzt in e jedes Vorkommen von x_i durch den zugehörigen Wert v_i.
- Beispiel
  - $(!\texttt{flag})[\texttt{false}/\texttt{flag}] \equiv !\texttt{false}$
  - $(\texttt{m}+\texttt{n})[25/\texttt{m}, 17/\texttt{n}] \equiv 25+17$

Typkorrektheit informell

- Typkorrektheit: Wenn für e ein Typ herleitbar ist, dann liefert e in endlich vielen Berechnungsschritten einen Wert.
- Insbesondere ist dabei **kein** Laufzeitfehler passiert.
- Für die Minisprache JAUS gilt auch die Umkehrung, aber im Allgemeinen nicht.
- Beweis in zwei Schritten (nach Wright und Felleisen).
  Angenommen e hat einen Typ, dann gilt:

  Progress: Entweder ist e ein Wert oder es gibt einen Berechnungsschritt für e.

  Preservation: Wenn $e \longrightarrow e'$, dann hat hat e' denselben Typ wie e.

Progress

Wenn $\vdash e : t$ herleitbar ist, dann ist e ein Wert oder es gibt e' mit $e \longrightarrow e'$.

Beweis

Induktion über den Herleitungsbaum von $\mathcal{J} = \vdash e : t$.

Falls $(\mathrm{INT})\ \dfrac{}{\vdash n : \texttt{int}}$ der letzte Schritt von $\mathcal{J}$ ist, dann ist *$e \equiv n$ ein Wert* (und $t \equiv \texttt{int}$).

Falls $(\mathrm{BOOL})\ \dfrac{}{\vdash b : \texttt{boolean}}$ der letzte Schritt von $\mathcal{J}$ ist, dann ist *$e \equiv b$ ein Wert* (und $t \equiv \texttt{boolean}$).

Progress: Addition

Falls $(\mathrm{ADD})\ \dfrac{\vdash e_1 : \mathtt{int} \quad \vdash e_2 : \mathtt{int}}{\vdash e_1{+}e_2 : \mathtt{int}}$ der letzte Schritt von $\mathcal{J}$ ist, dann ist $e \equiv e_1{+}e_2$ und $t \equiv \mathtt{int}$. Ferner sind $\vdash e_1 : \mathtt{int}$ und $\vdash e_2 : \mathtt{int}$ herleitbar.

Nach Induktionsvoraussetzung ist entweder e_1 ein Wert oder es gibt $e_1 \longrightarrow e_1'$. Falls $e_1 \longrightarrow e_1'$, dann gilt nach $(\mathrm{B\text{-}ADD\text{-}L})$ auch $\equiv e_1{+}e_2 \longrightarrow \equiv e_1'{+}e_2$.

Falls $e_1 \equiv v_1$ ein Wert ist, so betrachte $\vdash e_2 : \mathtt{int}$. Nach Induktionsvoraussetzung ist entweder e_2 ein Wert oder es gilt $e_2 \longrightarrow e_2'$. Falls $e_2 \longrightarrow e_2'$, dann gilt nach $(\mathrm{B\text{-}ADD\text{-}R})$ auch $\equiv v_1{+}e_2 \longrightarrow \equiv v_1{+}e_2'$.

Falls e_2 ein Wert v_2 ist, so ist leicht zu prüfen, dass $v_1 \equiv n_1$ und $v_2 \equiv n_2$ beide Zahlen sind und daher ein Berechnungsschritt nach Regel $(\mathrm{B\text{-}ADD})$ durchführbar ist.

Progress: Negation

Falls $(\mathrm{NOT})\ \dfrac{\vdash e_1 : \texttt{boolean}}{\vdash !e_1 : \texttt{boolean}}$ der letzte Schritt von $\mathcal{J}$ ist, dann ist $e \equiv !e_1$ und $t \equiv \texttt{boolean}$. Ferner ist $\vdash e_1 : \texttt{boolean}$ herleitbar.
Nach Induktionsvoraussetzung ist entweder e_1 ein Wert oder es gibt $e_1 \longrightarrow e_1'$. Falls $e_1 \longrightarrow e_1'$, dann gilt nach $(\mathrm{B\text{-}NEG})$ auch $!e_1 \longrightarrow !e_1'$.
Falls $e_1 \equiv v_1$ ein Wert ist, so ist leicht zu prüfen, dass $v_1 \equiv b_1$ ein Wahrheitswert ist und daher ein Berechnungsschritt nach Regel $(\mathrm{B\text{-}TRUE})$ oder $(\mathrm{B\text{-}FALSE})$ durchführbar ist.

QED

Preservation

Wenn $\vdash e : t$ und $e \longrightarrow e'$, dann $\vdash e' : t$.

Beweis

Induktion über die Herleitung von $e \longrightarrow e'$.

Falls $(\mathrm{B\text{-}ADD})\ \dfrac{}{\lceil n_1 \rceil + \lceil n_2 \rceil \longrightarrow \lceil n_1 + n_2 \rceil}$ der Berechnungsschritt ist, dann muss (wegen (ADD)) $t \equiv \mathtt{int}$ sein. Für $e' = \lceil n_1 + n_2 \rceil$ liefert die Regel (INT) sofort $\vdash \lceil n_1 + n_2 \rceil : \mathtt{int}$.

Falls $(\mathrm{B\text{-}TRUE})\ \dfrac{}{!\mathtt{true} \longrightarrow \mathtt{false}}$ der Berechnungsschritt ist, dann muss (wegen (NOT)) $t \equiv \mathtt{boolean}$ sein. Für $e' = \mathtt{false}$ liefert die Regel (BOOL) sofort $\vdash \mathtt{false} : \mathtt{boolean}$.
Der Fall der Regel $\mathrm{B\text{-}FALSE}$ ist analog.

Preservation: Addition

Falls $(\text{B-ADD-L})\ \dfrac{e_1 \longrightarrow e_1'}{e_1{+}e_2 \longrightarrow e_1'{+}e_2}$ den Berechnungsschritt begründet, dann muss der letzte Schritt von $\vdash e : t$ gerade

$$(\text{ADD})\ \frac{\vdash e_1 : \texttt{int} \quad \vdash e_2 : \texttt{int}}{\vdash e_1{+}e_2 : \texttt{int}}$$

mit $e \equiv e_1{+}e_2$ und $t \equiv \texttt{int}$ sein. Aus $\vdash e_1 : \texttt{int}$ und $e_1 \longrightarrow e_1'$ folgt nach Induktion $\vdash e_1' : \texttt{int}$, so dass eine erneute Anwendung von (ADD) auf $\vdash e_1' : \texttt{int}$ und $\vdash e_2 : \texttt{int}$ genau $\vdash e_1'{+}e_2 : \texttt{int}$ liefert.
Der Fall der Regel (B-ADD-R) ist analog.

Preservation: Negation

Falls (B-NEG) $\dfrac{e_1 \longrightarrow e_1'}{!e_1 \longrightarrow !e_1'}$ den Berechnungsschritt begründet, dann muss der letzte Schritt von $\vdash e : t$ gerade

$$(\mathrm{NOT})\ \frac{\vdash e_1 : \mathtt{boolean}}{\vdash !e_1 : \mathtt{boolean}}$$

mit $e \equiv !e_1$ und $t \equiv \mathtt{boolean}$ sein. Aus $\vdash e_1 : \mathtt{boolean}$ und $e_1 \longrightarrow e_1'$ folgt nach Induktion $\vdash e_1' : \mathtt{boolean}$, so dass eine erneute Anwendung von (NOT) auf $\vdash e_1' : \mathtt{boolean}$ genau $\vdash !e_1' : \mathtt{boolean}$ liefert.

QED

Elimination von Variablen durch Substitution

Ziel

Wenn $x_1 : t_1, \dots, x_n : t_n \vdash e : t$ und $\vdash v_i : t_i$ (für alle i), dann gilt $\vdash e[v_1/x_1, \dots, v_1/x_1] : t$.

Aussage

Wenn $A', x_0 : t_0 \vdash e : t$ und $A' \vdash e_0 : t_0$, dann gilt $A' \vdash e[e_0/x_0] : t$.

Beweis

Induktion über die Herleitung von $A \vdash e : t$, wobei $A \equiv A', x_0 : t_0$.

Falls $(\mathrm{VAR})\ \dfrac{x : t \in A}{A \vdash x : t}$ der letzte Schritt der Herleitung ist, gibt es zwei Fälle: Entweder ist $x \equiv x_0$ oder nicht.

Falls $x \equiv x_0$ ist, dann ist $e[e_0/x_0] \equiv e_0$ und nach (VAR) ist $t \equiv t_0$. Nach Voraussetzung gilt dann sofort $A' \vdash e_0 : t_0$.

Falls $x \not\equiv x_0$ ist, dann ist $e[e_0/x_0] \equiv x$ und es gilt $x : t \in A'$. Nach Regel (VAR) gilt nun $A' \vdash x : t$.

Substitution: Konstanten

Falls $(\mathrm{INT})\ \dfrac{}{A \vdash n : \mathtt{int}}$ der letzte Schritt ist, so gilt auch

$(\mathrm{INT})\ \dfrac{}{A' \vdash n : \mathtt{int}}$.

Falls $(\mathrm{BOOL})\ \dfrac{}{A \vdash b : \mathtt{boolean}}$ der letzte Schritt ist, so gilt auch

$(\mathrm{BOOL})\ \dfrac{}{A' \vdash b : \mathtt{boolean}}$.

Substitution: Addition

Falls $(\mathrm{ADD})\ \dfrac{A \vdash e_1 : \mathtt{int} \quad A \vdash e_2 : \mathtt{int}}{A \vdash e_1 + e_2 : \mathtt{int}}$ der letzte Schritt ist, so liefert die Induktionsvoraussetzung, dass $A' \vdash e_1[e_0/x_0] : \mathtt{int}$ und $A' \vdash e_2[e_0/x_0] : \mathtt{int}$. Darauf lässt sich (ADD) anwenden und liefert $A' \vdash (e_1 + e_2)[e_0/x_0] : \mathtt{int}$.

Substitution: Negation

Falls $(\mathrm{NOT})\ \dfrac{A \vdash e_1 : \mathtt{boolean}}{A \vdash !e_1 : \mathtt{boolean}}$ der letzte Schritt ist, so liefert die Induktionsvoraussetzung, dass $A' \vdash e_1[e_0/x_0] : \mathtt{boolean}$. Darauf lässt sich wieder Regel (NOT) anwenden und liefert $A' \vdash (!e_1)[e_0/x_0] : \mathtt{boolean}$.

QED

Satz: Typkorrektheit für JAUS

- Wenn $\vdash e : t$, dann gibt es einen Wert v mit $\vdash v : t$ und Berechnungsschritte

$$e_0 \longrightarrow e_1, e_1 \longrightarrow e_2, \ldots, e_{n-1} \longrightarrow e_n$$

  so dass $e \equiv e_0$ und $e_n \equiv v$ ist.
- Wenn Variable vorhanden sind, so muss für sie zunächst typkorrekt eingesetzt werden.