1 Konfluenz und Terminierung

Wir definieren ein TES über dem unären Funktionssymbol $-$, wir schreiben $\bar{x}$ anstelle für $-(x)$ und einem ternären FUnktionsymbol [], wir schreiben $[x, y, z]$ für $[(x, y, z)]$ mit folgender Signatur Σ:

$$[x, \bar{y}, z] \to_0 [x, z, y] \tag{1}$$
$$\overline{[x, y, z]} \to_0 [\bar{x}, \bar{y}, \bar{z}] \tag{2}$$

Achtung: x, y, z sind hier Variablen!

1. Ist das System stark normalisierend? Beweisen Sie dies ggf. mittels Polynomordnungen oder geben Sie ein Gegenbeispiel an.

2. Wie viele kritische Paare gibt es? Enrscheiden Sie für jedes kritisches Paar, ob es zusammenführbar ist oder nicht. Ist das System konfluent?

2 System F

1. Man erinnere sich an die folgende Typkodering in System F:

$$\begin{aligned} 1 &:= \forall r.r \to r \\ (a + b) &:= \forall r.(a \to r) \to (b \to r) \to r \\ \mathbb{L}a &:= \forall r.r \to (a \to r \to r) \to r) \end{aligned}$$

$\mathbb{L}$ steht kier kurz für Listen. Zeigen Sie unter der Annahme:
inl hat den Typ $\forall ab.a \to (a + b)$
inr hat den Typ $\forall ab.b \to (a + b)$,
dass der Term $t = \lambda l.(l(inl(\lambda x.x))(\lambda xy.inr(inrx))$ der Typ: $\forall z.\mathbb{L}z \to (1 + z)$ zugeordnet werden kann. Das heißt: Geben Sie eine Typherleitung in System F an für:
$\{inl : \forall ab.a \to (a + b), inr : \forall ab.b \to (a + b)\} \vdash \forall z.\mathbb{L}z \to (1 + z)$

2. Ist der aus der Vorlesung bekannte Algorithmus W vfür ML-Polymorphie von Hindley und Milner in der Lage, den Typ von t zu identifizieren? Begrüden Sie ihre Antwort.

3 Strukturelle Induktion und Folds

Wir betrachten den DatenTyp Nat der natürluchen Zahlen und die dazugehörige fold-Funktion foldn:

data Nat = Zero|Succ Zero
$foldn :: a \to (a \to a) \to Nat \to a$
$foldn\ x\ f\ Zero = x$
$foldn\ x\ f\ (Succ\ n)\ x = f\ (foldn\ x\ f\ n)$

1. Nehmen Sie für die folgenden Aufgaben an, dass die Multiplikation auf natürlichen Zahlen bereits implementiert ist:

$mult :: Nat \to Nat \to Nat$

a) Definieren Sie einen Datentyp NL a von nichtleeren Listen über einem Typ a

b) Nutzen Sie foldn, um eine Funktion genList:: Nat $\rightarrow NL\ Nat$ zu definieren, die zu einer Zahl n die Liste $[1, 2, 3, \ldots, n + 1]$ berechnet. Sie dürfen zusätzliche Hilfsfunktionen definieren.

c) Geben Sie den Typ und die Definition der zu NL a gehörigen Funktion foldnl an.

d) Vervollständigen Sie unter Verwendung der vorigen Teilaufgaben die folgende Vorlage einer Fakultätsfunktion , d.h. geben Sie Definitionen der Terme f, g und h an.
(Achtung! Natürlich muss zumindest einer dieser Terme von n abhängen), sodass fac n zazsächlich n berechnet. (ohne Beweis)

```
fac :: Nat ⇒ Nat
fac Zero = Succ Zero
fac (Succ Zero) = foldnl f g h
```

2. Die Definitionen der oben angenommen Multiplikation natürlicher Zahlen und der dazu benötigten Additionsfunktion seien wie folgt:

```
plus :: Nat → Nat → Nat
plus Zero n = n
plus (Succ n) m = Succ (plus n m)

mult :: Nat → Nat → Nat
mult Zero n = n
mult (Succ n) m = Succ (plus n m)
```

Zeigen Sie mittels struktureller Induktion, dass folgende Aussage gilt:

$$\forall n\text{: Nat. mult Zero n = mult n Zero}$$

4 Korekursion und Koinduktion

Wir erinnern uns an den Kodatentyp der Streams über einem Datentyp a:

```
codata Stream a where
  hd: Stream a → a
  tl: Stream a → Stream a
```

Desweitern benutzen wir im Folgenden den Datentyp Nat aus Aufgabe 3.

1. Gegeben sei die korekursive definierte Funktion add, die eine konstante natürliche Zahl auf einen Stream natürlicher Zahlen addiert:

```
add :: Stream Nat → Stream Nat → Stream Nat
hd (add x s) = plus( x ( hd s)
tl (add x s) = add x (tl s) )
```

Definieren Sie mithilfe von add korekursiv einen Streams nat, der die natürlichen Zahlen (Zero, Succ Zero, ...) repräsentiert.

Die Folgende Aufgabe verwendet den Paartypen (a,b). Zudem seien folgende Funktionen gegeben:

```
unfold :: (b → a) → (b → b) → b → Stream a
hd (unfold h t x) = hx
tl (unfold h t x) = unfold h t ( t x)

flip :: (a,b) → (b, a)
```

flip (x,y) = (y,x)

fst :: (a,b) → a fst (x,y) = x

alt :: a → a → Stream a
hd (alt x y) = x
tl (alt x y) = alt y x

Beweisen Sie die folgende Eigenschaft durch Koinduktion:

unfold fst flip (0,1) = alt 0 1,

wobei wir 0 für Zero und für 1 Succ Zero schreiben. Rechtfertigen Sie alle Beweisschritte.

5 Reguläre Sprachen

TODO: