

## Notwendigkeit bestimmter Bedingungen?

Wir haben, mit **fix**:

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

für jedes  $g :: (\alpha \rightarrow \text{Bool}) \rightarrow [\alpha] \rightarrow [\alpha]$ , wenn

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

## Notwendigkeit bestimmter Bedingungen?

Wir haben, mit **fix**:

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

für jedes  $g :: (\alpha \rightarrow \text{Bool}) \rightarrow [\alpha] \rightarrow [\alpha]$ , wenn

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

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

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

## Notwendigkeit bestimmter Bedingungen?

Wir haben, mit **fix**:

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

für jedes  $g :: (\alpha \rightarrow \text{Bool}) \rightarrow [\alpha] \rightarrow [\alpha]$ , wenn

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

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

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

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

## Notwendigkeit bestimmter Bedingungen?

Wir haben, mit **fix**:

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

für jedes  $g :: (\alpha \rightarrow \text{Bool}) \rightarrow [\alpha] \rightarrow [\alpha]$ , wenn

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

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

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

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

Naheliegende Fragen jeweils:

1. Sind die Bedingungen für jedes  $g$  notwendig?

## Notwendigkeit bestimmter Bedingungen?

Wir haben, mit **fix**:

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

für jedes  $g :: (\alpha \rightarrow \text{Bool}) \rightarrow [\alpha] \rightarrow [\alpha]$ , wenn

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

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

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

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

Naheliegende Fragen jeweils:

1. Sind die Bedingungen für jedes  $g$  notwendig?
2. Sind sie es für irgendein  $g$ ?

## 1. Frage, für (nur) **fix**

Sind alle Striktheitsbedingungen für jedes  $g$  notwendig?

## 1. Frage, für (nur) **fix**

Sind alle Striktheitsbedingungen für jedes  $g$  notwendig? **Nein!**

## 1. Frage, für (nur) **fix**

Sind alle Striktheitsbedingungen für jedes  $g$  notwendig? Nein!

Systematischer Ansatz: ersetze

$$\frac{\Gamma \vdash t : \tau \rightarrow \tau}{\Gamma \vdash (\mathbf{fix} \ t) : \tau}$$

durch

$$\frac{\Gamma \vdash \tau \in \mathbf{Pointed} \quad \Gamma \vdash t : \tau \rightarrow \tau}{\Gamma \vdash (\mathbf{fix} \ t) : \tau}$$



## 1. Frage, für (nur) **fix**

Sind alle Striktheitsbedingungen für jedes  $g$  notwendig? Nein!

Systematischer Ansatz: ersetze

$$\frac{\Gamma \vdash t : \tau \rightarrow \tau}{\Gamma \vdash (\mathbf{fix} \ t) : \tau}$$

durch

$$\frac{\Gamma \vdash \tau \in \mathbf{Pointed} \quad \Gamma \vdash t : \tau \rightarrow \tau}{\Gamma \vdash (\mathbf{fix} \ t) : \tau},$$

wobei

$$\mathbf{Pointed} \ \alpha, \Gamma \vdash \alpha \in \mathbf{Pointed}$$

$$\frac{\Gamma \vdash \tau_2 \in \mathbf{Pointed}}{\Gamma \vdash \tau_1 \rightarrow \tau_2 \in \mathbf{Pointed}}$$

$$\Gamma \vdash \mathbf{Bool} \in \mathbf{Pointed}$$

$$\Gamma \vdash [\tau] \in \mathbf{Pointed}$$

## 1. Frage, für (nur) **fix**

Sind alle Striktheitsbedingungen für jedes **g** notwendig? Nein!

Systematischer Ansatz: ersetze

$$\frac{\Gamma \vdash t : \tau \rightarrow \tau}{\Gamma \vdash (\mathbf{fix} \ t) : \tau}$$

durch

$$\frac{\Gamma \vdash \tau \in \text{Pointed} \quad \Gamma \vdash t : \tau \rightarrow \tau}{\Gamma \vdash (\mathbf{fix} \ t) : \tau},$$

wobei

Pointed  $\alpha, \Gamma \vdash \alpha \in \text{Pointed}$

$$\frac{\Gamma \vdash \tau_2 \in \text{Pointed}}{\Gamma \vdash \tau_1 \rightarrow \tau_2 \in \text{Pointed}}$$

$\Gamma \vdash \text{Bool} \in \text{Pointed}$

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

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

# 1. Frage, für (nur) **fix**

Zum Beispiel erhalten wir:

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

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

wenn  $h$  strikt.

# 1. Frage, für (nur) **fix**

Zum Beispiel erhalten wir:

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

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

wenn  $h$  strikt.

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

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

ohne Bedingung an  $h$ .

## 2. Frage, für (nur) **fix**

Zu gegebenem Typ, gibt es ein **g**, so dass die Striktheitsbedingungen notwendig sind?

## 2. Frage, für (nur) **fix**

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

## 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  $g :: (\alpha \rightarrow \text{Bool}) \rightarrow [\alpha] \rightarrow [\alpha]$ .

