
An Introduction to Relational Hylomorphisms

DI/UM, 2002

José N. Oliveira

Dept. Informatica

Universidade do Minho, 4700 Braga, Portugal

jno@di.uminho.pt

“How” does one specify?

General problem solving strategy?



Problem

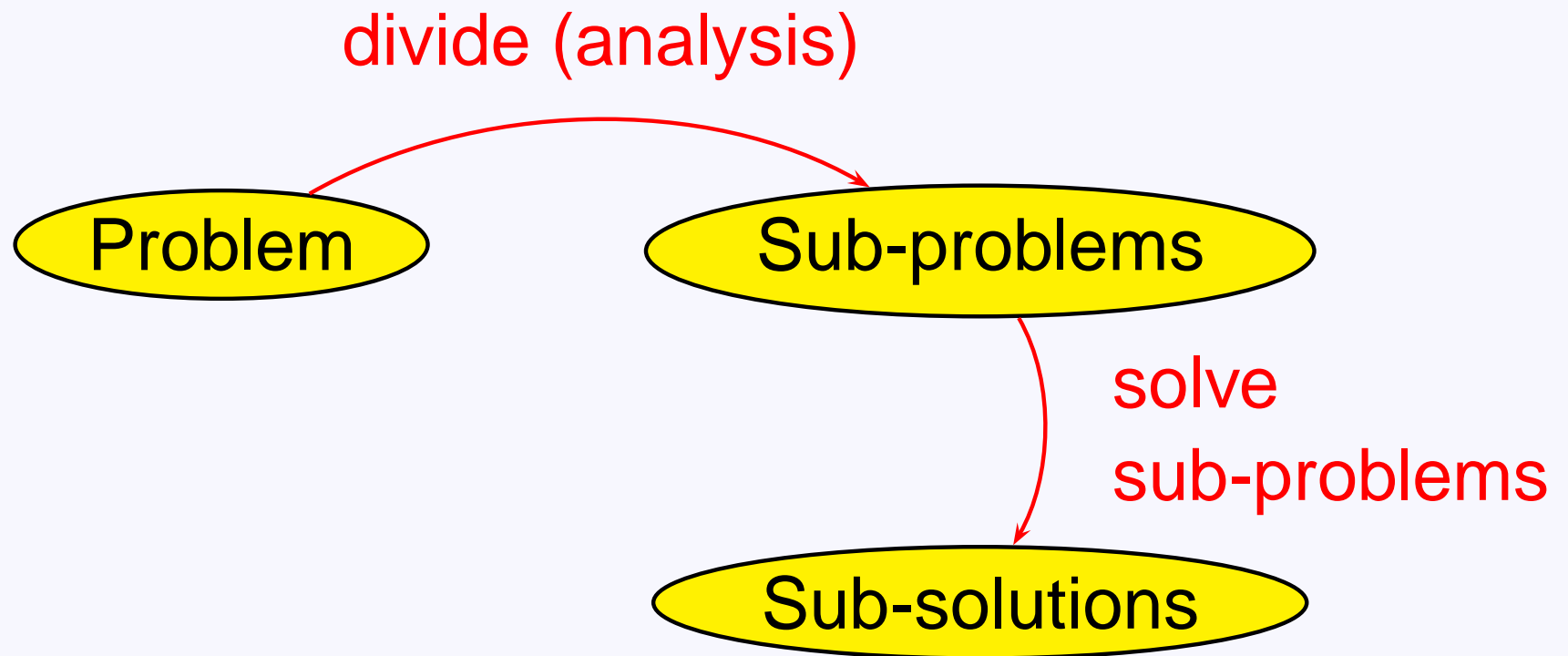
“How” does one specify?

Divide-and-conquer:



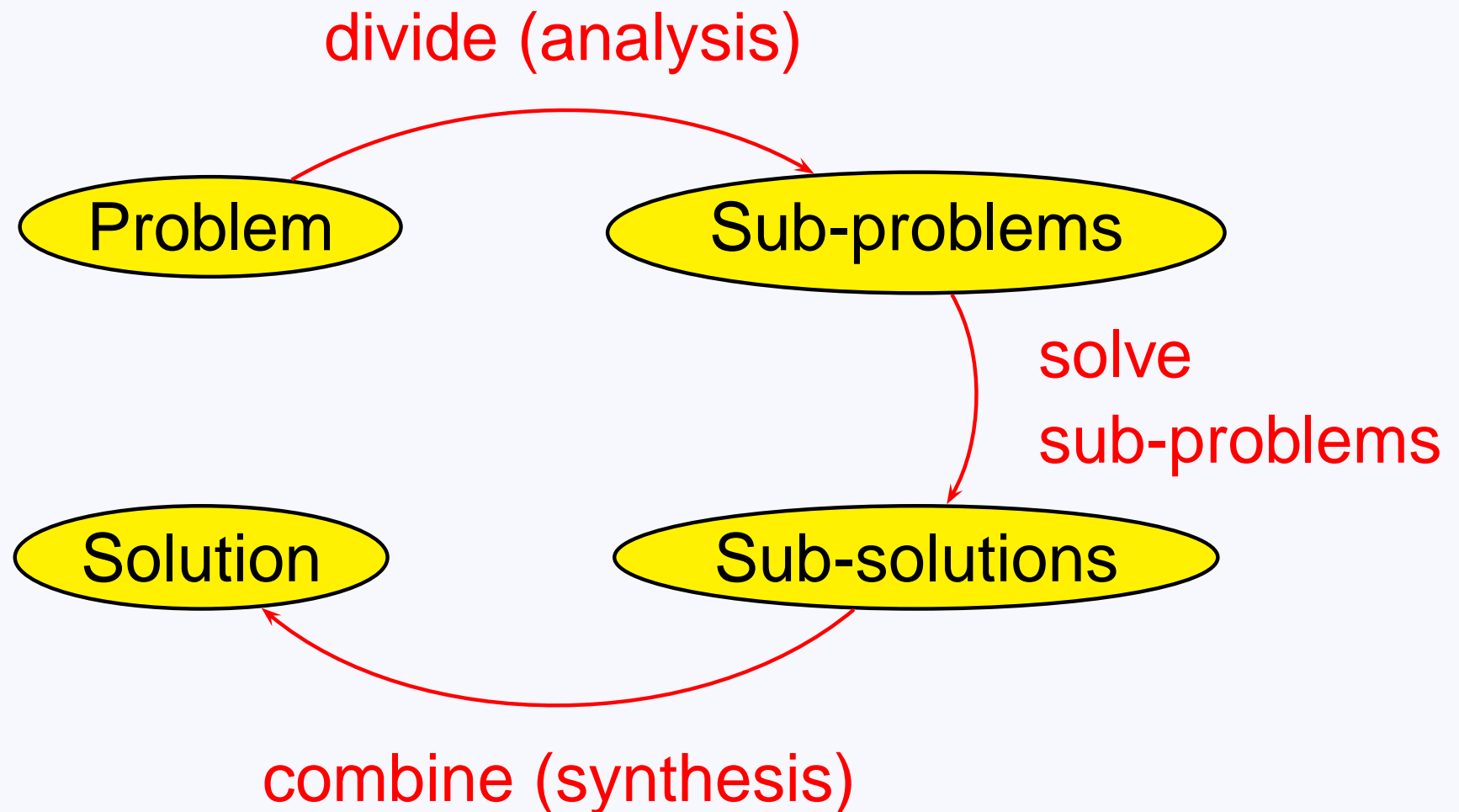
“How” does one specify?

Divide-and-conquer:



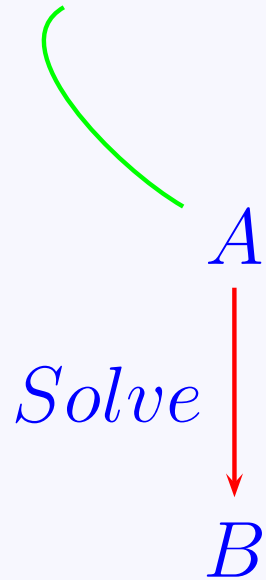
“How” does one specify?

