Prof. Dr. Jasmin Blanchette
Jannis Limperg
Lydia Kondylidou

Ludwig-Maximilians-Universität München
Institut für Informatik
Abgabe bis 28. Juni 2023, 10:00 Uhr

Übung 10 zur Vorlesung

Formale Sprachen und Komplexität

Erlaubte Konstrukte für

LOOP-Programme	WHILE-Programme	GOTO-Programme
$x_i := x_j + c$	$x_i := x_j + c$	$M_k : x_i := x_j + c$
$x_i := x_j - c$	$x_i := x_j - c$	$M_k : x_i := x_j - c$
$x_i := c$	$x_i := c$	$M_k : x_i := c$
$P_1; P_2$	$P_1; P_2$	$P_1; P_2$
LOOP x_i **DO** P **END**	**LOOP** x_i **DO** P **END**	$M_k :$ **GOTO** M_j
	WHILE $x_i \neq 0$ **DO** P **END**	$M_k :$ **IF** $x_i = c$ **THEN GOTO** M_j
		$M_k :$ **HALT**

Dabei darf jede Marke M_k nur einmal vorkommen

FSK10-1 ***WHILE-, LOOP- und GOTO-Programme*** (2 Punkte)

In den folgenden Teilaufgaben sollen Sie jeweils ein Programm angeben. Verwenden Sie dabei nur die erlaubten Anweisungen aus der obigen Tabelle. Kommentieren Sie außerdem Ihre Programme, sodass klar wird, wie sie funktionieren sollen. Nicht kommentierte Programme werden eventuell nicht gewertet.

a) Schreiben Sie ein GOTO-Programm, das der Variablen x_0 den Wert von $x_1 + x_2$ zuweist.

b) Schreiben Sie ein WHILE-Programm, das die folgende Funktion berechnet:

$$f : \mathbb{N}^2 \to \mathbb{N}, \quad f(x, y) = x * y + 1$$

c) Schreiben Sie ein LOOP-Programm, das den folgenden Pseudocode implementiert:

IF $x_1 \leq x_2$ **THEN** $x_0 := x_2 - x_1$ **ELSE** $x_0 := x_1 - x_2$ **END**

FSK10-2 ***LOOPY-Programme*** (2 Punkte)

Wir betrachten LOOPY-Programme, die aus folgenden Anweisungen bestehen:

- $x_i := x_j + c$, $x_i := x_j - c$ und $P_1; P_2$ wie bei LOOP-Programmen.

- $(P_1 \mid P_2)$, wobei P_1 und P_2 LOOPY-Programme sind. Diese Anweisung führt nichtdeterministisch entweder P_1 oder P_2 aus.
- **LOOPY** P **END**. Diese Anweisung führt das LOOPY-Programm P nichtdeterministisch 0 oder mehr Male aus.

Beispielsweise führt das LOOPY-Programm

$$\textbf{LOOPY}\ (x_0 := x_0 + 2 \mid x_0 := x_0 - 1)\ \textbf{END}$$

beliebig oft entweder die Anweisung $x_0 := x_0 + 2$ oder die Anweisung $x_0 := x_0 - 1$ aus, wobei es in jeder Iteration eine andere Entscheidung treffen kann. Der Wert von x_0 am Ende des Programms kann also jede natürliche Zahl sein.

a) Die Semantik eines LOOPY-Programms können wir als eine Relation $\xrightarrow[\text{LOOPY}]{}$ definieren. Diese Relation ist ähnlich der für WHILE-Programme, aber LOOPY-Programme können nichtdeterministisch verschiedene Ergebnisse haben. Dementsprechend kann ein LOOPY-Programm P mit einer Variablenbelegung ρ mehrere Nachfolge-Variablenbelegungen ρ' und mehrere Nachfolge-Programme P' haben, je nachdem, welche nichtdeterministischen Entscheidungen das Programm trifft. Für alle diese ρ' und P' soll gelten:

$$(\rho, P) \xrightarrow[\text{LOOPY}]{} (\rho', P')$$

Definieren Sie die Semantik von LOOPY-Programmen.

b) Zeigen Sie, dass mit Ihrer Semantik gilt:

$$(\{x_0 \mapsto 0\}, \textbf{LOOPY}\ (x_0 := x_0 + 2 \mid x_0 := x_0 - 1)\ \textbf{END}) \xrightarrow[\text{LOOPY}]{}{}^{*} (\{x_0 \mapsto 1\}, \varepsilon)$$

c) Eine Funktion $f\colon \mathbb{N}^k \to \mathbb{N}$ ist LOOPY-berechenbar, wenn es ein LOOPY-Programm P gibt sodass es für alle $n_1, \ldots, n_k \in \mathbb{N}$ eine Variablenbelegung ρ gibt, für die gilt:

$$(\{x_1 \mapsto n_1, \ldots, x_k \mapsto n_k\}, P) \xrightarrow[\text{LOOPY}]{}{}^{*} (\rho, \varepsilon)$$

und $\rho(x_0) = f(n_1, \ldots, n_k)$.

Zeigen Sie: jede Funktion, die LOOP-berechenbar ist, ist auch LOOPY-berechenbar.

FSK10-3 ***FOR-Programme*** (0 Punkte)

a) Wir betrachten FOR-Programme. Diese haben dieselbe Syntax wie LOOP-Programme ohne das **LOOP**-Konstrukt, dafür gibt es ein **FOR**-Konstrukt:

$$\textbf{FOR } x_i := 1 \textbf{ TO } x_k \textbf{ DO } P \textbf{ END}$$

wobei P ein FOR-Programm ist, welches die Variable x_i nicht neu setzt (d.h. $x_i := \ldots$ kommt in P nicht vor, aber lesende Zugriffe $x_j := x_i + c$ und $x_j := x_i - c$ (mit $j \neq i$) sind erlaubt). Die informelle Semantik der **FOR**-Schleife ist, dass sie das Programm P N-Mal durchläuft, wobei N der Wert von x_k vor Eintritt in die FOR-Schleife ist und im j-ten Durchlauf $x_i := j$ gilt. Nach Beenden der Durchläufe gilt $x_i := N$.

Definieren Sie die operationale Semantik der FOR-Programme formal (analog zur operationalen Semantik der LOOP-Programme, siehe Abschnitt 10.1.2 im Skript), indem Sie einen Berechnungsschritt $(\rho, P) \underset{\text{FOR}}{\longrightarrow} (\rho', P')$ definieren, wobei ρ eine Variablenbelegung und P ein FOR-Programm ist. Dabei muss die Semantik der **FOR**-Schleife der obigen informellen Semantik entsprechen und alle anderen Konstrukte müssen dieselbe Semantik wie in LOOP-Programmen haben.

b) Eine Funktion $f : \mathbb{N}^k \to \mathbb{N}$ heißt *FOR-berechenbar*, wenn es ein FOR-Programm P gibt, sodass für alle $n_1, \ldots, n_k \in \mathbb{N}$ und Variablenbelegungen $\rho = \{x_1 \mapsto n_1, \ldots, x_k \mapsto n_k\}$ gilt: $(\rho, P) \underset{\text{FOR}}{\longrightarrow}^* (\rho', \varepsilon)$ und $\rho'(x_0) = f(n_1, \ldots, n_k)$, wobei $\underset{\text{FOR}}{\longrightarrow}^*$ für 0 oder beliebig viele Berechnungsschritte steht und ε das leere Programm ist.

Zeigen Sie, dass jede FOR-berechenbare Funktionen auch LOOP-berechenbar ist. Hierfür müssen Sie zeigen, dass jedes FOR-Programm als LOOP-Programm kodiert werden kann, wobei die berechnete Funktion dieselbe ist.

c) Zeigen sie, dass jede LOOP-berechenbare Funktion auch FOR-berechenbar ist. Hierfür müssen Sie zeigen, dass jedes LOOP-Programm als FOR-Programm kodiert werden kann, wobei die berechnete Funktion dieselbe ist.

FSK10-4 ***Turingmaschinen*** (0 Punkte)

a) Seien $M = \{w \mid \#_a(w) = \#_b(w)\}$ und $N = L(a^*b^*)$ Sprachen über $\Sigma = \{a, b\}$.
Zeigen Sie, dass es eine Funktion f gibt mit $\forall x \in \Sigma^* \,.\, x \in M \iff f(x) \in N$.

b) Geben Sie eine deterministische Turingmaschine an, welche N erkennt.

c) Was muss für f gelten, damit man daraus folgern kann, dass M von einer Turingmaschine erkannt werden kann (ohne die Turingberechenbarkeit von M direkt zu zeigen)?