

# Der erweiterte Satz von Rice

Raphael Reitzig

Sebastian Wild

Markus E. Nebel

13. März 2012

## 1 Notation:

- Für (partielle) Funktionen  $f$  und  $g$  definieren wir  $f \subseteq g$ , » $f$  ist eine Teilfunktion von  $g$ «, wenn

(1)  $D_f \subseteq D_g$  und

(2)  $\forall x \in D_f : f(x) = g(x)$

Wir schreiben  $f \subsetneq g$  für  $f \subseteq g \wedge D_f \neq D_g$ .

( Das entspricht genau der Mengeninklusion auf den Graphen der Funktionen und wurde in ähnlicher Weise im Semantik-Kapitel des Buchs für Variablenbelegungen verwendet. )

- Definiere  $f \cong g$ , „ $f$  und  $g$  sind äquivalent/gleich“, wenn  $f \subseteq g$  und  $g \subseteq f$ .
- Für eine beliebige  $n + 1$ -stellige Funktion  $g$  schreiben wir  $F_i^g$  für die Funktion definiert durch  $F_i^g(\vec{x}) := g(i, \vec{x})$ .<sup>1</sup>
- Schreibe  $\mathcal{I}P := \{i \in \mathbb{N}_0 : F_i^g \in P\}$  für die Indexmenge einer Funktionenklasse  $P$ .

## 2 Der erweiterte Satz von Rice

Der Beweis zum erweiterten Satz von Rice verwendet ein recht nützliches Lemma, das wir hier zuerst betrachten:

---

<sup>1</sup>Damit ist  $g$  eine universelle Funktion für die Klasse  $K_n := \{F_i^g : i \in \mathbb{N}_0\}$  und induziert damit eine (nicht injektive) Gödelisierung  $G$  für  $K_n$ .

**Lemma Compiler:** Sei  $\mathcal{G}$  eine Standard-Gödelisierung für  $K$  und  $g \in K_{n+1}$  eine Funktion. Dann gibt es eine totale Funktion  $c \in K_1$  mit

$$\forall i \in \mathbb{N}_0 \quad F_{c(i)}^{\mathcal{G}} \cong F_i^g.$$

**Beweis:** Wegen  $g \in K$  und  $\mathcal{G}$  Standard-Gödelisierung existiert  $j \in \mathbb{N}_0$  mit  $F_j^{\mathcal{G}} = g$ . Damit gibt es nach smn-Eigenschaft für  $\mathcal{G}$  eine *totale* Funktion  $s_{1,n} \in K_2$  mit

$$\forall i \in \mathbb{N}_0 \quad \forall x \in \mathbb{N}_0, \vec{y} \in \mathbb{N}^n \quad F_{s_{1,n}(i,x)}^{\mathcal{G}}(\vec{y}) = F_i^{\mathcal{G}}(x, \vec{y}).$$

Das gilt natürlich insbesondere für  $i = j$ , d. h.  $F_{s_{1,n}(j,x)}^{\mathcal{G}}(\vec{y}) \cong g(x, \vec{y})$ . Setze jetzt  $c(i) := s_{1,n}(j, i)$ .<sup>2</sup> Dann gilt

$$\forall x \in \mathbb{N}_0, \vec{y} \in \mathbb{N}^n \quad F_{c(x)}^{\mathcal{G}}(\vec{y}) = F_{s_{1,n}(j,x)}^{\mathcal{G}}(\vec{y}) = g(x, \vec{y}) = F_x^g(\vec{y}).$$

□

Satz (Erweiterter Satz von Rice):

Sei  $\mathcal{G}$  eine Standard-Gödelisierung für  $\mathcal{R}$  und  $A \subseteq \mathbb{N}_0$  eine Menge. Gibt es  $f, g \in \mathcal{R}_n$  mit

- (1)  $f \subsetneq g$ ,
- (2)  $\mathcal{I}\{f\} \subseteq A$ , d. h. alle Indizes von  $f$  sind in  $A$ , und
- (3)  $\mathcal{I}\{g\} \cap A = \emptyset$ , d. h. kein Index von  $g$  ist in  $A$ ,

so ist  $A$  nicht rekursiv aufzählbar.

**Beweis:** Reduktion auf  $\bar{D} = \mathbb{N}_0 \setminus D$  nicht rekursiv aufzählbar, mit  $D = \{i : F_i^{\mathcal{G}}(i) \text{ def.}\}$ .

**Annahme:**  $A$  sei rekursiv aufzählbar.

Wir definieren die Funktion  $h$  durch

$$h(x, \vec{y}) := \begin{cases} g(\vec{y}) & \text{für } x \in D \\ f(\vec{y}) & \text{sonst} \end{cases}.$$

**Behauptung 1:**  $h \in \mathcal{R}_{n+1}$

<sup>2</sup>Wir behaupten hier (stillschweigend), dass  $c \in K_1$ . Das gilt aber nicht für beliebige  $K$ , sondern nur solche, die alle konstanten Funktionen enthalten und gegen Einsetzen abgeschlossen sind und gehört selbstredend zu den Voraussetzungen des Lemmas; der Übersichtlichkeit halber dort weggelassen. Anwenden werden wir das Lemma nur für  $K = \mathcal{R}$ , was diese Anforderung natürlich erfüllt.

Beweis via Timesharing: Wir berechnen abwechselnd  $F_x^{\mathcal{G}}(x)$  und  $f(\vec{y})$ . Terminiert ersteres, berechnen wir anschließend  $g(\vec{y})$ , ist  $f(\vec{y})$  als Erstes berechnet, so geben wir  $f(\vec{y})$  aus. Das beschriebene Vorgehen lässt sich beispielsweise auf Turing Maschinen realisieren, ist also berechenbar.

