Notwendigkeit bestimmter Bedingungen?

Wir haben, mit **fix**:

$$\texttt{g}\ p\ (\texttt{map}\ h\ l) \quad = \quad \texttt{map}\ h\ (\texttt{g}\ (p \circ h)\ l)$$

für jedes $\texttt{g} :: (\alpha \to \mathrm{Bool}) \to [\alpha] \to [\alpha]$, wenn

- h strikt ($h\ \bot = \bot$).

Wir haben, mit **fix** und **seq**: . . . , wenn

- $p \neq \bot$,
- h strikt ($h\ \bot = \bot$) und
- h total ($\forall x \neq \bot.\ h\ x \neq \bot$).

Wir haben, mit . . . , wenn . . .

Naheliegende Fragen jeweils:

1. Sind die Bedingungen für jedes $\texttt{g}$ notwendig?
2. Sind sie es für irgendein $\texttt{g}$?

1. Frage, für (nur) **fix**

Sind alle Striktheitsbedingungen für jedes g notwendig? Nein!

Systematischer Ansatz: ersetze

$$\frac{\Gamma \vdash t : \tau \to \tau}{\Gamma \vdash (\textbf{fix}\ t) : \tau}$$

durch

$$\frac{\Gamma \vdash \tau \in \text{Pointed} \qquad \Gamma \vdash t : \tau \to \tau}{\Gamma \vdash (\textbf{fix}\ t) : \tau},$$

wobei

$$\text{Pointed}\ \alpha, \Gamma \vdash \alpha \in \text{Pointed} \qquad \frac{\Gamma \vdash \tau_2 \in \text{Pointed}}{\Gamma \vdash \tau_1 \to \tau_2 \in \text{Pointed}}$$

$$\Gamma \vdash \text{Bool} \in \text{Pointed} \qquad \Gamma \vdash [\tau] \in \text{Pointed}$$

Gewinn: Selbst wenn Relationen für un-Pointed Typen nicht mehr strikt, gilt Parametrizitäts-Theorem weiter! [Launchbury & Paterson 1996]

1. Frage, für (nur) **fix**

Zum Beispiel erhalten wir:

- Für jedes $\mathtt{g} :: \text{Pointed } \alpha \Rightarrow (\alpha \to \text{Bool}) \to [\alpha] \to [\alpha]$,

  $$\mathtt{g}\ p\ (\mathtt{map}\ h\ l) \quad = \quad \mathtt{map}\ h\ (\mathtt{g}\ (p \circ h)\ l)$$

  wenn h strikt.

- Für jedes $\mathtt{g} :: (\alpha \to \text{Bool}) \to [\alpha] \to [\alpha]$ (im neuen Typsystem),

  $$\mathtt{g}\ p\ (\mathtt{map}\ h\ l) \quad = \quad \mathtt{map}\ h\ (\mathtt{g}\ (p \circ h)\ l)$$

  ohne Bedingung an h.

2. Frage, für (nur) **fix**

Zu gegebenem Typ, gibt es ein g, so dass die Striktheitsbedingungen notwendig sind? Nicht immer!

Idealszenario, mit automatischer Unterstützung:

- Ich gebe einen Typ vor, etwa $\mathtt{g} :: (\alpha \to \mathrm{Bool}) \to [\alpha] \to [\alpha]$.
- Ich erhalte ein freies Theorem. Hier:

  für jedes strikte h, $\quad \mathtt{g}\ p\ (\mathtt{map}\ h\ l) \quad = \quad \mathtt{map}\ h\ (\mathtt{g}\ (p \circ h)\ l)$

- Ich frage: warum muss h strikt sein? Was wäre sonst?
- Ich erhalte ein konkretes g, sowie p, l, und (nichtstriktes) h für welche die Aussage dann falsch wäre.

Das entsprechende Tool an einem Beispiel

The Free Theorem

The theorem generated for functions of the type

```
f :: (a -> Int) -> Int
```

is:

```
forall t1,t2 in TYPES, g :: t1 -> t2, g strict.
 forall p :: t1 -> Int.
  forall q :: t2 -> Int.
   (forall x :: t1. p x = q (g x)) ==> (f p = f q)
```

The Counterexample

By disregarding the strictness condition on `g` the theorem becomes wrong. The term

```
f = (\x1 -> (x1 _|_))
```

is a counterexample.

By setting `t1 = t2 = ... = ()` and

```
g = const ()
```

the following would be a consequence of the thus "naivified" free theorem:

```
(f p) = (f q)

where

p        = (\x1 -> 0)
q        = (\x1 -> (case x1 of {() -> 0}))
```

But this is wrong since with the above `f` it reduces to:

```
0 = _|_
```

Noch ein Beispiel

The Free Theorem

The theorem generated for functions of the type

```
f :: [a] -> Int
```

is:

```
forall t1,t2 in TYPES, g :: t1 -> t2, g strict.
 forall x :: [t1]. f x = f (map g x)
```

The Counterexample

Disregarding the strictness condition on `g` the algorithm found no counterexample.

