8. Aufgabenblatt zu Funktionale Programmierung vom Mi, 05.12.2018.
Fällig: Mi, 12.12.2018 (15:00 Uhr)

Themen: *Rechnen mit Funktionen und Funktionstransformatoren, Modellierung und Simulation von Turingmaschinen*

Zur Frist der Zweitabgabe: Siehe „Hinweise zu Organisation und Ablauf der Übung“ auf der Homepage der LVA.

Aufgabe

Für dieses Aufgabenblatt sollen Sie Haskell-Rechenvorschriften für die Lösung der unten angegebenen Aufgabenstellungen entwickeln und für die Abgabe in einer Datei namens `Aufgabe8.hs` ablegen. Sie sollen für die Lösung dieses Aufgabenblatts also wieder ein ‘gewöhnliches’ Haskell-Skript schreiben.

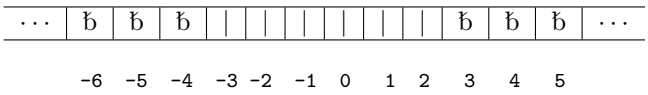
Versehen Sie wieder wie auf den bisherigen Aufgabenblättern alle Funktionen, die Sie zur Lösung benötigen, mit ihren Typdeklarationen und kommentieren Sie Ihre Programme aussagekräftig. Benutzen Sie, wo sinnvoll, Hilfsfunktionen und (Wertvereinbarungen für) Konstanten.

1. In Anhang B.1 der Vorlesungsunterlagen sind Turing-Maschinen mit einem unendlichen Rechenband eingeführt.

   Das unendliche Rechenband einer Turing-Maschine kann durch zwei halbseitig unendliche Rechenbänder modelliert werden. Dabei nimmt das linke Halbband die Felder mit den echt negativen Indizes -1, -2, -3, -4,... des unendlichen Bandes auf, das rechte Halbband die Felder mit den positiven Indizes 0, 1, 2, 3,...

   Diese Modellierung ist nachstehend am Beispiel eines unendlichen Rechenbands illustriert, auf dem auf den Feldern -3, -2, -1, 0, 1, 2 das Zeichen | steht, auf allen anderen Feldern das Leerzeichen, dargestellt durch das Symbol ƀ:

   Unendliches Rechenband:

   Modellierung durch zwei halbseitig unendliche Rechenbänder:

   Halbseitig unendliche Rechenbänder können wir in Haskell auf einfache Weise durch Listen implementieren, da Listen keiner *a priori* vorgegebenen Längenbeschränkung unterliegen. Damit erlaubt uns die Modellierung eines unendlichen Rechenbands durch zwei halbseitig unendliche Rechenbänder, Turing-Maschinen insgesamt in Haskell zu modellieren:

```
type Nat0 = Int

-- Zeichenvorrat und Leerzeichen
data Zeichenvorrat = A | B | C | D | E | F | G | H | I | J | K deriving (Eq,Show)
data Bandalphabet  = BA Zeichenvorrat | Leer deriving (Eq,Show)

-- Rechenband
type Halbband         = [Bandalphabet]
type Linkes_Halbband  = Halbband
type Rechtes_Halbband = Halbband
type Rechenband       = (Linkes_Halbband,Rechtes_Halbband)
```

```
-- Lese- und Schreibkopf
type Index                          = Int
type Lese_Schreibkopf_Position      = Index
type Zeichen_unter_Lese_Schreibkopf = Bandalphabet
data Richtung                       = Links | Rechts deriving (Eq,Show)
data Befehl                         = Drucke Bandalphabet
                                    | Bewege_Lese_Schreibkopf_nach Richtung
                                      deriving (Eq,Show)

-- Abkuerzungen
type LSk_Position = Lese_Schreibkopf_Position
type LSk_Zeichen  = Zeichen_unter_Lese_Schreibkopf

-- Interne Turingmaschinenzustaende
type Zustand               = Nat0
type Interner_Zustand      = Zustand
type Interner_Folgezustand = Zustand

-- Turing-Tafel
type Zeile        = (Interner_Zustand,LSk_Zeichen,Befehl,Interner_Folgezustand)
type Turingtafel  = [Zeile]

-- Globale Turingmaschinenzustaende
data Turingmaschinenzustand = TMZ Turingtafel Interner_Zustand Rechenband LSk_Position

-- Turingmaschinensimulatoreingabe
type Initiales_Rechenband = Rechenband
data Sim_Eingabe          = SE Turingtafel Initiales_Rechenband

-- Turingmaschinensimulatorausgabe
type Finaler_interner_Zustand         = Zustand
type Finale_Lese_Schreibkopf_Position = Index
type Finales_Rechenband               = Rechenband

-- Abkuerzungen
type Finaler_iZ  = Finaler_interner_Zustand
type Finale_LSkP = Finale_Lese_Schreibkopf_Position
type Finales_Rb  = Finales_Rechenband

-- Ausgabe
data Sim_Ausgabe = SA Finaler_iZ Finales_Rb Finale_LSkP

-- Zulaessige Turingtafeln
ist_zulaessige_Turingtafel :: Turingtafel -> Bool
ist_zulaessige_Turingtafel ...

-- Zulaessige initiale Rechenbaender
ist_zulaessiges_initiales_Rechenband :: Rechenband -> Bool
ist_zulaessiges_initiales_Rechenband ...

-- Turingmaschinenzustandsuebergangsfunktion, kurz Transitionsfunktion
tm_transition :: Turingmaschinenzustand -> Turingmaschinenzustand
tm_transition ...

-- Spurfunktionen
type Spur  = [Turingmaschinenzustand]
tm_spur :: Turingmaschinenzustand -> Spur
tm_spur ...
zeige_zustand :: Turingmaschinenzustand -> String
zeige_zustand ...
```

```
zeige_spur :: Spur -> String
zeige_spur ...

-- Turingmaschinensimulator
tm_sim :: Sim_Eingabe -> Sim_Ausgabe
tm_sim ...
```

Vervollständigen Sie die Implementierungen der Rechenvorschriften:

(a) `ist_zulaessige_Turingtafel :: Turingtafel -> Bool`

(b) `ist_zulaessiges_initiales_Rechenband :: Rechenband -> Bool`

(c) `tm_transition :: Turingmaschinenzustand -> Turingmaschinenzustand`

(d) `tm_spur :: Turingmaschinenzustand -> Spur`

(e) `zeige_zustand :: Turingmaschinenzustand -> String`

(f) `zeige_spur :: Spur -> String`

(g) `tm_sim :: Sim_Eingabe -> Sim_Ausgabe`

Nehmen Sie dazu bei Bedarf durch Instanzbildungen weitere Zuordnungen von Typen zu Typklassen vor, mit `deriving`-Klauseln, wo möglich, mit `instance`-Deklarationen, wo nicht.

Dabei soll gelten:

(a) *Zulässigkeit Turingtafel:* Eine Turingtafel ist *zulässig* gdw.: Sind `(iz,z,b,ifz)`, `(iz',z',b',ifz')` zwei verschiedene Zeilen der Turingtafel, so gilt: `iz` $\neq$ `iz'` oder `z` $\neq$ `z'`.

(b) *Zulässigkeit Rechenband:* Ein Rechenband ist als initiales Rechenband *zulässig* gdw. die linke Bandhälfte leer ist (d.h. Länge 0 hat). (Nicht vorhandene Felder der konzeptuell unendlichen Halbbänder sind implizit mit dem Leerzeichen beschriftet zu denken.)

(c) *Transition:* Angewendet auf einen Turingmaschinenzustand liefert die Funktion `tm_transition` den Nachfolgezustand entsprechend der Beschreibung der Arbeitsweise einer Turingmaschine in Anhang B.1. Führt die Maschine dabei den Befehl "gehe nach links" oder "gehe nach rechts" aus und das entsprechende Halbband enthält noch kein Feld für die entsprechend neue Position des Lese/Schreibkopfs, so wird das entsprechende Halbband um das neue Feld mit Beschriftung `Leer` erweitert. Ist kein Zustandsübergang möglich (weil die Turingtafel keine Zeile für das Paar aus aktuellem internen Zustand und Zeichen unter dem Lese/Schreibkopf enthält), so liefert die Transitionsfunktion den Argumentzustand als Ergebnis.

(d) *Spur:* Angewendet auf einen Turingmaschinenanfangszustand liefert die Funktion `tm_spur` die (möglicherweise nicht endliche) Liste von Turingmaschinenfolgezuständen, die die Transitionsfunktion angesetzt auf diesen Anfangszustand erzeugt (vgl. die Funktion `ggt_spur` von Aufgabenblatt 6). Die Funktion `tm_spur` terminiert, wenn der aktuelle interne Zustand und das aktuelle Zeichen unter dem Lese/Schreibkopf nicht als Anfang einer Zeile in der Turingtafel vorkommen, wodurch sich die Turingmaschine selbsttätig abschaltet (siehe Arbeitsweise einer Turingmaschine in Anhang B.1).

