# Einführung in die Informatik 2

Prof. Dr. Seidl, J. Kranz, N. Hartmann, J. Brunner
WS 2016/17

**Übungsblatt 12**
Abgabefrist: 30.1.2017

### Aufgabe 12.1 (P) Erste große Schritte

Gegeben seien folgende MiniOCaml-Definitionen:

```
let f = fun x -> x+1*2
let g = fun x -> x+42
let h = 3
```

Konstruieren Sie Beweise für folgende Aussagen:

1. `match` $[1;2;3]$ `with` $[\,] \texttt{ -> } 0 \mid x :: xs \texttt{ -> } f\ x \Rightarrow 3$
2. `let` $x = 5$ `in` $g\ x + h \Rightarrow 50$

### Lösungsvorschlag 12.1

Um die Beweisbäume möglichst lesbar zu halten, wurde auf die Benutzung von $v \Rightarrow v$ verzichtet.

1.

$$\text{PM}\ \frac{[1;2;3] \Rightarrow 1 :: [2;3] \qquad \text{APP}\ \dfrac{\text{GD}\ \dfrac{f = \texttt{fun}\ x\ \texttt{->}\ x + 1 * 2}{f \Rightarrow \texttt{fun}\ x\ \texttt{->}\ x + 1 * 2} \qquad \text{OP}\ \dfrac{\text{OP}\ \dfrac{1 * 2 \Rightarrow 2}{1 * 2 \Rightarrow 2} \qquad 1 + 2 \Rightarrow 3}{1 + 1 * 2 \Rightarrow 3}}{f\ 1 \Rightarrow 3}}{\texttt{match}\ [1;2;3]\ \texttt{with}\ [\,] \texttt{ -> } 0 \mid x :: xs \texttt{ -> } f\ x \Rightarrow 3}$$

2.

$$\text{LD}\ \frac{\text{OP}\ \dfrac{\text{APP}\ \dfrac{\text{GD}\ \dfrac{g = \texttt{fun}\ x\ \texttt{->}\ x + 42}{g \Rightarrow \texttt{fun}\ x\ \texttt{->}\ x + 42} \qquad \text{OP}\ \dfrac{5 + 42 \Rightarrow 47}{5 + 42 \Rightarrow 47}}{g\ 5 \Rightarrow 47} \qquad \text{GD}\ \dfrac{h = 3}{h \Rightarrow 3} \qquad 47 + 3 \Rightarrow 50}{g\ 5 + h \Rightarrow 50}}{\texttt{let}\ x = 5\ \texttt{in}\ g\ x + h \Rightarrow 50}$$

### Aufgabe 12.2 (P) Big-Step Konstruktoren

In dieser Aufgabe sollen Sie die MiniOCaml Sprache um Konstrukturen erweitern.

1. Überlegen Sie sich, welche neuen Werte die Ausdrücke der erweiterten Sprache dadurch annehmen können.

2. Definieren Sie die notwendigen Regeln der Big-Step-Semantik um Beweise über Ausdrücke der erweiterten Sprache führen zu können. Verändern Sie die existierenden Regeln und Axiome nicht.

3. Gegeben sei die folgende Definition:

```
let rec f = fun x -> Expr (0, 1 - x)
and g = fun x -> match x with Const a -> 42 | Expr (a, b) -> a - b
```

   Zeigen Sie, dass der Ausdruck `2 + g (f 6)` zu 7 auswertet.

**Lösungsvorschlag 12.2**

1. Für beliebige Werte $v_0, \ldots, v_n$ und einen beliebigen Konstruktor $Cons$ können Ausdrücke der erweiterten Sprache die Werte der Form $Cons\ (v_0, \ldots, v_n)$ annehmen.

2. Es muss lediglich die neue Regel

$$\text{CO}\ \frac{e_0 \Rightarrow v_0 \qquad \ldots \qquad e_n \Rightarrow v_n}{Cons\ (e_0, \ldots, e_n) \Rightarrow Cons\ (v_0, \ldots, v_n)}$$

   hinzugefügt werden.

3. Der Beweisbaum lässt sich wie folgt konstruieren:

APP
GD
g = fun x -> match x with Const a -> 42 | Expr (a,b) -> a − b
g ⇒ fun x -> match x with Const a -> 42 | Expr (a,b) -> a − b
APP
GD
f = fun x -> Expr (0,1 − x)
f ⇒ fun x -> Expr (0,1 − x)
CO
Expr (0,1 − 6) ⇒ Expr (0,−5)
OP
1 − 6 ⇒ −5
f 6 ⇒ Expr (0,−5)
PM
match Expr (0,−5) with Const a -> 42 | Expr (a,b) -> a − b ⇒ 5
OP
0 − (−5) ⇒ 5
Expr (0,−5) ⇒ Expr (0,−5)
g (f 6) ⇒ 5
OP
2 + 5 ⇒ 7
2 + g (f 6) ⇒ 7

### Aufgabe 12.3 (P) Terminierung

Gegeben sei nun:

```
let rec doit a x =
  if x = 0
    then a
  else
    doit (a + 2*x - 1) (x - 1)
let square x = doit 0 x
```

Zeigen Sie, dass `square` für alle positiven `x` terminiert.

### Lösungsvorschlag 12.3

Wir nehmen folgende Setzungen vor:

$$
\begin{aligned}
e_{match} &= \texttt{match}\ x\ \texttt{with}\ 0\ \texttt{->}\ a\ \mid\ x\ \texttt{->}\ \texttt{doit}\ (a + 2 * x - 1)\ (x - 1)\\
e_{funx} &= \texttt{fun x ->}\ e_{match}\\
\pi_a &= \text{GD}\ \frac{\texttt{doit = fun a x ->}\ e_{funx}}{\texttt{doit} \Rightarrow \texttt{fun a x ->}\ e_{funx}}\\
\pi_b &= \text{GD}\ \frac{\texttt{square = fun x ->\ doit 0 x}}{\texttt{square} \Rightarrow \texttt{fun x ->\ doit 0 x}}
\end{aligned}
$$

Wir zeigen hier eine stärke Aussage, nämlich dass `square` bzw. `doit` für $x \geq 0$ terminieren. Induktionsanfang $x = 0$:

$$
\text{App}\ \frac{\pi_b \qquad \text{App}\ \dfrac{\pi_a \qquad \text{PM}\ \dfrac{0 \Rightarrow 0 \equiv 0 \qquad 0 \Rightarrow 0}{e_{match}[0/x] \Rightarrow 0}}{\texttt{doit 0 0} \Rightarrow 0}}{\texttt{square 0} \Rightarrow 0}
$$

Induktionsschritt $x > 0$:

$$
\text{App}\ \frac{\pi_b \qquad \text{App}\ \dfrac{\pi_a \qquad \text{PM}\ \dfrac{x \Rightarrow x \equiv x \qquad \text{App}\ \dfrac{\pi_a \qquad (0 + 2 * x - 1) \Rightarrow v' \qquad \text{Op}\ \dfrac{x \Rightarrow x \qquad 1 \Rightarrow 1 \qquad x - 1 \Rightarrow x'}{(x - 1) \Rightarrow x'} \qquad \texttt{doit v' x'} \Rightarrow v}{\texttt{doit (0+2*x-1) (x-1)} \Rightarrow v}}{e_{match}[0/a, x/x] \Rightarrow v}}{\texttt{doit 0 x} \Rightarrow v}}{\texttt{square x} \Rightarrow v}
$$

Es wurde bei der Auswahl der Pattern-Regel verwendet, dass $x \neq 0$ gilt. Da $x' < x$ gilt, können wir entsprechend der Induktionsannahme (`doit` terminiert für alle $k$ mit $0 \leq k < x$) auf Terminierung schließen.

### Allgemeine Hinweise zur Hausaufgabenabgabe

Die Hausaufgabenabgabe bezüglich dieses Blattes erfolgt ausschließlich über Moodle. Die Abgabefrist finden Sie ebenfalls in Moodle. Bitte vergewissern Sie sich nach der Abgabe selbstständig, dass alle Dateien erfolgreich auf Moodle eingestellt wurden. Die Abgaben der Theorieaufgaben[1] müssen in Form einer einzigen PDF-Datei erfolgen. Nutzen Sie z.B. ImageMagick[2], um aus eingescannten Bildern ein PDF zu erzeugen. Achten Sie bei eingescannten Bildern darauf, dass diese im PDF-Dokument eine korrekte Orientierung haben. **Abgaben, die sich nicht in beschriebenen Formaten befinden, können nicht korrigiert oder bewertet werden!**

### Aufgabe 12.4 (H) Beweisen mit Big-Step [5 Punkte]

Zeigen oder widerlegen Sie die folgenden Aussagen:

1. Der MiniOCaml-Ausdruck `let a = 2 in (a + 3) * a` wertet zu 10 aus.
2. Der MiniOCaml-Ausdruck

   ```
   match (6-1, 4::[]) with (a, b::c) -> b+b | (a, b) -> 9
   ```

   wertet zu 9 aus.
3. Der MiniOCaml-Ausdruck

   ```
   let w = fun x -> fun y -> fun z -> y x + z in
     w 2 (fun a -> a - 1)
   ```

   wertet zu `fun z -> (fun a -> a - 1) 2 + z` aus.
4. Der MiniOCaml-Ausdruck

   ```
   let f = fun x -> 3 + x * 4 in f (f 3) + 4
   ```

   wertet zu 67 aus.

### Lösungsvorschlag 12.4

1. Der Beweisbaum für den Wert 10 lässt sich widerspruchsfrei konstruieren, daher gilt die Aussage.

$$\text{LD}\ \frac{2 \Rightarrow 2 \qquad \text{OP}\ \dfrac{\text{OP}\ \dfrac{2 \Rightarrow 2 \qquad 3 \Rightarrow 3 \qquad 2 + 3 \Rightarrow 5}{2 + 3 \Rightarrow 5} \qquad 2 \Rightarrow 2 \qquad 5 * 2 \Rightarrow 10}{(2 + 3) * 2 \Rightarrow 10}}{\texttt{let}\ a\ \texttt{=}\ 2\ \texttt{in}\ (a + 3) * a \Rightarrow 10}$$

2. Nur für den Wert 8 lässt sich ein korrekter Beweisbaum erzeugen. Alle anderen Werte (so auch 9) führen zu einem Widerspruch.

$$\text{PM}\ \frac{\text{TU}\ \dfrac{\text{OP}\ \dfrac{6 \Rightarrow 6 \qquad 1 \Rightarrow 1 \qquad 6 - 1 \Rightarrow 5}{6 - 1 \Rightarrow 5} \qquad \text{LI}\ \dfrac{4 \Rightarrow 4 \qquad [\,] \Rightarrow [\,]}{4 :: [] \Rightarrow 4 :: [\,]}}{(6 - 1, 4 :: []) \Rightarrow (5, 4 :: [\,])} \qquad \text{OP}\ \dfrac{4 \Rightarrow 4 \qquad 4 \Rightarrow 4 \qquad 4 + 4 \Rightarrow 8}{4 + 4 \Rightarrow 8}}{\texttt{match}\ (6 - 1, 4 :: [])\ \texttt{with}\ (a, b :: c)\ \texttt{->}\ b + b\ \mid\ (a, b)\ \texttt{->}\ 9 \Rightarrow 8}$$

3. Der Beweisbaum lässt sich widerspruchsfrei konstruieren, daher gilt die Aussage. Man beachte, dass Funktionsrümpfe nicht weiter berechnet werden.

$$\pi = \quad \text{APP}\ \frac{\texttt{fun}\ x\ \texttt{->}\ \texttt{fun}\ y\ \texttt{->}\ \texttt{fun}\ z\ \texttt{->}\ y\,x + z \Rightarrow \texttt{fun}\ x\ \texttt{->}\ \texttt{fun}\ y\ \texttt{->}\ \texttt{fun}\ z\ \texttt{->}\ y\,x + z \qquad 2 \Rightarrow 2 \qquad \texttt{fun}\ y\ \texttt{->}\ \texttt{fun}\ z\ \texttt{->}\ y\,2 + z \Rightarrow \texttt{fun}\ y\ \texttt{->}\ \texttt{fun}\ z\ \texttt{->}\ y\,2 + z}{(\texttt{fun}\ x\ \texttt{->}\ \texttt{fun}\ y\ \texttt{->}\ \texttt{fun}\ z\ \texttt{->}\ y\,x + z)\,2 \Rightarrow \texttt{fun}\ y\ \texttt{->}\ \texttt{fun}\ z\ \texttt{->}\ y\,2 + z}$$

$$\text{LD}\ \frac{\texttt{fun}\ x\ \texttt{->}\ \texttt{fun}\ y\ \texttt{->}\ \texttt{fun}\ z\ \texttt{->}\ y\,x + z \Rightarrow \texttt{fun}\ x\ \texttt{->}\ \texttt{fun}\ y\ \texttt{->}\ \texttt{fun}\ z\ \texttt{->}\ y\,x + z \qquad \text{APP}\ \dfrac{\pi \qquad \texttt{fun}\ a\ \texttt{->}\ a - 1 \Rightarrow \texttt{fun}\ a\ \texttt{->}\ a - 1 \qquad \texttt{fun}\ z\ \texttt{->}\ (\texttt{fun}\ a\ \texttt{->}\ a - 1)\,2 + z \Rightarrow \texttt{fun}\ z\ \texttt{->}\ (\texttt{fun}\ a\ \texttt{->}\ a - 1)\,2 + z}{(\texttt{fun}\ x\ \texttt{->}\ \texttt{fun}\ y\ \texttt{->}\ \texttt{fun}\ z\ \texttt{->}\ y\,x + z)\,2\,(\texttt{fun}\ a\ \texttt{->}\ a - 1) \Rightarrow \texttt{fun}\ z\ \texttt{->}\ (\texttt{fun}\ a\ \texttt{->}\ a - 1)\,2 + z}}{\texttt{let}\ w\ \texttt{=}\ \texttt{fun}\ x\ \texttt{->}\ \texttt{fun}\ y\ \texttt{->}\ \texttt{fun}\ z\ \texttt{->}\ y\,x + z\ \texttt{in}\ w\,2\,(\texttt{fun}\ a\ \texttt{->}\ a - 1) \Rightarrow \texttt{fun}\ z\ \texttt{->}\ (\texttt{fun}\ a\ \texttt{->}\ a - 1)\,2 + z}$$

4. Der Beweisbaum für den Wert 67 lässt sich widerspruchsfrei konstruieren, daher gilt die Aussage.

$$\frac{\texttt{fun}\ x\ \texttt{->}\ 3 + x * 4 \Rightarrow \texttt{fun}\ x\ \texttt{->}\ 3 + x * 4 \qquad \dfrac{\texttt{fun}\ x\ \texttt{->}\ 3 + x * 4 \Rightarrow \texttt{fun}\ x\ \texttt{->}\ 3 + x * 4 \qquad 3 \Rightarrow 3 \qquad \dfrac{3 \Rightarrow 3 \qquad \dfrac{3 \Rightarrow 3 \qquad 4 \Rightarrow 4 \qquad 3 * 4 \Rightarrow 12}{3 * 4 \Rightarrow 12} \qquad 3 + 12 \Rightarrow 15}{3 + 3 * 4 \Rightarrow 15} \qquad 3 \Rightarrow 3 \qquad \dfrac{15 \Rightarrow 15 \qquad \dfrac{4 \Rightarrow 4 \qquad 15 * 4 \Rightarrow 60}{15 * 4 \Rightarrow 60} \qquad 3 + 60 \Rightarrow 63}{3 + 15 * 4 \Rightarrow 63}}{(\texttt{fun}\ x\ \texttt{->}\ 3 + x * 4)\ (\texttt{fun}\ x\ \texttt{->}\ 3 + x * 4)\ \ 3) \Rightarrow 63} \qquad 4 \Rightarrow 4 \qquad 63 + 4 \Rightarrow 67}{(\texttt{fun}\ x\ \texttt{->}\ 3 + x * 4)\ (\texttt{fun}\ x\ \texttt{->}\ 3 + x * 4)\ \ 3) + 4 \Rightarrow 67}$$

$$\texttt{let}\ f\ \texttt{=}\ \texttt{fun}\ x\ \texttt{->}\ 3 + x * 4\ \texttt{in}\ f\ (f\ \ 3) + 4 \Rightarrow 67$$

**Aufgabe 12.5 (H) Mit großen Schritten zum Rekord** [7 Punkte]

In dieser Aufgabe sollen Sie die MiniOCaml Sprache um Records erweitern.

1. Überlegen Sie sich dazu zunächst, welche neuen Werte ein Ausdruck dadurch annehmen kann. Definieren Sie die notwendigen Regeln der Big-Step-Semantik, um die Korrektheit von Programmen mit Records zu beweisen. Verändern Sie die existierenden Beweisregeln und Axiome nicht.

2. Überlegen Sie, welchen Wert der Ausdruck

   ```
   { x=(1, true); y=3::[]; }.x :: { x=(2, false); y=[]; }.y
   ```

   der erweiterten MiniOCaml Sprache berechnet. Beweisen Sie dies.

3. Zeigen Sie, dass

   ```
   let x = { a = 21; b = fun n -> 2::n; } in
   let y = { p = []; q = x } in
     3 :: y.q.b y.p
   ```

   den Wert `3::2::[]` berechnet.

**Lösungsvorschlag 12.5**

Ausdrücke der erweiterten Sprachen können, neben den bisher zugelassenen Werten, auch Werte der Form $\{\ x_0 = v_0;\ \ldots\ x_n = v_n;\ \}$ annehmen. Um Programme dieser erweiterten Sprache beweisen zu können, werden zwei neue Regeln benötigt:

- Die erste für die Berechnung eines Record-Wertes (RE):

$$\text{RE}\ \frac{e_0 \Rightarrow v_0 \qquad \ldots \qquad e_n \Rightarrow v_n}{\{\ x_0 = e_0;\ \ldots\ x_n = e_n\ \} \Rightarrow \{\ x_0 = v_0;\ \ldots\ x_n = v_n\ \}}$$

- Und die zweite für den Zugriff auf ein Element (RA=Record Access):

$$\text{RA}\ \frac{e \Rightarrow \{\ \ldots\ \ x = v;\ \ldots\ \}}{e.x \Rightarrow v}$$

$$\text{LI}\;\dfrac{\text{RA}\;\dfrac{\text{RE}\;\dfrac{(1,true)\Rightarrow(1,true)\qquad 3::[\,]\Rightarrow 3::[\,]}{\{\;x=(1,true);\;\;y=3::[\,];\;\}\Rightarrow\{\;x=(1,true);\;\;y=3::[\,];\;\}}}{\{\;x=(1,true);\;\;y=3::[\,];\;\}.x\Rightarrow(1,true)}\qquad\text{RA}\;\dfrac{\text{RE}\;\dfrac{(2,false)\Rightarrow(2,false)\qquad [\,]\Rightarrow[\,]}{\{\;x=(2,false);\;\;y=[\,];\;\}\Rightarrow\{\;x=(2,false);\;\;y=[\,];\;\}}}{\{\;x=(2,false);\;\;y=[\,];\;\}.y\Rightarrow[\,]}}{\{\;x=(1,true);\;y=3::[\,];\;\}.x::\{\;x=(2,false);\;y=[\,];\;\}.y\Rightarrow(1,true)::[\,]}$$

Sei im Folgenden: $V_x = \{\;a=21;\;b=\texttt{fun}\;\;n\;\;\texttt{->}\;\;2::n;\;\}$ und $V_y = \{\;p=[\,];\;q=V_x;\;\}$

$$\text{LD}\;\dfrac{V_x\Rightarrow V_x\;\text{LD}\;\dfrac{V_y\Rightarrow V_y\;\text{LI}\;\dfrac{3\Rightarrow 3\;\text{APP}\;\dfrac{\text{RA}\;\dfrac{\text{RE}\;\dfrac{[\,]\Rightarrow[\,]\quad\text{RE}\;\dfrac{21\Rightarrow 21\qquad\texttt{fun}\;\;n\;\;\texttt{->}\;\;2::n\Rightarrow\texttt{fun}\;\;n\;\;\texttt{->}\;\;2::n}{V_x\Rightarrow V_x}}{V_y\Rightarrow V_y}}{\{\;p=[\,];\;\;q=\{\;a=21;\;\;b=\texttt{fun}\;\;n\;\;\texttt{->}\;\;2::n;\;\};\;\}.q\Rightarrow\{\;a=21;\;\;b=\texttt{fun}\;\;n\;\;\texttt{->}\;\;2::n;\;\}}\quad\text{RA}\;\dfrac{\text{RE}\;\dfrac{[\,]\Rightarrow[\,]\quad\text{RE}\;\dfrac{21\Rightarrow 21\qquad\texttt{fun}\;\;n\;\;\texttt{->}\;\;2::n\Rightarrow\texttt{fun}\;\;n\;\;\texttt{->}\;\;2::n}{V_x\Rightarrow V_x}}{V_y\Rightarrow V_y}}{\{\;p=[\,];\;\;q=\{\;a=21;\;\;b=\texttt{fun}\;\;n\;\;\texttt{->}\;\;2::n;\;\};\;\}.p\Rightarrow[\,]\qquad 2::[\,]\Rightarrow 2::[\,]}}{\{\;p=[\,];\;\;q=\{\;a=21;\;\;b=\texttt{fun}\;\;n\;\;\texttt{->}\;\;2::n;\;\};\;\}.q.b\;\{\;p=[\,];\;\;q=\{\;a=21;\;\;b=\texttt{fun}\;\;n\;\;\texttt{->}\;\;2::n;\;\};\;\}.p\Rightarrow 2::[\,]}}{3::\{\;p=[\,];\;\;q=\{\;a=21;\;\;b=\texttt{fun}\;\;n\;\;\texttt{->}\;\;2::n;\;\};\;\}.q.b\;\{\;p=[\,];\;\;q=\{\;a=21;\;\;b=\texttt{fun}\;\;n\;\;\texttt{->}\;\;2::n;\;\};\;\}.p\Rightarrow 3::2::[\,]}}{\texttt{let}\;\;y\;\texttt{=}\;\{\;p=[\,];\;q=\{\;a=21;\;b=\texttt{fun}\;\;n\;\;\texttt{->}\;\;2::n;\;\};\;\}\;\;\texttt{in}\;\;3\;\;::\;\;y.q.b\;y.p\Rightarrow 3::2::[\,]}}{\texttt{let}\;\;x\;\texttt{=}\;\{\;a=21;\;b=\texttt{fun}\;\;n\;\;\texttt{->}\;\;2::n;\;\}\;\;\texttt{in}\;\;\texttt{let}\;\;y\;\texttt{=}\;\{\;p=[\,];\;q=x\;\}\;\;\texttt{in}\;\;3\;\;::\;\;y.q.b\;y.p\Rightarrow 3::2::[\,]}$$

**Aufgabe 12.6 (H) Terminierung** [8 Punkte]

Gegeben sei folgendes MiniOcaml-Programm:

```
let rec rev_zip x y acc = match (x, y) with
    ([], []) -> acc
  | ([], y::ys) -> rev_zip ys x (y::acc)
  | (x::xs, _) -> rev_zip y xs (x::acc)
```

Nutzen Sie die Big-Step-Semantik, um folgende Aussage durch Induktion zu zeigen:

`rev_zip x y []` terminiert für alle Listen `x` und `y`.

