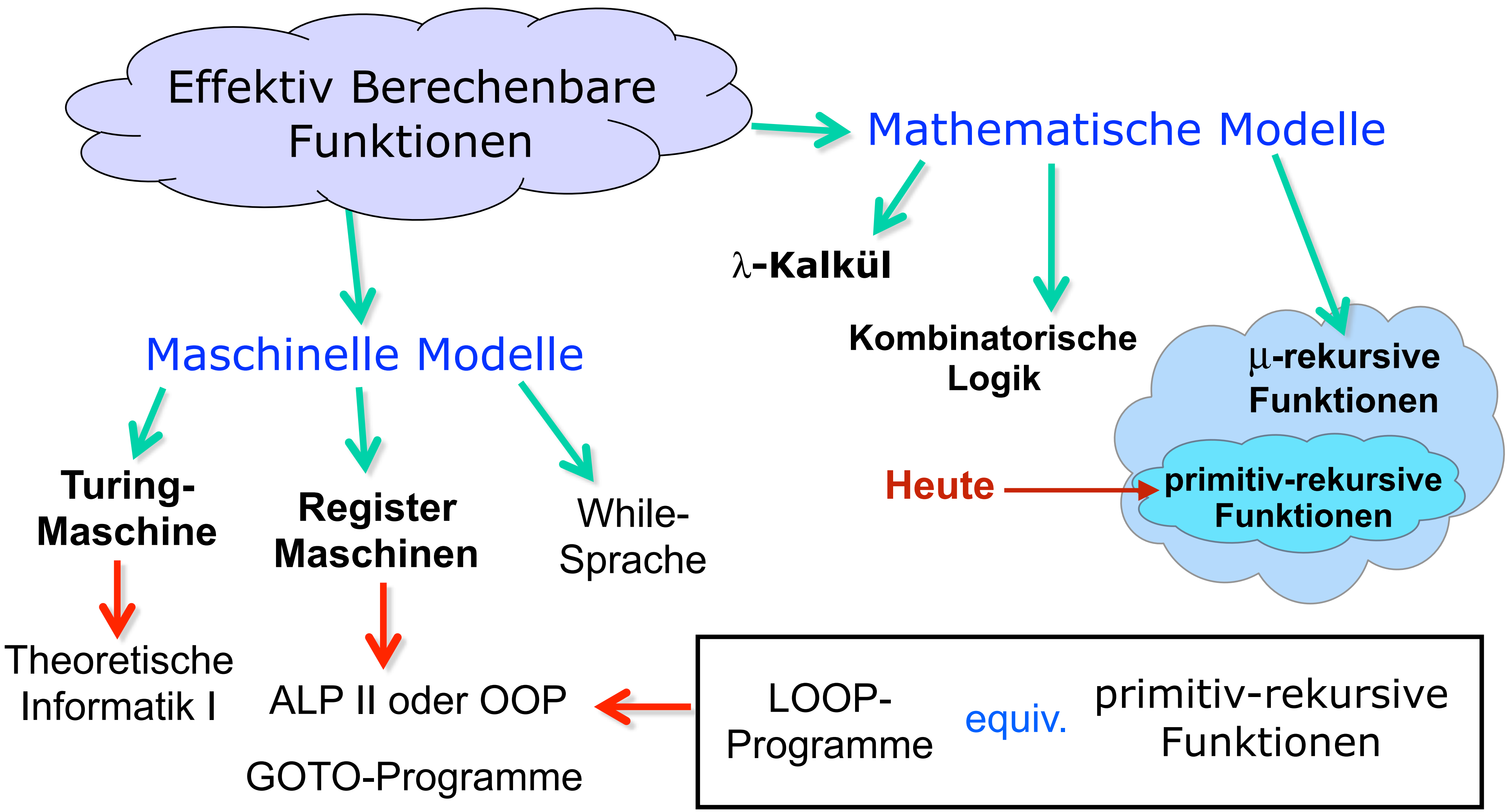
Funktionale Programmierung

Primitiv-Rekursive Funktionen

WS 2019/20

Prof. Dr. Margarita Esponda

Äquivalenz vieler Berechnungsmodelle



Primitiv-rekursive Funktionen

Kurt Gödel (1906-1978) und Rózsa Péter (1905-1977) haben sich mit der Theorie der rekursiven Funktionen auseinandergesetzt sowie diese stark geprägt.

Was kann alles mit primitiv-rekursiven Funktionen berechnet werden?

Gibt es ein allgemeines Schema zur Definition beliebiger berechenbarer Funktionen?

Primitiv-rekursive Funktionen

Der Begriff primitiv-rekursive Funktion wurde von Rózsa Péter geprägt.

Rózsa Péter

- Ungarische Mathematikerin (1905-1977)
- 1951, Buch "Rekursive Funktionen"
- 1955, Vereinfachung der Ackermann-Funktion

  "Ackermann-Péter-Funktion"

Primitiv-rekursive Funktionen

Fast alle Beispiele, die wir in die Vorlesungen diskutiert haben, sind Beispiele aus der Menge der sogenannten primitiv-rekursiven Funktionen **PR**.

Die primitiv-rekursiven Funktionen sind die einfachste Klasse von rekursiven Funktionen.

Die PR-Funktionen sind eine Untermenge der Effektiv Berechenbaren Funktionen. Das bedeutet, es gibt rekursive Funktionen, die berechenbar sind aber nicht primitiv-rekursiv sind.

Primitiv-rekursive Funktionen (PR)

Das Schema zur Definition beliebiger primitiv-rekursiver Funktionen besteht aus folgenden drei Hauptteilen:

1. Eine Reihe von Grundfunktionen.

2. Ein Ersetzungsmechanismus zur Definition von Funktionen ohne Rekursion (Funktionskomposition).

3. Ein Mechanismus zur Definition von primitiv-rekursiven Funktionen mit Rekursion (PR).

Primitiv-rekursive Funktionen (PR)

Die Klasse der **PR**-Funktionen $N^m \to N$ wird wie folgt induktiv definiert.

I Grundfunktionen

1. Die Nullfunktion $Z : N^m \to N$ ist primitiv rekursiv

   $$Z(x_1,\ldots,x_m) = 0$$

2. Die Nachfolgerfunktion $S : N \to N$ ist primitiv rekursiv

   $$S(n) = n+1$$

3. Die Projektionsfunktionen $\pi_i^m : N^m \to N$

   definiert durch $\pi_i^m (x_1, \ldots ,x_i, \ldots ,x_m) = x_i$, mit $1 \leq i \leq m$

   sind primitiv-rekursiv

   Beispiel: $\pi_2^3 (a, b, c) = b$

Primitiv-rekursive Funktionen (PR)

II Kompositionsschema

Die Funktionskomposition ist primitiv-rekursiv.

Das bedeutet, für alle primitiv-rekursiven Funktionen

$f : N^m \to N$ und $g_1, \ldots, g_m : N^n \to N$

ist die Funktion $C : N^n \to N$, definiert durch

$$C(x_1,\ldots,x_n) = f(g_1(x_1,\ldots,x_n),\ldots, g_m(x_1,\ldots,x_n))$$

auch primitiv rekursiv.

Primitiv-rekursive Funktionen

III Rekursionsschema

Jede Funktion, die sich durch primitive Rekursion (Induktion) aus primitiv-rekursiven Funktionen definieren lässt, ist auch primitiv-rekursiv.

Das bedeutet:

Wenn $g : N^m \rightarrow N$ und $h : N^{m+2} \rightarrow N$ primitiv-rekursive Funktionen sind, dann ist die folgende (induktiv definierte) Funktion $R : N^{m+1} \rightarrow N$

$$R\,(0, x_1,\ldots,x_m) = g(x_1,\ldots,x_m)$$

$$R\,(S(n), x_1,\ldots,x_m) = h(\,R(n, x_1,\ldots,x_m), n, x_1,\ldots,x_m\,)$$

ebenfalls primitiv rekursiv.

mit S = Nachfolgerfunktion

Primitiv-rekursive Funktionen

Eine Funktion heißt primitiv rekursiv, wenn sie zu den Grundfunktionen gehört oder aus diesen durch endliche viele Anwendungen von Komposition und primitiver Rekursion definiert werden kann.

Beispiele:

add2 = (+2) ist primitiv rekursiv:

$$c(x_1) = f(g_1(x_1),\ldots,g_n(x_1))$$

$$\boxed{add2(x_1) = (S \circ S)(\pi_1^1(x_1))}$$

✓ . . . aus II Kompositionsschema

Primitiv-rekursive Funktionen

Eine konstante Funktion, die eine beliebige Zahl n in der Konstante 3 abbildet, sieht wie folgt aus:

$$k3 : N \rightarrow N$$

aus I: S und Z sind primitiv-rekursiv

aus II: $(S \circ S \circ S \circ Z)$ ist primitiv-rekursiv

$$\boxed{k3(m) = S(S(S(Z(m)))) = 3}$$

✓

Freie Universität Berlin

Primitiv-rekursive Funktionen

Alle ***n***-stellige konstante Funktion mit Wert ***k*** sind primitiv-rekursive Funktionen.

$$C_k^n(x_1, x_2, \ldots, x_n) = k \quad \text{mit} \quad x_i\text{, k} \in \mathbb{N}$$

Hierbei lassen wir auch die 0-stelligen Konstanten C_k^0 zu, die man mit der Zahl ***k*** identifizieren kann.

Primitiv-rekursive Funktionen

Die Identitätsfunktion kann wie folgt definiert werden:

$id : N \rightarrow N$

$$\boxed{id(m) = \pi^1_1(m) = m}$$

✓

Freie Universität Berlin

Primitiv-rekursive Funktionen

Können wir die Additionsfunktion auf eine Definition, die nur aus primitiv-rekursiven Funktionen besteht, zurückführen?

$add : N^2 \rightarrow N$

haskell:

```
add    0      m  =   m
add  (n+1)  m  =   succ (add n m)
```

Wir müssen g und h (primitiv-rekursive Funktionen) finden.

III:

$$add(0, m) = g(m)$$
$$add(S(n), m) = h(add(n, m), n, m)$$

wir wählen $g = \pi_1^1$ und $h = S \circ \pi_1^3$

I, II und III:

$$add(0, m) = \pi_1^1(m)$$
$$add(S(n), m) = (S \circ \pi_1^3)(add(n, m), n, m))$$

✓

Primitiv-rekursive Funktionen

Vorgänger-Funktion: *pred* : $N \rightarrow N$

Weil die primitiv-rekursiven Funktionen nur über die natürlichen Zahlen definierbar sind, wird der Vorgänger von 0 als gleich 0 definiert.

haskell:

```
pred  0     =  0
pred (n+1)  =  n
```

primitiv-rekursiv:

$$\begin{aligned} pred(0) &= C_0^0 \\ pred(S(n)) &= \pi_2^2(pred(n), n) \end{aligned}$$

✓

Primitiv-rekursive Funktionen

Die Multiplikation kann rekursiv über die Addition definiert werden.

haskell:
```
mult  0   m          =  0
mult  (n+1)  m     =  (mult  n  m ) + m
```

$$\text{mult}(0, m) = g(m)$$
$$\text{mult}(S(n), m) = h(\text{mult}(n, m), n, m)$$

$$\text{mult}(0, m) = C_0^1$$
$$\text{mult}(S(n), m) = \text{add}(\pi_1^3(\text{mult}(n, m), n, m), \pi_3^3(\text{mult}(n, m), n, m))$$

$g = Z$ $\quad h = \underbrace{add}_{f} \circ [\underbrace{\pi_1^3}_{g_1}, \underbrace{\pi_3^3}_{g_2}]$ $\quad R = \text{mult}$

I Grundfunktionen

II Kompositionsschema

III Rekursionsschema

$$\text{mult}(0, m) = Z(m)$$
$$\text{mult}(S(n), m) = \left(add \circ [\pi_1^3, \pi_3^3]\right)(\text{mult}(n, m), n, m)$$

✓

Primitiv-rekursive Funktionen

Eine primitiv-rekursive Definition der Subtraktion sieht wie folgt aus:

haskell:
```haskell
sub m 0       = m
sub m (n+1) =  pred (sub m (pred (n+1)))
```

```
sub 2 1   ⇒ pred (sub 2 (pred 1))
          ⇒ pred (sub 2 0)
          ⇒ pred 2
          ⇒ 1
```

```
sub 1 3   ⇒ pred (sub 1 (pred 3))
          ⇒ pred (sub 1 2)
          ⇒ pred (pred (sub 1  (pred 2)))
          ⇒ pred (pred (sub 1  1))
          ⇒ pred (pred (pred (sub 1 (pred 1)))
          ⇒ pred (pred (pred (sub 1 0)))
          ⇒ pred (pred (pred 1))
          ⇒ pred (pred 0)
          ⇒ pred 0
          ⇒ 0
```

Primitiv-rekursive Funktionen

haskell:
```haskell
sub m 0       = m
sub m (n+1) =  pred (sub m (pred (n+1)))
```

Eine primitiv-rekursive Definition der Subtraktion sieht wie folgt aus:

$$sub\ (m,n) = sub'\ (\pi_2^2(m,n),\ \pi_1^2(m,n))$$

$$\begin{aligned} sub'\ (0,m) &= \pi_1^1(m) \\ sub'\ (S(n),m) &= pred\ (\pi_1^3(sub'(n,m),n,m)) \end{aligned}$$

✓

Primitiv-rekursive Funktionen

… weitere Beispiele an der Tafel . . .