Für den Nachweis, dass wir auch tatsächlich  $h(x, \vec{y})$  berechnen unterscheiden wir:

- $x \in D$  und  $f(\vec{y})$  undefiniert  
Die Berechnung von  $f(\vec{y})$  terminiert nicht,  $F_x^{\mathcal{G}}(x)$  schon. Also tun wir das, was  $g(\vec{y})$  getan hätte. Egal ob das terminiert oder nicht, es ist das richtige.
- $x \notin D$  und  $f(\vec{y}) = z$   
Hier wird sicher  $f(\vec{y})$  als einziges fertig, die Ausgabe  $z$  ist genau das Gewünschte.
- $x \notin D$  und  $f(\vec{y})$  undefiniert  
Hier ist  $h(x, \vec{y})$  nicht definiert und unsere Berechnung simuliert abwechselnd zwei nicht terminierende Auswertungen, passt.
- $x \in D$  und  $f(\vec{y}) = z$   
Das ist der spannendste Fall, denn wir haben zwei terminierende Berechnungen, von denen wir nicht wissen, welcher zuerst fertig wird. Ist  $F_x^{\mathcal{G}}(x)$  zuerst durch, machen wir mit der weiteren Auswertung von  $g(\vec{y})$  das Richtige. Ist aber  $f(\vec{y})$  zuerst fertig, so ist unsere Ausgabe  $z = f(\vec{y})$ , aber  $h(x, \vec{y}) = g(\vec{y})$ . Problem? Nein, denn mit  $f \subsetneq g$  und  $z \in D_f$  folgt  $z \in D_g$  und  $z = f(\vec{y}) = g(\vec{y})$ .

Die (angedeutete) Turing Maschine berechnet also tatsächlich  $h$ .

□ [Behauptung 1]

Mit  $h \in \mathcal{R}$  und  $\mathcal{G}$  Standard-Gödelisierung existiert nun nach Lemma Compiler  $c \in \mathcal{T}_1$  mit  $\forall i \in \mathbb{N}_0$   $F_{c(i)}^{\mathcal{G}} \cong F_i^h$ .

*Beachte:* Die Klasse von Funktionen, für die  $h$  eine universelle Funktion ist, ist ziemlich langweilig:

$$\{F_i^h : i \in \mathbb{N}_0\} = \{f, g\}. \quad (*)$$

Damit haben wir alles zusammen: Sei  $x \in \mathbb{N}_0$  beliebig so gilt

$$x \in \bar{D} \quad \xLeftrightarrow[\text{Def } h] \quad f \cong F_x^h \quad \xrightarrow[\text{Compiler}]{\cong} \quad F_{c(x)}^{\mathcal{G}} \quad \xLeftrightarrow[\text{(*)}] \quad c(x) \in A.$$

Nach Annahme ist  $A$  rekursiv aufzählbar, d. h. es gibt nach Satz 4.23 eine Funktion  $\pi_A \in \mathcal{R}_1$  mit  $D_{\pi_A} = A$ . Folglich ist auch die Funktion  $\pi_{\bar{D}} := \pi_A \circ c$  berechenbar und es gilt  $D_{\pi_{\bar{D}}} = \bar{D}$ , also wäre – wieder nach 4.23 –  $\bar{D}$  rekursiv aufzählbar.

⚡ da wir wissen, dass  $\bar{D}$  nicht rekursiv aufzählbar ist (Satz 4.24 (b)).

□

### 3 Anwendungsbeispiel

Behauptung: Die Menge  $A := \{x \in \mathbb{N}_0 : \exists y \in \mathbb{N}_0 F_x^{\mathcal{G}_1}(y) \text{ undef.}\}$ , d. h. die Menge aller Indizes von berechenbaren, nicht-totalen Funktionen, ist nicht rekursiv aufzählbar.

**Beweis:** via erweitertem Satz von Rice. Prüfen wir also die Voraussetzungen:

Dazu müssen wir ›passende‹ Funktionen  $f$  und  $g$  finden. Wir wählen  $f = \Phi_1$ , die einstellige nirgends definierte Funktion, und  $g = C_1^0$ , die einstellige konstante 0-Funktion. Wir prüfen:

- (1)  $f \subsetneq g$   
 $\emptyset = D_{\Phi_1} = D_f \subsetneq D_g = D_{C_1^0} = \mathbb{N}_0$  und trivialerweise  $\forall x \in \emptyset : f(x) = g(x)$ .
- (2)  $\mathcal{I}\{f\} \subseteq A$   
 Da  $f = \Phi$  nicht total, gilt für jeden Index  $x \in \mathcal{I}\{f\}$ :  $\exists y : F_x^{\mathcal{G}_1}(y) \text{ undef.}$  Also ist  $x \in A$ .
- (3)  $\mathcal{I}\{g\} \cap A = \emptyset$   
 $g$  total, also gilt für beliebiges  $x \in \mathcal{I}\{g\}$ :  $\nexists y : F_x^{\mathcal{G}_1}(y) \text{ undef.}$  und damit  $x \notin A$ .

Damit ist der erweiterte Satz von Rice anwendbar und es folgt, dass  $A$  nicht rekursiv aufzählbar ist. □



<http://www.springer.com/978-3-8348-1889-8>

Formale Grundlagen der Programmierung

Nebel, M.

2012, VIII, 194 S., Softcover

ISBN: 978-3-8348-1889-8