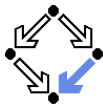


# Berechenbarkeit und Komplexität

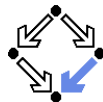
## Rekursive Funktionen

Wolfgang Schreiner  
Wolfgang.Schreiner@risc.jku.at

Research Institute for Symbolic Computation (RISC)  
Johannes Kepler University, Linz, Austria  
<http://www.risc.jku.at>



# Rekursive Funktionen



Rein mathematischer Zugang zum Begriff der berechenbaren Funktionen.

- Die **Grundfunktionen**:

- Die **Nullfunktion**  $\text{null}() := 0$ .

- Die **Nachfolgerfunktion**  $\text{succ}(x) := x + 1$ .

- Die **Projektionsfunktionen**  $\text{proj}_i^n(x_1, \dots, x_n) = x_i, 1 \leq i \leq n$ .

- Die aus der **Komposition** von  $f$  und  $g_1, \dots, g_k$  entstandene Funktion:

$$h(x_1, \dots, x_n) := f(g_1(x_1, \dots, x_n), \dots, g_k(x_1, \dots, x_n))$$

- $f$  ... eine Funktion in  $k$  Variablen.

- $(g_1, \dots, g_k)$  ...  $k$  Funktionen in  $n$  Variablen.

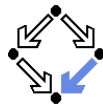
- Die durch **Rekursion** aus  $f$  und  $g$  entstandene Funktion:

$$h(m, x_1, \dots, x_n) := \begin{cases} f(x_1, \dots, x_n), & \text{wenn } m = 0 \\ g(m-1, h(m-1, x_1, \dots, x_n), x_1, \dots, x_n), & \text{wenn } m > 0 \end{cases}$$

- $f$  ... eine Funktion in  $n$  Variablen.

- $g$  ... eine Funktionen in  $n + 2$  Variablen.

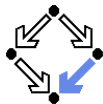
# Primitiv Rekursive Funktionen



- Die Klasse  $\mathcal{PR}$  der **primitiv rekursiven Funktionen** ist wie folgt induktiv definiert:
  - Die Grundfunktionen gehören zu  $\mathcal{PR}$ .
  - Alle Funktionen, welche durch Komposition und Rekursion aus Funktionen in  $\mathcal{PR}$  entstehen, gehören zu  $\mathcal{PR}$ .(Keine andere Funktion gehört zu  $\mathcal{PR}$ .)
- **Satz:** jede primitiv rekursive Funktion ist Turing-berechenbar.
  - Das Konzept der "Turing-Berechenbarkeit" ist mindestens so mächtig wie das der primitiv rekursiven Funktionen.(Das umgekehrte gilt nicht, wie wir später sehen werden.)

**Eine Klasse von Funktionen, deren Konstruktion ihre Berechenbarkeit sicher stellt.**

# Beispiel



Viele bekannte zahlentheoretischen Funktionen sind primitiv rekursiv.

- Die Additionsfunktion  $f(x, y) := x + y$  ist primitiv rekursiv:

$$f(0, y) := y$$

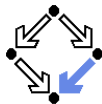
$$f(x + 1, y) := f(x, y) + 1$$

$$f(x, y) := \begin{cases} \text{proj}_1^1(y), & \text{wenn } x = 0 \\ s(x - 1, f(x - 1, y), y), & \text{wenn } x > 0 \end{cases}$$

$$s(u, v, w) = \text{succ}(\text{proj}_2^3(u, v, w))$$

Konstruktion aus Grundfunktionen durch Rekursion und Komposition.

# Beispiel



- Die Multiplikationsfunktion  $g(x, y) := x \cdot y$  ist primitiv rekursiv:

$$g(0, y) := 0$$

$$g(x + 1, y) := g(x, y) + y$$

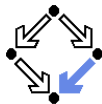
$$g(x, y) := \begin{cases} n(y), & \text{wenn } x = 0 \\ r(x - 1, g(x - 1, y), y), & \text{wenn } x > 0 \end{cases}$$

$$n(x) = \begin{cases} \text{null}(), & \text{wenn } x = 0 \\ \text{proj}_2^2(x - 1, n(x - 1)), & \text{wenn } x > 0 \end{cases}$$

$$r(u, v, w) = f(\text{proj}_2^3(u, v, w), \text{proj}_3^3(u, v, w))$$

Konstruktion aus Grundfunktionen und bereits konstruierten primitiv rekursiven Funktionen durch Rekursion und Komposition.

# Beispiel



- Die Faktoriellenfunktion  $h(x) := x!$  ist primitiv rekursiv:

$$h(0) := 1$$

$$h(x + 1) := h(x) \cdot (x + 1)$$

$$h(x) := \begin{cases} o(), & \text{wenn } x = 0 \\ s(x - 1, h(x - 1)), & \text{wenn } x > 0 \end{cases}$$

$$o() := \text{succ}(\text{null}())$$

$$s(x_1, x_2) := g(\text{succ}(\text{proj}_1^2(x_1, x_2)), \text{proj}_2^2(x_1, x_2))$$

Konstruktion aus Grundfunktionen und einer bereits konstruierten primitiv rekursiven Funktion durch Rekursion und Komposition.



# Beispiel

- Die Exponentialfunktion  $x^y$ :

$$x^0 := 1$$

$$x^{y+1} := x^y \cdot x$$

- Die Vorgängerfunktion  $\text{pred}(x) := x - 1$ :

$$\text{pred}(0) := 0$$

$$\text{pred}(x + 1) := x$$

- Die Differenz  $x \dot{-} y$  auf  $\mathbb{N}$ :

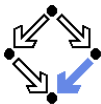
$$x \dot{-} 0 := x$$

$$x \dot{-} (y + 1) := \text{pred}(x \dot{-} y)$$

- Die Abstandsfunktion  $|x - y|$ :

$$|x - y| := (x \dot{-} y) + (y \dot{-} x)$$

Die formal genaue Definition als primitiv rekursive Funktion ist nicht weiters schwierig (Übung!).



# Primitiv Rekursive Prädikate

Primitiv rekursive Funktionen, die nur die Werte 0 (“falsch”) oder 1 (“wahr”) annehmen.

- Das Prädikat  $x = 0$

$$\text{istnull}(x) := \begin{cases} 1, & \text{wenn } x = 0 \\ 0, & \text{sonst} \end{cases} = 1 - x$$

- Das Prädikat  $x = y$

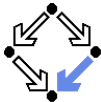
$$d(x, y) := \begin{cases} 1, & \text{wenn } x = y \\ 0, & \text{sonst} \end{cases} = \text{istnull}(|x - y|)$$

- Das Prädikat  $x \leq y$

$$l(x, y) := \begin{cases} 1, & \text{wenn } x \leq y \\ 0, & \text{sonst} \end{cases} = \text{istnull}(x - y)$$

Komposition von bereits definierten primitiv rekursiven Funktionen.





# Weitere Konstruktionen

- **Satz:** sind  $P$  und  $Q$  primitiv rekursive Prädikate, dann sind auch die folgenden Prädikate primitiv rekursiv:

$$\neg P, P \vee Q, P \wedge Q$$

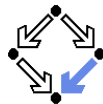
- Beweis:

- $(\neg P)(x_1, \dots, x_n) = \text{istnull}(P(x_1, \dots, x_n))$
- $(P \wedge Q)(x_1, \dots, x_n) = P(x_1, \dots, x_n) \cdot Q(x_1, \dots, x_n)$
- $(P \vee Q)(x_1, \dots, x_n) = \neg(\neg P(x_1, \dots, x_n) \wedge \neg Q(x_1, \dots, x_n))$

- **Satz:** sind  $g$  und  $h$  primitiv rekursive Funktionen und  $P$  ein primitiv rekursives Prädikat, dann ist auch die folgendermaßen definierte Funktion  $f$  primitiv rekursiv:

$$f(x_1, \dots, x_n) := \begin{cases} g(x_1, \dots, x_n), & \text{wenn } P(x_1, \dots, x_n) \\ h(x_1, \dots, x_n), & \text{sonst} \end{cases}$$

- Beweis:  $f(x_1, \dots, x_n) = g(x_1, \dots, x_n) \cdot P(x_1, \dots, x_n) + h(x_1, \dots, x_n) \cdot (\neg P(x_1, \dots, x_n))$



- **Satz:** Sei  $f(t, x_1, \dots, x_n)$  eine primitiv rekursive Funktion. Dann sind auch die folgenden Funktionen  $g$  und  $h$  primitiv rekursiv:

$$g(y, x_1, \dots, x_n) := \sum_{t=0}^y f(t, x_1, \dots, x_n)$$

$$h(y, x_1, \dots, x_n) := \prod_{t=0}^y f(t, x_1, \dots, x_n)$$

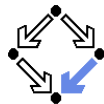
- **Beweis:**

$$g(0, x_1, \dots, x_n) := f(0, x_1, \dots, x_n)$$

$$g(t+1, x_1, \dots, x_n) := g(t, x_1, \dots, x_n) + f(t+1, x_1, \dots, x_n)$$

$$h(0, x_1, \dots, x_n) := f(0, x_1, \dots, x_n)$$

$$h(t+1, x_1, \dots, x_n) := h(t, x_1, \dots, x_n) \cdot f(t+1, x_1, \dots, x_n)$$



- **Satz:** Sei  $P(t, x_1, \dots, x_n)$  ein primitiv rekursives Prädikat. Dann sind auch die folgenden Prädikate primitiv rekursiv:

$$\forall t : t \leq y \Rightarrow P(t, x_1, \dots, x_n)$$

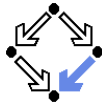
$$\exists t : t \leq y \wedge P(t, x_1, \dots, x_n)$$

- **Beweis:**

$$(\forall t : t \leq y \Rightarrow P(t, x_1, \dots, x_n)) \Leftrightarrow \left( \prod_{t=0}^y P(t, x_1, \dots, x_n) \right) = 1$$

$$(\exists t : t \leq y \wedge P(t, x_1, \dots, x_n)) \Leftrightarrow \left( \sum_{t=0}^y P(t, x_1, \dots, x_n) \right) \neq 0$$

# Beispiel



- Das Prädikat  $y|x$  (“ $y$  teilt  $x$ ”) ist primitiv rekursiv:

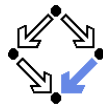
$$y|x :\Leftrightarrow \exists t : t \leq x \wedge y \cdot t = x$$

- Das Prädikat  $\text{Prime}(x)$  (“ $x$  ist eine Primzahl”) ist primitiv rekursiv.

$$\text{Prime}(x) :\Leftrightarrow (x \geq 2) \wedge \forall t : t \leq x \Rightarrow (t = 1 \vee t = x \vee \neg(t|x))$$

In der Praxis sind fast alle interessierenden Funktionen und Prädikate primitiv rekursiv.

# Primitiv Rekursive Funktionen



Aber nicht alle Turing-berechenbaren Funktionen sind primitiv rekursiv.

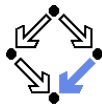
- Die *Ackermann-Funktion*:

$$\text{ack}(0, y) := y + 1$$

$$\text{ack}(x, 0) := \text{ack}(x - 1, 1)$$

$$\text{ack}(x, y) := \text{ack}(x - 1, \text{ack}(x, y - 1)), x > 0 \wedge y > 0.$$

Für die Behandlung derartiger Funktionen ist ein zusätzliches Konstruktionsprinzip notwendig.



# $\mu$ -rekursive Funktionen

## ■ Minimalisierung eines Prädikats:

- Sei  $P(y, x_1, \dots, x_n)$  ein Prädikat. Dann ist

$$\min_y P(y, x_1, \dots, x_n)$$

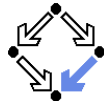
diejenige (partielle) Funktion, die  $x_1, \dots, x_n$  auf den kleinsten Wert  $y$  abbildet, für den  $P(y, x_1, \dots, x_n)$  wahr ist. Falls kein solches  $y$  existiert, ist das Ergebnis der Funktion undefiniert.

- Die Klasse  $\mathcal{R}$  der  $\mu$ -rekursiven Funktionen ist wie folgt definiert:

- Die Grundfunktionen gehören zu  $\mathcal{R}$ .
- Alle Funktionen, welche durch Komposition, Rekursion, und Minimalisierung aus Funktionen (und Prädikaten) in  $\mathcal{R}$  entstehen, gehören zu  $\mathcal{R}$ .

(Keine andere Funktion gehört zu  $\mathcal{R}$ .)

Die Klasse der  $\mu$ -rekursiven Funktionen ist genau die Klasse der Turing-berechenbaren Funktionen.



Wie kann man sich den Unterschied zwischen  $\mathcal{PR}$  und  $\mathcal{R}$  vorstellen?

## ■ **Primitiv rekursive Funktionen:**

- Ein Argument, das in jedem “rekursiven Aufruf” kleiner wird.
  - “Termination” der Rekursion ist garantiert.
- Entspricht Programmen, die mit Zählschleifen (aber nicht mit allgemeinen while-Schleifen) operieren.

Jede primitiv rekursive Funktion kann durch ein Programm realisiert werden, das nur mit (Zuweisungen, Verzweigungen und) Zählschleifen arbeitet.

## ■ **$\mu$ -rekursive Funktionen:**

- Die “Minimalisierung” entspricht einer unbegrenzten “Suche”.
  - “Termination” der Suche ist nicht garantiert.
- Entspricht Programmen, die mit allgemeinen while-Schleifen operieren (deren Termination nicht von vornherein garantiert ist).

Jede  $\mu$ -rekursive (aber nicht primitiv rekursive) Funktion kann nur durch ein Programm realisiert werden, das mit allgemeinen Schleifen (oder Rekursion) arbeitet.