

Übungen zur Vorlesung
„Grundlagen der Programm- und Systementwicklung“

Aufgabe 1 (H) Noethersche Induktion

Die Funktion sum auf den natürlichen Zahlen ist algebraisch wie folgt spezifiziert:

$$\text{sum}(x, 0) = x$$

$$\text{sum}(x, s(y)) = \text{sum}(s(x), y)$$

Zeigen Sie mittels Noetherscher Induktion, daß sum kommutativ ist!

(**Hinweis:** Kombinieren sie die Noethersche mit einer strukturellen Induktion! Zeigen Sie zunächst $\text{sum}(0, a) = \text{sum}(a, 0)$ und dann $\text{sum}(x, y) = \text{sum}(y, x)$. Die zu konstruierende Noethersche Ordnung ergibt sich schnell, wenn Sie den Beweis aufschreiben.)

Aufgabe 2 vollständige Induktion

(für die Zentralübung)

Ein abstraktes Reduktionssystem (A, \rightarrow) besteht aus einer Menge A und einer binären Relation $\rightarrow \subseteq A \times A$, die meistens infix notiert wird. Beispiele für Reduktionssysteme sind Wort- und Termersetzungssysteme, wobei letzere eine zentrale Rolle in der Theorie der funktionalen Programmierung spielen: Berechnungen/Auswertungen in der funktionalen Programmierungen entsprechen Reduktionsketten $a_0 \rightarrow a_1 \rightarrow \dots \rightarrow a_n$. Wenn es kein a_{n+1} mit $a_n \rightarrow a_{n+1}$ gibt, dann heißt a_n *irreduzibel* oder *in Normalform*. Eine Definition der *Gleichheit* besagt nun, daß zwei Terme (oder Wörter; Elemente aus A) genau dann gleich sind, wenn sie dieselbe Normalform besitzen. Eine wünschenswerte Eigenschaft des Reduktionssystems ist dann natürlich die, daß Normalformen eindeutig sind. Dafür hinreichende Kriterien sind *Konfluenz* bzw. *Church-Rosser* und *Terminierung*: (A, \rightarrow) (oder auch nur \rightarrow) heißt *konfluent*, falls $x \rightarrow^* y_1$ und $x \rightarrow^* y_2$ die Existenz eines z mit $y_1 \rightarrow^* z$ und $y_2 \rightarrow^* z$ gewährleistet. (A, \rightarrow) heißt *Church-Rosser*, falls $x \leftrightarrow^* y$ die Existenz eines z mit $x \rightarrow^* z$ und $y \rightarrow^* z$ impliziert. (Zur Erinnerung: \rightarrow^* ist der reflexive und transitive Abschluß von \rightarrow , und \leftrightarrow ist der symmetrische Abschluß von \rightarrow .) (A, \rightarrow) heißt *terminierend*, falls es keine unendlichen Ketten $a_0 \rightarrow a_1 \rightarrow \dots$ gibt.

Zeigen Sie:

1. Konfluenz und Church-Rosser sind äquivalente Eigenschaften. (**Hinweis:** Verdeutlichen Sie sich diese Konzepte graphisch, und führen Sie den Beweis der Richtung „Konfluenz \Rightarrow Church-Rosser“ durch vollständige Induktion *graphisch*.)
2. Wenn ein Reduktionssystem konfluent und terminierend ist, dann besitzt jedes Element genau eine Normalform.

Aufgabe 3 (H) Spezifikation eines Rings

Ein Ring ist eine Datenstruktur, in der man sich relativ zu einem ausgezeichneten „aktuellen“ Element rechts- und linksherum bewegen kann (wobei sich das aktuelle Element ggf. ändert). Enthält der Ring n Elemente, so verändert ihn n -faches Bewegen nach links oder n -faches Bewegen nach rechts nicht (aber für $n > 1$ ändert er sich bei $n-1$ -facher Bewegung; daher der Name „Ring“). Das aktuelle Element kann gelöscht werden, wobei das Element rechts des aktuellen Elements die Position des alten aktuellen Elements einnimmt. Außerdem kann ein Element eingefügt werden, das dann das neue aktuelle Element wird; dabei rutscht das alte aktuelle Element um eine Position nach rechts. Schließlich besteht die Möglichkeit, sich das aktuelle Element anzusehen.

1. Können Sie sich Anwendungen von Ringen vorstellen?
2. Gibt es Zweideutigkeiten in der umgangssprachlichen Beschreibung?
3. Erstellen Sie anhand dieser umgangssprachlichen Beschreibung ein Modell und eine algebraische Spezifikation! In letzterer dürfen Sie die n -fache Funktionsiteration f^n verwenden ($f^0 = \text{id}$; $f^{n+1} = f \circ f^n$ für die Funktionskomposition \circ).
4. Implementieren Sie Ringe in ML, und vergleichen Sie die unter (3.) erstellten Spezifikationen mit Ihrem Programm!