Synthesizing Set Functions: Eine prototypische Implementierung

Niels Bunkenburg

23. Januar 2019

Arbeitsgruppe für Programmiersprachen und Übersetzerkonstruktion
Institut für Informatik
Christian-Albrechts-Universität zu Kiel

Aufgabe

- Set Functions im KiCS2 verhalten sich anders als im PAKCS
- Set Functions kapseln Nichtdeterminismus einer Funktion
  `coin = 0 ? 1` $\Rightarrow$ `coin`$_S$ `= {0,1}`
- Nichtdeterminismus in Argumenten wird nicht gekapselt
  `double x = x + x` $\Rightarrow$ `double`$_S$ `coin = {0}, {2}`
- Neuer Ansatz: Synthesizing Set Functions [Antoy et al., 2018]

Idee: Plural Functions

- Set Functions bestehen aus Plural Functions
- Plural Function für $f :: a \to b$ ist $f_P :: \{a\} \to \{b\}$
- Implementierung von $\{.\}$ als Suchbaum

```
data ST a = Val a | Fail | Choice (ST a) (ST a)
```

- Anpassung der Ausdrücke

```
applyST :: (a -> ST b) -> ST a -> ST b

notP :: ST Bool -> ST Bool
notP = applyST $ \x ->
                   case x of False -> Val True
                             True  -> Val False
```

Datentypen

- Komponenten in Datentypen werden zu Suchbäumen

```
data STList a = Nil | Cons (ST a) (ST (STList a))
```

- Normalform (”Hochziehen” von Nichtdeterminismus)

```
class NF a where
  nf :: a -> ST a
```

- Konvertierung zwischen ursprünglichem und ST-Datentyp

```
class ConvertST a b where
  toValST :: a -> b
  fromValST :: b -> a
```

Implementierung

FlatCurry Funktion $\Rightarrow$ AbstractCurry Programm

Transformationsphasen

1. Aufsammeln der verwendeten Definitionen
2. Lifting von verschachtelten Case-Ausdrücken
3. Plural Function Transformation (Funktionstyp und Regeln)
   $\Rightarrow$ Vorkommende Datentypen werden transformiert
4. Normalform Instanzen
5. ConvertST Instanzen
6. Zusammenbauen der Set Function

Ergebnisse

- Implementierung der Transformation grundsätzlich möglich
- Einschränkungen
  - Verschachtelte Set Functions
  - Polymorphie/Higher-Order/Typsynonyme
  - Externe Funktionen
  - Freie Variablen
- Effizienz und Korrektheit?
- Integration in Präprozessor?

Literatur

S. Antoy, M. Hanus, and F. Teegen. Synthesizing Set Functions. *Proceedings of the 26th International Workshop on Functional and (constraint) Logic Programming (WFLP 2018)*, August 2018.