(e) *Zustandsausgabe:* Die Funktion `zeige_zustand` stellt einen Turingmaschinenzustand als Zeichenreihe dar, die den internen Zustand, den Bandinhalt und die Lese/Schreibkopfposition beschreibt, wobei das Feld mit Index 0 des Bandinhalts zusätzlich durch `"0:"` markiert wird:

```
tmz = TMZ _ 42 ([BA B,Leer,BA A,Leer],[BA C,BA D,Leer,BA E,Leer,Leer]) (-3)
zeige_zustand tmz ->> "(42,[Leer,A,Leer,B,0:C,D,Leer,E,Leer,Leer],-3)"
```

(f) *Spurausgabe:* Die Funktion `zeige_spur` überführt einen Spurwert in eine Zeichenreihendarstellung. Ein Spurwert wird dabei als Folge von Turingmaschinenzuständen dargestellt, jeweils getrennt durch `" ->> "` (d.h. je ein Leerzeichen vor und nach dem Pfeil):

```
spur = [tmz,tmz,tmz]
text = "(42,[Leer,A,Leer,B,0:C,D,Leer,E,Leer,Leer],-3)"
zeige_spur spur ->> "text ->> text ->> text"
```

(g) *Simulator:* Angewendet auf eine Eingabe mit zulässiger Turingtafel und zulässigem initialen Rechenband schaltet `tm_sim` die (simulierte) Turingmaschine wie in Anhang B.1 beschrieben ein. Terminiert die Turingmaschine, d.h. ist die Spur (die Folge von Zustandsübergängen), die die Turingmaschine angesetzt auf Turingtafel- und Rechenbandargument der Eingabe erzeugt, endlich, so liefert `tm_sim` den letzten Zustand der erzeugten Spur als Wert vom Typ `Sim_Ausgabe`; ist die erzeugte Spur nicht endlich, so terminiert die Funktion `tm_sim` nicht.

(h) Machen Sie den Typ `Sim_Ausgabe` zu einer Instanz der Typklasse `Show`, so dass `Sim_Ausgabe`-Werte wie nachstehend illustriert als Zeichenreihen dargestellt werden. Dabei gilt, dass mit der Zeichenreihe `"0:"` wieder die Position des Bandfelds mit Index 0 markiert wird, das unabhängig vom Wert des darauf befindlichen Zeichens stets ausgegeben wird. Weiters gilt, dass `"iZ:"`, `"Band:"`, `"LS-Pos:"` und `";"` von je einem Leerzeichen gefolgt und Leerfelder zu Beginn und am Ende unterdrückt werden (mit Ausnahme des stets ausgegebenen Bandfelds 0):

```
sa1 = SA 42 ([BA B,Leer,BA A,Leer],[BA C,BA D,Leer,BA E,Leer,Leer]) (-3)
sa2 = SA 42 ([Leer,Leer,BA A,Leer],[Leer,Leer,Leer]) (-3)
sa3 = SA 42 ([Leer,Leer,Leer],[Leer,Leer,Leer]) (-3)
show sa1 ->> "iZ: 42; Band: [A,Leer,B,0:C,D,Leer,E]; LS-Pos: -3"
show sa2 ->> "iZ: 42; Band: [A,Leer,Leer,0:Leer]; LS-Pos: -3"
show sa3 ->> "iZ: 42; Band: [0:Leer]; LS-Pos: -3"
```

Testen Sie Ihre Implementierung (ohne Abgabe!) für verschiedene Turingtafeln und Anfangsbandinhalte, z.B. für die im Anhang B.1 angegebenen Turingtafeln und Anfangsbandinhalte. Verwenden Sie dabei das Zeichen `I` für `|` und `Leer` für ƀ, die Richtungswerte `Links` und `Rechts` für L und R.

Haskell Live

Der nächste *Haskell Live*-Termin ist am Freitag, den 07.12.2018.

Haskell Private

Anmeldungen zu *Haskell Private* sind noch möglich. Nutzen Sie die Möglichkeit. Nähere Hinweise und die URL zur Anmeldungsseite finden Sie auf der Homepage der Lehrveranstaltung.