Einführung in die Theoretische Informatik

Woche 12

Harald Zankl

Institut für Informatik @ UIBK

Wintersemester 2014/2015

Zusammenfassung der letzten LV

Definition

Eine Registermaschine (RM) R ist ein Paar $R = ((x_i)_{1 \leqslant i \leqslant n}, P)$ sodass

1. $(x_i)_{1 \leqslant i \leqslant n}$ Sequenz von Registern x_i, die natürliche Zahlen beinhalten
2. P ein Programm

Befehle:

1. $x_i := x_i + 1$
2. $x_i := x_i - 1$
3. while $x_i \neq 0$ do P_1 end

Satz

Jede partielle Funktion $f \colon \mathbb{N}^k \to \mathbb{N}$, die berechenbar auf einer RM ist, ist auf einer TM berechenbar und umgekehrt.

Inhalte der Lehrveranstaltung

Einführung in die Logik

Syntax & Semantik der Aussagenlogik, Formales Beweisen, Konjunktive und Disjunktive Normalformen

Einführung in die Algebra

Boolesche Algebra, Universelle Algebra, Logische Schaltkreise

Einführung in die Theorie der Formalen Sprachen

Grammatiken und Formale Sprachen, Reguläre Sprachen, Kontextfreie Sprachen

Einführung in die Berechenbarkeitstheorie

Algorithmisch unlösbare Probleme, Turing Maschinen, Registermaschinen

Einführung in die Programmverifikation

Prinzipien der Analyse von Programmen, Verifikation nach Hoare, Verschlüsselung und Sicherheit

Wozu Programmverifikation

- Ariane-5
- Fehler in der Datenkonvertierung
- USD 370 Millionen

- Intel Pentium FDIV-Bug
- Falsche Berechnungen
- USD 475 Millionen

- Gepäckverteilung (Denver)
- Desaster
- USD 560 Millionen

- Blue Screen of Death
- ziemlich lästig

Prinzipien der Analyse von Programmen

Begutachtung

- Code wird von ähnlich qualifizierten Programmierern kontrolliert
- subtile Fehler werden leicht übersehen

Testen

- dynamische Technik, bei der das Programm ausgeführt wird
- Wie wird die richtige Testumgebung geschaffen?

Formal Methoden

- statische Technik, Eigenschaft von Programm wird formal bewiesen
- Vorteile
  - frühe Integration der Verifikation in die Softwareentwicklung
  - effektiv (höhere Erkennensrate von Fehlern als andere Methoden)
  - effizient (im Besonderen wenn automatisierbar)

Prädikatenlogik

Prädikatenlogik (informell)

- Die Prädikatenlogik ist eine Logik, deren Ausdruckskraft über die der Aussagenlogik weit hinausgeht
- Die wichtigste Erweiterung sind Prädikatensymbole
- Prädikatensymbole erlauben es uns, über Elemente einer Menge Aussagen zu treffen

Sprache einer Prädikatenlogik

Im Allgemeinen wird eine Prädikatenlogik durch eine Sprache beschrieben, diese Sprache enthält:

1. Funktionssymbole und Prädikatensymbole (mit Stelligkeit)
2. Variablen
3. $\underbrace{=}_{\text{Gleichheit}}, \underbrace{\neg, \wedge, \vee, \rightarrow}_{\text{Junktoren}}, \underbrace{\forall, \exists}_{\text{Quantoren}}$

Definition

Ein Ausdruck der mit Hilfe von Variablen und Funktionssymbolen gebildet wird, heißt Term.

Beispiel

Seien $0, 1, 2, \ldots$ Konstanten (Funktionssymbole) und $+$ ein Funktionssymbol mit Stelligkeit 2. Dann sind 0, $0+1$, $(0+1)+2$ Terme.

Definition

Sei P ein Prädikatensymbol mit Stelligkeit n und seien $t_1, \ldots, t_n$ Terme. Wir nennen Ausdrücke $P(t_1, \ldots, t_n)$ und $t_1 = t_2$ Atome.

Beispiel

- Sei 7 eine Konstante und ist_prim ein Prädikatensymbol.
- Dann ist ist_prim(7) ein Atom.

Zusicherungen

Definition

Wir definieren Zusicherungen induktiv:

1. Atome sind Zusicherungen
2. Wenn A und B Zusicherungen sind, dann sind auch die folgenden Ausdrücke, Zusicherungen:

$$\neg A \quad (A \wedge B) \quad (A \vee B) \quad (A \rightarrow B)$$

Konvention

Zusicherungen werden Formeln genannt

Definition

Interpretationen $\mathcal{I}$ werden verwendet, um den Ausdrücken der Prädikatenlogik eine Bedeutung zu geben.

Wahrheit einer Zusicherung

Beispiel

- Betrachte Konstante 7, Prädikatsymbol ist_prim, Funktionssymbol $+$
- Interpretation $\mathcal{I}$ legt z.B. fest, dass
  - 7 als die Zahl sieben zu verstehen ist,
  - ist_prim(n) genau dann wahr ist, wenn n eine Primzahl,
  - $+$ die Addition in den natürlichen Zahlen ist.

Beobachtung

1. Durch Interpretationen wird die Bedeutung von Atomen fixiert.
2. Ist die Bedeutung der Atome durch $\mathcal{I}$ fixiert, wird der Wahrheitswert einer Formel durch die Bedeutung der Junktoren bestimmt.

Beispiel

In der Interpretation $\mathcal{I}$ wird die Formel $\text{ist\_prim}(x) \wedge x = 7$ wahr, $\text{ist\_prim}(x + x) \wedge x = 7$ aber falsch.

Definition

Sei $\mathcal{I}$ eine Interpretation und F eine Formel. Wir schreiben $\mathcal{I} \models F$, wenn die Formel F in der Interpretation $\mathcal{I}$ wahr ist.

Beispiel

Es gilt $\mathcal{I} \models x = 7 \rightarrow \text{ist\_prim}(x)$, aber $\mathcal{I} \not\models \text{ist\_prim}(x) \rightarrow x = 7$.

Definition

Die Konsequenzrelation $A \models B$ gilt, gdw. für alle Interpretationen $\mathcal{I}$:

$$\mathcal{I} \models A \text{ impliziert } \mathcal{I} \models B$$

Im Folgenden schränken wir $\mathcal{I}$ auf die *natürliche Interpretation* ein, d.h. die Symbole werden interpretiert wie erwartet.

Beispiel

$x_1 > 4 \models x_1 + 1 > 5$ ✓ $\quad x_1 > 4 \models x_1 + 1 > 4$ ✓ $\quad x_1 > 4 \models x_1 > 5$ ✗

$x_1 + 1 > 5 \models x_1 > 4$ ✓ $\quad x_1 + 1 > 4 \models x_1 > 4$ ✗ $\quad x_1 > 5 \models x_1 > 4$ ✓

Hoare-Tripel

Definition

- Sei P ein while-Programm (ein Programm einer Registermaschine)
- Seien Q und R Zusicherungen
- Ein Hoare-Tripel ist wie folgt definiert:

$$\{Q\}\ P\ \{R\}$$

- Q wird Vorbedingung
- R wird Nachbedingung genannt

Beispiel

$\{x_1 > 4\}\ x_1 := x_1 + 1\ \{x_1 > 5\}$ ist ein Hoare-Tripel.

Definition

- Ein Hoare-Tripel $\{Q\}\ P\ \{R\}$ ist wahr, wenn:
  Wenn die Vorbedingung Q vor der Ausführung von P gilt, dann gilt die Nachbedingung R nach der Ausführung von P.
- Wenn $\{Q\}\ P\ \{R\}$ wahr, dann ist P korrekt in Bezug auf Q und R.
- Dann sagen wir auch P ist partiell korrekt.
- P ist total korrekt, wenn partiell korrekt und terminierend.

Beispiel

Die folgenden Hoare-Tripel

$$\{x_1 > 4\}\ x_1 := x_1 + 1\ \{x_1 > 5\} \qquad \{x_2 = 0\}\ x_2 := x_2 - 1\ \{x_2 = 0\}$$

sind wahr und die jeweiligen Programme total korrekt.

Hoare-Kalkül

Definition

Die Regeln des Hoare-Kalkül sind wie folgt definiert:

$$\frac{}{\{Q\{x \mapsto t\}\}\ x := t\ \{Q\}}\ [z] \qquad \frac{\{Q'\}\ P\ \{R'\}}{\{Q\}\ P\ \{R\}}\ [a], Q \models Q', R' \models R$$

$$\frac{\{Q\}\ P_1\ \{R\} \quad \{R\}\ P_2\ \{S\}}{\{Q\}\ P_1; P_2\ \{S\}}\ [s] \qquad \frac{\{I \wedge B\}\ P\ \{I\}}{\{I\}\ \text{while } B \text{ do } P \text{ end } \{I \wedge \neg B\}}\ [w]$$

Satz

Ist ein Hoare-Tripel in diesem Kalkül ableitbar, dann ist es wahr.

Beispiel

$$\cfrac{\cfrac{}{\{x_1 + 1 > 5\}\ x_1 := x_1 + 1\ \{x_1 > 5\}}\ [z]}{\{x_1 > 4\}\ x_1 := x_1 + 1\ \{x_1 > 5\}}\ [a], x_1 > 4 \models x_1 + 1 > 5$$

Beispiel

Wir betrachten das folgende while-Programm P:

while $x_1 \neq 0$ do
  $x_1 := x_1 - 1$
end

und zeigen $\{x_1 \geqslant 0\}\ P\ \{x_1 = 0\}$

$$\cfrac{\cfrac{\cfrac{\cfrac{}{\{x_1 - 1 \geqslant 0\}\ x_1 := x_1 - 1\ \{x_1 \geqslant 0\}}\ [z]}{\{x_1 \geqslant 0 \wedge x_1 \neq 0\}\ x_1 := x_1 - 1\ \{x_1 \geqslant 0\}}\ [a]}{\{x_1 \geqslant 0\}\ P\ \{x_1 \geqslant 0 \wedge x_1 = 0\}}\ [w]}{\{x_1 \geqslant 0\}\ P\ \{x_1 = 0\}}\ [a]$$

wir verwenden:

1. $x_1 \geqslant 0 \wedge x_1 = 0 \models x_1 = 0$
2. die Schleifeninvariante $x_1 \geqslant 0$
3. $x_1 \geqslant 0 \wedge x_1 \neq 0 \models x_1 - 1 \geqslant 0$

Automatisierung von Verifikationstechniken

Frage

Kann Verifikation vollständig automatisiert werden?

Antwort

Nein

- für totale Korrekheit muss Termination nachgewiesen werden
- Termination ist ein unentscheidbares Problem (Halteproblem)

Frage

Was tun?

Antwort

- Termination per Hand beweisen, oder
- der Verifikator für Termination ist partiell

Bemerkung

Termination von (imperativen) Programmen wird von folgenden Tools untersucht:

AProVE, COSTA, Julia, SPEED, Terminator, T_TT_2, ...

Bemerkung

Es können auch andere (unentscheidbare) Eigenschaften von (manchen) Programmen automatisch verifiziert werden, etwa der Speicherbedarf, die Effizienz, etc.

AProVE, COSTA, LOOPUS, RaML, SPEED, T_CT, ...

Nebenbemerkung

Diese Tools müssen natürlich unvollständig sein.

Planänderung

Letzte Vorlesung

Mittwoch, 28.01.2015, 12:15–14:00, HS A