**Lösungsvorschlag 12.6**

Wir beweisen die allgemeinere Aussage

`rev_zip x y acc` terminiert für alle Listen `x`, `y` und `acc`.

per Induktion über die Summe der Längen der Listen `x` und `y`.

1. **Induktionsanfang**: Summe ist 0, also gilt `x` = `y` = `[]`.

$$\text{APP}\ \frac{\text{GD}\ \dfrac{\texttt{rev\_zip} = \texttt{fun x y acc -> ...}}{\texttt{rev\_zip} \Rightarrow \texttt{fun x y acc -> ...}} \qquad \texttt{[]} \Rightarrow \texttt{[]} \qquad \texttt{[]} \Rightarrow \texttt{[]} \qquad \texttt{acc} \Rightarrow \texttt{acc} \qquad \text{PM}\ \dfrac{\texttt{([], [])} \Rightarrow \texttt{([], [])} \equiv \texttt{([], [])}[] \qquad \texttt{acc} \Rightarrow \texttt{acc}}{\texttt{match ([], []) with ([], []) -> acc ...} \Rightarrow \texttt{acc}}}{\texttt{rez\_zip [] [] acc} \Rightarrow \texttt{acc}}$$

2. **Induktionsschritt**: Wir nehmen nun an, `rev_zip` terminiert für alle Eingabeliste der Summenlänge $l \geq 0$ mit $l < t$. Sei nun die Summe der Längen der Listen `x` und `y` gleich $t$. Wir unterscheiden zwei Fälle:

   (a) Es gilt `x` = `[]` und `y` = `hdy::ys`:

$$\text{APP}\ \frac{\text{GD}\ \dfrac{\texttt{rev\_zip} = \texttt{fun x y acc -> ...}}{\texttt{rev\_zip} \Rightarrow \texttt{fun x y acc -> ...}} \qquad \texttt{[]} \Rightarrow \texttt{[]} \qquad \texttt{hdy::ys} \Rightarrow \texttt{hdy::ys} \qquad \texttt{acc} \Rightarrow \texttt{acc} \qquad \text{PM}\ \dfrac{\texttt{([], hdy::ys)} \Rightarrow \texttt{([], hdy::ys)} \equiv \texttt{([], y::ys)}[\texttt{hdy/y, ys/ys}] \qquad \texttt{rev\_zip ys [] (hdy::acc)} \Rightarrow \texttt{v'}}{\texttt{match ([], hdy::ys) with ... ([], y::ys) -> rev\_zip ys [] (y::acc) ...} \Rightarrow \texttt{v'}}}{\texttt{rez\_zip [] hdy::ys acc} \Rightarrow \texttt{v'}}$$

   (b) Es gilt `x` = `hdx::xs`:

$$\text{APP}\ \frac{\text{GD}\ \dfrac{\texttt{rev\_zip} = \texttt{fun x y acc -> ...}}{\texttt{rev\_zip} \Rightarrow \texttt{fun x y acc -> ...}} \qquad \texttt{hdx::xs} \Rightarrow \texttt{hdx::xs} \qquad \texttt{y} \Rightarrow \texttt{y} \qquad \texttt{acc} \Rightarrow \texttt{acc} \qquad \text{PM}\ \dfrac{\texttt{(hdx::xs, [])} \Rightarrow \texttt{(hdx::xs, [])} \equiv \texttt{(x::xs, \_)}[\texttt{hdx/x, xs/xs, y/\_}] \qquad \texttt{rev\_zip y xs (hdx::acc)} \Rightarrow \texttt{v'}}{\texttt{match (hdx::xs, y) with ... (x::xs, \_) -> rev\_zip y xs (x::acc)} \Rightarrow \texttt{v'}}}{\texttt{rez\_zip hdx::xs y acc} \Rightarrow \texttt{v'}}$$

   Aus der Induktionsannahme folgt, dass die Aufrufe `rev_zip ys [] (hdy::acc)` bzw. `rev_zip y xs (hdx::acc)` terminieren, da die Summe der Längen von `x` und `y` abgenommen hat.