Karlsruher Institut für Technologie

Tutorium 42, #14

Max Göckel– *uzkns@student.kit.edu*

Institut für Theoretische Informatik - Grundbegriffe der Informatik

Hoare-Kalkül: Tripel

Ein Hoare-Tripel besteht aus zwei Zusicherungen und einem Stück Programmcode:

- {P}: Vorbedingung
  - P soll wahr sein bevor der Programmcode beginnt
- S: Codesegment
  - Ein Stück Algorithmus - eine oder mehrere Zeilen vom Code
- {Q}: Nachbedingung
  - Ist P wahr und wurde S ausgeführt so soll nun Q wahr sein.

Eine Hoare-Zusicherung hat also die $\{P\}S\{Q\}$

Fakultät für Informatik
Institut für Theoretische Informatik

Hoare-Kalkül: Tripel

Für die Tripel gelten bestimmte Regeln (HT_1, HT_2, HT_3):

- HT_1 Gilt $P' \rightarrow P \wedge Q \rightarrow Q'$ und ist $\{P\}S\{Q\}$ korrekt, so ist auch $\{P'\}S\{Q'\}$ korrekt
- HT_2 $\{P\}S_1\{Q\}$ und $\{Q\}S_2\{R\}$ sind korrekt $\rightarrow$ $\{P\}S_1S_2\{R\}$ ist korrekt
- HT_3 Ist eine eine Zuweisung $x \leftarrow \beta$, so kann man aus der Nachbed. eine Vorbed. erstellen indem man x mit β substituiert

Hoare-Kalkül: Verzweigungen

In Algortihmen gibt es auch Verzweigungen der Form `if B then S else T`. Diese können mit Hoare-Tripeln ausgewertet werden.

$\{P\}$ Verzweigung $\{Q\}$ wahr $\Leftrightarrow$ $\{P \land B\}S\{Q\}$ u. $\{P \land \neg B\}T\{Q\}$ wahr
d.h. für eine Verzweigung sind beide Fälle wahr wenn man die Bedigung im `if`-Teil umdreht.

Hoare-Kalkül: Beweise

"Zeigen sie, dass das Hoare-Tripel $\{P\}S\{Q\}$ korrekt ist".

Vorgehen:

- S in atomare Befehle zerlegen (dabei unten Anfang und dann hocharbeiten)
- Für jeden Befehl ein Tripel finden und angeben
- Tripel und Code verbinden
- Sonderfall Verzweigungen:
  - Jeden Fall abarbeiten
  - Fallbedigung B herausfinden
  - Einzelne Fälle in $\{P \wedge B\}S\{Q\}$ u. $\{P \wedge \neg B\}T\{Q\}$ einsetzen
- Umformen, bis $\{P\}S\{Q\}$ herauskommt
- Fertig.

Gezeigt am Beispiel von Aufgabe 6.2. an der Tafel

Formale Sprachen

Gehen wir zurück zu Tutorium #3 so sind die formalen Sprachen wie folgt definiert:

Definition

- Eine formale Sprache F über einem Alphabet A ist eine Teilmenge der Kleenschen Hülle A^*

Aber was heißt das?

Fakultät für Informatik
Institut für Theoretische Informatik

Die Menge A^*

A^* ist die Menge aller Wörter beliebiger Länge über einem Alphabet A.

Ist $A = \{a, b, c\}$ so sind zum Beispiel

- a
- abc
- cab
- abcabcabaacabcabcabcacccccabcabcabcabcabcabcbbabcbcabc
- ϵ

alle in A^*

Formale Sprachen als Teilmenge von A^*

Eine formale Sprache ist dabei eine Teilmenge von A^*, also ausgesuchte Elemente aus dieser Menge.

Sei A unser bekanntes Alphabet (a-zA-Z) inkl. $0, 1, 2, 3, 4, 5, 6, 7, 8, 9$ und B alle Sonderzeichen mit Punkt und Komma.

Beispielhafte Sprachen über $A \cup B$ sind definiert als:

- $ABC = \{a, b, c\}$
- $website = \{www. \cdot w_1 \cdot . \cdot w_2 | w_1, w_2 \in A \wedge |w_2| \leq 3\}$
- $DCIM = \{\text{IMG\_ number } .jpeg | \text{number} \in \{0, ..., 9\} *\}$
- $count_2 = \{a^n b^n | n \in \mathbb{N}\}$

Formale Sprachen: Operationen

Auf formalen Sprachen gibt es Produkte, Potenzen und Konkatenationsabschlüsse:

- L_1, L_2 form. Sprachen, so ist $L_1 \cdot L_2 = \{w_1 \cdot w_2 | w_1 \in L_1, w_2 \in L_2\}$ also
- $\{a, b\} \cdot \{c, d\} = \{ac, ad, bc, bd\}$

- $L^0 = \{\epsilon\}$ und
- $\forall k \in \mathbb{N}_0 : L^{k+1} = L \cdot L^k$

- $L^* = \bigcup_{i \in \mathbb{N}_0} L^i$
- Jedes Wort aus L bel. oft mit jedem anderen Wort aus L kombiniert

Formale Sprachen: Operationen

Auf formalen Sprachen gibt es Produkte, Potenzen und Konkatenationsabschlüsse:

- L_1, L_2 form. Sprachen, so ist $L_1 \cdot L_2 = \{w_1 \cdot w_2 | w_1 \in L_1, w_2 \in L_2\}$ also
- $\{a, b\} \cdot \{c, d\} = \{ac, ad, bc, bd\}$
- $L^0 = \{\epsilon\}$ und
- $\forall k \in \mathbb{N}_0 : L^{k+1} = L \cdot L^k$
- $L^* = \bigcup_{i \in \mathbb{N}_0} L^i$
- Jedes Wort aus L bel. oft mit jedem anderen Wort aus L kombiniert

Klasusurthemen

Alle Themen aus GBI:

- Mengen
- Relationen
- Wörter
- Formale Sprachen
- Kontextfreie Grammatiken
- Induktion
- Aussagenlogik
- Prädikatenlogik
- Huffman
- Speicher
- MIMA
- Hoare-Kalkül
- Graphen
- O-Kalkül/Mastertheorem
- Automaten
- RegEx
- Turingmaschinen

Speicher

Karlsruher Institut für Technologie

Definition

- $memwrite : Val^{Adr} x Adr x Val \rightarrow Val^{Adr}$.
  - $(m, a, v) \mapsto m'$

Definition

- $memread : Val^{Adr} x Adr \rightarrow Val$
  - $(m, a) \mapsto m(a)$

Die Operationen können auch verkettet werden. an jedem Val^{Adr} kann *memwrite* einsetzt werden um die Tabelle vor dem Lesen zu verändern.