PD Dr. Jan Johannsen
Elisabeth Lempa
Luca Maio

Ludwig-Maximilians-Universität München
Institut für Informatik
Besprechung 25.06.2026 bis 29.06.2026
Abgabe bis 07.07.2026, 14:00 Uhr

Übung 9 zur Vorlesung

Formale Sprachen und Komplexität

Hinweis:
Die letzte Aufgabe auf diesem Blatt ist eine Aufgabe zur Klausurvorbereitung. Diese Aufgabe orientiert sich in Form und inhaltlichen Schwerpunkten an den Klausuraufgaben. Dies bedeutet jedoch nicht, dass die anderen Aufgaben nicht klausurrelevant sind.

Die Lösungen der Klausurvorbereitungs-Aufgaben werden am Ende der Beabeitungszeit gesondert veröffentlicht, aber **nicht** im Tutorium besprochen. Die Lösungen der anderen Aufgaben werden bereits zu Beginn der Bearbeitungszeit veröffentlicht und können Ihnen bei der Bearbeitung helfen.

Wenn Sie Ihre Lösung innerhalb der Bearbeitungszeit über Moodle abgeben, erhalten Sie eine individuelle Korrektur. Die Abgabe ist freiwillig (aber nachdrücklich empfohlen).

Wenn Sie Automaten angeben, tun Sie dies immer in Form eines Zustandsgraphen. Andere Formen der Darstellung (z.B. als Liste von Übergängen) werden nicht gewertet, da sie sehr viel aufwändiger zu korrigieren sind. Vergessen Sie nicht, im Zustandsgraph Start- und Endzustände zu markieren.

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

FSK9-1 ***WHILE- 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 $3 \cdot x_1 \cdot x_2$ zuweist.

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

$$f : \mathbb{N}^2 \to \mathbb{N}, \quad f(x, y) = 5 * (x + y)$$

FSK9-2 ***LOOPY-Programme***

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.

FSK9-3 ***μ-Rekursion***

Nehmen Sie für diese Aufgabe an, dass Addition, Subtraktion, Multiplikation, Division, Exponentiation und absolute Differenz $absdiff(x_1, x_2) = |x_1 - x_2|$ primitiv rekursiv sind.

Zur Erinnerung: Wir unterscheiden hier die (modifizierte) Subtraktion und die absoluten Differenz *absdiff*. (Wir schreiben in der folgenden Aufgabe zur besseren Unterscheidung $\dot{-}$ für die modifizierte Subtraktion.) Bei der (modifizierten) Subtraktion $x_1 \dot{-} x_2$ wird im Fall $x_2 > x_1$ auf 0 abgebildet, d.h. $3 \dot{-} 4 = 0$. Bei der absoluten Differenz werden dagegen x_1 und x_2 als ganze Zahlen subtrahiert, d.h. $|3 - 4| = 1$.

a) Berechnen Sie μg für folgende Funktionen g.

- $g\colon \mathbb{N} \to \mathbb{N}$, $g(x) = 3x^2 + 5x + 3$
- $g\colon \mathbb{N} \to \mathbb{N}$, $g(x) = (|x/2 - 4|)^2 \dot{-} 2$.
- $g\colon \mathbb{N}^2 \to \mathbb{N}$, $g(x_1, x_2) = (x_1 + 1) \cdot x_2$

b) Zeigen Sie, dass die Quadratwurzelfunktion

$$sqrt\colon \mathbb{N} \to \mathbb{N}, \quad sqrt(x_1) = \sqrt{x_1}$$

μ-rekursiv ist. Beachten Sie, dass die Quadratwurzel für natürliche Zahlen nicht überall definiert ist. Es gilt:

$$sqrt(x_1) = n \iff n^2 = x_1$$

FSK9-4 ***Primitiv rekursive Prädikate***

Primitiv rekursive Funktionen, die auf $\{0, 1\}$ abbilden, können auch als Prädikate aufgefasst werden. Wir betrachten in dieser Aufgabe nur einstellige Prädikate p, wobei $p(x) = 0$ bedeutet, dass x die Eigenschaft p nicht besitzt, und $p(x) = 1$ bedeutet, dass x die Eigenschaft p besitzt.

a) Zeigen Sie, dass die Funktion *iszero* ein primitiv rekursives Prädikat ist.

$$iszero(x_1) = \begin{cases} 0 & \text{für } x_1 > 0 \\ 1 & \text{für } x_1 = 0 \end{cases}$$

b) Zeigen Sie, dass die Funktion *even* ein primitiv rekursives Prädikat ist.

$$even(x_1) = \begin{cases} 1 & \text{für } x_1 \text{ gerade} \\ 0 & \text{für } x_1 \text{ ungerade} \end{cases}$$

c) Zeigen Sie, dass die Funktion *ifnotzero* primitiv rekursiv ist.

$$ifnotzero(x_1, x_2, x_3) = \begin{cases} x_2 & \text{falls } x_1 > 0 \\ x_3 & \text{sonst} \end{cases}$$

d) Zeigen Sie, dass die Funktion *ifthenelse* primitiv rekursiv ist, angenommen, dass $p(x)$ ein primitiv rekursives Prädikat ist.

$$ifthenelse(x_1, x_2, x_3) = \begin{cases} x_2 & \text{falls } p(x_1) \\ x_3 & \text{sonst} \end{cases}$$

e) Zeigen Sie, dass, gegeben zwei primitiv rekursive Prädikate p und q, auch die Konjunktion $p \wedge q$ primitiv rekursiv ist.

Klausurvorbereitung FSK-9-K

Zeigen Sie, dass die Funktion *odd* ein primitiv rekursives Prädikat ist:

$$odd(x_1) = \begin{cases} 0 & \text{für } x_1 \text{ gerade} \\ 1 & \text{für } x_1 \text{ ungerade} \end{cases}$$