(Master) Berechenbarkeit

v4.0.4.2.4.3.2 Ackermannfunktion ist nicht primitiv rekursiv (JavaScript)

Kategory GmbH & Co. KG
Präsentiert von Jörg Kunze

Beschreibung

Inhalt. Die Ackermannfunktion ist nicht primitiv rekursiv. Wir haben aber schon gesehen, dass wir sie programmieren können.

Der Beweis ist konstruktiv: wir geben zu jeder primitiv-rekursiven Funktion eine Ackermann-Funktion an, welche die primitiv-rekursiven Funktion majorisiert.

Dies bilden wir programmatisch nach: wir Erweitern unser Programm zur Erzeugung primitiv-rekursiver Funktionen aus den Atomen um das Ermitteln und das Ausgeben der entsprechenden majorisierenden Ackermann-Funktion.

Was ist hier der Unterschied zwischen einer und der Ackermann-Funktion? Currying: Die Ackermann-Funktion ist eine Funktion mit zwei Argumenten. Wir betrachten den ersten als Parameter einer Schar von Funktionen in einer Variablen, indem wir zu jeder festen Wahl des ersten, nur den zweiten variieren.

Präsentiert. Von Jörg Kunze

Voraussetzungen. Primitiv rekursive Funktionen, Ackermann-Funktion und ihre Eigenschaften, Ackermann-Funktion ist nicht primitiv-rekursiv.

Text. Der Begleittext als PDF und als LaTeX findet sich unter `https://github.com/kategory/kategoryMathematik/tree/main/v4%20Master/v4.0%20Berechenbarkeit/v4.0.4.2.4.3.2%20Ackermannfunktion%20ist%20nicht%20primitiv%20rekursiv%20(JavaScript)`.

Quelltext. Der JavaScript-Quelltext ist hier zu finden: `https://github.com/kategory/kategoryEducation/blob/main/mathematics/computabilityInJavaScript/v4.0.4.2.4.3.2%20(Master)%20Berechenbarkeit%20-%20Ackermannfunktion%20ist%20nicht%20primitiv%20rekursiv%20in%20JavaScript.js`

Meine Videos. Siehe auch in den folgenden Videos:
v4.0.4.2.4.3 (Master) Berechenbarkeit - Ackermannfunktion ist nicht primitiv rekursiv
`https://youtu.be/7S2bJ6LGekM`

4.0.4.2.1 (Master) Berechenbarkeit - Rekursive Funktionen in JavaScript
`https://youtu.be/19a1OfW1bLk`

v4.0.4.2.4.2 (Master) Berechenbarkeit - Ackermannfunktion Eigenschaften
`https://youtu.be/MzDubMBlQ20`

v4.0.4.2.4.1 (Master) Berechenbarkeit - Ackermannfunktion ist mu-rekursiv
`https://youtu.be/cbZ8EJ4U-tg`

v4.0.4.2.4 (Master) Berechenbarkeit - Ackermannfunktionen
`https://youtu.be/XE-7YdHi2Vw`

v4.0.4.2 (Master) Berechenbarkeit - Rekursive Funktionen
`https://youtu.be/tFEn2GoLLEQ`

Quellen. Siehe auch in den folgenden Seiten:
`https://de.wikipedia.org/wiki/Ackermannfunktion`
`https://en.wikipedia.org/wiki/Ackermann_function`
`https://planetmath.org/ackermannfunctionisnotprimitiverecursive`
`https://planetmath.org/PropertiesOfAckermannFunction`

Buch. Grundlage ist folgendes Buch:
Computability
A Mathematical Sketchbook
Douglas S. Bridges
Springer-Verlag New York Inc. 2013
978-1-4612-6925-0 (ISBN)

1. Ackermannfunktion ist nicht primitiv rekursiv

1.1. Ackermann-Funktion.

Definition 1.1.1. (Ackermann-Funktion): Die Ackermann-Funktion $A(k,n)\colon \mathbb{N}^2 \to \mathbb{N}$, mit $\mathbb{N} = \{0,1,2,3,\cdot\}$, ist definiert durch

$$A(0,n) := n+1 \tag{1}$$

$$A(k+1,0) := A(k,1) \tag{2}$$

$$A(k+1,n+1) := A(k,A(k+1,n)). \tag{3}$$

1.2. Currying.

Wir machen aus der Funktion $A(k,n)\colon \mathbb{N}^2 \to \mathbb{N}$ für jedes k eine Funktion:

$$A(k,\_)\colon \mathbb{N} \to \mathbb{N} \tag{4}$$

$$n \mapsto A(k,n). \tag{5}$$

Dieses Vorgehen nennen wir Currying. Dass Currying liefert eine Bijektion:

$$\mathrm{Hom}(A \times B, C) \cong \mathrm{Hom}(A, \mathrm{Hom}(B,C)) \tag{6}$$

$$[f\colon A \times B \to C] \mapsto [f^*\colon A \to \mathrm{Hom}(B,C)] \tag{7}$$

$$[(a,b) \mapsto f(a,b)] \mapsto \begin{bmatrix} f^*(a)\colon & B & \to & C \\ & b & \mapsto & f(a,b) \end{bmatrix}. \tag{8}$$

1.3. Primitiv rekursiv.

Definition 1.3.1. (Primitiv rekursiv): Wir definieren induktiv als primitiv rekursiv:

- (**PR1**): $f \circ g$, falls f, g primitiv rekursiv sind
- (**PR2**): h, mit $h(0,x_1,\cdots,x_s) := f(x_1,\cdots,x_s)$ und h, mit $h(n+1,x_1,\cdots,x_s) := g(n,h(n,x_1,\cdots,x_s),x_1,\cdots,x_s)$, falls f, g primitiv rekursiv sind
- (**PR3**): Die konstanten Funktionen $c(n) := c$ für alle $c \in \mathbb{N}$
- (**PR4**): Die Projektionen $P_j(x_1,\cdots,x_s) := x_j$
- (**PR5**): Die Nachfolger-Funktion $s(n) := n+1$.

1.4. Der Parameter.

Gemäß des Beweises im Video „v4.0.4.2.4.3 (Master) Berechenbarkeit - Ackermannfunktion ist nicht primitiv rekursiv“ haben wir folgende Tabelle für den Parameter k.

- (**PR1**): $k_{f\circ g} := k_f + k_g + 2$
- (**PR2**): $k_{R(f,g)} := 5 + \max\{k_f, k_g\}$
- (**PR3**): $k_c := c$
- (**PR4**): $k_{P_j} := 0$
- (**PR5**): $k_s := 1$

Literatur

[Douglas2013] Douglas S. Bridges, *Computability, A Mathematical Sketchbook*, Springer, Berlin Heidelberg New York 2013, ISBN 978-1-4612-6925-0 (ISBN).

Symbolverzeichnis

$\mathbb{N}$	Die Menge der natürlichen Zahlen (mit Null): $\{0, 1, 2, 3, \cdots\}$
n, k, l, s, i, x_i, q	Natürliche Zahlen
$A(k, n)$	Ackermannfunktion
f, g, h	Funktionen
$s()$	Nachfolgerfunktion: $s(n) := n + 1$
$\mathbf{x}$	Vektor natürlicher Zahlen $(x_1, \cdots, x_s)$
$\|\mathbf{x}\|$	$\max\{x_1, \cdots, x_s\}$
$g > f$	Die Funktion g majorisiert die Funktion f
P_j	Die Projektionen $P_j(x_1, \cdots, x_s) := x_j$