Vorlesung vom 16.05.06: Generierung von Verifikationsbedingungen

Wo sind wir?

- *Vorletzte Woche*: Korrektheit des Hoare-Kalküls
- *Letzte Woche*: Vollständigkeit des Hoare-Kalküls
- *Diese Woche*: Verifikationsbedingungen

Ziel

- *Ziel*: maschinelle Verifikation von Code

  Eingabe: $\{A\}c\{B\}$

  Ausgabe: `True` wenn $\models \{A\}c\{B\}$
  `False` wenn $\neg \models \{A\}c\{B\}$
- Warum nicht $w[\![c, B]\!]$ *nutzen*?

$$\models \{A\}c\{B\} \Longleftrightarrow \models w[\![c, B]\!]$$

- Problem: nicht *handhabbar*.

Annotierte Programme

- Idee: Programme mit *Zusicherungen* annotieren
  - An der Schleife: *Invariante*
- *Annotierte Anweisungen* ($\mathbf{Com}_A$):

$$\begin{aligned} c ::= \;& \texttt{skip} \mid x := a \mid \\ & c; (x := a) \mid c; \{D\}c_0 \\ & \texttt{if } b \texttt{ then } c \texttt{ else } c \texttt{ fi} \mid \texttt{while } b \texttt{ do } \{D\}c \texttt{ od} \end{aligned}$$

  mit c_0 *keine* Zuweisung
- Idee: *Zusicherung* ist an dieser Stelle immer *wahr*

Verifikationsbedingungen

$vc : \mathbf{Assn} \times \mathbf{Com}_A \times \mathbf{Assn} \to \mathbb{P}(\mathbf{Assn})$

$$\begin{aligned} vc(\{A\}\texttt{skip}\{B\}) &= \{A \Longrightarrow B\} \\ vc(\{A\}X := a\{B\}) &= \{A \Longrightarrow B\,[a/X]\} \\ vc(\{A\}c; X := a\{B\}) &= vc(\{A\}c\{B\,[a/X]\}) \\ vc(\{A\}c; \{D\}c_0\{B\}) &= vc(\{A\}c\{D\}) \cup vc(\{D\}c_0\{B\}) \\ vc(\{A\}\texttt{if } b \texttt{ then } c_1 \texttt{ else } c_2 \texttt{ fi}\{B\}) &= vc(\{A \wedge b\}c_1\{B\}) \\ &\quad \cup vc(\{A \wedge \neg b\}c_2\{B\}) \\ vc(\{A\}\texttt{while } b \texttt{ do } \{D\}c \texttt{ od}\{B\}) &= vc(\{D \wedge b\}c\{D\}) \\ &\quad \cup \{A \Longrightarrow D\} \\ &\quad \cup \{D \wedge \neg b \Longrightarrow B\} \end{aligned}$$

Korrektheit der Verifikationsbedingungen

*Satz (*Korrektheit*):* Wenn für alle $D \in vc\{A\}c\{B\}$, $\models D$, dann $\models \{A\}c\{B\}$.

Beweis: Wir zeigen $\vdash \{A\}c\{B\}$ mit struktureller Induktion über $c \in \mathbf{Com}_A$.

- $c \equiv \texttt{skip}$
- $c \equiv X := a$
- $c \equiv c_0; \{D\}c_1$
- $c \equiv \texttt{if } b \texttt{ then } c_1 \texttt{ else } c_2 \texttt{ fi}$
- $c \equiv \texttt{while } b \texttt{ do } \{D\}c \texttt{ od}$

Zusammenfassung

- *Konstruktion* von $w[\![c, B]\!]$ aus *Vollständigkeitsnbeweis* unbrauchbar.
- Daher: mit *Zusicherungen annotierte Programme*
- Daraus: Generierung von *Verifikationsbedingungen*
- Wenn alle *Verifikationsbedingungen* beweisbar, Programm *korrekt*.
- Aber: Verifikationsbedingungen können *unbeweisbar* sein, wenn
  - Programm *falsch*
  - Zusicherungen *unerfüllbar*
- Schwachpunkt: *Grammatik*