## 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  $g :: (\alpha \rightarrow \text{Bool}) \rightarrow [\alpha] \rightarrow [\alpha]$ .
- Ich erhalte ein freies Theorem. Hier:

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



## 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  $g :: (\alpha \rightarrow \text{Bool}) \rightarrow [\alpha] \rightarrow [\alpha]$ .
- Ich erhalte ein freies Theorem. Hier:  
für jedes strikte  $h$ ,  $g \ p \ (\text{map } h \ l) = \text{map } h \ (g \ (p \circ h) \ l)$
- Ich frage: warum muss  $h$  strikt sein? Was wäre sonst?

## 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  $g :: (\alpha \rightarrow \text{Bool}) \rightarrow [\alpha] \rightarrow [\alpha]$ .
- Ich erhalte ein freies Theorem. Hier:  
für jedes strikte  $h$ ,  $g\ p\ (\text{map}\ h\ l) = \text{map}\ h\ (g\ (p \circ h)\ l)$
- Ich frage: warum muss  $h$  strikt sein? Was wäre sonst?
- Ich erhalte ein konkretes **g**, sowie  $p$ ,  $l$ , und (nichstriktes)  $h$  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  $t1 = t2 = \dots = ()$  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

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

aber nicht

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

## Idee 1: Verwende den Pointed-Ansatz

Zum Beispiel, suche ein  $g$  so dass

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

aber nicht

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

Natürlicher „Anfang“:

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

## Idee 1: Verwende den Pointed-Ansatz

Zum Beispiel, suche ein  $g$  so dass

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

aber nicht

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

Natürlicher „Anfang“:

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

Ansonsten, abhängig vom Typ weitersuchen.

## Idee 1: Verwende den Pointed-Ansatz

Zum Beispiel, suche ein  $g$  so dass

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

aber nicht

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

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“.



## Idee 1: Verwende den Pointed-Ansatz

Zum Beispiel, suche ein  $g$  so dass

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

aber nicht

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

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:

$$\frac{\Gamma \vdash t : \tau_1 \rightarrow \tau_2 \quad \Gamma \vdash u : \tau_1}{\Gamma \vdash (t u) : \tau_2}$$

## Idee 1: Verwende den Pointed-Ansatz

Zum Beispiel, suche ein  $g$  so dass

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

aber nicht

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

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:

$$\frac{\Gamma \vdash t : \tau_1 \rightarrow \tau_2 \quad \Gamma \vdash u : \tau_1}{\Gamma \vdash (t u) : \tau_2}$$

## Idee 1: Verwende den Pointed-Ansatz

Zum Beispiel, suche ein  $g$  so dass

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

aber nicht

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

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:

$$\frac{\Gamma \vdash \tau_1 \rightarrow \tau_2 \quad \Gamma \vdash \tau_1}{\Gamma \vdash \tau_2}$$

## Idee 2: Curry/Howard-Isomorphismus

- [Dyckhoff 1992] stellt eine Beweisprozedur für intuitionistische Propositional-Logik vor.

## 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]).

## 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

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

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

## 1. Frage, für (**fix** und) **seq**

Sind alle Totalitäts- und „ $\neq \perp$ “-bedingungen für jedes **g** notwendig?

## 1. Frage, für (**fix** und) **seq**

Sind alle Totalitäts- und „ $\neq \perp$ “-bedingungen für jedes **g** notwendig?

**Nein!**



# 1. Frage, für (**fix** und) **seq**

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

Nein!

Naheliegender Ansatz: ersetze

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

durch

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

# 1. Frage, für (**fix** und) **seq**

Sind alle Totalitäts- und „ $\neq \perp$ “-bedingungen für jedes **g** notwendig?

Nein!

Naheliegender Ansatz: ersetze

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

durch

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

wobei

$\mathbf{Seqable} \ \alpha, \Gamma \vdash \alpha \in \mathbf{Seqable}$

$$\frac{\mathbf{???}}{\Gamma \vdash (\tau_1 \rightarrow \tau_2) \in \mathbf{Seqable}}$$

$\Gamma \vdash \mathbf{Bool} \in \mathbf{Seqable}$

$\Gamma \vdash [\tau] \in \mathbf{Seqable}$

# 1. Frage, für (**fix** und) **seq**

Sind alle Totalitäts- und „ $\neq \perp$ “-bedingungen für jedes **g** notwendig?

Nein!

Naheliegender Ansatz: ersetze

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

durch

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

wobei

$\mathbf{Seqable} \ \alpha, \Gamma \vdash \alpha \in \mathbf{Seqable}$

$$\frac{\mathbf{???}}{\Gamma \vdash (\tau_1 \rightarrow \tau_2) \in \mathbf{Seqable}}$$

$\Gamma \vdash \mathbf{Bool} \in \mathbf{Seqable}$

$\Gamma \vdash [\tau] \in \mathbf{Seqable}$

**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: