

# Operations on Bounded Natural Functors

Jasmin Christian Blanchette      Andrei Popescu      Dmitriy Traytel

February 6, 2026

## Abstract

This entry formalizes the closure property of bounded natural functors (BNFs) under seven operations. These operations and the corresponding proofs constitute the core of Isabelle’s (co)datatype package. To be close to the implemented tactics, the proofs are deliberately formulated as detailed apply scripts. The (co)datatypes together with (co)induction principles and (co)recursors are byproducts of the fixpoint operations LFP and GFP. Composition of BNFs is subdivided into four simpler operations: Compose, Kill, Lift, and Permute. The N2M operation provides mutual (co)induction principles and (co)recursors for nested (co)datatypes.

## Contents

<b>1</b>	<b>Least Fixpoint (a.k.a. Datatype)</b>	<b>1</b>
1.1	Algebra	2
1.2	Morphism	2
1.3	Bounds	2
1.4	Minimal Algebras	3
1.5	Initiality	5
1.6	Initial Algebras	5
1.7	The datatype	6
1.8	The Result as an BNF	8
<b>2</b>	<b>Greatest Fixpoint (a.k.a. Codatatype)</b>	<b>11</b>
2.1	Coalgebra	12
2.2	Type-coalgebra	13
2.3	Morphism	13
2.4	Bisimulations	14
2.5	The Tree Coalgebra	15
2.6	Quotient Coalgebra	19
2.7	Coinduction	22
2.8	The Result as an BNF	22
<b>3</b>	<b>Normalized Composition of BNFs</b>	<b>26</b>
<b>4</b>	<b>Removing Live Variables</b>	<b>28</b>
<b>5</b>	<b>Adding New Live Variables</b>	<b>30</b>
<b>6</b>	<b>Changing the Order of Live Variables</b>	<b>32</b>
<b>7</b>	<b>Mutual View on Nested Datatypes</b>	<b>33</b>
7.1	Nested Definition	33
7.2	Isomorphic Mutual Definition	33
7.3	Mutualization	34
7.3.1	Iterators	34
7.3.2	Recursors	34
7.3.3	Induction	35

<b>8</b>	<b>Mutual View on Nested Coatypes</b>	<b>35</b>
8.1	Nested definition . . . . .	35
8.2	Isomorphic Mutual Definition . . . . .	35
8.3	Mutualization . . . . .	36
8.3.1	Coiterators . . . . .	36
8.3.2	Corecursors . . . . .	36
8.3.3	Coinduction . . . . .	37

## 1 Least Fixpoint (a.k.a. Datatype)

**unbundle** *cardinal\_syntax*

$\langle ML \rangle$

**notation** *BNF\_Def.convolver* ( $\langle \_ , \_ \rangle$ )

'b1 = ('a, 'b1, 'b2) F1

'b2 = ('a, 'b1, 'b2) F2

To build a witness scenario, let us assume

('a, 'b1, 'b2) F1 = 'a \* 'b1 + 'a \* 'b2

('a, 'b1, 'b2) F2 = unit + 'b1 \* 'b2

**declare** *[[bnf\_internals]]*

**bnf-axiomatization** (*F1set1: 'a, F1set2: 'b1, F1set3: 'b2*) *F1*

[*wits: 'a  $\Rightarrow$  'b1  $\Rightarrow$  ('a, 'b1, 'b2) F1 'a  $\Rightarrow$  'b2  $\Rightarrow$  ('a, 'b1, 'b2) F1*]

**for** *map: F1map rel: F1rel*

**bnf-axiomatization** (*F2set1: 'a, F2set2: 'b1, F2set3: 'b2*) *F2*

[*wits: ('a, 'b1, 'b2) F2*]

**for** *map: F2map rel: F2rel*

**abbreviation** *F1in :: 'a1 set  $\Rightarrow$  'a2 set  $\Rightarrow$  'a3 set  $\Rightarrow$  (('a1, 'a2, 'a3) F1) set* **where**

*F1in A1 A2 A3  $\equiv$  {x. F1set1 x  $\subseteq$  A1  $\wedge$  F1set2 x  $\subseteq$  A2  $\wedge$  F1set3 x  $\subseteq$  A3}*

**abbreviation** *F2in :: 'a1 set  $\Rightarrow$  'a2 set  $\Rightarrow$  'a3 set  $\Rightarrow$  (('a1, 'a2, 'a3) F2) set* **where**

*F2in A1 A2 A3  $\equiv$  {x. F2set1 x  $\subseteq$  A1  $\wedge$  F2set2 x  $\subseteq$  A2  $\wedge$  F2set3 x  $\subseteq$  A3}*

**lemma** *F1map\_comp\_id: F1map g1 g2 g3 (F1map id f2 f3 x) = F1map g1 (g2 o f2) (g3 o f3) x*

$\langle proof \rangle$

**lemmas** *F1in\_mono23 = F1.in\_mono[OF subset\_refl]*

**lemma** *F1map\_congL:  $\llbracket \forall a \in F1set2 x. f a = a; \forall a \in F1set3 x. g a = a \rrbracket \Longrightarrow$*

*F1map id f g x = x*

$\langle proof \rangle$

**lemma** *F2map\_comp\_id: F2map g1 g2 g3 (F2map id f2 f3 x) = F2map g1 (g2 o f2) (g3 o f3) x*

$\langle proof \rangle$

**lemmas** *F2in\_mono23 = F2.in\_mono[OF subset\_refl]*

**lemma** *F2map\_congL:  $\llbracket \forall a \in F2set2 x. f a = a; \forall a \in F2set3 x. g a = a \rrbracket \Longrightarrow$*

*F2map id f g x = x*

$\langle proof \rangle$

### 1.1 Algebra

**definition** *alg where*

*alg B1 B2 s1 s2 =*

$((\forall x \in F1in (UNIV :: 'a set) B1 B2. s1 x \in B1) \wedge (\forall y \in F2in (UNIV :: 'a set) B1 B2. s2 y \in B2))$

**lemma** *alg\_F1set:  $\llbracket alg B1 B2 s1 s2; F1set2 x \subseteq B1; F1set3 x \subseteq B2 \rrbracket \Longrightarrow s1 x \in B1$*

$\langle proof \rangle$

**lemma** *alg\_F2set*:  $\llbracket \text{alg } B1 \ B2 \ s1 \ s2; \text{F2set2 } x \subseteq B1; \text{F2set3 } x \subseteq B2 \rrbracket \implies s2 \ x \in B2$   
 ⟨proof⟩

**lemma** *alg\_not\_empty*:  
 $\text{alg } B1 \ B2 \ s1 \ s2 \implies B1 \neq \{\} \wedge B2 \neq \{\}$   
 ⟨proof⟩

## 1.2 Morphism

**definition** *mor where*

$\text{mor } B1 \ B2 \ s1 \ s2 \ B1' \ B2' \ s1' \ s2' \ f \ g =$   
 $((\forall a \in B1. f \ a \in B1') \wedge (\forall a \in B2. g \ a \in B2')) \wedge$   
 $((\forall z \in \text{F1in } (\text{UNIV} :: 'a \ \text{set}) \ B1 \ B2. f \ (s1 \ z) = s1' \ (F1map \ \text{id} \ f \ g \ z)) \wedge$   
 $(\forall z \in \text{F2in } (\text{UNIV} :: 'a \ \text{set}) \ B1 \ B2. g \ (s2 \ z) = s2' \ (F2map \ \text{id} \ f \ g \ z))))$

**lemma** *morE1*:  $\llbracket \text{mor } B1 \ B2 \ s1 \ s2 \ B1' \ B2' \ s1' \ s2' \ f \ g; \ z \in \text{F1in } \text{UNIV } B1 \ B2 \rrbracket$   
 $\implies f \ (s1 \ z) = s1' \ (F1map \ \text{id} \ f \ g \ z)$   
 ⟨proof⟩

**lemma** *morE2*:  $\llbracket \text{mor } B1 \ B2 \ s1 \ s2 \ B1' \ B2' \ s1' \ s2' \ f \ g; \ z \in \text{F2in } \text{UNIV } B1 \ B2 \rrbracket$   
 $\implies g \ (s2 \ z) = s2' \ (F2map \ \text{id} \ f \ g \ z)$   
 ⟨proof⟩

**lemma** *mor\_incl*:  $\llbracket B1 \subseteq B1'; \ B2 \subseteq B2' \rrbracket \implies \text{mor } B1 \ B2 \ s1 \ s2 \ B1' \ B2' \ s1 \ s2 \ \text{id} \ \text{id}$   
 ⟨proof⟩

**lemma** *mor\_comp*:  
 $\llbracket \text{mor } B1 \ B2 \ s1 \ s2 \ B1' \ B2' \ s1' \ s2' \ f \ g;$   
 $\text{mor } B1' \ B2' \ s1' \ s2' \ B1'' \ B2'' \ s1'' \ s2'' \ f' \ g' \rrbracket \implies$   
 $\text{mor } B1 \ B2 \ s1 \ s2 \ B1'' \ B2'' \ s1'' \ s2'' \ (f' \ o \ f) \ (g' \ o \ g)$   
 ⟨proof⟩

**lemma** *mor\_cong*:  $\llbracket f' = f; \ g' = g; \ \text{mor } B1 \ B2 \ s1 \ s2 \ B1' \ B2' \ s1' \ s2' \ f \ g \rrbracket \implies$   
 $\text{mor } B1 \ B2 \ s1 \ s2 \ B1' \ B2' \ s1' \ s2' \ f' \ g'$   
 ⟨proof⟩

**lemma** *mor\_str*:  
 $\text{mor } \text{UNIV } \text{UNIV} \ (F1map \ \text{id} \ s1 \ s2) \ (F2map \ \text{id} \ s1 \ s2) \ \text{UNIV } \text{UNIV} \ s1 \ s2 \ s1 \ s2$   
 ⟨proof⟩

## 1.3 Bounds

**type-synonym**  $\text{bd\_type\_F1}' = \text{bd\_type\_F1} + (\text{bd\_type\_F1}, \text{bd\_type\_F1}, \text{bd\_type\_F1}) \ \text{F1}$

**type-synonym**  $\text{bd\_type\_F2}' = \text{bd\_type\_F2} + (\text{bd\_type\_F2}, \text{bd\_type\_F2}, \text{bd\_type\_F2}) \ \text{F2}$

**type-synonym**  $\text{SucFbd\_type} = ((\text{bd\_type\_F1}' + \text{bd\_type\_F2}') \ \text{set})$

**type-synonym**  $'a1 \ \text{ASucFbd\_type} = (\text{SucFbd\_type} \Rightarrow ('a1 + \text{bool}))$

**abbreviation**  $\text{F1bd}' \equiv \text{bd\_F1} + c \ | \ \text{UNIV} :: (\text{bd\_type\_F1}, \text{bd\_type\_F1}, \text{bd\_type\_F1}) \ \text{F1} \ \text{set}$

**lemma** *F1set1\_bd\_incr*:  $\bigwedge x. |\text{F1set1 } x| < o \ \text{F1bd}'$   
 ⟨proof⟩

**lemma** *F1set2\_bd\_incr*:  $\bigwedge x. |\text{F1set2 } x| < o \ \text{F1bd}'$   
 ⟨proof⟩

**lemma** *F1set3\_bd\_incr*:  $\bigwedge x. |\text{F1set3 } x| < o \ \text{F1bd}'$   
 ⟨proof⟩

**lemmas** *F1bd'\_Card\_order* = *Card\_order\_csum*

**lemmas** *F1bd'\_Cinfinite* = *Cinfinite\_csum1*[OF *F1.bd\_Cinfinite*]

**lemmas** *F1bd'\_Cnotzero* = *Cinfinite\_Cnotzero*[OF *F1bd'\_Cinfinite*]

**lemmas** *F1bd'\_card\_order* = *card\_order\_csum*[OF *F1.bd\_card\_order card\_of\_card\_order\_on*]

**abbreviation**  $\text{F2bd}' \equiv \text{bd\_F2} + c \ | \ \text{UNIV} :: (\text{bd\_type\_F2}, \text{bd\_type\_F2}, \text{bd\_type\_F2}) \ \text{F2} \ \text{set}$

**lemma** *F2set1\_bd\_incr*:  $\bigwedge x. |\text{F2set1 } x| < o \ \text{F2bd}'$   
 ⟨proof⟩

**lemma**  $F2set2\_bd\_incr$ :  $\bigwedge x. |F2set2\ x| <_o F2bd'$

*<proof>*

**lemma**  $F2set3\_bd\_incr$ :  $\bigwedge x. |F2set3\ x| <_o F2bd'$

*<proof>*

**lemmas**  $F2bd'\_Card\_order = Card\_order\_csum$

**lemmas**  $F2bd'\_Cinfinite = Cinfinite\_csum1[OF\ F2.bd\_Cinfinite]$

**lemmas**  $F2bd'\_Cnotzero = Cinfinite\_Cnotzero[OF\ F2bd'\_Cinfinite]$

**lemmas**  $F2bd'\_card\_order = card\_order\_csum[OF\ F2.bd\_card\_order\ card\_of\_card\_order\_on]$

**abbreviation**  $SucFbd$  **where**  $SucFbd \equiv cardSuc\ (F1bd' +_c\ F2bd')$

**abbreviation**  $ASucFbd$  **where**  $ASucFbd \equiv (|UNIV| +_c\ ctwo) \hat{c}\ SucFbd$

**lemma**  $F1set1\_bd$ :  $|F1set1\ x| <_o\ bd\_F1 +_c\ bd\_F2$

*<proof>*

**lemma**  $F1set2\_bd$ :  $|F1set2\ x| <_o\ bd\_F1 +_c\ bd\_F2$

*<proof>*

**lemma**  $F1set3\_bd$ :  $|F1set3\ x| <_o\ bd\_F1 +_c\ bd\_F2$

*<proof>*

**lemma**  $F2set1\_bd$ :  $|F2set1\ x| <_o\ bd\_F1 +_c\ bd\_F2$

*<proof>*

**lemma**  $F2set2\_bd$ :  $|F2set2\ x| <_o\ bd\_F1 +_c\ bd\_F2$

*<proof>*

**lemma**  $F2set3\_bd$ :  $|F2set3\ x| <_o\ bd\_F1 +_c\ bd\_F2$

*<proof>*

**lemmas**  $SucFbd\_Card\_order = cardSuc\_Card\_order[OF\ Card\_order\_csum]$

**lemmas**  $SucFbd\_Cinfinite = Cinfinite\_cardSuc[OF\ Cinfinite\_csum1[OF\ F1bd'\_Cinfinite]]$

**lemmas**  $SucFbd\_Cnotzero = Cinfinite\_Cnotzero[OF\ SucFbd\_Cinfinite]$

**lemmas**  $worel\_SucFbd = Card\_order\_wo\_rel[OF\ SucFbd\_Card\_order]$

**lemmas**  $ASucFbd\_Cinfinite = Cinfinite\_cexp[OF\ ordLeq\_csum2[OF\ Card\_order\_ctwo]\ SucFbd\_Cinfinite]$

## 1.4 Minimal Algebras

**abbreviation**  $min\_G1$  **where**

$min\_G1\ As1\_As2\ i \equiv (\bigcup j \in underS\ SucFbd\ i.\ fst\ (As1\_As2\ j))$

**abbreviation**  $min\_G2$  **where**

$min\_G2\ As1\_As2\ i \equiv (\bigcup j \in underS\ SucFbd\ i.\ snd\ (As1\_As2\ j))$

**abbreviation**  $min\_H$  **where**

$min\_H\ s1\ s2\ As1\_As2\ i \equiv$

$(min\_G1\ As1\_As2\ i \cup s1 \text{ ' } (F1in\ (UNIV :: 'a\ set)\ (min\_G1\ As1\_As2\ i)\ (min\_G2\ As1\_As2\ i)),$   
 $min\_G2\ As1\_As2\ i \cup s2 \text{ ' } (F2in\ (UNIV :: 'a\ set)\ (min\_G1\ As1\_As2\ i)\ (min\_G2\ As1\_As2\ i)))$

**abbreviation**  $min\_algs$  **where**

$min\_algs\ s1\ s2 \equiv wo\_rel.worec\ SucFbd\ (min\_H\ s1\ s2)$

**definition**  $min\_alg1$  **where**

$min\_alg1\ s1\ s2 = (\bigcup i \in Field\ SucFbd.\ fst\ (min\_algs\ s1\ s2\ i))$

**definition**  $min\_alg2$  **where**

$min\_alg2\ s1\ s2 = (\bigcup i \in Field\ SucFbd.\ snd\ (min\_algs\ s1\ s2\ i))$

**lemma**  $min\_algs$ :

$i \in Field\ SucFbd \implies min\_algs\ s1\ s2\ i = min\_H\ s1\ s2\ (min\_algs\ s1\ s2)\ i$

*<proof>*

**corollary**  $min\_algs1$ :  $i \in Field\ SucFbd \implies fst\ (min\_algs\ s1\ s2\ i) =$

$\text{min\_G1 } (\text{min\_algs } s1 \ s2) \ i \cup$   
 $s1 \ ' (F1 \text{in UNIV } (\text{min\_G1 } (\text{min\_algs } s1 \ s2) \ i) \ (\text{min\_G2 } (\text{min\_algs } s1 \ s2) \ i))$   
 $\langle \text{proof} \rangle$

**corollary**  $\text{min\_algs2}: i \in \text{Field SucFbd} \implies \text{snd } (\text{min\_algs } s1 \ s2 \ i) =$   
 $\text{min\_G2 } (\text{min\_algs } s1 \ s2) \ i \cup$   
 $s2 \ ' (F2 \text{in UNIV } (\text{min\_G1 } (\text{min\_algs } s1 \ s2) \ i) \ (\text{min\_G2 } (\text{min\_algs } s1 \ s2) \ i))$   
 $\langle \text{proof} \rangle$

**lemma**  $\text{min\_algs\_mono1}: \text{relChain SucFbd } (\%i. \text{fst } (\text{min\_algs } s1 \ s2 \ i))$   
 $\langle \text{proof} \rangle$

**lemma**  $\text{min\_algs\_mono2}: \text{relChain SucFbd } (\%i. \text{snd } (\text{min\_algs } s1 \ s2 \ i))$   
 $\langle \text{proof} \rangle$

**lemma**  $\text{SucFbd\_limit}: \llbracket x1 \in \text{Field SucFbd} \ \& \ x2 \in \text{Field SucFbd} \rrbracket$   
 $\implies \exists y \in \text{Field SucFbd}. (x1 \neq y \wedge (x1, y) \in \text{SucFbd}) \wedge (x2 \neq y \wedge (x2, y) \in \text{SucFbd})$   
 $\langle \text{proof} \rangle$

**lemma**  $\text{alg\_min\_alg}: \text{alg } (\text{min\_alg1 } s1 \ s2) \ (\text{min\_alg2 } s1 \ s2) \ s1 \ s2$   
 $\langle \text{proof} \rangle$

**lemmas**  $\text{SucFbd\_ASucFbd} = \text{ordLess\_ordLeq\_trans}[OF$   
 $\text{ordLess\_ctwo\_cexp}$   
 $\text{cexp\_mono1}[OF \text{ordLeq\_csum2}[OF \text{Card\_order\_ctwo}]],$   
 $OF \text{SucFbd\_Card\_order SucFbd\_Card\_order}]$

**lemma**  $\text{card\_of\_min\_algs}:$   
**fixes**  $s1 :: ('a, 'b, 'c) F1 \Rightarrow 'b$  **and**  $s2 :: ('a, 'b, 'c) F2 \Rightarrow 'c$   
**shows**  $i \in \text{Field SucFbd} \longrightarrow$   
 $(|\text{fst } (\text{min\_algs } s1 \ s2 \ i)| \leq_o (\text{ASucFbd} :: 'a \ \text{ASucFbd\_type rel}) \wedge |\text{snd } (\text{min\_algs } s1 \ s2 \ i)| \leq_o (\text{ASucFbd} :: 'a \ \text{ASucFbd\_type rel}))$   
 $\langle \text{proof} \rangle$

**lemma**  $\text{card\_of\_min\_alg1}:$   
**fixes**  $s1 :: ('a, 'b, 'c) F1 \Rightarrow 'b$  **and**  $s2 :: ('a, 'b, 'c) F2 \Rightarrow 'c$   
**shows**  $|\text{min\_alg1 } s1 \ s2| \leq_o (\text{ASucFbd} :: 'a \ \text{ASucFbd\_type rel})$   
 $\langle \text{proof} \rangle$

**lemma**  $\text{card\_of\_min\_alg2}:$   
**fixes**  $s1 :: ('a, 'b, 'c) F1 \Rightarrow 'b$  **and**  $s2 :: ('a, 'b, 'c) F2 \Rightarrow 'c$   
**shows**  $|\text{min\_alg2 } s1 \ s2| \leq_o (\text{ASucFbd} :: 'a \ \text{ASucFbd\_type rel})$   
 $\langle \text{proof} \rangle$

**lemma**  $\text{least\_min\_algs}: \text{alg } B1 \ B2 \ s1 \ s2 \implies$   
 $i \in \text{Field SucFbd} \longrightarrow$   
 $\text{fst } (\text{min\_algs } s1 \ s2 \ i) \subseteq B1 \wedge \text{snd } (\text{min\_algs } s1 \ s2 \ i) \subseteq B2$   
 $\langle \text{proof} \rangle$

**lemma**  $\text{least\_min\_alg1}: \text{alg } B1 \ B2 \ s1 \ s2 \implies \text{min\_alg1 } s1 \ s2 \subseteq B1$   
 $\langle \text{proof} \rangle$

**lemma**  $\text{least\_min\_alg2}: \text{alg } B1 \ B2 \ s1 \ s2 \implies \text{min\_alg2 } s1 \ s2 \subseteq B2$   
 $\langle \text{proof} \rangle$

**lemma**  $\text{mor\_incl\_min\_alg}:$   
 $\text{alg } B1 \ B2 \ s1 \ s2 \implies$   
 $\text{mor } (\text{min\_alg1 } s1 \ s2) \ (\text{min\_alg2 } s1 \ s2) \ s1 \ s2 \ B1 \ B2 \ s1 \ s2 \ \text{id id}$   
 $\langle \text{proof} \rangle$

## 1.5 Initiality

The following "happens" to be the type (for our particular construction) of the initial algebra carrier:

**type-synonym** 'a1 F1init\_type = ('a1, 'a1 ASucFbd\_type, 'a1 ASucFbd\_type) F1  $\Rightarrow$  'a1 ASucFbd\_type  
**type-synonym** 'a1 F2init\_type = ('a1, 'a1 ASucFbd\_type, 'a1 ASucFbd\_type) F2  $\Rightarrow$  'a1 ASucFbd\_type

**typedef** 'a1 IIT =  
 UNIV ::  
 (('a1 ASucFbd\_type set  $\times$  'a1 ASucFbd\_type set)  $\times$  ('a1 F1init\_type  $\times$  'a1 F2init\_type)) set  
 <proof>

## 1.6 Initial Algebras

**abbreviation** II :: 'a1 IIT set where  
 II  $\equiv$  {Abs\_IIT ((B1, B2), (s1, s2)) | B1 B2 s1 s2. alg B1 B2 s1 s2}

**definition** str\_init1 where  
 str\_init1 (dummy :: 'a1)  
 (y::('a1, 'a1 IIT  $\Rightarrow$  'a1 ASucFbd\_type, 'a1 IIT  $\Rightarrow$  'a1 ASucFbd\_type) F1)  
 (i :: 'a1 IIT) =  
 fst (snd (Rep\_IIT i))  
 (F1map id ( $\lambda$ f :: 'a1 IIT  $\Rightarrow$  'a1 ASucFbd\_type. f i) ( $\lambda$ f. f i) y)

**definition** str\_init2 where  
 str\_init2 (dummy :: 'a1) y (i :: 'a1 IIT) =  
 snd (snd (Rep\_IIT i)) (F2map id ( $\lambda$ f. f i) ( $\lambda$ f. f i) y)

**abbreviation** car\_init1 where  
 car\_init1 dummy  $\equiv$  min\_alg1 (str\_init1 dummy) (str\_init2 dummy)

**abbreviation** car\_init2 where  
 car\_init2 dummy  $\equiv$  min\_alg2 (str\_init1 dummy) (str\_init2 dummy)

**lemma** alg\_select:  
 $\forall i \in II. \text{alg} (\text{fst} (\text{fst} (\text{Rep\_IIT } i))) (\text{snd} (\text{fst} (\text{Rep\_IIT } i)))$   
 $(\text{fst} (\text{snd} (\text{Rep\_IIT } i))) (\text{snd} (\text{snd} (\text{Rep\_IIT } i)))$   
 <proof>

**lemma** mor\_select:  
 $\llbracket i \in II;$   
 $\text{mor} (\text{fst} (\text{fst} (\text{Rep\_IIT } i))) (\text{snd} (\text{fst} (\text{Rep\_IIT } i)))$   
 $(\text{fst} (\text{snd} (\text{Rep\_IIT } i))) (\text{snd} (\text{snd} (\text{Rep\_IIT } i))) \text{UNIV UNIV } s1' s2' f g \rrbracket \Longrightarrow$   
 $\text{mor} (\text{car\_init1 } \text{dummy}) (\text{car\_init2 } \text{dummy}) (\text{str\_init1 } \text{dummy}) (\text{str\_init2 } \text{dummy}) \text{UNIV UNIV } s1' s2' (f \circ (\lambda h.$   
 $h \ i)) (g \circ (\lambda h. h \ i))$   
 <proof>

**lemma** init\_unique\_mor:  
 $\llbracket a1 \in \text{car\_init1 } \text{dummy}; a2 \in \text{car\_init2 } \text{dummy};$   
 $\text{mor} (\text{car\_init1 } \text{dummy}) (\text{car\_init2 } \text{dummy}) (\text{str\_init1 } \text{dummy}) (\text{str\_init2 } \text{dummy}) B1 B2 s1 s2 f1 f2;$   
 $\text{mor} (\text{car\_init1 } \text{dummy}) (\text{car\_init2 } \text{dummy}) (\text{str\_init1 } \text{dummy}) (\text{str\_init2 } \text{dummy}) B1 B2 s1 s2 g1 g2 \rrbracket \Longrightarrow$   
 $f1 \ a1 = g1 \ a1 \wedge f2 \ a2 = g2 \ a2$   
 <proof>

**abbreviation** closed where  
 closed dummy phi1 phi2  $\equiv$  (( $\forall x \in F1in \ \text{UNIV} (\text{car\_init1 } \text{dummy}) (\text{car\_init2 } \text{dummy}).$   
 ( $\forall z \in F1set2 \ x. \ \text{phi1 } z$ )  $\wedge$  ( $\forall z \in F1set3 \ x. \ \text{phi2 } z$ )  $\longrightarrow$  phi1 (str\_init1 dummy x))  $\wedge$   
 ( $\forall x \in F2in \ \text{UNIV} (\text{car\_init1 } \text{dummy}) (\text{car\_init2 } \text{dummy}).$   
 ( $\forall z \in F2set2 \ x. \ \text{phi1 } z$ )  $\wedge$  ( $\forall z \in F2set3 \ x. \ \text{phi2 } z$ )  $\longrightarrow$  phi2 (str\_init2 dummy x)))

**lemma** init\_induct: closed dummy phi1 phi2  $\Longrightarrow$   
 ( $\forall x \in \text{car\_init1 } \text{dummy}. \ \text{phi1 } x$ )  $\wedge$  ( $\forall x \in \text{car\_init2 } \text{dummy}. \ \text{phi2 } x$ )  
 <proof>

## 1.7 The datatype

**typedef** (overloaded) 'a1 IF1 = car\_init1 (undefined :: 'a1)  
 <proof>

**typedef** (overloaded) 'a1 IF2 = car\_init2 (undefined :: 'a1)  
 <proof>

**definition** *ctor1* **where** *ctor1* = *Abs\_IF1* o *str\_init1* undefined o *F1map id* *Rep\_IF1* *Rep\_IF2*

**definition** *ctor2* **where** *ctor2* = *Abs\_IF2* o *str\_init2* undefined o *F2map id* *Rep\_IF1* *Rep\_IF2*

**lemma** *mor\_Rep\_IF*:

*mor* (*UNIV* :: 'a *IF1* set) (*UNIV* :: 'a *IF2* set) *ctor1* *ctor2*  
(*car\_init1* undefined) (*car\_init2* undefined) (*str\_init1* undefined) (*str\_init2* undefined) *Rep\_IF1* *Rep\_IF2*  
<proof>

**lemma** *mor\_Abs\_IF*:

*mor* (*car\_init1* undefined) (*car\_init2* undefined)  
(*str\_init1* undefined) (*str\_init2* undefined) *UNIV* *UNIV* *ctor1* *ctor2* *Abs\_IF1* *Abs\_IF2*  
<proof>

**lemma** *copy*:

$\llbracket \text{alg } B1 \ B2 \ s1 \ s2; \text{bij\_betw } f \ B1' \ B1; \text{bij\_betw } g \ B2' \ B2 \rrbracket \implies$   
 $\exists f' \ g'. \text{alg } B1' \ B2' \ f' \ g' \wedge \text{mor } B1' \ B2' \ f' \ g' \ B1 \ B2 \ s1 \ s2 \ f \ g$   
<proof>

**lemma** *init\_ex\_mor*:

$\exists f \ g. \text{mor } UNIV \ UNIV \ \text{ctor1} \ \text{ctor2} \ UNIV \ UNIV \ s1 \ s2 \ f \ g$   
<proof>

Iteration

**abbreviation** *fold* **where**

*fold* *s1* *s2*  $\equiv$  (*SOME* *f*. *mor* *UNIV* *UNIV* *ctor1* *ctor2* *UNIV* *UNIV* *s1* *s2* (*fst* *f*) (*snd* *f*))

**definition** *fold1* **where** *fold1* *s1* *s2* = *fst* (*fold* *s1* *s2*)

**definition** *fold2* **where** *fold2* *s1* *s2* = *snd* (*fold* *s1* *s2*)

**lemma** *mor\_fold*:

*mor* *UNIV* *UNIV* *ctor1* *ctor2* *UNIV* *UNIV* *s1* *s2* (*fold1* *s1* *s2*) (*fold2* *s1* *s2*)  
<proof>

<ML>

**theorem** *fold1*:

(*fold1* *s1* *s2*) (*ctor1* *x*) = *s1* (*F1map id* (*fold1* *s1* *s2*) (*fold2* *s1* *s2*) *x*)  
<proof>

**theorem** *fold2*:

(*fold2* *s1* *s2*) (*ctor2* *x*) = *s2* (*F2map id* (*fold1* *s1* *s2*) (*fold2* *s1* *s2*) *x*)  
<proof>

**lemma** *mor\_UNIV*: *mor* *UNIV* *UNIV* *s1* *s2* *UNIV* *UNIV* *s1'* *s2'* *f* *g*  $\longleftrightarrow$

*f* o *s1* = *s1'* o *F1map id* *f* *g*  $\wedge$  *g* o *s2* = *s2'* o *F2map id* *f* *g*  
<proof>

**lemma** *fold\_unique\_mor*: *mor* *UNIV* *UNIV* *ctor1* *ctor2* *UNIV* *UNIV* *s1* *s2* *f* *g*  $\implies$

*f* = *fold1* *s1* *s2*  $\wedge$  *g* = *fold2* *s1* *s2*  
<proof>

**lemmas** *fold\_unique* = *fold\_unique\_mor*[*OF iffD2*[*OF mor\_UNIV*], *OF conjI*]

**lemmas** *fold1\_ctor* = *sym*[*OF conjunct1*[*OF fold\_unique\_mor*[*OF mor\_incl*[*OF subset\_UNIV subset\_UNIV*]]]]

**lemmas** *fold2\_ctor* = *sym*[*OF conjunct2*[*OF fold\_unique\_mor*[*OF mor\_incl*[*OF subset\_UNIV subset\_UNIV*]]]]

Case distinction

**lemmas** *ctor1\_o\_fold1* =

*trans*[*OF conjunct1*[*OF fold\_unique\_mor*[*OF mor\_comp*[*OF mor\_fold mor\_str*]]]] *fold1\_ctor*

**lemmas** *ctor2\_o\_fold2* =

*trans*[*OF conjunct2*[*OF fold\_unique\_mor*[*OF mor\_comp*[*OF mor\_fold mor\_str*]]]] *fold2\_ctor*

**definition**  $dtor1 = fold1 (F1map\ id\ ctor1\ ctor2) (F2map\ id\ ctor1\ ctor2)$

**definition**  $dtor2 = fold2 (F1map\ id\ ctor1\ ctor2) (F2map\ id\ ctor1\ ctor2)$

$\langle ML \rangle$

**lemma**  $ctor1\_o\_dtor1: ctor1\ o\ dtor1 = id$   
 $\langle proof \rangle$

**lemma**  $ctor2\_o\_dtor2: ctor2\ o\ dtor2 = id$   
 $\langle proof \rangle$

**lemma**  $dtor1\_o\_ctor1: dtor1\ o\ ctor1 = id$   
 $\langle proof \rangle$

**lemma**  $dtor2\_o\_ctor2: dtor2\ o\ ctor2 = id$   
 $\langle proof \rangle$

**lemmas**  $dtor1\_ctor1 = pointfree\_idE[OF\ dtor1\_o\_ctor1]$

**lemmas**  $dtor2\_ctor2 = pointfree\_idE[OF\ dtor2\_o\_ctor2]$

**lemmas**  $ctor1\_dtor1 = pointfree\_idE[OF\ ctor1\_o\_dtor1]$

**lemmas**  $ctor2\_dtor2 = pointfree\_idE[OF\ ctor2\_o\_dtor2]$

**lemmas**  $bij\_dtor1 = o\_bij[OF\ ctor1\_o\_dtor1\ dtor1\_o\_ctor1]$

**lemmas**  $inj\_dtor1 = bij\_is\_inj[OF\ bij\_dtor1]$

**lemmas**  $surj\_dtor1 = bij\_is\_surj[OF\ bij\_dtor1]$

**lemmas**  $dtor1\_nchotomy = surjD[OF\ surj\_dtor1]$

**lemmas**  $dtor1\_diff = inj\_eq[OF\ inj\_dtor1]$

**lemmas**  $dtor1\_cases = exE[OF\ dtor1\_nchotomy]$

**lemmas**  $bij\_dtor2 = o\_bij[OF\ ctor2\_o\_dtor2\ dtor2\_o\_ctor2]$

**lemmas**  $inj\_dtor2 = bij\_is\_inj[OF\ bij\_dtor2]$

**lemmas**  $surj\_dtor2 = bij\_is\_surj[OF\ bij\_dtor2]$

**lemmas**  $dtor2\_nchotomy = surjD[OF\ surj\_dtor2]$

**lemmas**  $dtor2\_diff = inj\_eq[OF\ inj\_dtor2]$

**lemmas**  $dtor2\_cases = exE[OF\ dtor2\_nchotomy]$

**lemmas**  $bij\_ctor1 = o\_bij[OF\ dtor1\_o\_ctor1\ ctor1\_o\_dtor1]$

**lemmas**  $inj\_ctor1 = bij\_is\_inj[OF\ bij\_ctor1]$

**lemmas**  $surj\_ctor1 = bij\_is\_surj[OF\ bij\_ctor1]$

**lemmas**  $ctor1\_nchotomy = surjD[OF\ surj\_ctor1]$

**lemmas**  $ctor1\_diff = inj\_eq[OF\ inj\_ctor1]$

**lemmas**  $ctor1\_cases = exE[OF\ ctor1\_nchotomy]$

**lemmas**  $bij\_ctor2 = o\_bij[OF\ dtor2\_o\_ctor2\ ctor2\_o\_dtor2]$

**lemmas**  $inj\_ctor2 = bij\_is\_inj[OF\ bij\_ctor2]$

**lemmas**  $surj\_ctor2 = bij\_is\_surj[OF\ bij\_ctor2]$

**lemmas**  $ctor2\_nchotomy = surjD[OF\ surj\_ctor2]$

**lemmas**  $ctor2\_diff = inj\_eq[OF\ inj\_ctor2]$

**lemmas**  $ctor2\_cases = exE[OF\ ctor2\_nchotomy]$

Primitive recursion

**definition**  $rec1$  **where**

$rec1\ s1\ s2 = snd\ o\ fold1\ (<ctor1\ o\ F1map\ id\ fst\ fst,\ s1>) (<ctor2\ o\ F2map\ id\ fst\ fst,\ s2>)$

**definition**  $rec2$  **where**

$rec2\ s1\ s2 = snd\ o\ fold2\ (<ctor1\ o\ F1map\ id\ fst\ fst,\ s1>) (<ctor2\ o\ F2map\ id\ fst\ fst,\ s2>)$

**lemma**  $fold1\_o\_ctor1: fold1\ s1\ s2\ o\ ctor1 = s1\ o\ F1map\ id\ (fold1\ s1\ s2)\ (fold2\ s1\ s2)$

$\langle proof \rangle$

**lemma**  $fold2\_o\_ctor2: fold2\ s1\ s2\ o\ ctor2 = s2\ o\ F2map\ id\ (fold1\ s1\ s2)\ (fold2\ s1\ s2)$

$\langle proof \rangle$

**lemmas**  $fst\_rec1\_pair =$

$trans[OF\ conjunct1[OF\ fold\_unique[OF$   
 $trans[OF\ o\_assoc[symmetric]\ trans[OF\ arg\_cong2[of\ \_\ \_\ \_\ \_\ (o),\ OF\ refl$   
 $trans[OF\ fold1\_o\_ctor1\ convol\_o]],\ OF\ trans[OF\ fst\_convol]]$   
 $trans[OF\ o\_assoc[symmetric]\ trans[OF\ arg\_cong2[of\ \_\ \_\ \_\ \_\ (o),\ OF\ refl$



**theorem** *IF2map\_id*:  $IF2map\ id = id$   
 ⟨proof⟩

**theorem** *IF1map\_comp*:  $IF1map\ (g\ o\ f) = IF1map\ g\ o\ IF1map\ f$   
 ⟨proof⟩

**theorem** *IF2map\_comp*:  $IF2map\ (g\ o\ f) = IF2map\ g\ o\ IF2map\ f$   
 ⟨proof⟩

The bound

**abbreviation** *IFbd* **where**  $IFbd \equiv bd\_F1 + c\ bd\_F2$

**theorem** *IFbd\_card\_order*:  $card\_order\ IFbd$   
 ⟨proof⟩

**lemma** *IFbd\_Cinfinite*:  $Cinfinite\ IFbd$   
 ⟨proof⟩

**lemma** *IFbd\_regularCard*:  $regularCard\ IFbd$   
 ⟨proof⟩

**lemmas** *IFbd\_cinfinite* =  $conjunct1[OF\ IFbd\_Cinfinite]$

The set operator

**abbreviation** *IF1col* **where**  $IF1col \equiv (\lambda X. F1set1\ X \cup (\bigcup (F1set2\ X) \cup \bigcup (F1set3\ X)))$

**abbreviation** *IF2col* **where**  $IF2col \equiv (\lambda X. F2set1\ X \cup (\bigcup (F2set2\ X) \cup \bigcup (F2set3\ X)))$

**abbreviation** *IF1set* **where**  $IF1set \equiv fold1\ IF1col\ IF2col$

**abbreviation** *IF2set* **where**  $IF2set \equiv fold2\ IF1col\ IF2col$

**abbreviation** *IF1in* **where**  $IF1in\ A \equiv \{x. IF1set\ x \subseteq A\}$

**abbreviation** *IF2in* **where**  $IF2in\ A \equiv \{x. IF2set\ x \subseteq A\}$

**lemma** *IF1set*:  $IF1set\ o\ ctor1 = IF1col\ o\ (F1map\ id\ IF1set\ IF2set)$   
 ⟨proof⟩

**lemma** *IF2set*:  $IF2set\ o\ ctor2 = IF2col\ o\ (F2map\ id\ IF1set\ IF2set)$   
 ⟨proof⟩

**theorem** *IF1set\_simps*:  
 $IF1set\ (ctor1\ x) = F1set1\ x \cup ((\bigcup a \in F1set2\ x. IF1set\ a) \cup (\bigcup a \in F1set3\ x. IF2set\ a))$   
 ⟨proof⟩

**theorem** *IF2set\_simps*:  
 $IF2set\ (ctor2\ x) = F2set1\ x \cup ((\bigcup a \in F2set2\ x. IF1set\ a) \cup (\bigcup a \in F2set3\ x. IF2set\ a))$   
 ⟨proof⟩

**lemmas** *F1set1\_IF1set* =  $xt1(3)[OF\ IF1set\_simps\ Un\_upper1]$

**lemmas** *F1set2\_IF1set* =  $subset\_trans[OF\ UN\_upper\ subset\_trans[OF\ Un\_upper1\ xt1(3)[OF\ IF1set\_simps\ Un\_upper2]]]$

**lemmas** *F1set3\_IF1set* =  $subset\_trans[OF\ UN\_upper\ subset\_trans[OF\ Un\_upper2\ xt1(3)[OF\ IF1set\_simps\ Un\_upper2]]]$

**lemmas** *F2set1\_IF2set* =  $xt1(3)[OF\ IF2set\_simps\ Un\_upper1]$

**lemmas** *F2set2\_IF2set* =  $subset\_trans[OF\ UN\_upper\ subset\_trans[OF\ Un\_upper1\ xt1(3)[OF\ IF2set\_simps\ Un\_upper2]]]$

**lemmas** *F2set3\_IF2set* =  $subset\_trans[OF\ UN\_upper\ subset\_trans[OF\ Un\_upper2\ xt1(3)[OF\ IF2set\_simps\ Un\_upper2]]]$

The BNF conditions for IF

**lemma** *IFset\_natural*:  
 $f' (IF1set\ x) = IF1set\ (IF1map\ f\ x) \wedge f' (IF2set\ y) = IF2set\ (IF2map\ f\ y)$   
 ⟨proof⟩

**theorem** *IF1set\_natural*:  $IF1set\ o\ (IF1map\ f) = image\ f\ o\ IF1set$   
 ⟨proof⟩

**theorem** *IF2set\_natural*:  $IF2set\ o\ (IF2map\ f) = image\ f\ o\ IF2set$   
 ⟨proof⟩

**lemma** *IFmap\_cong*:  
 $((\forall a \in IF1set\ x.\ f\ a = g\ a) \longrightarrow IF1map\ f\ x = IF1map\ g\ x) \wedge$   
 $((\forall a \in IF2set\ y.\ f\ a = g\ a) \longrightarrow IF2map\ f\ y = IF2map\ g\ y)$   
 ⟨proof⟩

**theorem** *IF1map\_cong*:  
 $(\bigwedge a.\ a \in IF1set\ x \implies f\ a = g\ a) \implies IF1map\ f\ x = IF1map\ g\ x$   
 ⟨proof⟩

**theorem** *IF2map\_cong*:  
 $(\bigwedge a.\ a \in IF2set\ x \implies f\ a = g\ a) \implies IF2map\ f\ x = IF2map\ g\ x$   
 ⟨proof⟩

**lemma** *IFset\_bd*:  
 $|IF1set\ (x :: 'a\ IF1)| < o\ IFbd \wedge |IF2set\ (y :: 'a\ IF2)| < o\ IFbd$   
 ⟨proof⟩

**lemmas** *IF1set\_bd = conjunct1[OF IFset\_bd]*

**lemmas** *IF2set\_bd = conjunct2[OF IFset\_bd]*

**definition** *IF1rel where*

$IF1rel\ R =$   
 $(BNF\_Def.Grp\ (IF1in\ (Collect\ (case\_prod\ R)))\ (IF1map\ fst)) \hat{\ } \text{--} 1\ OO$   
 $(BNF\_Def.Grp\ (IF1in\ (Collect\ (case\_prod\ R)))\ (IF1map\ snd))$

**definition** *IF2rel where*

$IF2rel\ R =$   
 $(BNF\_Def.Grp\ (IF2in\ (Collect\ (case\_prod\ R)))\ (IF2map\ fst)) \hat{\ } \text{--} 1\ OO$   
 $(BNF\_Def.Grp\ (IF2in\ (Collect\ (case\_prod\ R)))\ (IF2map\ snd))$

**lemma** *in\_IF1rel*:  
 $IF1rel\ R\ x\ y \longleftrightarrow (\exists z.\ z \in IF1in\ (Collect\ (case\_prod\ R)) \wedge IF1map\ fst\ z = x \wedge IF1map\ snd\ z = y)$   
 ⟨proof⟩

**lemma** *in\_IF2rel*:  
 $IF2rel\ R\ x\ y \longleftrightarrow (\exists z.\ z \in IF2in\ (Collect\ (case\_prod\ R)) \wedge IF2map\ fst\ z = x \wedge IF2map\ snd\ z = y)$   
 ⟨proof⟩

**lemma** *IF1rel\_F1rel*:  $IF1rel\ R\ (ctor1\ a)\ (ctor1\ b) \longleftrightarrow F1rel\ R\ (IF1rel\ R)\ (IF2rel\ R)\ a\ b$   
 ⟨proof⟩

**lemma** *IF2rel\_F2rel*:  $IF2rel\ R\ (ctor2\ a)\ (ctor2\ b) \longleftrightarrow F2rel\ R\ (IF1rel\ R)\ (IF2rel\ R)\ a\ b$   
 ⟨proof⟩

**lemma** *Irel\_induct*:

**assumes**  $IH1: \forall x\ y.\ F1rel\ P1\ P2\ P3\ x\ y \longrightarrow P2\ (ctor1\ x)\ (ctor1\ y)$   
**and**  $IH2: \forall x\ y.\ F2rel\ P1\ P2\ P3\ x\ y \longrightarrow P3\ (ctor2\ x)\ (ctor2\ y)$   
**shows**  $IF1rel\ P1 \leq P2 \wedge IF2rel\ P1 \leq P3$   
 ⟨proof⟩

**lemma** *le\_IFrel\_Comp*:

$((IF1rel\ R\ OO\ IF1rel\ S)\ x1\ y1 \longrightarrow IF1rel\ (R\ OO\ S)\ x1\ y1) \wedge$   
 $((IF2rel\ R\ OO\ IF2rel\ S)\ x2\ y2 \longrightarrow IF2rel\ (R\ OO\ S)\ x2\ y2)$   
 ⟨proof⟩

**lemma** *le\_IF1rel\_Comp*:  $IF1rel\ R1\ OO\ IF1rel\ R2 \leq IF1rel\ (R1\ OO\ R2)$   
 ⟨proof⟩

**lemma** *le\_IF2rel\_Comp*:  $IF2rel\ R1\ OO\ IF2rel\ R2 \leq IF2rel\ (R1\ OO\ R2)$   
 ⟨proof⟩

context includes *lifting\_syntax*

begin

lemma *fold\_transfer*:

$((F1rel\ R\ S\ T\ ==>\ S) ==>\ (F2rel\ R\ S\ T\ ==>\ T) ==>\ IF1rel\ R\ ==>\ S)\ fold1\ fold1\ \wedge$   
 $((F1rel\ R\ S\ T\ ==>\ S) ==>\ (F2rel\ R\ S\ T\ ==>\ T) ==>\ IF2rel\ R\ ==>\ T)\ fold2\ fold2$   
<proof>

end

definition *IF1wit*  $x = ctor1\ (wit2\_F1\ x\ (ctor2\ wit\_F2))$

definition *IF2wit*  $= ctor2\ wit\_F2$

lemma *IF1wit*:  $x \in IF1set\ (IF1wit\ y) \implies x = y$

<proof>

lemma *IF2wit*:  $x \in IF2set\ IF2wit \implies False$

<proof>

<ML>

bnf '*a* *IF1*

map: *IF1map*

sets: *IF1set*

bd: *IFbd*

wits: *IF1wit*

rel: *IF1rel*

<proof>

bnf '*a* *IF2*

map: *IF2map*

sets: *IF2set*

bd: *IFbd*

wits: *IF2wit*

rel: *IF2rel*

<proof>

## 2 Greatest Fixpoint (a.k.a. Codatatype)

unbundle *cardinal\_syntax*

'b1 = ('a, 'b1, 'b2) F1

'b2 = ('a, 'b1, 'b2) F2

To build a witness scenario, let us assume

$(\ 'a, \ 'b1, \ 'b2) \ F1 = \ 'a * \ 'b1 + \ 'a * \ 'b2$

$(\ 'a, \ 'b1, \ 'b2) \ F2 = \ unit + \ 'b1 * \ 'b2$

<ML>

declare [[*bnf\_internals*]]

bnf-axiomatization (*F1set1*: '*a*, *F1set2*: '*b1*, *F1set3*: '*b2*) *F1*

[*wits*: '*a*  $\Rightarrow$  '*b1*  $\Rightarrow$  ('*a*, '*b1*, '*b2*) *F1* '*a*  $\Rightarrow$  '*b2*  $\Rightarrow$  ('*a*, '*b1*, '*b2*) *F1*]

for map: *F1map* rel: *F1rel*

bnf-axiomatization (*F2set1*: '*a*, *F2set2*: '*b1*, *F2set3*: '*b2*) *F2*

[*wits*: ('*a*, '*b1*, '*b2*) *F2*]

for map: *F2map* rel: *F2rel*

lemma *F1rel\_cong*:  $[[R1 = S1; R2 = S2; R3 = S3]] \implies F1rel\ R1\ R2\ R3 = F1rel\ S1\ S2\ S3$

<proof>

lemma *F2rel\_cong*:  $[[R1 = S1; R2 = S2; R3 = S3]] \implies F2rel\ R1\ R2\ R3 = F2rel\ S1\ S2\ S3$

*<proof>*

**abbreviation**  $F1in :: 'a1\ set \Rightarrow 'a2\ set \Rightarrow 'a3\ set \Rightarrow (('a1, 'a2, 'a3)\ F1)\ set$  **where**  
 $F1in\ A1\ A2\ A3 \equiv \{x.\ F1set1\ x \subseteq A1 \wedge F1set2\ x \subseteq A2 \wedge F1set3\ x \subseteq A3\}$

**abbreviation**  $F2in :: 'a1\ set \Rightarrow 'a2\ set \Rightarrow 'a3\ set \Rightarrow (('a1, 'a2, 'a3)\ F2)\ set$  **where**  
 $F2in\ A1\ A2\ A3 \equiv \{x.\ F2set1\ x \subseteq A1 \wedge F2set2\ x \subseteq A2 \wedge F2set3\ x \subseteq A3\}$

**lemma**  $F1map\_comp\_id: F1map\ g1\ g2\ g3\ (F1map\ id\ f2\ f3\ x) = F1map\ g1\ (g2\ o\ f2)\ (g3\ o\ f3)\ x$   
*<proof>*

**lemmas**  $F1in\_mono23 = F1.in\_mono[OF\ subset\_refl]$

**lemmas**  $F1in\_mono23' = subsetD[OF\ F1in\_mono23]$

**lemma**  $F1map\_congL: [\forall a \in F1set2\ x.\ f\ a = a; \forall a \in F1set3\ x.\ g\ a = a] \Longrightarrow$   
 $F1map\ id\ f\ g\ x = x$   
*<proof>*

**lemma**  $F2map\_comp\_id: F2map\ g1\ g2\ g3\ (F2map\ id\ f2\ f3\ x) = F2map\ g1\ (g2\ o\ f2)\ (g3\ o\ f3)\ x$   
*<proof>*

**lemmas**  $F2in\_mono23 = F2.in\_mono[OF\ subset\_refl]$

**lemmas**  $F2in\_mono23' = subsetD[OF\ F2in\_mono23]$

**lemma**  $F2map\_congL: [\forall a \in F2set2\ x.\ f\ a = a; \forall a \in F2set3\ x.\ g\ a = a] \Longrightarrow$   
 $F2map\ id\ f\ g\ x = x$   
*<proof>*

## 2.1 Coalgebra

**definition**  $coalg$  **where**

$coalg\ B1\ B2\ s1\ s2 =$

$((\forall a \in B1.\ s1\ a \in F1in\ (UNIV :: 'a\ set)\ B1\ B2) \wedge (\forall a \in B2.\ s2\ a \in F2in\ (UNIV :: 'a\ set)\ B1\ B2))$

**lemmas**  $coalg\_F1in = bspec[OF\ conjunct1[OF\ iffD1[OF\ coalg\_def]]]$

**lemmas**  $coalg\_F2in = bspec[OF\ conjunct2[OF\ iffD1[OF\ coalg\_def]]]$

**lemma**  $coalg\_F1set2:$

$[[coalg\ B1\ B2\ s1\ s2; a \in B1]] \Longrightarrow F1set2\ (s1\ a) \subseteq B1$   
*<proof>*

**lemma**  $coalg\_F1set3:$

$[[coalg\ B1\ B2\ s1\ s2; a \in B1]] \Longrightarrow F1set3\ (s1\ a) \subseteq B2$   
*<proof>*

**lemma**  $coalg\_F2set2:$

$[[coalg\ B1\ B2\ s1\ s2; a \in B2]] \Longrightarrow F2set2\ (s2\ a) \subseteq B1$   
*<proof>*

**lemma**  $coalg\_F2set3:$

$[[coalg\ B1\ B2\ s1\ s2; a \in B2]] \Longrightarrow F2set3\ (s2\ a) \subseteq B2$   
*<proof>*

## 2.2 Type-coalgebra

**abbreviation**  $tcoalg\ s1\ s2 \equiv coalg\ UNIV\ UNIV\ s1\ s2$

**lemma**  $tcoalg: tcoalg\ s1\ s2$

*<proof>*

## 2.3 Morphism

**definition**  $mor$  **where**

$mor\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1'\ s2'\ f\ g =$

$((\forall a \in B1.\ f\ a \in B1') \wedge (\forall a \in B2.\ g\ a \in B2')) \wedge$

$$((\forall z \in B1. F1map (id :: 'a \Rightarrow 'a) f g (s1 z) = s1' (f z)) \wedge (\forall z \in B2. F2map (id :: 'a \Rightarrow 'a) f g (s2 z) = s2' (g z)))$$

**lemma** *mor\_image1*:  $mor\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1'\ s2'\ f\ g \implies f' \cdot B1 \subseteq B1'$   
 ⟨proof⟩

**lemma** *mor\_image2*:  $mor\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1'\ s2'\ f\ g \implies g' \cdot B2 \subseteq B2'$   
 ⟨proof⟩

**lemmas** *mor\_image1'* = *subsetD*[*OF mor\_image1 imageI*]  
**lemmas** *mor\_image2'* = *subsetD*[*OF mor\_image2 imageI*]

**lemma** *morE1*:  $\llbracket mor\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1'\ s2'\ f\ g; z \in B1 \rrbracket \implies F1map\ id\ f\ g\ (s1\ z) = s1'\ (f\ z)$   
 ⟨proof⟩

**lemma** *morE2*:  $\llbracket mor\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1'\ s2'\ f\ g; z \in B2 \rrbracket \implies F2map\ id\ f\ g\ (s2\ z) = s2'\ (g\ z)$   
 ⟨proof⟩

**lemma** *mor\_incl*:  $\llbracket B1 \subseteq B1'; B2 \subseteq B2' \rrbracket \implies mor\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1\ s2\ id\ id$   
 ⟨proof⟩

**lemmas** *mor\_id* = *mor\_incl*[*OF subset\_refl subset\_refl*]

**lemma** *mor\_comp*:  
 $\llbracket mor\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1'\ s2'\ f\ g; mor\ B1'\ B2'\ s1'\ s2'\ B1''\ B2''\ s1''\ s2''\ f'\ g' \rrbracket \implies mor\ B1\ B2\ s1\ s2\ B1''\ B2''\ s1''\ s2''\ (f' \circ f)\ (g' \circ g)$   
 ⟨proof⟩

**lemma** *mor\_cong*:  $\llbracket f' = f; g' = g; mor\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1'\ s2'\ f\ g \rrbracket \implies mor\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1'\ s2'\ f'\ g'$   
 ⟨proof⟩

**lemma** *mor\_UNIV*:  $mor\ UNIV\ UNIV\ s1\ s2\ UNIV\ UNIV\ s1'\ s2'\ f1\ f2 \longleftrightarrow F1map\ id\ f1\ f2\ o\ s1 = s1' \circ f1 \wedge F2map\ id\ f1\ f2\ o\ s2 = s2' \circ f2$   
 ⟨proof⟩

**lemma** *mor\_str*:  
 $mor\ UNIV\ UNIV\ s1\ s2\ UNIV\ UNIV\ (F1map\ id\ s1\ s2)\ (F2map\ id\ s1\ s2)\ s1\ s2$   
 ⟨proof⟩

**lemma** *mor\_case\_sum*:  
 $mor\ UNIV\ UNIV\ s1\ s2\ UNIV\ UNIV\ (case\_sum\ (F1map\ id\ Inl\ Inl\ o\ s1)\ s1')\ (case\_sum\ (F2map\ id\ Inl\ Inl\ o\ s2)\ s2')\ Inl\ Inl$   
 ⟨proof⟩

## 2.4 Bisimulations

**definition** *bis* where

$$bis\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1'\ s2'\ R1\ R2 = ((R1 \subseteq B1 \times B1' \wedge R2 \subseteq B2 \times B2') \wedge ((\forall b1\ b1'. (b1, b1') \in R1 \longrightarrow (\exists z \in F1in\ UNIV\ R1\ R2. F1map\ id\ fst\ fst\ z = s1\ b1 \wedge F1map\ id\ snd\ snd\ z = s1'\ b1')) \wedge (\forall b2\ b2'. (b2, b2') \in R2 \longrightarrow (\exists z \in F2in\ UNIV\ R1\ R2. F2map\ id\ fst\ fst\ z = s2\ b2 \wedge F2map\ id\ snd\ snd\ z = s2'\ b2')))))$$

**lemma** *bis\_cong*:  $\llbracket bis\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1'\ s2'\ R1\ R2; R1' = R1; R2' = R2 \rrbracket \implies bis\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1'\ s2'\ R1'\ R2'$   
 ⟨proof⟩

**lemma** *bis\_Frel*:

$bis\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1'\ s2'\ R1\ R2 \longleftrightarrow$   
 $(R1 \subseteq B1 \times B1' \wedge R2 \subseteq B2 \times B2') \wedge$   
 $((\forall\ b1\ b1'. (b1, b1') \in R1 \longrightarrow F1rel (=) (in\_rel\ R1) (in\_rel\ R2) (s1\ b1) (s1'\ b1')) \wedge$   
 $(\forall\ b2\ b2'. (b2, b2') \in R2 \longrightarrow F2rel (=) (in\_rel\ R1) (in\_rel\ R2) (s2\ b2) (s2'\ b2'))))$   
 $\langle proof \rangle$

**lemma** *bis\_converse*:

$bis\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1'\ s2'\ R1\ R2 \implies$   
 $bis\ B1'\ B2'\ s1'\ s2'\ B1\ B2\ s1\ s2\ (R1^{\wedge-1})\ (R2^{\wedge-1})$   
 $\langle proof \rangle$

**lemma** *bis\_Comp*:

$\llbracket bis\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1'\ s2'\ P1\ P2;$   
 $bis\ B1'\ B2'\ s1'\ s2'\ B1''\ B2''\ s1''\ s2''\ Q1\ Q2 \rrbracket \implies$   
 $bis\ B1\ B2\ s1\ s2\ B1''\ B2''\ s1''\ s2''\ (P1\ O\ Q1)\ (P2\ O\ Q2)$   
 $\langle proof \rangle$

**lemma** *bis\_Gr*:  $\llbracket coalg\ B1\ B2\ s1\ s2; mor\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1'\ s2'\ f1\ f2 \rrbracket \implies$

$bis\ B1\ B2\ s1\ s2\ B1'\ B2'\ s1'\ s2'\ (BNF\_Def.Gr\ B1\ f1)\ (BNF\_Def.Gr\ B2\ f2)$   
 $\langle proof \rangle$

**lemmas** *bis\_image2* =  $bis\_cong[OF\ bis\_Comp[OF\ bis\_converse[OF\ bis\_Gr]\ bis\_Gr]\ image2\_Gr\ image2\_Gr]$

**lemmas** *bis\_diag* =  $bis\_cong[OF\ bis\_Gr[OF\ \_mor\_id]\ Id\_on\_Gr\ Id\_on\_Gr]$

**lemma** *bis\_Union*:  $\forall i \in I. bis\ B1\ B2\ s1\ s2\ B1\ B2\ s1\ s2\ (R1i\ i)\ (R2i\ i) \implies$

$bis\ B1\ B2\ s1\ s2\ B1\ B2\ s1\ s2\ (\bigcup_{i \in I} R1i\ i)\ (\bigcup_{i \in I} R2i\ i)$   
 $\langle proof \rangle$

**abbreviation** *sbis*  $B1\ B2\ s1\ s2\ R1\ R2 \equiv bis\ B1\ B2\ s1\ s2\ B1\ B2\ s1\ s2\ R1\ R2$

**definition** *lsbis1* **where**  $lsbis1\ B1\ B2\ s1\ s2 =$

$(\bigcup R \in \{(R1, R2) \mid R1\ R2 \cdot sbis\ B1\ B2\ s1\ s2\ R1\ R2\}. fst\ R)$

**definition** *lsbis2* **where**  $lsbis2\ B1\ B2\ s1\ s2 =$

$(\bigcup R \in \{(R1, R2) \mid R1\ R2 \cdot sbis\ B1\ B2\ s1\ s2\ R1\ R2\}. snd\ R)$

**lemma** *sbis\_lsbis*:

$sbis\ B1\ B2\ s1\ s2\ (lsbis1\ B1\ B2\ s1\ s2)\ (lsbis2\ B1\ B2\ s1\ s2)$   
 $\langle proof \rangle$

**lemmas** *lsbis1\_incl* =  $conjunct1[OF\ conjunct1[OF\ iffD1[OF\ bis\_def]], OF\ sbis\_lsbis]$

**lemmas** *lsbis2\_incl* =  $conjunct2[OF\ conjunct1[OF\ iffD1[OF\ bis\_def]], OF\ sbis\_lsbis]$

**lemmas** *lsbisE1* =

$mp[OF\ spec[OF\ spec[OF\ conjunct1[OF\ conjunct2[OF\ iffD1[OF\ bis\_def]], OF\ sbis\_lsbis]]]$

**lemmas** *lsbisE2* =

$mp[OF\ spec[OF\ spec[OF\ conjunct2[OF\ conjunct2[OF\ iffD1[OF\ bis\_def]], OF\ sbis\_lsbis]]]$

**lemma** *incl\_lsbis1*:  $sbis\ B1\ B2\ s1\ s2\ R1\ R2 \implies R1 \subseteq lsbis1\ B1\ B2\ s1\ s2$

$\langle proof \rangle$

**lemma** *incl\_lsbis2*:  $sbis\ B1\ B2\ s1\ s2\ R1\ R2 \implies R2 \subseteq lsbis2\ B1\ B2\ s1\ s2$

$\langle proof \rangle$

**lemma** *equiv\_lsbis1*:  $coalg\ B1\ B2\ s1\ s2 \implies equiv\ B1\ (lsbis1\ B1\ B2\ s1\ s2)$

$\langle proof \rangle$

**lemma** *equiv\_lsbis2*:  $coalg\ B1\ B2\ s1\ s2 \implies equiv\ B2\ (lsbis2\ B1\ B2\ s1\ s2)$

$\langle proof \rangle$

## 2.5 The Tree Coalgebra

**typedef**  $bd\_type\_F = UNIV :: (bd\_type\_F1 + bd\_type\_F2) \text{ suc set}$   
 ⟨proof⟩

**type-synonym**  $'a \text{ carrier} = ((bd\_type\_F + bd\_type\_F) \text{ list set} \times$   
 $((bd\_type\_F + bd\_type\_F) \text{ list} \Rightarrow (('a, bd\_type\_F, bd\_type\_F) F1 + ('a, bd\_type\_F, bd\_type\_F) F2)))$

**abbreviation**  $bd\_F \equiv dir\_image (card\_suc (bd\_F1 + c bd\_F2)) Abs\_bd\_type\_F$

**lemmas**  $sum\_card\_order = card\_order\_csum[OF F1.bd\_card\_order F2.bd\_card\_order]$

**lemmas**  $sum\_Cinfinite = Cinfinite\_csum1[OF F1.bd\_Cinfinite]$

**lemmas**  $bd\_F = dir\_image[OF Abs\_bd\_type\_F\_inject[OF UNIV\_I UNIV\_I] Card\_order\_card\_suc[OF sum\_card\_order]]$

**lemmas**  $bd\_F\_Cinfinite = Cinfinite\_cong[OF bd\_F\_Cinfinite\_card\_suc[OF sum\_Cinfinite sum\_card\_order]]$

**lemmas**  $bd\_F\_Card\_order = Card\_order\_ordIso[OF Card\_order\_card\_suc[OF sum\_card\_order] ordIso\_symmetric[OF bd\_F]]$

**lemma**  $bd\_F\_card\_order: card\_order bd\_F$

⟨proof⟩

**lemmas**  $bd\_F\_regularCard = regularCard\_ordIso[OF bd\_F\_Cinfinite\_card\_suc[OF sum\_Cinfinite sum\_card\_order]$   
 $regularCard\_card\_suc[OF sum\_card\_order sum\_Cinfinite]$

]

**lemmas**  $F1set1\_bd' = ordLess\_transitive[OF F1.set\_bd(1) ordLess\_ordIso\_trans[OF$   
 $ordLeq\_ordLess\_trans[OF ordLeq\_csum1[OF F1.bd\_Card\_order] card\_suc\_greater[OF sum\_card\_order]]$   
 $bd\_F]]$

**lemmas**  $F1set2\_bd' = ordLess\_transitive[OF F1.set\_bd(2) ordLess\_ordIso\_trans[OF$   
 $ordLeq\_ordLess\_trans[OF ordLeq\_csum1[OF F1.bd\_Card\_order] card\_suc\_greater[OF sum\_card\_order]]$   
 $bd\_F]]$

**lemmas**  $F1set3\_bd' = ordLess\_transitive[OF F1.set\_bd(3) ordLess\_ordIso\_trans[OF$   
 $ordLeq\_ordLess\_trans[OF ordLeq\_csum1[OF F1.bd\_Card\_order] card\_suc\_greater[OF sum\_card\_order]]$   
 $bd\_F]]$

**lemmas**  $F2set1\_bd' = ordLess\_transitive[OF F2.set\_bd(1) ordLess\_ordIso\_trans[OF$   
 $ordLeq\_ordLess\_trans[OF ordLeq\_csum2[OF F2.bd\_Card\_order] card\_suc\_greater[OF sum\_card\_order]]$   
 $bd\_F]]$

**lemmas**  $F2set2\_bd' = ordLess\_transitive[OF F2.set\_bd(2) ordLess\_ordIso\_trans[OF$   
 $ordLeq\_ordLess\_trans[OF ordLeq\_csum2[OF F2.bd\_Card\_order] card\_suc\_greater[OF sum\_card\_order]]$   
 $bd\_F]]$

**lemmas**  $F2set3\_bd' = ordLess\_transitive[OF F2.set\_bd(3) ordLess\_ordIso\_trans[OF$   
 $ordLeq\_ordLess\_trans[OF ordLeq\_csum2[OF F2.bd\_Card\_order] card\_suc\_greater[OF sum\_card\_order]]$   
 $bd\_F]]$

**abbreviation**  $Succ1 Kl kl \equiv \{k1. Inl k1 \in BNF\_Greatest\_Fixpoint.Succ Kl kl\}$

**abbreviation**  $Succ2 Kl kl \equiv \{k2. Inr k2 \in BNF\_Greatest\_Fixpoint.Succ Kl kl\}$

**definition**  $isNode1$  **where**

$isNode1 Kl lab kl = (\exists x1. lab kl = Inl x1 \wedge F1set2 x1 = Succ1 Kl kl \wedge F1set3 x1 = Succ2 Kl kl)$

**definition**  $isNode2$  **where**

$isNode2 Kl lab kl = (\exists x2. lab kl = Inr x2 \wedge F2set2 x2 = Succ1 Kl kl \wedge F2set3 x2 = Succ2 Kl kl)$

**abbreviation**  $isTree$  **where**

$isTree Kl lab \equiv ([\ ] \in Kl \wedge$   
 $(\forall kl \in Kl. (\forall k1 \in Succ1 Kl kl. isNode1 Kl lab (kl @ [Inl k1])) \wedge$   
 $(\forall k2 \in Succ2 Kl kl. isNode2 Kl lab (kl @ [Inr k2])))$

**definition**  $carT1$  **where**

$carT1 = \{(Kl :: (bd\_type\_F + bd\_type\_F) \text{ list set}, lab) \mid Kl \text{ lab. } isTree Kl lab \wedge isNode1 Kl lab [\ ]\}$

**definition**  $carT2$  **where**

$carT2 = \{(Kl :: (bd\_type\_F + bd\_type\_F) \text{ list set}, lab) \mid Kl \text{ lab. } isTree Kl lab \wedge isNode2 Kl lab [\ ]\}$

**definition**  $strT1$  **where**

$strT1 = (case\_prod (\%Kl \text{ lab. case\_sum (F1map id$

( $\lambda k1. (BNF\_Greatest\_Fixpoint.Shift\ Kl\ (Inl\ k1),\ BNF\_Greatest\_Fixpoint.shift\ lab\ (Inl\ k1))$ )  
( $\lambda k2. (BNF\_Greatest\_Fixpoint.Shift\ Kl\ (Inr\ k2),\ BNF\_Greatest\_Fixpoint.shift\ lab\ (Inr\ k2))$ ) undefined (lab  
[]))

**definition** *strT2* **where**

*strT2* = (case\_prod (%Kl lab. case\_sum undefined (F2map id  
( $\lambda k1. (BNF\_Greatest\_Fixpoint.Shift\ Kl\ (Inl\ k1),\ BNF\_Greatest\_Fixpoint.shift\ lab\ (Inl\ k1))$ )  
( $\lambda k2. (BNF\_Greatest\_Fixpoint.Shift\ Kl\ (Inr\ k2),\ BNF\_Greatest\_Fixpoint.shift\ lab\ (Inr\ k2))$ ))) (lab []))

**lemma** *coalg\_T*: *coalg carT1 carT2 strT1 strT2*

(*proof*)

**abbreviation** *tobd\_F12* **where** *tobd\_F12 s1 x*  $\equiv$  *toCard (F1set2 (s1 x)) bd\_F*

**abbreviation** *tobd\_F13* **where** *tobd\_F13 s1 x*  $\equiv$  *toCard (F1set3 (s1 x)) bd\_F*

**abbreviation** *tobd\_F22* **where** *tobd\_F22 s2 x*  $\equiv$  *toCard (F2set2 (s2 x)) bd\_F*

**abbreviation** *tobd\_F23* **where** *tobd\_F23 s2 x*  $\equiv$  *toCard (F2set3 (s2 x)) bd\_F*

**abbreviation** *frombd\_F12* **where** *frombd\_F12 s1 x*  $\equiv$  *fromCard (F1set2 (s1 x)) bd\_F*

**abbreviation** *frombd\_F13* **where** *frombd\_F13 s1 x*  $\equiv$  *fromCard (F1set3 (s1 x)) bd\_F*

**abbreviation** *frombd\_F22* **where** *frombd\_F22 s2 x*  $\equiv$  *fromCard (F2set2 (s2 x)) bd\_F*

**abbreviation** *frombd\_F23* **where** *frombd\_F23 s2 x*  $\equiv$  *fromCard (F2set3 (s2 x)) bd\_F*

**lemmas** *tobd\_F12\_inj* = *toCard\_inj[OF ordLess\_imp\_ordLeq[OF F1set2\_bd'] bd\_F\_Card\_order]*

**lemmas** *tobd\_F13\_inj* = *toCard\_inj[OF ordLess\_imp\_ordLeq[OF F1set3\_bd'] bd\_F\_Card\_order]*

**lemmas** *tobd\_F22\_inj* = *toCard\_inj[OF ordLess\_imp\_ordLeq[OF F2set2\_bd'] bd\_F\_Card\_order]*

**lemmas** *tobd\_F23\_inj* = *toCard\_inj[OF ordLess\_imp\_ordLeq[OF F2set3\_bd'] bd\_F\_Card\_order]*

**lemmas** *frombd\_F12\_tobd\_F12* = *fromCard\_toCard[OF ordLess\_imp\_ordLeq[OF F1set2\_bd'] bd\_F\_Card\_order]*

**lemmas** *frombd\_F13\_tobd\_F13* = *fromCard\_toCard[OF ordLess\_imp\_ordLeq[OF F1set3\_bd'] bd\_F\_Card\_order]*

**lemmas** *frombd\_F22\_tobd\_F22* = *fromCard\_toCard[OF ordLess\_imp\_ordLeq[OF F2set2\_bd'] bd\_F\_Card\_order]*

**lemmas** *frombd\_F23\_tobd\_F23* = *fromCard\_toCard[OF ordLess\_imp\_ordLeq[OF F2set3\_bd'] bd\_F\_Card\_order]*

**definition** *Lev* **where**

*Lev s1 s2* = *rec\_nat (%a. {[]}, %b. {[]})*

(%n rec.

(%a1.

{*Inl (tobd\_F12 s1 a1 b1) # kl | b1 kl. b1  $\in$  F1set2 (s1 a1)  $\wedge$  kl  $\in$  fst rec b1}*  $\cup$   
{i*Inr (tobd\_F13 s1 a1 b2) # kl | b2 kl. b2  $\in$  F1set3 (s1 a1)  $\wedge$  kl  $\in$  snd rec b2*},

%a2.

{*Inl (tobd\_F22 s2 a2 b1) # kl | b1 kl. b1  $\in$  F2set2 (s2 a2)  $\wedge$  kl  $\in$  fst rec b1}*  $\cup$   
{i*Inr (tobd\_F23 s2 a2 b2) # kl | b2 kl. b2  $\in$  F2set3 (s2 a2)  $\wedge$  kl  $\in$  snd rec b2*})

**abbreviation** *Lev1* **where** *Lev1 s1 s2 n*  $\equiv$  *fst (Lev s1 s2 n)*

**abbreviation** *Lev2* **where** *Lev2 s1 s2 n*  $\equiv$  *snd (Lev s1 s2 n)*

**lemmas** *Lev1\_0* = *fun\_cong[OF fstI[OF rec\_nat\_0\_imp[OF Lev\_def]]]*

**lemmas** *Lev2\_0* = *fun\_cong[OF sndI[OF rec\_nat\_0\_imp[OF Lev\_def]]]*

**lemmas** *Lev1\_Suc* = *fun\_cong[OF fstI[OF rec\_nat\_Suc\_imp[OF Lev\_def]]]*

**lemmas** *Lev2\_Suc* = *fun\_cong[OF sndI[OF rec\_nat\_Suc\_imp[OF Lev\_def]]]*

**definition** *rv* **where**

*rv s1 s2* = *rec\_list (%b1. Inl b1, %b2. Inr b2)*

(%k kl rec.

(%b1. case\_sum (%k1. fst rec (frombd\_F12 s1 b1 k1)) (%k2. snd rec (frombd\_F13 s1 b1 k2)) k,  
%b2. case\_sum (%k1. fst rec (frombd\_F22 s2 b2 k1)) (%k2. snd rec (frombd\_F23 s2 b2 k2)) k))

**abbreviation** *rv1* **where** *rv1 s1 s2 kl*  $\equiv$  *fst (rv s1 s2 kl)*

**abbreviation** *rv2* **where** *rv2 s1 s2 kl*  $\equiv$  *snd (rv s1 s2 kl)*

**lemmas** *rv1\_Nil* = *fun\_cong[OF fstI[OF rec\_list\_Nil\_imp[OF rv\_def]]]*

**lemmas** *rv2\_Nil* = *fun\_cong[OF sndI[OF rec\_list\_Nil\_imp[OF rv\_def]]]*

**lemmas** *rv1\_Cons* = *fun\_cong[OF fstI[OF rec\_list\_Cons\_imp[OF rv\_def]]]*

**lemmas** *rv2\_Cons* = *fun\_cong[OF sndI[OF rec\_list\_Cons\_imp[OF rv\_def]]]*

**abbreviation**  $Lab1\ s1\ s2\ b1\ kl \equiv$

$$(case\_sum\ (\%k.\ Inl\ (F1map\ id\ (tobd\_F12\ s1\ k)\ (tobd\_F13\ s1\ k)\ (s1\ k))) \\ (\%k.\ Inr\ (F2map\ id\ (tobd\_F22\ s2\ k)\ (tobd\_F23\ s2\ k)\ (s2\ k)))\ (rv1\ s1\ s2\ kl\ b1))$$

**abbreviation**  $Lab2\ s1\ s2\ b2\ kl \equiv$

$$(case\_sum\ (\%k.\ Inl\ (F1map\ id\ (tobd\_F12\ s1\ k)\ (tobd\_F13\ s1\ k)\ (s1\ k))) \\ (\%k.\ Inr\ (F2map\ id\ (tobd\_F22\ s2\ k)\ (tobd\_F23\ s2\ k)\ (s2\ k)))\ (rv2\ s1\ s2\ kl\ b2))$$

**definition**  $beh1\ s1\ s2\ a = (\bigcup n.\ Lev1\ s1\ s2\ n\ a,\ Lab1\ s1\ s2\ a)$

**definition**  $beh2\ s1\ s2\ a = (\bigcup n.\ Lev2\ s1\ s2\ n\ a,\ Lab2\ s1\ s2\ a)$

**lemma**  $length\_Lev:$

$$\forall kl\ b1\ b2.\ (kl \in Lev1\ s1\ s2\ n\ b1 \longrightarrow length\ kl = n) \wedge \\ (kl \in Lev2\ s1\ s2\ n\ b2 \longrightarrow length\ kl = n) \\ \langle proof \rangle$$

**lemmas**  $length\_Lev1 = mp[OF\ conjunct1[OF\ spec[OF\ spec[OF\ spec[OF\ length\_Lev]]]]]$

**lemmas**  $length\_Lev2 = mp[OF\ conjunct2[OF\ spec[OF\ spec[OF\ spec[OF\ length\_Lev]]]]]$

**lemma**  $length\_Lev1': kl \in Lev1\ s1\ s2\ n\ a \implies kl \in Lev1\ s1\ s2\ (length\ kl)\ a$

$\langle proof \rangle$

**lemma**  $length\_Lev2': kl \in Lev2\ s1\ s2\ n\ a \implies kl \in Lev2\ s1\ s2\ (length\ kl)\ a$

$\langle proof \rangle$

**lemma**  $rv\_last:$

$$\forall k\ b1\ b2.\ \\ ((\exists b1'.\ rv1\ s1\ s2\ (kl\ @\ [Inl\ k])\ b1 = Inl\ b1') \wedge \\ (\exists b1'.\ rv1\ s1\ s2\ (kl\ @\ [Inr\ k])\ b1 = Inr\ b1') \wedge \\ ((\exists b2'.\ rv2\ s1\ s2\ (kl\ @\ [Inl\ k])\ b2 = Inl\ b2') \wedge \\ (\exists b2'.\ rv2\ s1\ s2\ (kl\ @\ [Inr\ k])\ b2 = Inr\ b2'))) \\ \langle proof \rangle$$

**lemmas**  $rv\_last' = spec[OF\ spec[OF\ spec[OF\ rv\_last]]]$

**lemmas**  $rv1\_Inl\_last = conjunct1[OF\ conjunct1[OF\ rv\_last']]$

**lemmas**  $rv1\_Inr\_last = conjunct2[OF\ conjunct1[OF\ rv\_last']]$

**lemmas**  $rv2\_Inl\_last = conjunct1[OF\ conjunct2[OF\ rv\_last']]$

**lemmas**  $rv2\_Inr\_last = conjunct2[OF\ conjunct2[OF\ rv\_last']]$

**lemma**  $Fset\_Lev:$

$$\forall kl\ b1\ b2\ b1'\ b2'\ b1''\ b2''. \\ (kl \in Lev1\ s1\ s2\ n\ b1 \longrightarrow \\ ((rv1\ s1\ s2\ kl\ b1 = Inl\ b1' \longrightarrow \\ (b1'' \in F1set2\ (s1\ b1') \longrightarrow \\ kl\ @\ [Inl\ (tobd\_F12\ s1\ b1'\ b1'')]) \in Lev1\ s1\ s2\ (Suc\ n)\ b1) \wedge \\ (b2'' \in F1set3\ (s1\ b1') \longrightarrow \\ kl\ @\ [Inr\ (tobd\_F13\ s1\ b1'\ b2'')]) \in Lev1\ s1\ s2\ (Suc\ n)\ b1)) \wedge \\ (rv1\ s1\ s2\ kl\ b1 = Inr\ b2' \longrightarrow \\ (b1'' \in F2set2\ (s2\ b2') \longrightarrow \\ kl\ @\ [Inl\ (tobd\_F22\ s2\ b2'\ b1'')]) \in Lev1\ s1\ s2\ (Suc\ n)\ b1) \wedge \\ (b2'' \in F2set3\ (s2\ b2') \longrightarrow \\ kl\ @\ [Inr\ (tobd\_F23\ s2\ b2'\ b2'')]) \in Lev1\ s1\ s2\ (Suc\ n)\ b1)))) \wedge \\ (kl \in Lev2\ s1\ s2\ n\ b2 \longrightarrow \\ ((rv2\ s1\ s2\ kl\ b2 = Inl\ b1' \longrightarrow \\ (b1'' \in F1set2\ (s1\ b1') \longrightarrow \\ kl\ @\ [Inl\ (tobd\_F12\ s1\ b1'\ b1'')]) \in Lev2\ s1\ s2\ (Suc\ n)\ b2) \wedge \\ (b2'' \in F1set3\ (s1\ b1') \longrightarrow \\ kl\ @\ [Inr\ (tobd\_F13\ s1\ b1'\ b2'')]) \in Lev2\ s1\ s2\ (Suc\ n)\ b2)) \wedge \\ (rv2\ s1\ s2\ kl\ b2 = Inr\ b2' \longrightarrow \\ (b1'' \in F2set2\ (s2\ b2') \longrightarrow$$

$kl @ [Inl (tobd\_F22\ s2\ b2'\ b1'')] \in Lev2\ s1\ s2\ (Suc\ n)\ b2) \wedge$   
 $(b2'' \in F2set3\ (s2\ b2')) \longrightarrow$   
 $kl @ [Inr (tobd\_F23\ s2\ b2'\ b2'')] \in Lev2\ s1\ s2\ (Suc\ n)\ b2))$   
 <proof>

**lemmas**  $Fset\_Lev' = spec[OF\ spec[OF\ spec[OF\ spec[OF\ spec[OF\ spec[OF\ spec[OF\ Fset\_Lev]]]]]]]$   
**lemmas**  $F1set2\_Lev1 = mp[OF\ conjunct1[OF\ mp[OF\ conjunct1[OF\ mp[OF\ conjunct1[OF\ Fset\_Lev]]]]]]]$   
**lemmas**  $F1set2\_Lev2 = mp[OF\ conjunct1[OF\ mp[OF\ conjunct1[OF\ mp[OF\ conjunct2[OF\ Fset\_Lev]]]]]]]$   
**lemmas**  $F2set2\_Lev1 = mp[OF\ conjunct1[OF\ mp[OF\ conjunct2[OF\ mp[OF\ conjunct1[OF\ Fset\_Lev]]]]]]]$   
**lemmas**  $F2set2\_Lev2 = mp[OF\ conjunct1[OF\ mp[OF\ conjunct2[OF\ mp[OF\ conjunct2[OF\ Fset\_Lev]]]]]]]$   
**lemmas**  $F1set3\_Lev1 = mp[OF\ conjunct2[OF\ mp[OF\ conjunct1[OF\ mp[OF\ conjunct1[OF\ Fset\_Lev]]]]]]]$   
**lemmas**  $F1set3\_Lev2 = mp[OF\ conjunct2[OF\ mp[OF\ conjunct1[OF\ mp[OF\ conjunct2[OF\ Fset\_Lev]]]]]]]$   
**lemmas**  $F2set3\_Lev1 = mp[OF\ conjunct2[OF\ mp[OF\ conjunct2[OF\ mp[OF\ conjunct1[OF\ Fset\_Lev]]]]]]]$   
**lemmas**  $F2set3\_Lev2 = mp[OF\ conjunct2[OF\ mp[OF\ conjunct2[OF\ mp[OF\ conjunct2[OF\ Fset\_Lev]]]]]]]$

**lemma**  $Fset\_image\_Lev:$   
 $\forall kl\ k\ b1\ b2\ b1'\ b2'.$   
 $(kl \in Lev1\ s1\ s2\ n\ b1 \longrightarrow$   
 $(kl @ [Inl\ k] \in Lev1\ s1\ s2\ (Suc\ n)\ b1 \longrightarrow$   
 $(rv1\ s1\ s2\ kl\ b1 = Inl\ b1' \longrightarrow k \in tobd\_F12\ s1\ b1' \text{ ' } F1set2\ (s1\ b1')) \wedge$   
 $(rv1\ s1\ s2\ kl\ b1 = Inr\ b2' \longrightarrow k \in tobd\_F22\ s2\ b2' \text{ ' } F2set2\ (s2\ b2')))) \wedge$   
 $(kl @ [Inr\ k] \in Lev1\ s1\ s2\ (Suc\ n)\ b1 \longrightarrow$   
 $(rv1\ s1\ s2\ kl\ b1 = Inl\ b1' \longrightarrow k \in tobd\_F13\ s1\ b1' \text{ ' } F1set3\ (s1\ b1')) \wedge$   
 $(rv1\ s1\ s2\ kl\ b1 = Inr\ b2' \longrightarrow k \in tobd\_F23\ s2\ b2' \text{ ' } F2set3\ (s2\ b2')))) \wedge$   
 $(kl \in Lev2\ s1\ s2\ n\ b2 \longrightarrow$   
 $(kl @ [Inl\ k] \in Lev2\ s1\ s2\ (Suc\ n)\ b2 \longrightarrow$   
 $(rv2\ s1\ s2\ kl\ b2 = Inl\ b1' \longrightarrow k \in tobd\_F12\ s1\ b1' \text{ ' } F1set2\ (s1\ b1')) \wedge$   
 $(rv2\ s1\ s2\ kl\ b2 = Inr\ b2' \longrightarrow k \in tobd\_F22\ s2\ b2' \text{ ' } F2set2\ (s2\ b2')))) \wedge$   
 $(kl @ [Inr\ k] \in Lev2\ s1\ s2\ (Suc\ n)\ b2 \longrightarrow$   
 $(rv2\ s1\ s2\ kl\ b2 = Inl\ b1' \longrightarrow k \in tobd\_F13\ s1\ b1' \text{ ' } F1set3\ (s1\ b1')) \wedge$   
 $(rv2\ s1\ s2\ kl\ b2 = Inr\ b2' \longrightarrow k \in tobd\_F23\ s2\ b2' \text{ ' } F2set3\ (s2\ b2'))))$   
 <proof>

**lemmas**  $Fset\_image\_Lev' =$   
 $spec[OF\ spec[OF\ spec[OF\ spec[OF\ spec[OF\ spec[OF\ Fset\_image\_Lev]]]]]]]$   
**lemmas**  $F1set2\_image\_Lev1 =$   
 $mp[OF\ conjunct1[OF\ mp[OF\ conjunct1[OF\ mp[OF\ conjunct1[OF\ Fset\_image\_Lev]]]]]]]$   
**lemmas**  $F1set2\_image\_Lev2 =$   
 $mp[OF\ conjunct1[OF\ mp[OF\ conjunct1[OF\ mp[OF\ conjunct2[OF\ Fset\_image\_Lev]]]]]]]$   
**lemmas**  $F1set3\_image\_Lev1 =$   
 $mp[OF\ conjunct1[OF\ mp[OF\ conjunct2[OF\ mp[OF\ conjunct1[OF\ Fset\_image\_Lev]]]]]]]$   
**lemmas**  $F1set3\_image\_Lev2 =$   
 $mp[OF\ conjunct1[OF\ mp[OF\ conjunct2[OF\ mp[OF\ conjunct2[OF\ Fset\_image\_Lev]]]]]]]$   
**lemmas**  $F2set2\_image\_Lev1 =$   
 $mp[OF\ conjunct2[OF\ mp[OF\ conjunct1[OF\ mp[OF\ conjunct1[OF\ Fset\_image\_Lev]]]]]]]$   
**lemmas**  $F2set2\_image\_Lev2 =$   
 $mp[OF\ conjunct2[OF\ mp[OF\ conjunct1[OF\ mp[OF\ conjunct2[OF\ Fset\_image\_Lev]]]]]]]$   
**lemmas**  $F2set3\_image\_Lev1 =$   
 $mp[OF\ conjunct2[OF\ mp[OF\ conjunct2[OF\ mp[OF\ conjunct1[OF\ Fset\_image\_Lev]]]]]]]$   
**lemmas**  $F2set3\_image\_Lev2 =$   
 $mp[OF\ conjunct2[OF\ mp[OF\ conjunct2[OF\ mp[OF\ conjunct2[OF\ Fset\_image\_Lev]]]]]]]$

**lemma**  $mor\_beh:$   
 $mor\ UNIV\ UNIV\ s1\ s2\ carT1\ carT2\ strT1\ strT2\ (beh1\ s1\ s2)\ (beh2\ s1\ s2)$   
 <proof>

## 2.6 Quotient Coalgebra

**abbreviation**  $car\_final1$  **where**  
 $car\_final1 \equiv carT1\ /\ (lsbis1\ carT1\ carT2\ strT1\ strT2)$   
**abbreviation**  $car\_final2$  **where**  
 $car\_final2 \equiv carT2\ /\ (lsbis2\ carT1\ carT2\ strT1\ strT2)$   
**abbreviation**  $str\_final1$  **where**  
 $str\_final1 \equiv univ\ (F1map\ id)$

```

(Equiv_Relations.proj (lsbis1 carT1 carT2 strT1 strT2))
(Equiv_Relations.proj (lsbis2 carT1 carT2 strT1 strT2)) o strT1
abbreviation str_final2 where
  str_final2  $\equiv$  univ (F2map id
    (Equiv_Relations.proj (lsbis1 carT1 carT2 strT1 strT2))
    (Equiv_Relations.proj (lsbis2 carT1 carT2 strT1 strT2)) o strT2)

lemma congruent_strQ1: congruent (lsbis1 carT1 carT2 strT1 strT2 :: 'a carrier rel)
  (F1map id (Equiv_Relations.proj (lsbis1 carT1 carT2 strT1 strT2 :: 'a carrier rel))
    (Equiv_Relations.proj (lsbis2 carT1 carT2 strT1 strT2 :: 'a carrier rel)) o strT1)
  <proof>

lemma congruent_strQ2: congruent (lsbis2 carT1 carT2 strT1 strT2 :: 'a carrier rel)
  (F2map id (Equiv_Relations.proj (lsbis1 carT1 carT2 strT1 strT2 :: 'a carrier rel))
    (Equiv_Relations.proj (lsbis2 carT1 carT2 strT1 strT2 :: 'a carrier rel)) o strT2)
  <proof>

lemma coalg_final:
  coalg car_final1 car_final2 str_final1 str_final2
  <proof>

lemma mor_T_final:
  mor carT1 carT2 strT1 strT2 car_final1 car_final2 str_final1 str_final2
  (Equiv_Relations.proj (lsbis1 carT1 carT2 strT1 strT2))
  (Equiv_Relations.proj (lsbis2 carT1 carT2 strT1 strT2))
  <proof>

lemmas mor_final = mor_comp[OF mor_beh mor_T_final]
lemmas in_car_final1 = mor_image1'[OF mor_final UNIV_I]
lemmas in_car_final2 = mor_image2'[OF mor_final UNIV_I]

typedef (overloaded) 'a JF1 = car_final1 :: 'a carrier set set
  <proof>

typedef (overloaded) 'a JF2 = car_final2 :: 'a carrier set set
  <proof>

definition dtor1 where
  dtor1 x = F1map id Abs_JF1 Abs_JF2 (str_final1 (Rep_JF1 x))
definition dtor2 where
  dtor2 x = F2map id Abs_JF1 Abs_JF2 (str_final2 (Rep_JF2 x))

lemma mor_Rep_JF: mor UNIV UNIV dtor1 dtor2
  car_final1 car_final2 str_final1 str_final2
  Rep_JF1 Rep_JF2
  <proof>

lemma mor_Abs_JF: mor car_final1 car_final2 str_final1 str_final2
  UNIV UNIV dtor1 dtor2 Abs_JF1 Abs_JF2
  <proof>

definition unfold1 where
  unfold1 s1 s2 x =
    Abs_JF1 ((Equiv_Relations.proj (lsbis1 carT1 carT2 strT1 strT2)) o beh1 s1 s2) x)
definition unfold2 where
  unfold2 s1 s2 x =
    Abs_JF2 ((Equiv_Relations.proj (lsbis2 carT1 carT2 strT1 strT2)) o beh2 s1 s2) x)

lemma mor_unfold:

```

*mor UNIV UNIV s1 s2 UNIV UNIV dtor1 dtor2 (unfold1 s1 s2) (unfold2 s1 s2)*  
 ⟨proof⟩

**lemmas** *unfold1 = sym[OF morE1[OF mor\_unfold UNIV\_I]]*  
**lemmas** *unfold2 = sym[OF morE2[OF mor\_unfold UNIV\_I]]*

**lemma** *JF\_cind: sbis UNIV UNIV dtor1 dtor2 R1 R2  $\implies R1 \subseteq Id \wedge R2 \subseteq Id$*   
 ⟨proof⟩

**lemmas** *JF\_cind1 = conjunct1[OF JF\_cind]*  
**lemmas** *JF\_cind2 = conjunct2[OF JF\_cind]*

**lemma** *unfold\_unique\_mor:*  
*mor UNIV UNIV s1 s2 UNIV UNIV dtor1 dtor2 f1 f2  $\implies$*   
*f1 = unfold1 s1 s2  $\wedge$  f2 = unfold2 s1 s2*  
 ⟨proof⟩

**lemmas** *unfold\_unique = unfold\_unique\_mor[OF iffD2[OF mor\_UNIV], OF conjI]*  
**lemmas** *unfold1\_dtor = sym[OF conjunct1[OF unfold\_unique\_mor[OF mor\_id]]]*  
**lemmas** *unfold2\_dtor = sym[OF conjunct2[OF unfold\_unique\_mor[OF mor\_id]]]*

**lemmas** *unfold1\_o\_dtor1 =*  
*trans[OF conjunct1[OF unfold\_unique\_mor[OF mor\_comp[OF mor\_str mor\_unfold]]] unfold1\_dtor]*  
**lemmas** *unfold2\_o\_dtor2 =*  
*trans[OF conjunct2[OF unfold\_unique\_mor[OF mor\_comp[OF mor\_str mor\_unfold]]] unfold2\_dtor]*

**definition** *ctor1 where ctor1 = unfold1 (F1map id dtor1 dtor2) (F2map id dtor1 dtor2)*  
**definition** *ctor2 where ctor2 = unfold2 (F1map id dtor1 dtor2) (F2map id dtor1 dtor2)*

**lemma** *ctor1\_o\_dtor1:*  
*ctor1 o dtor1 = id*  
 ⟨proof⟩

**lemma** *ctor2\_o\_dtor2:*  
*ctor2 o dtor2 = id*  
 ⟨proof⟩

**lemma** *dtor1\_o\_ctor1:*  
*dtor1 o ctor1 = id*  
 ⟨proof⟩

**lemma** *dtor2\_o\_ctor2:*  
*dtor2 o ctor2 = id*  
 ⟨proof⟩

**lemmas** *dtor1\_ctor1 = pointfree\_idE[OF dtor1\_o\_ctor1]*  
**lemmas** *dtor2\_ctor2 = pointfree\_idE[OF dtor2\_o\_ctor2]*  
**lemmas** *ctor1\_dtor1 = pointfree\_idE[OF ctor1\_o\_dtor1]*  
**lemmas** *ctor2\_dtor2 = pointfree\_idE[OF ctor2\_o\_dtor2]*

**lemmas** *bij\_dtor1 = o\_bij[OF ctor1\_o\_dtor1 dtor1\_o\_ctor1]*  
**lemmas** *inj\_dtor1 = bij\_is\_inj[OF bij\_dtor1]*  
**lemmas** *surj\_dtor1 = bij\_is\_surj[OF bij\_dtor1]*  
**lemmas** *dtor1\_nchotomy = surjD[OF surj\_dtor1]*  
**lemmas** *dtor1\_diff = inj\_eq[OF inj\_dtor1]*  
**lemmas** *dtor1\_cases = exE[OF dtor1\_nchotomy]*  
**lemmas** *bij\_dtor2 = o\_bij[OF ctor2\_o\_dtor2 dtor2\_o\_ctor2]*  
**lemmas** *inj\_dtor2 = bij\_is\_inj[OF bij\_dtor2]*  
**lemmas** *surj\_dtor2 = bij\_is\_surj[OF bij\_dtor2]*  
**lemmas** *dtor2\_nchotomy = surjD[OF surj\_dtor2]*  
**lemmas** *dtor2\_diff = inj\_eq[OF inj\_dtor2]*  
**lemmas** *dtor2\_cases = exE[OF dtor2\_nchotomy]*

**lemmas**  $\text{bij\_ctor1} = \text{o\_bij}[OF \text{dtor1\_o\_ctor1} \text{ctor1\_o\_dtor1}]$   
**lemmas**  $\text{inj\_ctor1} = \text{bij\_is\_inj}[OF \text{bij\_ctor1}]$   
**lemmas**  $\text{surj\_ctor1} = \text{bij\_is\_surj}[OF \text{bij\_ctor1}]$   
**lemmas**  $\text{ctor1\_nchotomy} = \text{surjD}[OF \text{surj\_ctor1}]$   
**lemmas**  $\text{ctor1\_diff} = \text{inj\_eq}[OF \text{inj\_ctor1}]$   
**lemmas**  $\text{ctor1\_cases} = \text{exE}[OF \text{ctor1\_nchotomy}]$   
**lemmas**  $\text{bij\_ctor2} = \text{o\_bij}[OF \text{dtor2\_o\_ctor2} \text{ctor2\_o\_dtor2}]$   
**lemmas**  $\text{inj\_ctor2} = \text{bij\_is\_inj}[OF \text{bij\_ctor2}]$   
**lemmas**  $\text{surj\_ctor2} = \text{bij\_is\_surj}[OF \text{bij\_ctor2}]$   
**lemmas**  $\text{ctor2\_nchotomy} = \text{surjD}[OF \text{surj\_ctor2}]$   
**lemmas**  $\text{ctor2\_diff} = \text{inj\_eq}[OF \text{inj\_ctor2}]$   
**lemmas**  $\text{ctor2\_cases} = \text{exE}[OF \text{ctor2\_nchotomy}]$

**lemmas**  $\text{ctor1\_unfold1} = \text{iffD1}[OF \text{dtor1\_diff} \text{trans}[OF \text{unfold1} \text{sym}[OF \text{dtor1\_ctor1}]]]$   
**lemmas**  $\text{ctor2\_unfold2} = \text{iffD1}[OF \text{dtor2\_diff} \text{trans}[OF \text{unfold2} \text{sym}[OF \text{dtor2\_ctor2}]]]$

**definition**  $\text{corec1}$  **where**  $\text{corec1} \text{ s1 s2} =$   
 $\text{unfold1} (\text{case\_sum} (\text{F1map id Inl Inl o dtor1}) \text{ s1})$   
 $(\text{case\_sum} (\text{F2map id Inl Inl o dtor2}) \text{ s2}) \text{ o Inr}$

**definition**  $\text{corec2}$  **where**  $\text{corec2} \text{ s1 s2} =$   
 $\text{unfold2} (\text{case\_sum} (\text{F1map id Inl Inl o dtor1}) \text{ s1})$   
 $(\text{case\_sum} (\text{F2map id Inl Inl o dtor2}) \text{ s2}) \text{ o Inr}$

**lemma**  $\text{dtor1\_o\_unfold1}$ :  $\text{dtor1 o unfold1 s1 s2} = \text{F1map id} (\text{unfold1 s1 s2}) (\text{unfold2 s1 s2}) \text{ o s1}$   
 $\langle \text{proof} \rangle$

**lemma**  $\text{dtor2\_o\_unfold2}$ :  $\text{dtor2 o unfold2 s1 s2} = \text{F2map id} (\text{unfold1 s1 s2}) (\text{unfold2 s1 s2}) \text{ o s2}$   
 $\langle \text{proof} \rangle$

**lemma**  $\text{corec1\_Inl\_sum}$ :  
 $\text{unfold1} (\text{case\_sum} (\text{F1map id Inl Inl o dtor1}) \text{ s1}) (\text{case\_sum} (\text{F2map id Inl Inl o dtor2}) \text{ s2}) \text{ o Inl} = \text{id}$   
 $\langle \text{proof} \rangle$

**lemma**  $\text{corec2\_Inl\_sum}$ :  
 $\text{unfold2} (\text{case\_sum} (\text{F1map id Inl Inl o dtor1}) \text{ s1}) (\text{case\_sum} (\text{F2map id Inl Inl o dtor2}) \text{ s2}) \text{ o Inl} = \text{id}$   
 $\langle \text{proof} \rangle$

**lemma**  $\text{case\_sum\_expand\_Inr}$ :  $f \text{ o Inl} = g \implies \text{case\_sum } g (f \text{ o Inr}) = f$   
 $\langle \text{proof} \rangle$

**theorem**  $\text{corec1}$ :  
 $\text{dtor1} (\text{corec1} \text{ s1 s2 } a) =$   
 $\text{F1map id} (\text{case\_sum id} (\text{corec1} \text{ s1 s2})) (\text{case\_sum id} (\text{corec2} \text{ s1 s2})) (\text{s1 } a)$   
 $\langle \text{proof} \rangle$

**theorem**  $\text{corec2}$ :  
 $\text{dtor2} (\text{corec2} \text{ s1 s2 } a) =$   
 $\text{F2map id} (\text{case\_sum id} (\text{corec1} \text{ s1 s2})) (\text{case\_sum id} (\text{corec2} \text{ s1 s2})) (\text{s2 } a)$   
 $\langle \text{proof} \rangle$

**lemma**  $\text{corec\_unique}$ :  
 $\text{F1map id} (\text{case\_sum id } f1) (\text{case\_sum id } f2) \text{ o s1} = \text{dtor1 o f1} \implies$   
 $\text{F2map id} (\text{case\_sum id } f1) (\text{case\_sum id } f2) \text{ o s2} = \text{dtor2 o f2} \implies$   
 $f1 = \text{corec1 } s1 \text{ s2} \wedge f2 = \text{corec2 } s1 \text{ s2}$   
 $\langle \text{proof} \rangle$

## 2.7 Coinduction

**lemma**  $\text{Frel\_coind}$ :  
 $\llbracket \forall a \text{ b. } \text{phi1 } a \text{ b} \implies \text{F1rel} (=) \text{ phi1 } \text{ phi2} (\text{dtor1 } a) (\text{dtor1 } b);$   
 $\forall a \text{ b. } \text{phi2 } a \text{ b} \implies \text{F2rel} (=) \text{ phi1 } \text{ phi2} (\text{dtor2 } a) (\text{dtor2 } b) \rrbracket \implies$   
 $(\text{phi1 } a1 \text{ b1} \implies a1 = b1) \wedge (\text{phi2 } a2 \text{ b2} \implies a2 = b2)$

*<proof>*

## 2.8 The Result as an BNF

**abbreviation** *JF1map* **where**

$JF1map\ u \equiv unfold1\ (F1map\ u\ id\ id\ o\ dtor1)\ (F2map\ u\ id\ id\ o\ dtor2)$

**abbreviation** *JF2map* **where**

$JF2map\ u \equiv unfold2\ (F1map\ u\ id\ id\ o\ dtor1)\ (F2map\ u\ id\ id\ o\ dtor2)$

**lemma** *JF1map*:  $dtor1\ o\ JF1map\ u = F1map\ u\ (JF1map\ u)\ (JF2map\ u)\ o\ dtor1$

*<proof>*

**lemma** *JF2map*:  $dtor2\ o\ JF2map\ u = F2map\ u\ (JF1map\ u)\ (JF2map\ u)\ o\ dtor2$

*<proof>*

**lemmas** *JF1map\_simps* =  $o\_eq\_dest[OF\ JF1map]$

**lemmas** *JF2map\_simps* =  $o\_eq\_dest[OF\ JF2map]$

**theorem** *JF1map\_id*:  $JF1map\ id = id$

*<proof>*

**theorem** *JF2map\_id*:  $JF2map\ id = id$

*<proof>*

**lemma** *JFmap\_unique*:

$[[dtor1\ o\ u = F1map\ f\ u\ v\ o\ dtor1; dtor2\ o\ v = F2map\ f\ u\ v\ o\ dtor2]] \implies$

$u = JF1map\ f \wedge v = JF2map\ f$

*<proof>*

**theorem** *JF1map\_comp*:  $JF1map\ (g\ o\ f) = JF1map\ g\ o\ JF1map\ f$

*<proof>*

**theorem** *JF2map\_comp*:  $JF2map\ (g\ o\ f) = JF2map\ g\ o\ JF2map\ f$

*<proof>*

**definition** *JFcol* **where**

$JFcol = rec\_nat\ (\%a.\ \{\},\ \%b.\ \{\})$

$(\%n\ rec.$

$\%a.\ F1set1\ (dtor1\ a) \cup$

$((\bigcup a' \in F1set2\ (dtor1\ a).\ fst\ rec\ a') \cup$

$(\bigcup a' \in F1set3\ (dtor1\ a).\ snd\ rec\ a'))$ ,

$\%b.\ F2set1\ (dtor2\ b) \cup$

$((\bigcup b' \in F2set2\ (dtor2\ b).\ fst\ rec\ b') \cup$

$(\bigcup b' \in F2set3\ (dtor2\ b).\ snd\ rec\ b'))))$

**abbreviation** *JF1col* **where**  $JF1col\ n \equiv fst\ (JFcol\ n)$

**abbreviation** *JF2col* **where**  $JF2col\ n \equiv snd\ (JFcol\ n)$

**lemmas** *JF1col\_0* =  $fun\_cong[OF\ fstI[OF\ rec\_nat\_0\_imp[OF\ JFcol\_def]]]$

**lemmas** *JF2col\_0* =  $fun\_cong[OF\ sndI[OF\ rec\_nat\_0\_imp[OF\ JFcol\_def]]]$

**lemmas** *JF1col\_Suc* =  $fun\_cong[OF\ fstI[OF\ rec\_nat\_Suc\_imp[OF\ JFcol\_def]]]$

**lemmas** *JF2col\_Suc* =  $fun\_cong[OF\ sndI[OF\ rec\_nat\_Suc\_imp[OF\ JFcol\_def]]]$

**lemma** *JFcol\_minimal*:

$[[\bigwedge a.\ F1set1\ (dtor1\ a) \subseteq K1\ a;$

$\bigwedge b.\ F2set1\ (dtor2\ b) \subseteq K2\ b;$

$\bigwedge a\ a'.\ a' \in F1set2\ (dtor1\ a) \implies K1\ a' \subseteq K1\ a;$

$\bigwedge a\ b'.\ b' \in F1set3\ (dtor1\ a) \implies K2\ b' \subseteq K1\ a;$

$\bigwedge b\ a'.\ a' \in F2set2\ (dtor2\ b) \implies K1\ a' \subseteq K2\ b;$

$\bigwedge b\ b'.\ b' \in F2set3\ (dtor2\ b) \implies K2\ b' \subseteq K2\ b]] \implies$

$\forall a\ b.\ JF1col\ n\ a \subseteq K1\ a \wedge JF2col\ n\ b \subseteq K2\ b$

*<proof>*

**lemma** *JFset\_minimal*:

$$\begin{aligned} & \llbracket \bigwedge a. F1set1 \ (d\text{tor}1 \ a) \subseteq K1 \ a; \\ & \quad \bigwedge b. F2set1 \ (d\text{tor}2 \ b) \subseteq K2 \ b; \\ & \quad \bigwedge a \ a'. \ a' \in F1set2 \ (d\text{tor}1 \ a) \implies K1 \ a' \subseteq K1 \ a; \\ & \quad \bigwedge a \ b'. \ b' \in F1set3 \ (d\text{tor}1 \ a) \implies K2 \ b' \subseteq K1 \ a; \\ & \quad \bigwedge b \ a'. \ a' \in F2set2 \ (d\text{tor}2 \ b) \implies K1 \ a' \subseteq K2 \ b; \\ & \quad \bigwedge b \ b'. \ b' \in F2set3 \ (d\text{tor}2 \ b) \implies K2 \ b' \subseteq K2 \ b \rrbracket \implies \\ & (\bigcup n. JF1col \ n \ a) \subseteq K1 \ a \wedge (\bigcup n. JF2col \ n \ b) \subseteq K2 \ b \\ & \langle \text{proof} \rangle \end{aligned}$$

**abbreviation** *JF1set where*  $JF1set \ a \equiv (\bigcup n. JF1col \ n \ a)$

**abbreviation** *JF2set where*  $JF2set \ a \equiv (\bigcup n. JF2col \ n \ a)$

**lemma** *F1set1\_incl\_JF1set*:

$$F1set1 \ (d\text{tor}1 \ a) \subseteq JF1set \ a$$

*<proof>*

**lemma** *F2set1\_incl\_JF2set*:

$$F2set1 \ (d\text{tor}2 \ a) \subseteq JF2set \ a$$

*<proof>*

**lemma** *F1set2\_JF1set\_incl\_JF1set*:

$$a' \in F1set2 \ (d\text{tor}1 \ a) \implies JF1set \ a' \subseteq JF1set \ a$$

*<proof>*

**lemma** *F1set3\_JF2set\_incl\_JF1set*:

$$a' \in F1set3 \ (d\text{tor}1 \ a) \implies JF2set \ a' \subseteq JF1set \ a$$

*<proof>*

**lemma** *F2set2\_JF1set\_incl\_JF2set*:

$$a' \in F2set2 \ (d\text{tor}2 \ a) \implies JF1set \ a' \subseteq JF2set \ a$$

*<proof>*

**lemma** *F2set3\_JF2set\_incl\_JF2set*:

$$a' \in F2set3 \ (d\text{tor}2 \ a) \implies JF2set \ a' \subseteq JF2set \ a$$

*<proof>*

**lemmas** *F1set1\_JF1set = subsetD[OF F1set1\_incl\_JF1set]*

**lemmas** *F2set1\_JF2set = subsetD[OF F2set1\_incl\_JF2set]*

**lemmas** *F1set2\_JF1set\_JF1set = subsetD[OF F1set2\_JF1set\_incl\_JF1set]*

**lemmas** *F1set3\_JF2set\_JF1set = subsetD[OF F1set3\_JF2set\_incl\_JF1set]*

**lemmas** *F2set2\_JF1set\_JF2set = subsetD[OF F2set2\_JF1set\_incl\_JF2set]*

**lemmas** *F2set3\_JF2set\_JF2set = subsetD[OF F2set3\_JF2set\_incl\_JF2set]*

**lemma** *JFset\_le*:

**fixes**  $a :: 'a \ JF1$  **and**  $b :: 'a \ JF2$

**shows**

$$\begin{aligned} & JF1set \ a \subseteq F1set1 \ (d\text{tor}1 \ a) \cup (\bigcup (JF1set \ ' \ F1set2 \ (d\text{tor}1 \ a)) \cup \bigcup (JF2set \ ' \ F1set3 \ (d\text{tor}1 \ a))) \wedge \\ & JF2set \ b \subseteq F2set1 \ (d\text{tor}2 \ b) \cup (\bigcup (JF1set \ ' \ F2set2 \ (d\text{tor}2 \ b)) \cup \bigcup (JF2set \ ' \ F2set3 \ (d\text{tor}2 \ b))) \end{aligned}$$

*<proof>*

**theorem** *JF1set\_simps*:

$$\begin{aligned} JF1set \ a &= F1set1 \ (d\text{tor}1 \ a) \cup \\ & ((\bigcup b \in F1set2 \ (d\text{tor}1 \ a). JF1set \ b) \cup \\ & (\bigcup b \in F1set3 \ (d\text{tor}1 \ a). JF2set \ b)) \end{aligned}$$

*<proof>*

**theorem** *JF2set\_simps*:

$$\begin{aligned} JF2set \ a &= F2set1 \ (d\text{tor}2 \ a) \cup \\ & ((\bigcup b \in F2set2 \ (d\text{tor}2 \ a). JF1set \ b) \cup \\ & (\bigcup b \in F2set3 \ (d\text{tor}2 \ a). JF2set \ b)) \end{aligned}$$

*<proof>*

**lemma** *JFcol\_natural*:

$\forall b1\ b2. u \text{ ' } (JF1col\ n\ b1) = JF1col\ n\ (JF1map\ u\ b1) \wedge$   
 $u \text{ ' } (JF2col\ n\ b2) = JF2col\ n\ (JF2map\ u\ b2)$   
{proof}

**theorem** *JF1set\_natural*:  $JF1set\ o\ (JF1map\ u) = image\ u\ o\ JF1set$   
{proof}

**theorem** *JF2set\_natural*:  $JF2set\ o\ (JF2map\ u) = image\ u\ o\ JF2set$   
{proof}

**theorem** *JFmap\_cong0*:

$((\forall p \in JF1set\ a. u\ p = v\ p) \longrightarrow JF1map\ u\ a = JF1map\ v\ a) \wedge$   
 $((\forall p \in JF2set\ b. u\ p = v\ p) \longrightarrow JF2map\ u\ b = JF2map\ v\ b)$   
{proof}

**lemmas** *JF1map\_cong0* =  $mp[OF\ conjunct1[OF\ JFmap\_cong0]]$

**lemmas** *JF2map\_cong0* =  $mp[OF\ conjunct2[OF\ JFmap\_cong0]]$

**lemma** *JFcol\_bd*:  $\forall (j1 :: 'a\ JF1)\ (j2 :: 'a\ JF2). |JF1col\ n\ j1| < o\ bd\_F \wedge |JF2col\ n\ j2| < o\ bd\_F$   
{proof}

**theorem** *JF1set\_bd*:  $|JF1set\ j| < o\ bd\_F$   
{proof}

**theorem** *JF2set\_bd*:  $|JF2set\ j| < o\ bd\_F$   
{proof}

**abbreviation** *JF2wit*  $\equiv\ ctor2\ wit\_F2$

**theorem** *JF2wit*:  $\bigwedge x. x \in JF2set\ JF2wit \implies False$   
{proof}

**abbreviation** *JF1wit*  $\equiv\ (\%a. ctor1\ (wit2\_F1\ a\ JF2wit))$

**theorem** *JF1wit*:  $\bigwedge x. x \in JF1set\ (JF1wit\ a) \implies x = a$   
{proof}

**context**

**fixes** *phi1* :: 'a  $\Rightarrow$  'a *JF1*  $\Rightarrow$  bool **and** *phi2* :: 'a  $\Rightarrow$  'a *JF2*  $\Rightarrow$  bool  
**begin**

**lemmas** *JFset\_induct* =

*JFset\_minimal*[of %b1. {a  $\in$  *JF1set* b1 . *phi1* a b1} %b2. {a  $\in$  *JF2set* b2 . *phi2* a b2},  
*unfolded subset\_Collect\_iff*[*OF F1set1\_incl\_JF1set*] *subset\_Collect\_iff*[*OF F2set1\_incl\_JF2set*]  
*subset\_Collect\_iff*[*OF subset\_refl*],  
*OF ballI ballI*  
*subset\_CollectI*[*OF F1set2\_JF1set\_incl\_JF1set*]  
*subset\_CollectI*[*OF F1set3\_JF2set\_incl\_JF1set*]  
*subset\_CollectI*[*OF F2set2\_JF1set\_incl\_JF2set*]  
*subset\_CollectI*[*OF F2set3\_JF2set\_incl\_JF2set*]]

**end**

{ML}

**abbreviation** *JF1in* **where** *JF1in* B  $\equiv$  {a. *JF1set* a  $\subseteq$  B}

**abbreviation** *JF2in* **where** *JF2in* B  $\equiv$  {a. *JF2set* a  $\subseteq$  B}

**definition** *JF1rel* where

$$JF1rel\ R = (BNF\_Def.Grp (JF1in (Collect (case\_prod\ R))) (JF1map\ fst)) \hat{\ } - - 1\ OO \\ (BNF\_Def.Grp (JF1in (Collect (case\_prod\ R))) (JF1map\ snd))$$

**definition** *JF2rel* where

$$JF2rel\ R = (BNF\_Def.Grp (JF2in (Collect (case\_prod\ R))) (JF2map\ fst)) \hat{\ } - - 1\ OO \\ (BNF\_Def.Grp (JF2in (Collect (case\_prod\ R))) (JF2map\ snd))$$

**lemma** *in\_JF1rel*:

$$JF1rel\ R\ x\ y \longleftrightarrow (\exists\ z. z \in JF1in (Collect (case\_prod\ R)) \wedge JF1map\ fst\ z = x \wedge JF1map\ snd\ z = y) \\ \langle proof \rangle$$

**lemma** *in\_JF2rel*:

$$JF2rel\ R\ x\ y \longleftrightarrow (\exists\ z. z \in JF2in (Collect (case\_prod\ R)) \wedge JF2map\ fst\ z = x \wedge JF2map\ snd\ z = y) \\ \langle proof \rangle$$

**lemma** *J\_rel\_coind\_ind*:

$$\llbracket \forall x\ y. R2\ x\ y \longrightarrow (f\ x\ y \in F1in (Collect (case\_prod\ R1)) (Collect (case\_prod\ R2)) (Collect (case\_prod\ R3)) \wedge \\ F1map\ fst\ fst\ fst\ (f\ x\ y) = dtor1\ x \wedge \\ F1map\ snd\ snd\ snd\ (f\ x\ y) = dtor1\ y); \\ \forall x\ y. R3\ x\ y \longrightarrow (g\ x\ y \in F2in (Collect (case\_prod\ R1)) (Collect (case\_prod\ R2)) (Collect (case\_prod\ R3)) \wedge \\ F2map\ fst\ fst\ fst\ (g\ x\ y) = dtor2\ x \wedge \\ F2map\ snd\ snd\ snd\ (g\ x\ y) = dtor2\ y) \rrbracket \Longrightarrow \\ (\forall a \in JF1set\ z1. \forall x\ y. R2\ x\ y \wedge z1 = unfold1 (case\_prod\ f) (case\_prod\ g) (x, y) \longrightarrow R1 (fst\ a) (snd\ a)) \wedge \\ (\forall a \in JF2set\ z2. \forall x\ y. R3\ x\ y \wedge z2 = unfold2 (case\_prod\ f) (case\_prod\ g) (x, y) \longrightarrow R1 (fst\ a) (snd\ a)) \\ \langle proof \rangle$$

**lemma** *J\_rel\_coind\_coind1*:

$$\llbracket \forall x\ y. R2\ x\ y \longrightarrow (f\ x\ y \in F1in (Collect (case\_prod\ R1)) (Collect (case\_prod\ R2)) (Collect (case\_prod\ R3)) \wedge \\ F1map\ fst\ fst\ fst\ (f\ x\ y) = dtor1\ x \wedge \\ F1map\ snd\ snd\ snd\ (f\ x\ y) = dtor1\ y); \\ \forall x\ y. R3\ x\ y \longrightarrow (g\ x\ y \in F2in (Collect (case\_prod\ R1)) (Collect (case\_prod\ R2)) (Collect (case\_prod\ R3)) \wedge \\ F2map\ fst\ fst\ fst\ (g\ x\ y) = dtor2\ x \wedge \\ F2map\ snd\ snd\ snd\ (g\ x\ y) = dtor2\ y) \rrbracket \Longrightarrow \\ ((\exists y. R2\ x1\ y \wedge x1' = JF1map\ fst (unfold1 (case\_prod\ f) (case\_prod\ g) (x1, y))) \longrightarrow x1' = x1) \wedge \\ ((\exists y. R3\ x2\ y \wedge x2' = JF2map\ fst (unfold2 (case\_prod\ f) (case\_prod\ g) (x2, y))) \longrightarrow x2' = x2) \\ \langle proof \rangle$$

**lemma** *J\_rel\_coind\_coind2*:

$$\llbracket \forall x\ y. R2\ x\ y \longrightarrow (f\ x\ y \in F1in (Collect (case\_prod\ R1)) (Collect (case\_prod\ R2)) (Collect (case\_prod\ R3)) \wedge \\ F1map\ fst\ fst\ fst\ (f\ x\ y) = dtor1\ x \wedge \\ F1map\ snd\ snd\ snd\ (f\ x\ y) = dtor1\ y); \\ \forall x\ y. R3\ x\ y \longrightarrow (g\ x\ y \in F2in (Collect (case\_prod\ R1)) (Collect (case\_prod\ R2)) (Collect (case\_prod\ R3)) \wedge \\ F2map\ fst\ fst\ fst\ (g\ x\ y) = dtor2\ x \wedge \\ F2map\ snd\ snd\ snd\ (g\ x\ y) = dtor2\ y) \rrbracket \Longrightarrow \\ ((\exists x. R2\ x\ y1 \wedge y1' = JF1map\ snd (unfold1 (case\_prod\ f) (case\_prod\ g) (x, y1))) \longrightarrow y1' = y1) \wedge \\ ((\exists x. R3\ x\ y2 \wedge y2' = JF2map\ snd (unfold2 (case\_prod\ f) (case\_prod\ g) (x, y2))) \longrightarrow y2' = y2) \\ \langle proof \rangle$$

**lemma** *J\_rel\_coind*:

$$\text{assumes } CIH1: \forall x2\ y2. R2\ x2\ y2 \longrightarrow F1rel\ R1\ R2\ R3\ (dtor1\ x2)\ (dtor1\ y2) \\ \text{and } CIH2: \forall x3\ y3. R3\ x3\ y3 \longrightarrow F2rel\ R1\ R2\ R3\ (dtor2\ x3)\ (dtor2\ y3) \\ \text{shows } R2 \leq JF1rel\ R1 \wedge R3 \leq JF2rel\ R1 \\ \langle proof \rangle$$

**lemma** *JF1rel\_F1rel*:  $JF1rel\ R\ a\ b \longleftrightarrow F1rel\ R\ (JF1rel\ R)\ (JF2rel\ R)\ (dtor1\ a)\ (dtor1\ b)$

$\langle proof \rangle$

**lemma** *JF2rel\_F2rel*:  $JF2rel\ R\ a\ b \longleftrightarrow F2rel\ R\ (JF1rel\ R)\ (JF2rel\ R)\ (dtor2\ a)\ (dtor2\ b)$

$\langle proof \rangle$

**lemma** *JFrel\_Comp\_le*:

$JF1rel\ R1\ OO\ JF1rel\ R2 \leq JF1rel\ (R1\ OO\ R2) \wedge JF2rel\ R1\ OO\ JF2rel\ R2 \leq JF2rel\ (R1\ OO\ R2)$   
 <proof>

**context includes** *lifting\_syntax*

**begin**

**lemma** *unfold\_transfer*:

$((S \implies F1rel\ R\ S\ T) \implies (T \implies F2rel\ R\ S\ T) \implies S \implies JF1rel\ R)$  *unfold1* *unfold1*  $\wedge$   
 $((S \implies F1rel\ R\ S\ T) \implies (T \implies F2rel\ R\ S\ T) \implies T \implies JF2rel\ R)$  *unfold2* *unfold2*  
 <proof>

**end**

<ML>

**bnf** *'a* *JF1*

*map*: *JF1map*  
*sets*: *JF1set*  
*bd*: *bd\_F*  
*wits*: *JF1wit*  
*rel*: *JF1rel*  
 <proof>

**bnf** *'a* *JF2*

*map*: *JF2map*  
*sets*: *JF2set*  
*bd*: *bd\_F*  
*wits*: *JF2wit*  
*rel*: *JF2rel*  
 <proof>

### 3 Normalized Composition of BNFs

Expected normal form: outer m-ary BNF is composed with m inner n-ary BNFs.

**unbundle** *cardinal\_syntax*

**declare**  $[[bnf\_internals]]$

**bnf-axiomatization** (*dead* *'p1*, *F1set1*: *'a1*, *F1set2*: *'a2*) *F1*

[*wits*: (*'p1*, *'a1*, *'a2*) *F1*]

**for** *map*: *F1map* *rel*: *F1rel*

**bnf-axiomatization** (*dead* *'p2*, *F2set1*: *'a1*, *F2set2*: *'a2*) *F2*

[*wits*: *'a1*  $\Rightarrow$  (*'p2*, *'a1*, *'a2*) *F2* *'a2*  $\Rightarrow$  (*'p2*, *'a1*, *'a2*) *F2*]

**for** *map*: *F2map* *rel*: *F2rel*

**bnf-axiomatization** (*dead* *'p3*, *F3set1*: *'a1*, *F3set2*: *'a2*) *F3*

[*wits*: *'a1*  $\Rightarrow$  *'a2*  $\Rightarrow$  (*'p3*, *'a1*, *'a2*) *F3*]

**for** *map*: *F3map* *rel*: *F3rel*

**bnf-axiomatization** (*dead* *'p*, *Gset1*: *'b1*, *Gset2*: *'b2*, *Gset3*: *'b3*) *G*

[*wits*: *'b1*  $\Rightarrow$  *'b3*  $\Rightarrow$  (*'p*, *'b1*, *'b2*, *'b3*) *G* *'b2*  $\Rightarrow$  *'b3*  $\Rightarrow$  (*'p*, *'b1*, *'b2*, *'b3*) *G*]

**for** *map*: *Gmap* *rel*: *Grel*

**type-synonym** (*'p1*, *'p2*, *'p3*, *'p*, *'a1*, *'a2*) *H* =

(*'p*, (*'p1*, *'a1*, *'a2*) *F1*, (*'p2*, *'a1*, *'a2*) *F2*, (*'p3*, *'a1*, *'a2*) *F3*) *G*

**type-synonym** (*'p1*, *'p2*, *'p3*, *'p*) *Hbd\_type* =

(*'p1* *bd\_type\_F1* + *'p2* *bd\_type\_F2* + *'p3* *bd\_type\_F3*)  $\times$  *'p* *bd\_type\_G*

**abbreviation** *F1in* **where** *F1in* *A1* *A2*  $\equiv \{x. F1set1\ x \subseteq A1 \wedge F1set2\ x \subseteq A2\}$

**abbreviation** *F2in* **where** *F2in* *A1* *A2*  $\equiv \{x. F2set1\ x \subseteq A1 \wedge F2set2\ x \subseteq A2\}$

**abbreviation** *F3in* **where** *F3in* *A1* *A2*  $\equiv \{x. F3set1\ x \subseteq A1 \wedge F3set2\ x \subseteq A2\}$

**abbreviation** *Gin* **where** *Gin* *A1* *A2* *A3*  $\equiv \{x. Gset1\ x \subseteq A1 \wedge Gset2\ x \subseteq A2 \wedge Gset3\ x \subseteq A3\}$

**abbreviation** *Gset* where

$$Gset \equiv BNF\_Def.collect \{Gset1, Gset2, Gset3\}$$

**abbreviation** *Hmap* :: ('a1  $\Rightarrow$  'b1)  $\Rightarrow$  ('a2  $\Rightarrow$  'b2)  $\Rightarrow$   
( 'p1, 'p2, 'p3, 'p, 'a1, 'a2) H  $\Rightarrow$  ('p1, 'p2, 'p3, 'p, 'b1, 'b2) H where  
*Hmap* f g  $\equiv$  *Gmap* (*F1map* f g) (*F2map* f g) (*F3map* f g)

**abbreviation** *Hset1* :: ('p1, 'p2, 'p3, 'p, 'a1, 'a2) H  $\Rightarrow$  'a1 set where  
*Hset1*  $\equiv$  *Union* o *Gset* o *Gmap* *F1set1* *F2set1* *F3set1*

**abbreviation** *Hset2* :: ('p1, 'p2, 'p3, 'p, 'a1, 'a2) H  $\Rightarrow$  'a2 set where  
*Hset2*  $\equiv$  *Union* o *Gset* o *Gmap* *F1set2* *F2set2* *F3set2*

**lemma** *Hset1\_alt*:

$$Hset1 = Union \ o \ BNF\_Def.collect \ {image \ F1set1 \ o \ Gset1, image \ F2set1 \ o \ Gset2, image \ F3set1 \ o \ Gset3\}$$

*<proof>*

**lemma** *Hset2\_alt*:

$$Hset2 = Union \ o \ BNF\_Def.collect \ {image \ F1set2 \ o \ Gset1, image \ F2set2 \ o \ Gset2, image \ F3set2 \ o \ Gset3\}$$

*<proof>*

**abbreviation** *Hbd* where

$$Hbd \equiv (bd\_F1 +c \ bd\_F2 +c \ bd\_F3) *c \ bd\_G$$

**theorem** *Hmap\_id*: *Hmap* id id = id

*<proof>*

**theorem** *Hmap\_comp*: *Hmap* (f1 o g1) (f2 o g2) = *Hmap* f1 f2 o *Hmap* g1 g2

*<proof>*

**theorem** *Hmap\_cong*:  $\llbracket \bigwedge z. z \in Hset1 \ x \implies f1 \ z = g1 \ z; \bigwedge z. z \in Hset2 \ x \implies f2 \ z = g2 \ z \rrbracket \implies$

$$Hmap \ f1 \ f2 \ x = Hmap \ g1 \ g2 \ x$$

*<proof>*

**theorem** *Hset1\_natural*: *Hset1* o *Hmap* f1 f2 = *image* f1 o *Hset1*

*<proof>*

**theorem** *Hset2\_natural*: *Hset2* o *Hmap* f1 f2 = *image* f2 o *Hset2*

*<proof>*

**theorem** *Hbd\_card\_order*: *card\_order* *Hbd*

*<proof>*

**theorem** *Hbd\_cinfinite*: *cinfinite* *Hbd*

*<proof>*

**theorem** *Hbd\_regularCard*: *regularCard* *Hbd*

*<proof>*

**theorem** *Hset1\_bd*:  $|Hset1 \ (x :: ('p1, 'p2, 'p3, 'p, 'a1, 'a2) \ H) | < o$

$$(Hbd :: ('p1, 'p2, 'p3, 'p) \ Hbd\_type \ rel)$$

*<proof>*

**theorem** *Hset2\_bd*:  $|Hset2 \ (x :: ('p1, 'p2, 'p3, 'p, 'a1, 'a2) \ H) | < o$

$$(Hbd :: ('p1, 'p2, 'p3, 'p) \ Hbd\_type \ rel)$$

*<proof>*

**abbreviation** *Hin* where *Hin* A1 A2  $\equiv \{x. Hset1 \ x \subseteq A1 \wedge Hset2 \ x \subseteq A2\}$

**lemma** *Hin\_alt*: *Hin* A1 A2 = *Gin* (*F1in* A1 A2) (*F2in* A1 A2) (*F3in* A1 A2)

*<proof>*

**definition** *Hwit1* where *Hwit1* b c = *wit1\_G* *wit\_F1* (*wit\_F3* b c)

**definition** *Hwit21* **where** *Hwit21* *b c* = *wit2\_G* (*wit1\_F2* *b*) (*wit\_F3* *b c*)

**definition** *Hwit22* **where** *Hwit22* *b c* = *wit2\_G* (*wit2\_F2* *c*) (*wit\_F3* *b c*)

**lemma** *Hwit1*:

$\bigwedge x. x \in \text{Hset1 } (\text{Hwit1 } b c) \implies x = b$   
 $\bigwedge x. x \in \text{Hset2 } (\text{Hwit1 } b c) \implies x = c$   
(*proof*)

**lemma** *Hwit21*:

$\bigwedge x. x \in \text{Hset1 } (\text{Hwit21 } b c) \implies x = b$   
 $\bigwedge x. x \in \text{Hset2 } (\text{Hwit21 } b c) \implies x = c$   
(*proof*)

**lemma** *Hwit22*:

$\bigwedge x. x \in \text{Hset1 } (\text{Hwit22 } b c) \implies x = b$   
 $\bigwedge x. x \in \text{Hset2 } (\text{Hwit22 } b c) \implies x = c$   
(*proof*)

**lemma** *Grel\_cong*:  $\llbracket R1 = S1; R2 = S2; R3 = S3 \rrbracket \implies \text{Grel } R1 R2 R3 = \text{Grel } S1 S2 S3$

(*proof*)

**definition** *Hrel* **where**

*Hrel* *R1 R2* = (*BNF\_Def.Grp* (*Hin* (*Collect* (*case\_prod* *R1*)) (*Collect* (*case\_prod* *R2*)))) (*Hmap* *fst* *fst*)<sup>^--1</sup> *OO*  
(*BNF\_Def.Grp* (*Hin* (*Collect* (*case\_prod* *R1*)) (*Collect* (*case\_prod* *R2*)))) (*Hmap* *snd* *snd*)

**lemmas** *Hrel\_unfold* = *trans*[*OF* *Hrel\_def* *trans*[*OF* *OO\_Grp\_cong*[*OF* *Hin\_alt*

*trans*[*OF* *arg\_cong2*[*of* *\_\_ \_\_ \_\_* *relcompp*, *OF* *trans*[*OF* *arg\_cong*[*of* *\_\_ \_\_* *conversep*, *OF* *sym*[*OF* *G.rel\_Grp*]]

*G.rel\_conversep*[*symmetric*]] *sym*[*OF* *G.rel\_Grp*]]  
*trans*[*OF* *G.rel\_compp*[*symmetric*] *Grel\_cong*[*OF* *sym*[*OF* *F1.rel\_compp\_Grp*] *sym*[*OF* *F2.rel\_compp\_Grp*]

*sym*[*OF* *F3.rel\_compp\_Grp*]]]]]]

**bnf** *H*: (*'p1*, *'p2*, *'p3*, *'p*, *'a1*, *'a2*) *H*

*map*: *Hmap*

*sets*: *Hset1* *Hset2*

*bd*: *Hbd* :: (*'p1*, *'p2*, *'p3*, *'p*) *Hbd\_type* *rel*

*rel*: *Hrel*

(*proof*)

## 4 Removing Live Variables

**unbundle** *cardinal\_syntax*

**declare** [[*bnf\_internals*]]

**bnf-axiomatization** (*dead* *'p*, *Fset1*: *'a1*, *Fset2*: *'a2*, *Fset3*: *'a3*) *F* **for** *map*: *Fmap* *rel*: *Frel*

**abbreviation** *F1map* :: (*'a2*  $\Rightarrow$  *'b2*)  $\Rightarrow$  (*'a3*  $\Rightarrow$  *'b3*)  $\Rightarrow$  (*'p*, *'a1*, *'a2*, *'a3*) *F*  $\Rightarrow$  (*'p*, *'a1*, *'b2*, *'b3*) *F* **where**

*F1map*  $\equiv$  *Fmap* *id*

**abbreviation** *F2map* :: (*'a3*  $\Rightarrow$  *'b3*)  $\Rightarrow$  (*'p*, *'a1*, *'a2*, *'a3*) *F*  $\Rightarrow$  (*'p*, *'a1*, *'a2*, *'b3*) *F* **where**

*F2map*  $\equiv$  *Fmap* *id* *id*

**abbreviation** *F1set1*  $\equiv$  *Fset2*

**abbreviation** *F1set2*  $\equiv$  *Fset3*

**abbreviation** *F2set*  $\equiv$  *Fset3*

**theorem** *F1map\_id*: *F1map* *id* *id* = *id*

(*proof*)

**theorem** *F2map\_id*: *F2map* *id* = *id*

*<proof>*

**theorem** *F1map\_comp*:  $F1map (f1 \circ g1) (f2 \circ g2) = F1map f1 f2 \circ F1map g1 g2$   
*<proof>*

**theorem** *F2map\_comp*:  $F2map (f \circ g) = F2map f \circ F2map g$   
*<proof>*

**theorem** *F1map\_cong*:  $\llbracket \bigwedge z. z \in F1set1 \ x \implies f1 \ z = g1 \ z; \bigwedge z. z \in F1set2 \ x \implies f2 \ z = g2 \ z \rrbracket$   
 $\implies F1map f1 f2 \ x = F1map g1 g2 \ x$   
*<proof>*

**theorem** *F2map\_cong*:  $\llbracket \bigwedge z. z \in F2set \ x \implies f \ z = g \ z \rrbracket \implies F2map f \ x = F2map g \ x$   
*<proof>*

**theorem** *F1set1\_natural*:  $F1set1 \circ F1map f1 f2 = image f1 \circ F1set1$   
*<proof>*

**theorem** *F1set2\_natural*:  $F1set2 \circ F1map f1 f2 = image f2 \circ F1set2$   
*<proof>*

**theorem** *F2set\_natural*:  $F2set \circ F2map f = image f \circ F2set$   
*<proof>*

**abbreviation** *Fin* :: 'a1 set  $\Rightarrow$  'a2 set  $\Rightarrow$  'a3 set  $\Rightarrow$  (('p, 'a1, 'a2, 'a3) F) set **where**  
 $Fin \ A1 \ A2 \ A3 \equiv \{x. Fset1 \ x \subseteq A1 \ \wedge \ Fset2 \ x \subseteq A2 \ \wedge \ Fset3 \ x \subseteq A3\}$

**abbreviation** *F1in* :: 'a2 set  $\Rightarrow$  'a3 set  $\Rightarrow$  (('p, 'a1, 'a2, 'a3) F) set **where**  
 $F1in \ A1 \ A2 \equiv \{x. F1set1 \ x \subseteq A1 \ \wedge \ F1set2 \ x \subseteq A2\}$

**lemma** *F1in\_alt*:  $F1in \ A2 \ A3 = Fin \ UNIV \ A2 \ A3$   
*<proof>*

**abbreviation** *F2in* :: 'a3 set  $\Rightarrow$  (('p, 'a1, 'a2, 'a3) F) set **where**  
 $F2in \ A \equiv \{x. F2set \ x \subseteq A\}$

**lemma** *F2in\_alt*:  $F2in \ A3 = Fin \ UNIV \ UNIV \ A3$   
*<proof>*

**lemma** *Frel\_cong*:  $\llbracket R1 = S1; R2 = S2; R3 = S3 \rrbracket \implies Frel \ R1 \ R2 \ R3 = Frel \ S1 \ S2 \ S3$   
*<proof>*

**definition** *F1rel* **where**

$F1rel \ R1 \ R2 = (BNF\_Def.Grp (F1in (Collect (case\_prod \ R1))) (Collect (case\_prod \ R2))) (F1map \ fst \ fst)^{\hat{-}-1}$   
 $OO$   
 $(BNF\_Def.Grp (F1in (Collect (case\_prod \ R1))) (Collect (case\_prod \ R2))) (F1map \ snd \ snd))$

**lemmas** *F1rel\_unfold* =  $trans[OF \ F1rel\_def \ trans[OF \ OO\_Grp\_cong[OF \ F1in\_alt]$   
 $trans[OF \ arg\_cong2[of \ \_ \ \_ \ \_ \ relcompp, \ OF \ trans[OF \ arg\_cong[of \ \_ \ \_ \ conversep, \ OF \ sym[OF \ F.rel\_Grp]]$   
 $F.rel\_conversep[symmetric]] \ sym[OF \ F.rel\_Grp]]$   
 $trans[OF \ F.rel\_compp[symmetric] \ Frel\_cong[OF \ trans[OF \ Grp\_UNIV\_id[OF \ refl] \ eq\_alt[symmetric]] \ Grp\_fst\_snd$   
 $Grp\_fst\_snd]]]]]$

**definition** *F2rel* **where**

$F2rel \ R1 = (BNF\_Def.Grp (F2in (Collect (case\_prod \ R1))) (F2map \ fst))^{\hat{-}-1} \ OO$   
 $(BNF\_Def.Grp (F2in (Collect (case\_prod \ R1))) (F2map \ snd))$

**lemmas** *F2rel\_unfold* =  $trans[OF \ F2rel\_def \ trans[OF \ OO\_Grp\_cong[OF \ F2in\_alt]$   
 $trans[OF \ arg\_cong2[of \ \_ \ \_ \ \_ \ relcompp, \ OF \ trans[OF \ arg\_cong[of \ \_ \ \_ \ conversep, \ OF \ sym[OF \ F.rel\_Grp]]$   
 $F.rel\_conversep[symmetric]] \ sym[OF \ F.rel\_Grp]]$   
 $trans[OF \ F.rel\_compp[symmetric] \ Frel\_cong[OF \ trans[OF \ Grp\_UNIV\_id[OF \ refl] \ eq\_alt[symmetric]] \ trans[OF$   
 $Grp\_UNIV\_id[OF \ refl] \ eq\_alt[symmetric]] \ Grp\_fst\_snd]]]]]$

**bnf F1:** ('p, 'a1, 'a2, 'a3) F  
 map: F1map  
 sets: F1set1 F1set2  
 bd: bd\_F :: ('p bd\_type\_F) rel  
 rel: F1rel  
 ⟨proof⟩

**bnf F2:** ('p, 'a1, 'a2, 'a3) F  
 map: F2map  
 sets: F2set  
 bd: bd\_F :: ('p bd\_type\_F) rel  
 rel: F2rel  
 ⟨proof⟩

## 5 Adding New Live Variables

**unbundle** cardinal\_syntax

**declare** [[bnf\_internals]]

**bnf-axiomatization** (dead 'p, Fset1: 'a1, Fset2: 'a2) F

[wits: 'a1 ⇒ 'a2 ⇒ ('p, 'a1, 'a2) F]

**for** map: Fmap rel: Frel

**type-synonym** ('p, 'a1, 'a2, 'a3, 'a4) F' = ('p, 'a3, 'a4) F

**abbreviation** F'map :: ('a1 ⇒ 'b1) ⇒ ('a2 ⇒ 'b2) ⇒ ('a3 ⇒ 'b3) ⇒ ('a4 ⇒ 'b4) ⇒ ('p, 'a1, 'a2, 'a3, 'a4) F' ⇒ ('p, 'b1, 'b2, 'b3, 'b4) F' **where**  
 F'map f1 f2 f3 f4 ≡ Fmap f3 f4

**abbreviation** F'set1 :: ('p, 'a1, 'a2, 'a3, 'a4) F' ⇒ 'a1 set **where**  
 F'set1 ≡ λ\_. {}

**abbreviation** F'set2 :: ('p, 'a1, 'a2, 'a3, 'a4) F' ⇒ 'a2 set **where**  
 F'set2 ≡ λ\_. {}

**abbreviation** F'set3 :: ('p, 'a1, 'a2, 'a3, 'a4) F' ⇒ 'a3 set **where**  
 F'set3 ≡ Fset1

**abbreviation** F'set4 :: ('p, 'a1, 'a2, 'a3, 'a4) F' ⇒ 'a4 set **where**  
 F'set4 ≡ Fset2

**abbreviation** F'bd **where**  
 F'bd ≡ bd\_F

**theorem** F'map\_id: F'map id id id id = id  
 ⟨proof⟩

**theorem** F'map\_comp:

F'map (f1 o g1) (f2 o g2) (f3 o g3) (f4 o g4) = F'map f1 f2 f3 f4 o F'map g1 g2 g3 g4  
 ⟨proof⟩

**theorem** F'map\_cong:

[[Λz. z ∈ F'set1 x ⇒ f1 z = g1 z; Λz. z ∈ F'set2 x ⇒ f2 z = g2 z;  
 Λz. z ∈ F'set3 x ⇒ f3 z = g3 z; Λz. z ∈ F'set4 x ⇒ f4 z = g4 z]  
 ⇒ F'map f1 f2 f3 f4 x = F'map g1 g2 g3 g4 x  
 ⟨proof⟩

**theorem** F'set1\_natural: F'set1 o F'map f1 f2 f3 f4 = image f1 o F'set1  
 ⟨proof⟩

**theorem** F'set2\_natural: F'set2 o F'map f1 f2 f3 f4 = image f2 o F'set2  
 ⟨proof⟩

**theorem**  $F'set3\_natural$ :  $F'set3 \circ F'map\ f1\ f2\ f3\ f4 = image\ f3 \circ F'set3$   
 ⟨proof⟩

**theorem**  $F'set4\_natural$ :  $F'set4 \circ F'map\ f1\ f2\ f3\ f4 = image\ f4 \circ F'set4$   
 ⟨proof⟩

**theorem**  $F'bd\_card\_order$ :  $card\_order\ bd\_F$   
 ⟨proof⟩

**theorem**  $F'bd\_cinfinit$ :  $cinfinit\ bd\_F$   
 ⟨proof⟩

**theorem**  $F'bd\_regularCard$ :  $regularCard\ bd\_F$   
 ⟨proof⟩

**theorem**  $F'set1\_bd$ :  $|F'set1\ x| < o\ F'bd$   
 ⟨proof⟩

**theorem**  $F'set2\_bd$ :  $|F'set2\ x| < o\ F'bd$   
 ⟨proof⟩

**theorem**  $F'set3\_bd$ :  $|F'set3\ (x :: ('c, 'a, 'd)\ F)| < o\ (F'bd :: 'c\ bd\_type\_F\ rel)$   
 ⟨proof⟩

**theorem**  $F'set4\_bd$ :  $|F'set4\ (x :: ('c, 'a, 'd)\ F)| < o\ (F'bd :: 'c\ bd\_type\_F\ rel)$   
 ⟨proof⟩

**abbreviation**  $F'in :: 'a1\ set \Rightarrow 'a2\ set \Rightarrow 'a3\ set \Rightarrow 'a4\ set \Rightarrow ((p, 'a1, 'a2, 'a3, 'a4)\ F)\ set$  **where**  
 $F'in\ A1\ A2\ A3\ A4 \equiv \{x.\ F'set1\ x \subseteq A1 \wedge F'set2\ x \subseteq A2 \wedge F'set3\ x \subseteq A3 \wedge F'set4\ x \subseteq A4\}$

**definition**  $F'rel$  **where**

$F'rel\ R1\ R2\ R3\ R4 = (BNF\_Def.Grp\ (F'in\ (Collect\ (case\_prod\ R1))\ (Collect\ (case\_prod\ R2))\ (Collect\ (case\_prod\ R3))\ (Collect\ (case\_prod\ R4))))\ (F'map\ fst\ fst\ fst\ fst)^{-1}\ OO$   
 $(BNF\_Def.Grp\ (F'in\ (Collect\ (case\_prod\ R1))\ (Collect\ (case\_prod\ R2))\ (Collect\ (case\_prod\ R3))\ (Collect\ (case\_prod\ R4))))\ (F'map\ snd\ snd\ snd\ snd)$

**lemmas**  $F'rel\_unfold = trans[OF\ F'rel\_def[unfolded\ eqTrueI[OF\ empty\_subsetI]\ simp\_thms(22)]\ trans[OF\ OO\_Grp\_cong[OF\ refl]\ sym[OF\ F.rel\_compp\_Grp]]]$

**bnf**  $F'$ :  $(p, 'a1, 'a2, 'a3, 'a4)\ F'$   
 map:  $F'map$   
 sets:  $F'set1\ F'set2\ F'set3\ F'set4$   
 bd:  $F'bd :: 'p\ bd\_type\_F\ rel$   
 wits:  $wit\_F$   
 rel:  $F'rel$   
 plugins del: *lifting transfer*  
 ⟨proof⟩

## 6 Changing the Order of Live Variables

**unbundle**  $cardinal\_syntax$

**declare**  $[[bnf\_internals]]$

**bnf-axiomatization**  $(dead\ 'p, Fset1: 'a1, Fset2: 'a2, Fset3: 'a3)\ F$  **for** map:  $Fmap$  rel:  $Frel$

**type-synonym**  $(p, 'a1, 'a2, 'a3)\ F' = (p, 'a3, 'a1, 'a2)\ F$

**abbreviation**  $Fin :: 'a1\ set \Rightarrow 'a2\ set \Rightarrow 'a3\ set \Rightarrow ((p, 'a1, 'a2, 'a3)\ F)\ set$  **where**  
 $Fin\ A1\ A2\ A3 \equiv \{x.\ Fset1\ x \subseteq A1 \wedge Fset2\ x \subseteq A2 \wedge Fset3\ x \subseteq A3\}$

**abbreviation**  $F'map :: ('a1 \Rightarrow 'b1) \Rightarrow ('a2 \Rightarrow 'b2) \Rightarrow ('a3 \Rightarrow 'b3) \Rightarrow (p, 'a1, 'a2, 'a3)\ F' \Rightarrow (p, 'b1, 'b2, 'b3)\ F'$  **where**

$F'map\ f\ g\ h \equiv Fmap\ h\ f\ g$

**abbreviation**  $F'set1 :: ('p, 'a1, 'a2, 'a3) F' \Rightarrow 'a1 \text{ set}$  **where**  
 $F'set1 \equiv Fset2$

**abbreviation**  $F'set2 :: ('p, 'a1, 'a2, 'a3) F' \Rightarrow 'a2 \text{ set}$  **where**  
 $F'set2 \equiv Fset3$

**abbreviation**  $F'set3 :: ('p, 'a1, 'a2, 'a3) F' \Rightarrow 'a3 \text{ set}$  **where**  
 $F'set3 \equiv Fset1$

**abbreviation**  $F'bd$  **where**  
 $F'bd \equiv bd\_F$

**theorem**  $F'map\_id: F'map \text{ id id id} = \text{id}$   
 $\langle \text{proof} \rangle$

**theorem**  $F'map\_comp: F'map (f1 \circ g1) (f2 \circ g2) (f3 \circ g3) = F'map f1 f2 f3 \circ F'map g1 g2 g3$   
 $\langle \text{proof} \rangle$

**theorem**  $F'map\_cong: [\bigwedge z. z \in F'set1 \ x \Longrightarrow f1 \ z = g1 \ z; \bigwedge z. z \in F'set2 \ x \Longrightarrow f2 \ z = g2 \ z; \bigwedge z. z \in F'set3 \ x \Longrightarrow f3 \ z = g3 \ z]$   
 $\Longrightarrow F'map f1 f2 f3 \ x = F'map g1 g2 g3 \ x$   
 $\langle \text{proof} \rangle$

**theorem**  $F'set1\_natural: F'set1 \circ F'map f1 f2 f3 = \text{image } f1 \circ F'set1$   
 $\langle \text{proof} \rangle$

**theorem**  $F'set2\_natural: F'set2 \circ F'map f1 f2 f3 = \text{image } f2 \circ F'set2$   
 $\langle \text{proof} \rangle$

**theorem**  $F'set3\_natural: F'set3 \circ F'map f1 f2 f3 = \text{image } f3 \circ F'set3$   
 $\langle \text{proof} \rangle$

**theorem**  $F'bd\_card\_order: \text{card\_order } F'bd$   
 $\langle \text{proof} \rangle$

**theorem**  $F'bd\_cinfinte: \text{cinfinte } F'bd$   
 $\langle \text{proof} \rangle$

**theorem**  $F'bd\_regularCard: \text{regularCard } F'bd$   
 $\langle \text{proof} \rangle$

**theorem**  $F'set1\_bd: |F'set1 (x :: ('c, 'a, 'b, 'd) F)| < o (F'bd :: 'c \text{ bd\_type\_} F \text{ rel})$   
 $\langle \text{proof} \rangle$

**theorem**  $F'set2\_bd: |F'set2 (x :: ('c, 'a, 'b, 'd) F)| < o (F'bd :: 'c \text{ bd\_type\_} F \text{ rel})$   
 $\langle \text{proof} \rangle$

**theorem**  $F'set3\_bd: |F'set3 (x :: ('c, 'a, 'b, 'd) F)| < o (F'bd :: 'c \text{ bd\_type\_} F \text{ rel})$   
 $\langle \text{proof} \rangle$

**abbreviation**  $F'in :: 'a1 \text{ set} \Rightarrow 'a2 \text{ set} \Rightarrow 'a3 \text{ set} \Rightarrow ((p, 'a1, 'a2, 'a3) F') \text{ set}$  **where**  
 $F'in \ A1 \ A2 \ A3 \equiv \{x. F'set1 \ x \subseteq A1 \wedge F'set2 \ x \subseteq A2 \wedge F'set3 \ x \subseteq A3\}$

**lemma**  $F'in\_alt: F'in \ A1 \ A2 \ A3 = F'in \ A3 \ A1 \ A2$   
 $\langle \text{proof} \rangle$

**definition**  $F'rel$  **where**

$F'rel \ R1 \ R2 \ R3 = (\text{BNF\_Def.Grp } (F'in \ (\text{Collect } (\text{case\_prod } R1)) \ (\text{Collect } (\text{case\_prod } R2)) \ (\text{Collect } (\text{case\_prod } R3)))) \ (F'map \ \text{fst} \ \text{fst} \ \text{fst}) \hat{\ } - - 1 \ \text{OO}$   
 $(\text{BNF\_Def.Grp } (F'in \ (\text{Collect } (\text{case\_prod } R1)) \ (\text{Collect } (\text{case\_prod } R2)) \ (\text{Collect } (\text{case\_prod } R3)))) \ (F'map \ \text{snd} \ \text{snd} \ \text{snd}))$

lemmas  $F'_{rel\_unfold} = trans[OF F'_{rel\_def} trans[OF OO\_Grp\_cong[OF F'_{in\_alt}] sym[OF F_{rel\_compp\_Grp}]]]$

**bnf**  $F'$ : ( $'p, 'a1, 'a2, 'a3$ )  $F'$   
 map:  $F'_{map}$   
 sets:  $F'_{set1} F'_{set2} F'_{set3}$   
 bd:  $F'_{bd} :: 'p \text{ bd\_type\_} F \text{ rel}$   
 rel:  $F'_{rel}$   
 (proof)

## 7 Mutual View on Nested Datatypes

**notation**  $BNF\_Def.convolve (\langle \_ , \_ \rangle)$

**declare**  $[[bnf\_internals]]$

**declare**  $[[typedef\_overloaded]]$

**bnf-axiomatization** ( $'a, 'b$ )  $F0$  [wits:  $'a \Rightarrow ('a, 'b) F0$ ]

**bnf-axiomatization** ( $'a, 'b$ )  $G0$  [wits:  $'a \Rightarrow ('a, 'b) G0$ ]

### 7.1 Nested Definition

**datatype**  $'a F = CF ('a, 'a F) F0$

**datatype**  $'a G = CG ('a, ('a G) F) G0$

**type-synonym** ( $'b, 'c$ )  $F\_pre\_F = ('c, 'b) F0$

**type-synonym** ( $'c, 'a$ )  $G\_pre\_G = ('a, 'c F) G0$

**term**  $ctor\_fold\_F :: (('b, 'c) F\_pre\_F \Rightarrow 'b) \Rightarrow 'c F \Rightarrow 'b$

**term**  $ctor\_fold\_G :: (('c, 'a) G\_pre\_G \Rightarrow 'c) \Rightarrow 'a G \Rightarrow 'c$

**term**  $ctor\_rec\_F :: (('c F \times 'b, 'c) F\_pre\_F \Rightarrow 'b) \Rightarrow 'c F \Rightarrow 'b$

**term**  $ctor\_rec\_G :: (('a G \times 'c, 'a) G\_pre\_G \Rightarrow 'c) \Rightarrow 'a G \Rightarrow 'c$

**thm**  $F.ctor\_rel\_induct$

**thm**  $G.ctor\_rel\_induct[unfolded rel\_pre\_G\_def id\_apply]$

### 7.2 Isomorphic Mutual Definition

**datatype**  $'a G_M = CG ('a, 'a GF_M) G0$

**and**  $'a GF_M = CF ('a G_M, 'a GF_M) F0$

**type-synonym** ( $'b, 'c$ )  $GF_M\_pre\_GF_M = ('c, 'b) F0$

**type-synonym** ( $'c, 'a$ )  $G_M\_pre\_G_M = ('a, 'c) G0$

**term**  $ctor\_fold\_G_M :: (('c, 'a) G_M\_pre\_G_M \Rightarrow 'b) \Rightarrow (('c, 'b) GF_M\_pre\_GF_M \Rightarrow 'c) \Rightarrow 'a G_M \Rightarrow 'b$

**term**  $ctor\_fold\_GF_M :: (('c, 'a) G_M\_pre\_G_M \Rightarrow 'b) \Rightarrow (('c, 'b) GF_M\_pre\_GF_M \Rightarrow 'c) \Rightarrow 'a GF_M \Rightarrow 'c$

**term**  $ctor\_rec\_G_M :: (('a GF_M \times 'c, 'a) G_M\_pre\_G_M \Rightarrow 'b) \Rightarrow (('a GF_M \times 'c, 'a G_M \times 'b) GF_M\_pre\_GF_M \Rightarrow 'c) \Rightarrow 'a G_M \Rightarrow 'b$

**term**  $ctor\_rec\_GF_M :: (('a GF_M \times 'c, 'a) G_M\_pre\_G_M \Rightarrow 'b) \Rightarrow (('a GF_M \times 'c, 'a G_M \times 'b) GF_M\_pre\_GF_M \Rightarrow 'c) \Rightarrow 'a GF_M \Rightarrow 'c$

**thm**  $G_M\_GF_M.ctor\_rel\_induct[unfolded rel\_pre\_G_M\_def rel\_pre\_GF_M\_def]$

### 7.3 Mutualization

#### 7.3.1 Iterators

**definition**  $n2m\_ctor\_fold\_G :: (('c, 'a) G_M\_pre\_G_M \Rightarrow 'b) \Rightarrow (('c, 'b) GF_M\_pre\_GF_M \Rightarrow 'c) \Rightarrow 'a G \Rightarrow 'b$

**where**  $n2m\_ctor\_fold\_G \ s1 \ s2 = ctor\_fold\_G \ (s1 \ o$

$map\_pre\_G_M \ id \ (id :: unit \Rightarrow unit) \ (ctor\_fold\_F \ (s2 \ o \ BNF\_Composition.id\_bnf \ o \ BNF\_Composition.id\_bnf))$   
 $\ o \ BNF\_Composition.id\_bnf \ o \ BNF\_Composition.id\_bnf)$

**definition**  $n2m\_ctor\_fold\_G\_F :: (('c, 'a) G_M\_pre\_G_M \Rightarrow 'b) \Rightarrow (('c, 'b) GF_M\_pre\_GF_M \Rightarrow 'c) \Rightarrow 'a \ G \ F \Rightarrow 'c$

where  $n2m\_ctor\_fold\_G\_F\ s1\ s2 = ctor\_fold\_F\ (s2\ o\ map\_pre\_GF_M\ (id\ ::\ unit\ \Rightarrow\ unit))\ (n2m\_ctor\_fold\_G\ s1\ s2)\ id\ o\ BNF\_Composition.id\_bnf\ o\ BNF\_Composition.id\_bnf$

**lemma**  $G\_ctor\_o\_fold$ :  $ctor\_fold\_G\ s\ o\ ctor\_G = s\ o\ map\_pre\_G\ id\ (ctor\_fold\_G\ s)$   
 ⟨proof⟩

**lemma**  $F\_ctor\_o\_fold$ :  $ctor\_fold\_F\ s\ o\ ctor\_F = s\ o\ map\_pre\_F\ id\ (ctor\_fold\_F\ s)$   
 ⟨proof⟩

**lemma**  $G\_ctor\_o\_rec$ :  $ctor\_rec\_G\ s\ o\ ctor\_G = s\ o\ map\_pre\_G\ id\ (BNF\_Def.convolve\ id\ (ctor\_rec\_G\ s))$   
 ⟨proof⟩

**lemma**  $F\_ctor\_o\_rec$ :  $ctor\_rec\_F\ s\ o\ ctor\_F = s\ o\ map\_pre\_F\ id\ (BNF\_Def.convolve\ id\ (ctor\_rec\_F\ s))$   
 ⟨proof⟩

**lemma**  $n2m\_ctor\_fold\_G$ :  
 $n2m\_ctor\_fold\_G\ s1\ s2\ o\ ctor\_G = s1\ o\ map\_pre\_G_M\ id\ id\ (n2m\_ctor\_fold\_G\_F\ s1\ s2)\ o\ BNF\_Composition.id\_bnf\ o\ BNF\_Composition.id\_bnf$   
 ⟨proof⟩

**lemma**  $n2m\_ctor\_fold\_G\_F$ :  
 $n2m\_ctor\_fold\_G\_F\ s1\ s2\ o\ ctor\_F = s2\ o\ map\_pre\_GF_M\ id\ (n2m\_ctor\_fold\_G\ s1\ s2)\ (n2m\_ctor\_fold\_G\_F\ s1\ s2)\ o\ BNF\_Composition.id\_bnf\ o\ BNF\_Composition.id\_bnf$   
 ⟨proof⟩

### 7.3.2 Recursors

**definition**  $n2m\_ctor\_rec\_G\ ::$   
 $((a\ G\ F \times 'c, 'a)\ G_M\_pre\_G_M \Rightarrow 'b) \Rightarrow ((a\ G\ F \times 'c, 'a\ G \times 'b)\ GF_M\_pre\_GF_M \Rightarrow 'c) \Rightarrow 'a\ G \Rightarrow 'b$   
 where  $n2m\_ctor\_rec\_G\ s1\ s2 =$   
 $ctor\_rec\_G\ (s1\ o$   
 $map\_pre\_G_M\ id\ (id\ ::\ unit\ \Rightarrow\ unit))$   
 $(BNF\_Def.convolve\ (map\_F\ fst)\ (ctor\_rec\_F\ (s2\ o\ map\_pre\_GF_M\ (id\ ::\ unit\ \Rightarrow\ unit))\ id\ (map\_prod\ (map\_F\ fst)\ id)\ o\ BNF\_Composition.id\_bnf\ o\ BNF\_Composition.id\_bnf)))\ o$   
 $BNF\_Composition.id\_bnf\ o\ BNF\_Composition.id\_bnf$

**definition**  $n2m\_ctor\_rec\_G\_F\ ::$   
 $((a\ G\ F \times 'c, 'a)\ G_M\_pre\_G_M \Rightarrow 'b) \Rightarrow ((a\ G\ F \times 'c, 'a\ G \times 'b)\ GF_M\_pre\_GF_M \Rightarrow 'c) \Rightarrow 'a\ G\ F \Rightarrow 'c$   
 where  $n2m\_ctor\_rec\_G\_F\ s1\ s2 = ctor\_rec\_F\ (s2\ o\ map\_pre\_GF_M\ (id\ ::\ unit\ \Rightarrow\ unit))\ (BNF\_Def.convolve\ id\ (n2m\_ctor\_rec\_G\ s1\ s2))\ id\ o\ BNF\_Composition.id\_bnf\ o\ BNF\_Composition.id\_bnf$

**lemma**  $n2m\_ctor\_rec\_G$ :  
 $n2m\_ctor\_rec\_G\ s1\ s2\ o\ ctor\_G = s1\ o\ map\_pre\_G_M\ id\ id\ (BNF\_Def.convolve\ id\ (n2m\_ctor\_rec\_G\_F\ s1\ s2))$   
 $o\ BNF\_Composition.id\_bnf\ o\ BNF\_Composition.id\_bnf$   
 ⟨proof⟩

**lemma**  $n2m\_ctor\_rec\_G\_F$ :  
 $n2m\_ctor\_rec\_G\_F\ s1\ s2\ o\ ctor\_F = s2\ o\ map\_pre\_GF_M\ id\ (BNF\_Def.convolve\ id\ (n2m\_ctor\_rec\_G\ s1\ s2))$   
 $(BNF\_Def.convolve\ id\ (n2m\_ctor\_rec\_G\_F\ s1\ s2))\ o\ BNF\_Composition.id\_bnf\ o\ BNF\_Composition.id\_bnf$   
 ⟨proof⟩

### 7.3.3 Induction

**lemma**  $n2m\_rel\_induct\_G\_G\_F$ :  
**assumes**  $IH1: \forall x\ y. BNF\_Def.vimage2p\ (BNF\_Composition.id\_bnf\ o\ BNF\_Composition.id\_bnf)\ (BNF\_Composition.id\_bnf\ o\ BNF\_Composition.id\_bnf)\ (rel\_pre\_G_M\ P\ R\ S)\ x\ y \longrightarrow R\ (ctor\_G\ x)\ (ctor\_G\ y)$   
**and**  $IH2: \forall x\ y. BNF\_Def.vimage2p\ (BNF\_Composition.id\_bnf\ o\ BNF\_Composition.id\_bnf)\ (BNF\_Composition.id\_bnf\ o\ BNF\_Composition.id\_bnf)\ (rel\_pre\_GF_M\ P\ R\ S)\ x\ y \longrightarrow S\ (ctor\_F\ x)\ (ctor\_F\ y)$   
**shows**  $rel\_G\ P \leq R \wedge rel\_F\ (rel\_G\ P) \leq S$   
 ⟨proof⟩

**lemmas**  $n2m\_ctor\_induct\_G\_G\_F = spec[OF\ spec\ [OF$   
 $n2m\_rel\_induct\_G\_G\_F[of\ (=)\ BNF\_Def.Grp\ (Collect\ R)\ id\ BNF\_Def.Grp\ (Collect\ S)\ id\ \mathbf{for}\ R\ S,$   
 $unfolded\ G.rel\_eq\ F.rel\_eq\ eq\_le\_Grp\_id\_iff\ all\_simps(1,2)[symmetric]]],$   
 $unfolded\ eq\_alt\ pre\_G_M.rel\_Grp\ pre\_GF_M.rel\_Grp\ pre\_G_M.map\_id0\ pre\_GF_M.map\_id0,$   
 $unfolded\ vimage2p\_comp\ vimage2p\_id\ comp\_apply\ comp\_id\ Grp\_id\_mono\_subst$

$type\_copy\_vimage2p\_Grp\_Rep[OF\ BNF\_Composition.type\_definition\_id\_bnf\_UNIV]$   
 $type\_copy\_Abs\_o\_Rep[OF\ BNF\_Composition.type\_definition\_id\_bnf\_UNIV]$   
 $eqTrueI[OF\ subset\_UNIV]\ simp\_thms(22)$   
 $atomize\_conjL[symmetric]\ atomize\_all[symmetric]\ atomize\_imp[symmetric],$   
 $unfolded\ subset\_iff\ mem\_Collect\_eq]$

## 8 Mutual View on Nested Coatypes

**bnf-axiomatization** ('a, 'b) coF0

**bnf-axiomatization** ('a, 'b) coG0

### 8.1 Nested definition

**codatatype** 'a coF = CcoF ('a, 'a coF) coF0

**codatatype** 'a coG = CcoG ('a, ('a coG) coF) coG0

**type-synonym** ('b, 'c) coF\_pre\_coF = ('c, 'b) coF0

**type-synonym** ('c, 'a) coG\_pre\_coG = ('a, 'c coF) coG0

**term** dtor\_unfold\_coF :: ('b ⇒ ('b, 'c) coF\_pre\_coF) ⇒ 'b ⇒ 'c coF

**term** dtor\_unfold\_coG :: ('c ⇒ ('c, 'a) coG\_pre\_coG) ⇒ 'c ⇒ 'a coG

**term** dtor\_corec\_coF :: ('b ⇒ ('c coF + 'b, 'c) coF\_pre\_coF) ⇒ 'b ⇒ 'c coF

**term** dtor\_corec\_coG :: ('c ⇒ ('a coG + 'c, 'a) coG\_pre\_coG) ⇒ 'c ⇒ 'a coG

**thm** coF.dtor\_rel\_coinduct

**thm** coG.dtor\_rel\_coinduct[unfolded rel\_pre\_coG\_def id\_apply]

### 8.2 Isomorphic Mutual Definition

**codatatype** 'a coG<sub>M</sub> = CcoG ('a, 'a coGcoF<sub>M</sub>) coG0

**and** 'a coGcoF<sub>M</sub> = CcoF ('a coG<sub>M</sub>, 'a coGcoF<sub>M</sub>) coF0

**type-synonym** ('b, 'c) coGcoF<sub>M</sub>\_pre\_coGcoF<sub>M</sub> = ('c, 'b) coF0

**type-synonym** ('c, 'a) coG<sub>M</sub>\_pre\_coG<sub>M</sub> = ('a, 'c) coG0

**term** dtor\_unfold\_coG<sub>M</sub> :: ('b ⇒ ('c, 'a) coG<sub>M</sub>\_pre\_coG<sub>M</sub>) ⇒ ('c ⇒ ('c, 'b) coGcoF<sub>M</sub>\_pre\_coGcoF<sub>M</sub>) ⇒ 'b ⇒ 'a coG<sub>M</sub>

**term** dtor\_unfold\_coGcoF<sub>M</sub> :: ('b ⇒ ('c, 'a) coG<sub>M</sub>\_pre\_coG<sub>M</sub>) ⇒ ('c ⇒ ('c, 'b) coGcoF<sub>M</sub>\_pre\_coGcoF<sub>M</sub>) ⇒ 'c ⇒ 'a coGcoF<sub>M</sub>

**term** dtor\_corec\_coG<sub>M</sub> :: ('b ⇒ ('a coGcoF<sub>M</sub> + 'c, 'a) coG<sub>M</sub>\_pre\_coG<sub>M</sub>) ⇒ ('c ⇒ ('a coGcoF<sub>M</sub> + 'c, 'a coG<sub>M</sub> + 'b) coGcoF<sub>M</sub>\_pre\_coGcoF<sub>M</sub>) ⇒ 'b ⇒ 'a coG<sub>M</sub>

**term** dtor\_corec\_coGcoF<sub>M</sub> :: ('b ⇒ ('a coGcoF<sub>M</sub> + 'c, 'a) coG<sub>M</sub>\_pre\_coG<sub>M</sub>) ⇒ ('c ⇒ ('a coGcoF<sub>M</sub> + 'c, 'a coG<sub>M</sub> + 'b) coGcoF<sub>M</sub>\_pre\_coGcoF<sub>M</sub>) ⇒ 'c ⇒ 'a coGcoF<sub>M</sub>

**thm** coG<sub>M</sub>\_coGcoF<sub>M</sub>.dtor\_rel\_coinduct[unfolded rel\_pre\_coG<sub>M</sub>\_def rel\_pre\_coGcoF<sub>M</sub>\_def]

### 8.3 Mutualization

#### 8.3.1 Coiterators

**definition** n2m\_dtor\_unfold\_coG :: ('b ⇒ ('c, 'a) coG<sub>M</sub>\_pre\_coG<sub>M</sub>) ⇒ ('c ⇒ ('c, 'b) coGcoF<sub>M</sub>\_pre\_coGcoF<sub>M</sub>) ⇒ 'b ⇒ 'a coG

**where** n2m\_dtor\_unfold\_coG s1 s2 = dtor\_unfold\_coG (BNF\_Composition.id\_bnf o BNF\_Composition.id\_bnf o

map\_pre\_coG<sub>M</sub> id (id :: unit ⇒ unit) (dtor\_unfold\_coF (BNF\_Composition.id\_bnf o BNF\_Composition.id\_bnf o s2)) o s1)

**definition** n2m\_dtor\_unfold\_coG\_coF :: ('b ⇒ ('c, 'a) coG<sub>M</sub>\_pre\_coG<sub>M</sub>) ⇒ ('c ⇒ ('c, 'b) coGcoF<sub>M</sub>\_pre\_coGcoF<sub>M</sub>) ⇒ 'c ⇒ 'a coG coF

**where** n2m\_dtor\_unfold\_coG\_coF s1 s2 = dtor\_unfold\_coF (BNF\_Composition.id\_bnf o BNF\_Composition.id\_bnf o map\_pre\_coGcoF<sub>M</sub> (id :: unit ⇒ unit) (n2m\_dtor\_unfold\_coG s1 s2) id o s2)

**lemma** coG\_dtor\_o\_unfold: dtor\_coG o dtor\_unfold\_coG s = map\_pre\_coG id (dtor\_unfold\_coG s) o s  
<proof>

**lemma** coF\_dtor\_o\_unfold: dtor\_coF o dtor\_unfold\_coF s = map\_pre\_coF id (dtor\_unfold\_coF s) o s  
<proof>

**lemma** *coG\_dtor\_o\_corec*:  $\text{dtor\_coG } o \text{ dtor\_corec\_coG } s = \text{map\_pre\_coG } id \text{ (case\_sum } id \text{ (dtor\_corec\_coG } s))$   
 $o \ s$   
 ⟨proof⟩

**lemma** *coF\_dtor\_o\_corec*:  $\text{dtor\_coF } o \text{ dtor\_corec\_coF } s = \text{map\_pre\_coF } id \text{ (case\_sum } id \text{ (dtor\_corec\_coF } s))$   
 $o \ s$   
 ⟨proof⟩

**lemma** *n2m\_dtor\_unfold\_coG*:  
 $\text{dtor\_coG } o \text{ n2m\_dtor\_unfold\_coG } s1 \ s2 = \text{BNF\_Composition.id\_bnf } o \text{ BNF\_Composition.id\_bnf } o \text{ map\_pre\_coG}_M$   
 $id \text{ (n2m\_dtor\_unfold\_coG\_coF } s1 \ s2) \ o \ s1$   
 ⟨proof⟩

**lemma** *n2m\_dtor\_unfold\_coG\_coF*:  
 $\text{dtor\_coF } o \text{ n2m\_dtor\_unfold\_coG\_coF } s1 \ s2 = \text{BNF\_Composition.id\_bnf } o \text{ BNF\_Composition.id\_bnf } o \text{ map\_pre\_coGcoF}_M$   
 $id \text{ (n2m\_dtor\_unfold\_coG } s1 \ s2) \ (n2m\_dtor\_unfold\_coG\_coF } s1 \ s2) \ o \ s2$   
 ⟨proof⟩

### 8.3.2 Corecursors

**definition** *n2m\_dtor\_corec\_coG* ::  
 $( 'b \Rightarrow ('a \text{ coG } coF + 'c, 'a) \text{ coG}_M \text{pre\_coG}_M) \Rightarrow ('c \Rightarrow ('a \text{ coG } coF + 'c, 'a \text{ coG } + 'b) \text{ coGcoF}_M \text{pre\_coGcoF}_M)$   
 $\Rightarrow 'b \Rightarrow 'a \text{ coG}$   
**where** *n2m\_dtor\_corec\_coG*  $s1 \ s2 =$   
 $\text{dtor\_corec\_coG } (\text{BNF\_Composition.id\_bnf } o \text{ BNF\_Composition.id\_bnf } o$   
 $\text{map\_pre\_coG}_M \text{ id } (id :: \text{unit} \Rightarrow \text{unit})$   
 $(\text{case\_sum } (\text{map\_coF } \text{Inl}) \text{ (dtor\_corec\_coF } (\text{BNF\_Composition.id\_bnf } o \text{ BNF\_Composition.id\_bnf } o$   
 $\text{map\_pre\_coGcoF}_M \text{ (id :: \text{unit} \Rightarrow \text{unit}) } id \text{ (map\_sum } (\text{map\_coF } \text{Inl}) \text{ id } o \ s2))) \ o$   
 $s1)$

**definition** *n2m\_dtor\_corec\_coG\_coF* ::  
 $( 'b \Rightarrow ('a \text{ coG } coF + 'c, 'a) \text{ coG}_M \text{pre\_coG}_M) \Rightarrow ('c \Rightarrow ('a \text{ coG } coF + 'c, 'a \text{ coG } + 'b) \text{ coGcoF}_M \text{pre\_coGcoF}_M)$   
 $\Rightarrow 'c \Rightarrow 'a \text{ coG } coF$   
**where** *n2m\_dtor\_corec\_coG\_coF*  $s1 \ s2 = \text{dtor\_corec\_coF } (\text{BNF\_Composition.id\_bnf } o \text{ BNF\_Composition.id\_bnf } o$   
 $\text{map\_pre\_coGcoF}_M \text{ (id :: \text{unit} \Rightarrow \text{unit}) } (\text{case\_sum } id \text{ (n2m\_dtor\_corec\_coG } s1 \ s2)) \ id \ o \ s2)$

**lemma** *n2m\_dtor\_corec\_coG*:  
 $\text{dtor\_coG } o \text{ n2m\_dtor\_corec\_coG } s1 \ s2 = \text{BNF\_Composition.id\_bnf } o \text{ BNF\_Composition.id\_bnf } o \text{ map\_pre\_coG}_M$   
 $id \text{ id } (\text{case\_sum } id \text{ (n2m\_dtor\_corec\_coG\_coF } s1 \ s2)) \ o \ s1$   
 ⟨proof⟩

**lemma** *n2m\_dtor\_corec\_coG\_coF*:  
 $\text{dtor\_coF } o \text{ n2m\_dtor\_corec\_coG\_coF } s1 \ s2 = \text{BNF\_Composition.id\_bnf } o \text{ BNF\_Composition.id\_bnf } o \text{ map\_pre\_coGcoF}_M$   
 $id \text{ (case\_sum } id \text{ (n2m\_dtor\_corec\_coG } s1 \ s2)) \ (\text{case\_sum } id \text{ (n2m\_dtor\_corec\_coG\_coF } s1 \ s2)) \ o \ s2$   
 ⟨proof⟩

### 8.3.3 Coinduction

**lemma** *n2m\_rel\_coinduct\_coG\_coG\_coF*:  
**assumes** *CIH1*:  $\forall x \ y. \ R \ x \ y \longrightarrow \text{BNF\_Def.vimage2p } (\text{BNF\_Composition.id\_bnf } o \text{ BNF\_Composition.id\_bnf})$   
 $(\text{BNF\_Composition.id\_bnf } o \text{ BNF\_Composition.id\_bnf}) \text{ (rel\_pre\_coG}_M \ P \ R \ S) \text{ (dtor\_coG } x) \text{ (dtor\_coG } y)$   
**and** *CIH2*:  $\forall x \ y. \ S \ x \ y \longrightarrow \text{BNF\_Def.vimage2p } (\text{BNF\_Composition.id\_bnf } o \text{ BNF\_Composition.id\_bnf})$   
 $(\text{BNF\_Composition.id\_bnf } o \text{ BNF\_Composition.id\_bnf}) \text{ (rel\_pre\_coGcoF}_M \ P \ R \ S) \text{ (dtor\_coF } x) \text{ (dtor\_coF } y)$   
**shows**  $R \leq \text{rel\_coG } P \wedge S \leq \text{rel\_coF } (\text{rel\_coG } P)$   
 ⟨proof⟩

**lemmas** *n2m\_ctor\_induct\_coG\_coG\_coF* =  $\text{spec}[OF \ \text{spec}[OF \ \text{spec}[OF \ \text{spec}[OF$   
 $\text{n2m\_rel\_coinduct\_coG\_coG\_coF}[of \ \_ (=),$   
 $\text{unfolded } coG.\text{rel\_eq } coF.\text{rel\_eq } le\_fun\_def \ le\_bool\_def \ \text{all\_simps}(1,2)[\text{symmetric}]]]]]]$