Idee 1: Verwende den Pointed-Ansatz

Zum Beispiel, suche ein g so dass

$$\text{Pointed } \alpha \vdash \mathrm{g} : (\alpha \to \text{Bool}) \to [\alpha] \to [\alpha]$$

aber nicht

$$\alpha \vdash \mathrm{g} : (\alpha \to \text{Bool}) \to [\alpha] \to [\alpha]$$

Natürlicher „Anfang“:

$$\frac{\Gamma \vdash \tau \notin \text{Pointed}}{\Gamma \Vdash (\mathbf{fix}\ (\lambda x : \tau . x)) : \tau}$$

Ansonsten, abhängig vom Typ weitersuchen.

Problem: Nicht alle vorhandenen Regeln sind „syntax-gesteuert“.

Insbesondere:

$$\frac{\Gamma \vdash t : \tau_1 \to \tau_2 \qquad \Gamma \vdash u : \tau_1}{\Gamma \vdash (t\ u) : \tau_2}$$

Idee 2: Curry/Howard-Isomorphismus

- [Dyckhoff 1992] stellt eine Beweisprozedur für intuitionistische Propositional-Logik vor.
- Diese Prozedur kann man als Generator für **fix**-freie Terme zu vorgegebenen polymorphen Typen verwenden (Djinn, [Augustsson 2009]).
- Wir fügen unsere Regel

$$\frac{\Gamma \vdash \tau \notin \text{Pointed}}{\Gamma \Vdash (\textbf{fix}\ (\lambda x : \tau.x)) : \tau}$$

hinzu, führen weitere Anpassungen durch ... [Seidel & V. 2010]

1. Frage, für (fix und) seq

Sind alle Totalitäts- und „$\neq \bot$“-bedingungen für jedes g notwendig?
Nein!

Naheliegender Ansatz: ersetze

$$\frac{\Gamma \vdash t_1 : \tau_1 \qquad \Gamma \vdash t_2 : \tau_2}{\Gamma \vdash (\mathbf{seq}\ t_1\ t_2) : \tau_2}$$

durch

$$\frac{\Gamma \vdash \tau_1 \in \mathsf{Seqable} \qquad \Gamma \vdash t_1 : \tau_1 \qquad \Gamma \vdash t_2 : \tau_2}{\Gamma \vdash (\mathbf{seq}\ t_1\ t_2) : \tau_2},$$

wobei

$$\mathsf{Seqable}\ \alpha, \Gamma \vdash \alpha \in \mathsf{Seqable} \qquad \frac{???}{\Gamma \vdash (\tau_1 \to \tau_2) \in \mathsf{Seqable}}$$

$$\Gamma \vdash \mathsf{Bool} \in \mathsf{Seqable} \qquad \Gamma \vdash [\tau] \in \mathsf{Seqable}$$

Problem: Man braucht einen ganz neuen Ansatz für Funktionstypen.

. . . aber es geht [Seidel & V. 2009]

The term

```
t = (/\a.
     (/\b.
      (\c::(a -> (b -> a)).
       (fix (\h::(a -> ([b] -> a)).
             (\n::a.
              (\ys::[b].
               (seq (c n) (case ys of {[] -> n; x:xs ->
                           (seq xs (seq x (let n' = ((c n) x) in
                                           ((h n') xs))))})))))))))
```

can be typed to the optimal type

```
(forall^n a. (forall^e b. ((a ->^n (b ->^e a)) ->^e (a ->^e ([b] ->^e a)))))
```

with the free theorem

```
forall t1,t2 in TYPES, f :: t1 -> t2, f strict.
 forall t3,t4 in TYPES, g :: t3 -> t4, g strict and total.
  ((t_{t1}_{t3} /= _|_) <=> (t_{t2}_{t4} /= _|_))
  && (forall p :: t1 -> (t3 -> t1).
       forall q :: t2 -> (t4 -> t2).
        (forall x :: t1.
          ((p x /= _|_) <=> (q (f x) /= _|_))
          && (forall y :: t3. f (p x y) = q (f x) (g y)))
        ==> (((t_{t1}_{t3} p /= _|_) <=> (t_{t2}_{t4} q /= _|_))
             && (forall z :: t1.
                  ((t_{t1}_{t3} p z /= _|_) <=> (t_{t2}_{t4} q (f z) /= _|_))
                  && (forall v :: [t3].
                       f (t_{t1}_{t3} p z v) = t_{t2}_{t4} q (f z) (map_{t3}_{t4} g v)))))
```

The normal free theorem for the type without marks would be: