Theorie der Programmierung SoSe 2021

Klausur [Probe]

Bitte vermerken Sie auf Ihrer Abgabe Ihren Namen und Ihre Matrikelnummer.

Hinweis: Die Klausur ist ab 15 Punkten bestanden. Die Verteilung der Punkte auf die Aufgaben kann variieren.

Aufgabe 1 Konfluenz und Terminierung (9 Punkte)

Wir definieren ein Termersetzungssystem über der aus zwei binären Funktionssymbolen $\Uparrow$ und $\Downarrow$ (in Infixnotation geschrieben) bestehenden Signatur Σ durch

$$x \Uparrow (y \Uparrow z) \to_0 x \Uparrow (y \Downarrow y)$$
$$x \Downarrow (x \Downarrow y) \to_0 x \Downarrow y$$

1. Ist dieses System terminierend? Geben Sie einen Beweis mittels Polynomordnungen oder ein Gegenbeispiel an.

2. Ist das System konfluent? Geben Sie einen Beweis bzw. ein Gegenbeispiel an.

Aufgabe 2 System F (9 Punkte)

1. Man erinnere sich, dass die Church-Numerale in System F den Typ $\mathbb{N} := \forall a.(a \to a) \to a \to a$ haben. Zeigen Sie unter der Annahme, dass *one* und *two* beide den Typ $\mathbb{N}$ haben und dass *mult* den Typ $\mathbb{N} \to \mathbb{N} \to \mathbb{N}$ hat, dass der Term $\lambda n.\ n\ (mult\ two)\ one$ Typ $\mathbb{N} \to \mathbb{N}$ hat; d.h. geben sie eine Typherleitung in System F für

   $$\{mult : \mathbb{N} \to \mathbb{N} \to \mathbb{N}, one : \mathbb{N}, two : \mathbb{N}\} \vdash \lambda\ n.\ n\ (mult\ two)\ one : \mathbb{N} \to \mathbb{N}$$

   an.

2. Bestimmen Sie die Menge $PT(\Gamma; s; \alpha)$ aus dem Algorithmus von Hindley und Milner für den Term

   $$s = \text{let } (f = \lambda x\ y.\ y\ x) \text{ in } \lambda g\ z.\ f\ z\ (f\ g)$$

   im leeren Kontext. *Achtung:* Sie müssen **nicht** das Unifikationsproblem für $PT(\Gamma; s; \alpha)$ lösen. Zur Bestimmung des polymorphen Typs von f dürfen Sie zudem den Unifikator angeben, ohne die einzelnen Schritte des Unifikationsalgorithmus durchzuführen.

Aufgabe 3 Strukturelle Induktion und Folds (9 Punkte)

1. Wir erinnern an den Datentyp der Listen und einige hierauf rekursiv definierte Standardfunktionen:

data List $a = Nil \mid Cons\ a\ (\textbf{List}\ a)$

$$\begin{aligned} snoc\ Nil\ y &= Cons\ y\ Nil \\ snoc\ (Cons\ x\ xs)\ y &= Cons\ x\ (snoc\ xs\ y) \end{aligned}$$

$$\begin{aligned} Nil \oplus ys &= ys \\ (Cons\ x\ xs) \oplus ys &= Cons\ x\ (xs \oplus ys) \end{aligned}$$

Beweisen Sie mittels struktureller Induktion, dass

$$\forall e,\ xs,\ ys.\ xs \oplus (Cons\ e\ ys) = (snoc\ xs\ e) \oplus ys.$$

Geben Sie im Induktionsschritt die Induktionsannahme explizit an und erläutern Sie alle Schritte des Beweises.

2. Gegeben sei der folgende Datentyp, den man sich als nichtleere Liste von Paaren vorstellen kann:

   data Twins $a = End\ a\ a \mid More\ a\ (\textbf{Twins}\ a)\ a$

   Geben Sie die *fold*-Funktion für **Twins**, *foldtw*, inklusive ihres Typs an. Nutzen Sie diese, um eine Funktion

   $$identicalTwins : \textbf{Twins}\ a \to \textbf{List}\ a$$

   zu definieren, die Elemente nur dann in die Zielliste übernimmt, wenn die beiden Werte übereinstimmen. Beispielsweise soll gelten:

   $identicalTwins\ (More\ 1\ (More\ 2\ (More\ 1\ (End\ 2\ 0)\ 1)\ 2)\ 3) = [2,1]$

   Hinweis: Sie dürfen die üblichen Konstrukte wie λ-Abstraktion, $if \ldots then \ldots else$ verwenden und annehmen, dass Elemente mittels == auf Gleichheit überprüft werden können.

Aufgabe 4 Korekursion und Koinduktion (9 Punkte)

Ein (digitales) *Signal* ist ein zeitlich veränderlicher Wert zwischen $-\infty$ und $+\infty$, wobei wir die Zeit als diskret behandeln, entsprechend etwa fortgesetztem Sampling. Solche Signale lassen sich in natürlicher Weise durch einen Kodatentyp repräsentieren:

codata Signal where
  $currentSample : \textbf{Signal} \to \textbf{Int}$
  $discardSample : \textbf{Signal} \to \textbf{Signal}$

Z.B. repräsentiert gemäß den folgenden korekursiven Definitionen $flat\ x$ ein konstantes Signal mit Wert x und $square\ x\ y$ ein zwischen x und y alternierendes Signal:

$$\begin{aligned} currentSample\ (flat\ x) &= x \\ discardSample\ (flat\ x) &= flat\ x \end{aligned}$$

$$\begin{aligned} currentSample\ (square\ x\ y) &= x \\ discardSample\ (square\ x\ y) &= square\ y\ x \end{aligned}$$

1. Definieren Sie korekursiv eine Funktion $sampler : \textbf{Signal} \to \textbf{Signal} \to \textbf{Signal}$, so dass der Wert des Signals $sampler\ t\ s$ der Wert von s ist, wenn der Wert von t größer als 0 ist, und 0 sonst. Insbesondere sollten z.B. die Gleichungen

$$sampler\ (square\ 0\ 1)\ (square\ x\ 0) = flat\ 0 \tag{1}$$

und (für alle x)

$$sampler\ (square\ 1\ 0)\ (\ flat\ \ x) = square\ x\ 0 \tag{2}$$

gelten. *Hinweis:* Sie dürfen bei der Definition die üblichen Operationen auf Zahlen (Addition, Vergleich etc.) und auf Booleans (and, or, if-then-else etc.) als gegeben annehmen.

2. Beweisen Sie dann per Koinduktion, dass für Ihre Definition von *sampler* tatsächlich die beiden Gleichungen (1) und (2) gelten. Erläutern Sie alle Beweisschritte.

Aufgabe 5 Reguläre Sprachen (4 Punkte)

Minimieren Sie den abgebildeten deterministischen endlichen Automaten über dem Alphabet $\Sigma = \{0, 1\}$ mittels des tabellarischen Algorithmus aus der Vorlesung und zeichnen Sie den Minimalautomaten.