Lehrstuhl für Informatik 1
Prof. Dr. Gerhard Woeginger
Jan Böker, Tim Hartmann WS 2018/19

Übung zur Vorlesung
Berechenbarkeit und Komplexität

Lösung Blatt 7

Hausaufgabe 7.1 **(2 + 2 Punkte)**

Welche der folgenden Fragen über multivariate Polynome $p : \mathbb{Z}^k \to \mathbb{Z}$ (mit ganzzahligen Koeffizienten) sind entscheidbar? Beweisen Sie die Korrektheit Ihrer Antworten.

(a) Besitzt p eine Nullstelle, in der alle Variablen natürliche Werte annehmen?

Es bezeichne Dioph($\mathbb{Z}$) das ursprüngliche Problem, ob ein solches Polynom p eine ganzzahlige Nullstelle besitzt, und Dioph($\mathbb{N}$) die Einschränkung auf Nullstellen, die ausschließlich aus natürlichen Zahlen bestehen, wobei die Koeffizienten der Polynome natürlich weiterhin ganzzahlig sein dürfen. Indem man zeigt, dass Dioph($\mathbb{Z}$) $\leq$ Dioph($\mathbb{N}$) gilt, folgt die Unentscheidbarkeit des Problems aus der Unentscheidbarkeit von Dioph($\mathbb{Z}$).

Sei $p(x_1, \ldots, x_k)$ eine Instanz für das Problem Dioph($\mathbb{Z}$). Wir setzen

$$f(p(x_1, \ldots, x_k)) = p'(x_1, x'_1, x_2, x'_2, \ldots, x_k, x'_k),$$

wobei

$$p'(x_1, x'_1, x_2, x'_2, \ldots, x_k, x'_k) = p(x_1 - x'_1, x_2 - x'_2, \ldots, x_k - x'_k)$$

und p' eine diophantische Gleichung ist. Für syntaktisch inkorrekte Eingaben x setzen wir $f(x) = x$.

Offensichtlich ist die Funktion f berechenbar.

Korrektheit: Sei zunächst $(a_1, \ldots, a_k) \in \mathbb{Z}^k$ eine Nullstelle von p. Für $i \in \{1, \ldots, k\}$ wähle $b_i, b'_i \in \mathbb{N}$ mit $a_i = b_i - b'_i$. Dann ist $(b_1, b'_1, \ldots, b_k, b'_k)$ eine Nullstelle von $f(p)$.

Für die Rückrichtung sei nun $(b_1, b'_1, \ldots, b_k, b'_k) \in \mathbb{N}^{2k}$ eine Nullstelle von $p' = f(p)$. Wir setzen $a_i = b_i - b'_i$. Dann ist $(a_1, \ldots, a_k) \in \mathbb{Z}^k$ eine Nullstelle von p.

(b) Besitzt p eine ganzzahlige Nullstelle, in der alle Variablenwerte zwischen -10^6 und 10^6 liegen?

Entscheidbar, da man nur endlich viele Nullstellenkandidaten hat: Diese $(2 \cdot 10^6 + 1)^k$ Nullstellenkandidaten kann man einfach nacheinander ausprobieren.

Hausaufgabe 7.2 (2 + 2 Punkte)

Zeigen Sie, dass folgende arithmetische Befehle durch ein LOOP-Programm simuliert werden können:

(a) $x_i \coloneqq x_j$ DIV x_k (Division ohne Rest, gegeben $x_k > 0$)

Wir suchen das kleinste x_i so dass $(x_i + 1) \cdot x_k > x_j$. Dazu probieren wir alle x_i von 0 bis x_j aus.

Wir verwenden, dass nach Vorlesung die Addition und das IF-x_i-=-0-THEN-ELSE-Konstrukt LOOP-berechenbar ist. Zudem ist nach Tutoriumsaufgabe 7.2 (b) die modifizierte Vorgängerfunktion und die Subtraktion LOOP-berechenbar. Es seien y, z, d Variablen, die sonst nicht verwendet werden: y wird verwendet, um den Wert $x_i \cdot x_k$ zu speichern. z wird nur dafür verwendet, $x_j + 1$ (statt x_j) Iterationen der LOOP-Schleife durchzuführen. d wird für den Vergleich von $y = x_i \cdot x_k$ und x_j in der IF-Abfrage verwendet.

$y := 0;$
$x_i := 0;$
$d := x_j + 1;$
$d := d \dot{-} y;$
$z := x_j + 1;$
LOOP z DO
    IF $d = 0$ THEN ELSE
        $y := y + x_k;$
        $x_i := x_i + 1;$
        $d := x_j + 1;$
        $d := d \dot{-} y$
    ENDIF
ENDLOOP;
$x_i := x_i \dot{-} 1$

Der Einfachheit halber wurde hinter dem THEN eine Dummy-Anweisung ausgelassen. Für formale Korrektheit könnte man dort z. B. $x_i \coloneqq x_i$ ergänzen.

Es genügen $x_j + 1$ Iterationen der LOOP-Schleife, da $(x_j + 1) \cdot x_k > x_j$ gilt. Im Fall $x_k = 1$ ist diese Anzahl auch tatsächlich notwendig.

(b) $x_i \coloneqq x_j$ MOD x_k (Modulo, gegeben $x_k > 0$)

Wir benutzen, dass wir in (a) gezeigt haben, dass es ein LOOP-Programm gibt, das die Division ohne Rest berechnet. Zudem ist nach Vorlesung die Multiplikation LOOP-berechenbar und nach Tutoraufgabe 7.2 (b) auch die Subtraktion.

$y \coloneqq x_j$ DIV $x_k;$
$y \coloneqq y \cdot x_k;$
$x_i := x_j \dot{-} y$

Hausaufgabe 7.3 (2 Punkte)

Die Programmiersprache LOOP-WHILE ist eine Kombination der beiden Programmiersprachen LOOP und WHILE. Die syntaktischen Komponenten

von LOOP-WHILE sind genau die Komponenten von LOOP zusammen mit den Komponenten von WHILE: LOOP-WHILE-Programme sind Zuweisungen, die Hintereinanderausführung von zwei LOOP-WHILE-Programmen, das LOOP-Konstrukt um ein LOOP-WHILE-Programm oder das WHILE-Konstrukt um ein LOOP-WHILE-Programm. In einem LOOP-WHILE-Programm darf allerdings das WHILE-Konstrukt nur höchstens einmal benutzt werden.

Beweisen oder widerlegen Sie: Die Programmiersprache LOOP-WHILE ist Turing-mächtig.

Der in der Vorlesung gezeigte Beweis für die Turing-Mächtigkeit von WHILE verwendet nur eine einzige WHILE-Schleife: Die IF-Abfragen und die MOD-Befehle in dieser WHILE-Schleife können über LOOP-Schleifen realisiert werden, vgl. Folien und Hausaufgabe 7.2 (b). Mit diesem Wissen folgt die Aussage direkt.

Hausaufgabe 7.4 **(3 + 3 Punkte)**

Bestimmen Sie die Wachstumsfunktionen $F_P : \mathbb{N} \to \mathbb{N}$ für die folgenden LOOP-Programme. Bestimmen Sie für jedes dieser LOOP-Programme P eine natürliche Zahl m_P, sodass $F_P(n) < A(m_P, n)$ für alle $n \in \mathbb{N}$ gilt. Beachten Sie, dass die folgenden LOOP-Programme Kurzschreibweisen verwenden, z. B. ist $x_2 := x_3 + 2$ Kurzschreibweise für $x_2 := x_3 + 1; x_2 := x_2 + 1$.

(a) $\boldsymbol{x_3 := x_2 + 3;}$
$\boldsymbol{x_1 := x_2 + 1;}$
$\boldsymbol{x_2 := x_3 + 2}$

Langschreibweise:

$x_3 := x_2 + 1;$
$x_3 := x_3 + 1;$
$x_3 := x_3 + 1;$
$x_1 := x_2 + 1;$
$x_2 := x_3 + 1;$
$x_2 := x_2 + 1$

P übersetzt den Eingabevektor (a_1, a_2, a_3) in den Ausgabevektor (a_2+1, a_2+5, a_2+3). Folglich gilt $f_P(a_1, a_2, a_3) = 3a_2 + 9$, was

$$\begin{aligned} F_P(n) &= \max\{3a_2 + 9 \mid a_1, a_2, a_3 \in \mathbb{N} \text{ mit } a_1 + a_2 + a_3 \leq n\} \\ &= 3n + 9 \end{aligned}$$

liefert.

Verwende die „Regeln“ aus dem Induktionsbeweis aus der Vorlesung. Es wird ignoriert, wie genau die Befehle durch die Hintereinanderausführung verschachtelt sind; dies spielt offensichtlich für die berechnete Funktion keine Rolle.

$$\left.\begin{array}{l}\left.\begin{array}{l}\left.\begin{array}{r} x_3 := x_2 + 1;\,\} < A(2,n) \\ x_3 := x_3 + 1;\,\} < A(2,n) \end{array}\right\} < A(3,n) \\ \left.\begin{array}{r} x_3 := x_3 + 1;\,\} < A(2,n) \\ x_1 := x_2 + 1;\,\} < A(2,n) \end{array}\right\} < A(3,n) \end{array}\right\} < A(4,n) \\ \left.\begin{array}{r} x_2 := x_3 + 1;\,\} < A(2,n) \\ x_2 := x_2 + 1;\,\} < A(2,n) \end{array}\right\} < A(3,n) \leq A(4,n) \end{array}\right\} < A(5,n)$$

Also gilt $F_P(n) < A(5,n)$.

(b) $\boldsymbol{x_3 := x_2;}$
LOOP $\boldsymbol{x_1}$ DO
  LOOP $\boldsymbol{x_3}$ DO $\boldsymbol{x_2 := x_2 + 1}$ ENDLOOP
ENDLOOP

P übersetzt den Eingabevektor (a_1, a_2, a_3) in den Ausgabevektor $(a_1, a_2 + a_1 \cdot a_2, a_2)$ Folglich gilt $f_P(a_1, a_2, a_3) = a_1 + 2a_2 + a_1 \cdot a_2$, was

$$F_P(n) = \max\{a_1 + 2a_2 + a_1 \cdot a_2 \mid a_1, a_2, a_3 \in \mathbb{N} \text{ mit } a_1 + a_2 + a_3 \le n\}$$

liefert. Dieses Maximum muss nun berechnet werden. Da das Polynom $a_1+2a_2+a_1\cdot a_2$ nur positive Koeffizienten hat, wählt man a_1 und a_2 stets möglich groß, um den Wert zu maximieren, d. h., für gegebenes a_1 wählt man $a_2 = n - a_1$. Damit erhält man

$$a_1 + 2(n - a_1) + a_1(n - a_1) = -a_1^2 + (n-1)a_1 + 2n.$$

Die Ableitung lautet $-2a_1 + n - 1$ und hat die Nullstelle $\frac{n-1}{2}$; da es sich bei der Funktion um eine gespiegelte Parabel handelt, wird an diesem Punkt das Maximum angenommen. Für den Fall, dass n gerade ist, muss man allerdings $\frac{n}{2}$ (oder alternativ $\frac{n}{2} - 1$, dann aber nur für $n \ge 2$) betrachten. Damit erhält man

$$F_P(n) = \begin{cases} \frac{1}{4}n^2 + \frac{3}{2}n + \frac{1}{4} & \text{falls } n \text{ ungerade} \\ \frac{1}{4}n^2 + \frac{3}{2}n & \text{falls } n \text{ gerade} \end{cases}.$$

Verwende die „Regeln“ aus dem Induktionsbeweis aus der Vorlesung:

$$\left.\begin{array}{l} x_3 := x_2;\} < A(2,n) \le A(4,n) \\ \left.\begin{array}{l} \text{LOOP } x_1 \text{ DO} \\ \left.\begin{array}{l} \quad\text{LOOP } x_3 \text{ DO} \\ \qquad x_2 := x_2 + 1\} < A(2,n) \\ \quad\text{ENDLOOP} \end{array}\right\} < A(3,n) \\ \text{ENDLOOP} \end{array}\right\} < A(4,n) \end{array}\right\} < A(5,n)$$

Also gilt $F_P(n) < A(5,n)$.