Divide-and-conquer:



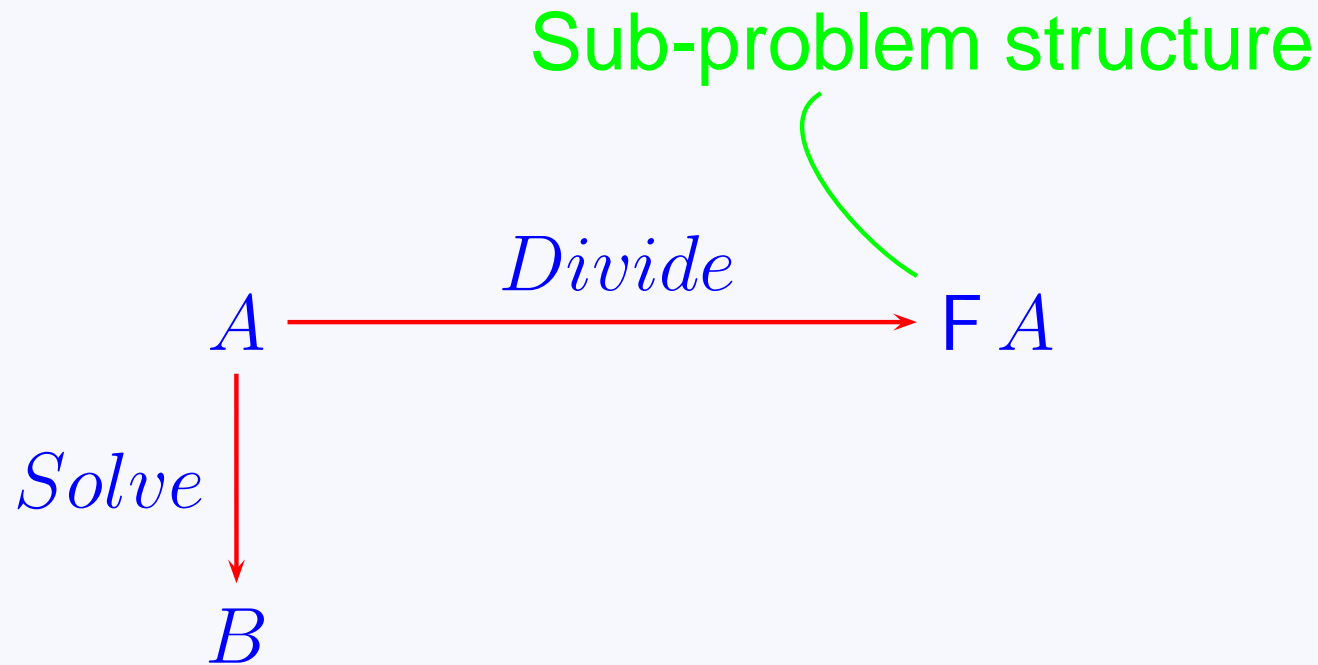
Divide-and-conquer (formally)

Problem space

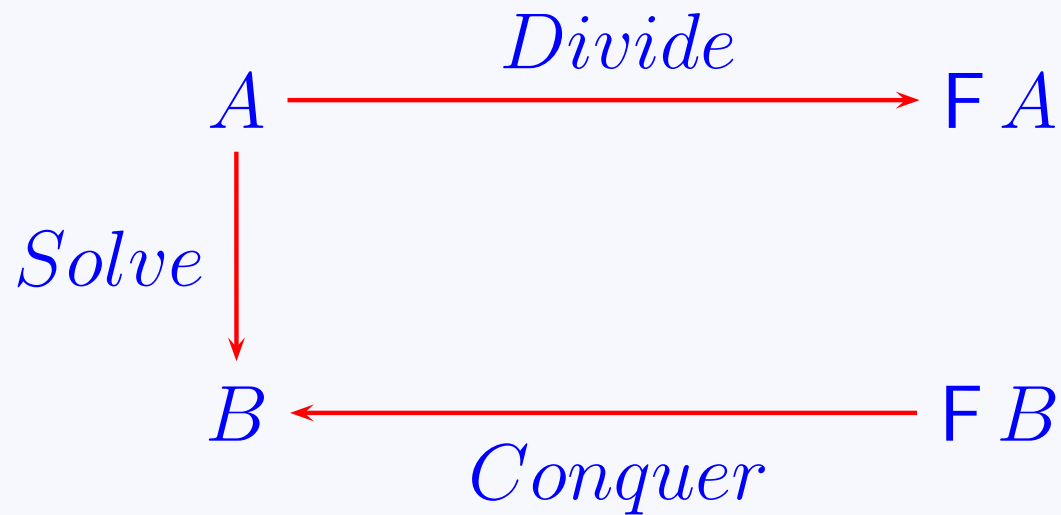


Solution space

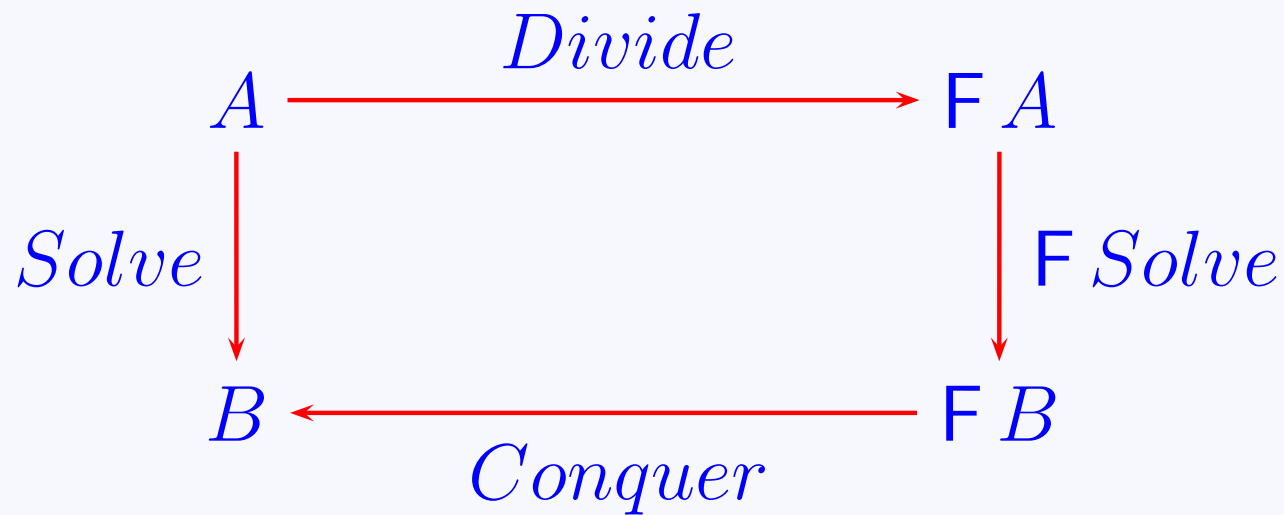
Divide-and-conquer (formally)



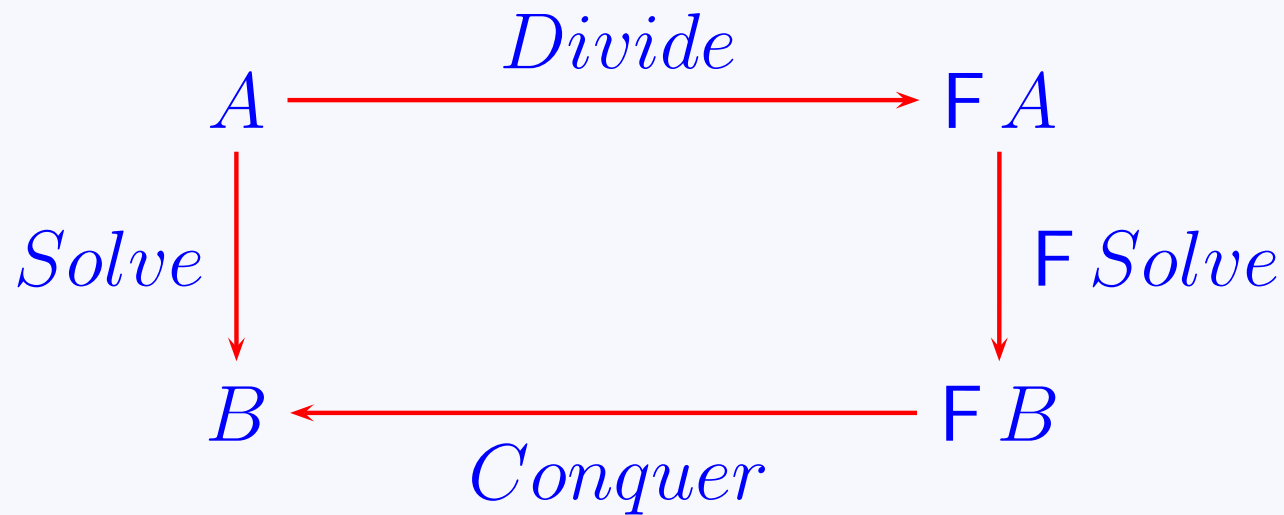
Divide-and-conquer (formally)



Divide-and-conquer (formally)



Divide-and-conquer (formally)



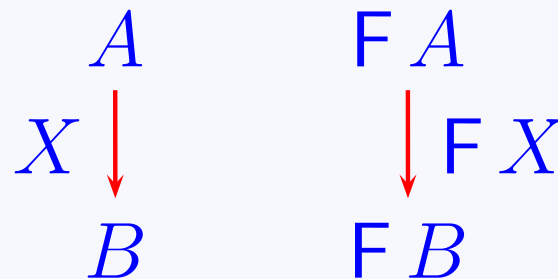
Questions:

- What are the mathematics of *Divide*, *Conquer*, *Solve*?
- What do $(F A)$, $(F \text{ Solve})$ mean?

Relators

Symbol F is overloaded:

- $F A$ means a (parametric) **datatype**, eg. A^* — seq of A in VDM-SL;
- $F X$ means a **relation**



Example: X^* will be such that

$$l(X^*)l' \equiv \text{len } l = \text{len } l' \wedge \forall i \in \text{inds } l. (l \ i)X(l' \ i)$$

Properties of relators

Every **relator** F is monotone,

$$R \subseteq S \Rightarrow (F R) \subseteq (F S)$$

and commutes with (\cdot) , $(_)^\circ$ and id :

$$F (R \cdot S) = (F R) \cdot (F S)$$

$$F id = id$$

$$F (R^\circ) = (F R)^\circ$$

Terminology:

Properties of relators

Every **relator** F is monotone,

$$R \subseteq S \Rightarrow (F R) \subseteq (F S)$$

and commutes with (\cdot) , $(_)^{\circ}$ and id :

$$F (R \cdot S) = (F R) \cdot (F S)$$

$$F id = id$$

$$F (R^{\circ}) = (F R)^{\circ}$$

Terminology:

$A \xleftarrow{R} F A$ is called an **F-algebra**

Properties of relators

Every **relator** F is monotone,

$$R \subseteq S \Rightarrow (F R) \subseteq (F S)$$

and commutes with (\cdot) , $(_)^\circ$ and id :

$$F (R \cdot S) = (F R) \cdot (F S)$$

$$F id = id$$

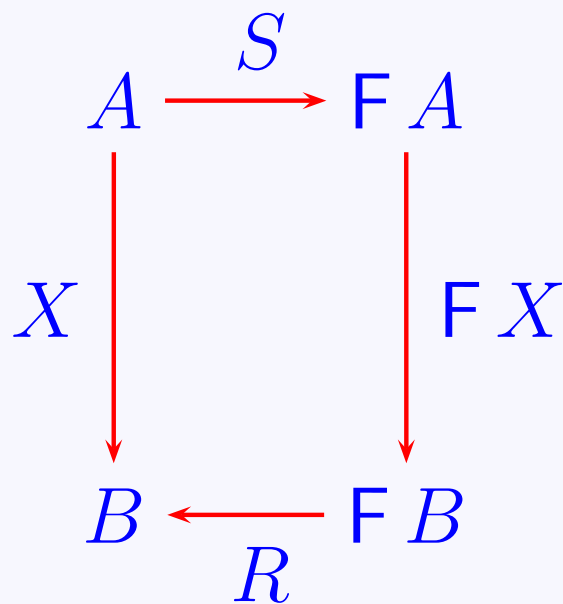
$$F (R^\circ) = (F R)^\circ$$

Terminology:

$A \xrightarrow{S} F A$ is called an **F-coalgebra**

Back to divide-and-conquer

Divide-and-conquer = relational **hylomorphism**:



that is,

$$X = R \cdot (F X) \cdot S$$

How do we solve this (hylo) equation for X ?

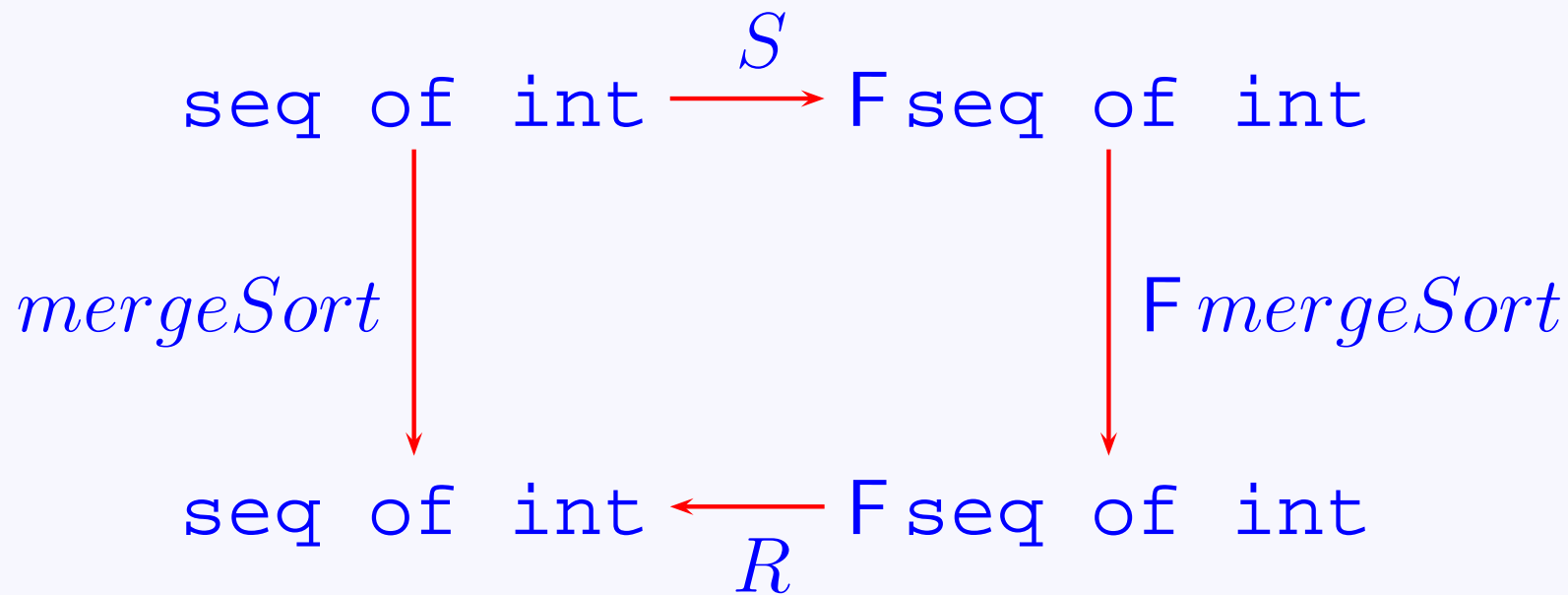
An example first

```
mergeSort: seq of int -> seq of int
mergeSort(l) ==
  cases l :
    [e] -> [e] ,
    others -> let l1 ^ l2
               in set {l} be st
               abs (len l1 - len l2) < 2 in
               lmerge(mergeSort(l1), mergeSort(l2))
  end;
```

is a **relational** hylomorphism for

$$F X = \text{int} + X \times X$$

In fact



that is,

$$\text{mergeSort} = R \cdot (F \text{ mergeSort}) \cdot S$$

where $R = [\text{singl}, \text{lmerge}]$, for $\text{singl} = \lambda e.[e]$

mergeSort algebra and coalgebra

and S is

```
S: seq of int -> ( int | seq of int * seq of int )
S(l) ==
  cases l :
    [e] -> e ,
    others -> let l1 ^ l2
               in set {l} be st
               abs (len l1 - len l2) < 2 in
               mk_(l1,l2)
  end;
```

Equations and fixpoints

Given an **equation** of pattern

$$x = f x$$

where $A \xleftarrow{f} A$ for some A , we will say that any **solution** to this equation — that is, any $a_0 \in A$ such that

$$a_0 = f a_0$$

is a **fixpoint** of f .

Equations versus recursion

Equation $x = f x$ can also be regarded as a “recursive” definition of its fixpoints, eg.

- $x = 1 + \frac{x}{2}$ is a recursive definition of number 2.

However,

- $x = \frac{x^2+3}{4}$ has two solutions (=fixpoints) 1 e 3.

What are we “recursively defining” here?

- Furthermore, $x = x$ defines any object!
- Last but not least, some equations don't have any solution at all. Think eg. of $x = x + 1$ in \mathbb{N} .

Solving (Fixpoint) Equations I

Let $A \xleftarrow{\leq_A} A$ be a **partial order**. Then, every $a \in A$ such that

$$a \leq_A f a$$

is said to be a **post-fixpoint** of f , and every $a \in A$ such that

$$a \geq_A f a$$

is said to be a **pre-fixpoint** of f . Clearly,

Every $a \in A$ which is both a pre-fixpoint and a post-fixpoint of f is a **fixpoint** of f .

Solving (Fixpoint) Equations II

Function $B \xleftarrow{f} A$ is **monotone** wherever

$$f \cdot \leq_A \subseteq \leq_B \cdot f$$

for partial orders \leq_A and \leq_B , that is:

$$f \cdot \leq_A \subseteq \leq_B \cdot f$$

$$\equiv \{ \text{shunting} \}$$

$$\leq_A \subseteq f^\circ \cdot \leq_B \cdot f$$

$$\equiv \{ \text{going pointwise} \}$$

$$a \leq_A a' \Rightarrow (f a) \leq_B (f a')$$

Solving (Fixpoint) Equations III

Pointwise ordering on functions $B \xleftarrow{f, g} A$:

$$f \leq_B^\cdot g \equiv f \subseteq \leq_B \cdot g$$

meaning

$$\begin{aligned} f \leq_B^\cdot g &\equiv f \subseteq \leq_B \cdot g \\ &\equiv \{ \text{shunting} \} \end{aligned}$$

$$\begin{aligned} &id \subseteq f^\circ \cdot \leq_B \cdot g \\ &\equiv \{ \text{going pointwise} \} \end{aligned}$$

$$\forall a. f a \leq_B g a$$

Solving (Fixpoint) Equations IV

Lattice fixpoint theorem (Tarski 1955) for **monotone** f as above and \leq_A defining a **complete lattice**:

- The set of all fixpoints of f ,

$$P = \{a \in A \mid a = f a\}$$

is non-empty and $\langle P; \leq_A \rangle$ is a complete (sub)lattice.

- The least of all fixpoints ($\bigwedge P$) and the greatest one ($\bigvee P$) are as follows:

$$\mu f = \bigwedge P = \bigwedge \{x \mid x \geq f x\}$$

$$\nu f = \bigvee P = \bigvee \{x \mid x \leq_A f x\}$$

Solving relational equations

Hylo-equation

$$X = \underbrace{R \cdot (F X) \cdot S}_{f \ X}$$

and other relational equations such as

$$X = \underbrace{R \cup R \cdot X}_{g \ X}$$

(cf. **transitive** closure) have least solutions

$$\mu f = \llbracket R, S \rrbracket$$

$$\mu g = R^+$$

because both f, g are monotone.

Laws of the Fixpoint Calculus

Computation rule:

$$\mu f = f \mu f$$

Example: hylo-cancellation law

$$\llbracket R, S \rrbracket = R \cdot F \llbracket R, S \rrbracket \cdot S$$

Rolling rule:

$$\mu(g \cdot h) = g(\mu(h \cdot g))$$

Example: $f = g \cdot h$ where $h X = R \cdot X$ and $g X = R \cup X$. Then

Rolling rule

$$\begin{aligned}\mu f &= \mu(g \cdot h) \\ &= \{ \text{rolling rule} \} \\ &\quad g(\mu(h \cdot g)) \\ &= \{ \text{definitions of } g, h \} \\ &\quad R \cup (\mu x.(R \cdot (R \cup x))) \\ &= \{ (R \cdot) \text{ is a lower-adjoint} \} \\ &\quad R \cup \mu x.(R^2 \cup R \cdot x)\end{aligned}$$

Further application of this rule will “factor out” R^2 , R^3 , etc., In the limit, $\mu f = \bigcup_{j=1}^{\infty} R^j = R^+$.

Hylo rolling rule

Let $f = g \cdot h$ where $h X = F X \cdot S$ and $g = (R \cdot)$. Then

$$\begin{aligned}
 \mu f &= \mu(g \cdot h) &= g(\mu(h \cdot g)) \\
 &= &\{ \text{definitions of } g, h \} \\
 &&R \cdot (\mu X. (F(R \cdot X) \cdot S)) \\
 &= &\{ \text{relators} \} \\
 &&R \cdot (\mu X. F R \cdot F X \cdot S)
 \end{aligned}$$

that is,

$$\llbracket R, S \rrbracket = R \cdot \llbracket F R, S \rrbracket$$

Other rules

Square rule:

$$\mu f = \mu(f^2)$$

Monotonicity:

$$f \dot{\leq} g \Rightarrow \mu f \leq \mu g$$

Thus

$$\llbracket T, U \rrbracket \subseteq \llbracket R, S \rrbracket \Leftarrow T \subseteq R \wedge U \subseteq S$$

Other rules

Induction rule:

$$\mu f \leq x \iff f x \leq x$$

Thus

$$\llbracket R, S \rrbracket \subseteq T \iff R \cdot FT \cdot S \subseteq T$$

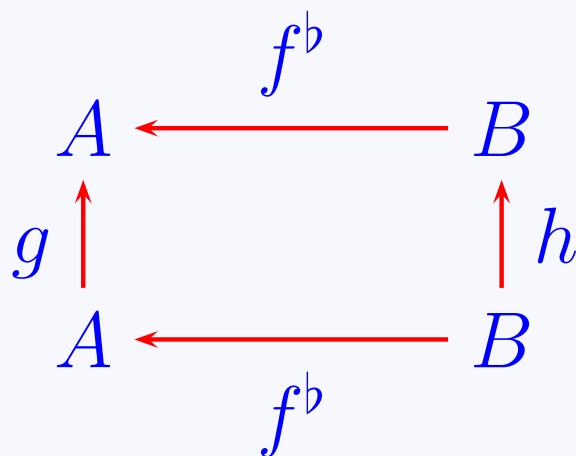
and, in particular (**coreflexive** hylos):

$$\begin{aligned} \llbracket R, S \rrbracket \subseteq id &\iff R \cdot S \subseteq id \\ \llbracket R, R^\circ \rrbracket \subseteq id &\iff R \text{ is simple} \end{aligned}$$

Last — but not least — **μ -fusion**:

μ -fusion theorem

Let



- h, g be monotonic,
- (A, \leq) and (B, \sqsubseteq) be complete **lattices**,
- f^b be a lower-adjoint.

Then

$$f^b(\mu h) = \mu g \iff f^b \cdot h = g \cdot f^b$$

Applications of μ -fusion theorem

Converse of a hylo

$$\llbracket S, R \rrbracket^\circ = \llbracket R^\circ, S^\circ \rrbracket$$

Proof: let $f^\flat = (-)^\circ$ and

$$h \ X \ = \ S \cdot \mathsf{F} \ X \cdot R$$

$$g \ X \ = \ T \cdot \mathsf{F} \ X \cdot U$$

that is,

$$\mu h = \llbracket S, R \rrbracket$$

$$\mu g = \llbracket T, U \rrbracket$$

Proof

Then

$$\llbracket S, R \rrbracket^\circ = \llbracket T, U \rrbracket$$

$$\Leftarrow \{ \mu\text{-fusion theorem} \}$$

$$(S \cdot \mathsf{F} X \cdot R)^\circ = T \cdot \mathsf{F} (X^\circ) \cdot U$$

$$\equiv \{ \text{converse and } \mathsf{F} \text{ is a relator} \}$$

$$R^\circ \cdot \mathsf{F} X^\circ \cdot S^\circ = T \cdot \mathsf{F} X^\circ \cdot U$$

$$\Leftarrow \{ \text{Leibniz} \}$$

$$R^\circ = T \wedge S^\circ = U$$

Hylo(cata)-fusion

$$V \cdot \llbracket S, R \rrbracket = \llbracket T, R \rrbracket \Leftarrow V \cdot S = T \cdot (FV)$$

Proof: since $(V \cdot) = (V \setminus)^b$,

$$\begin{aligned} & V \cdot \llbracket S, R \rrbracket = \llbracket T, R \rrbracket \\ \Leftarrow & \quad \{ \mu\text{-fusion theorem} \} \\ & V \cdot (S \cdot F X \cdot R) = T \cdot F (V \cdot X) \cdot R \\ \equiv & \quad \{ \text{associative } (\cdot) \text{ and relator } F \} \\ & (V \cdot S) \cdot F X \cdot R = T \cdot (FV) \cdot (F X) \cdot R \\ \Leftarrow & \quad \{ \text{Leibniz} \} \\ & V \cdot S = T \cdot (FV) \end{aligned}$$

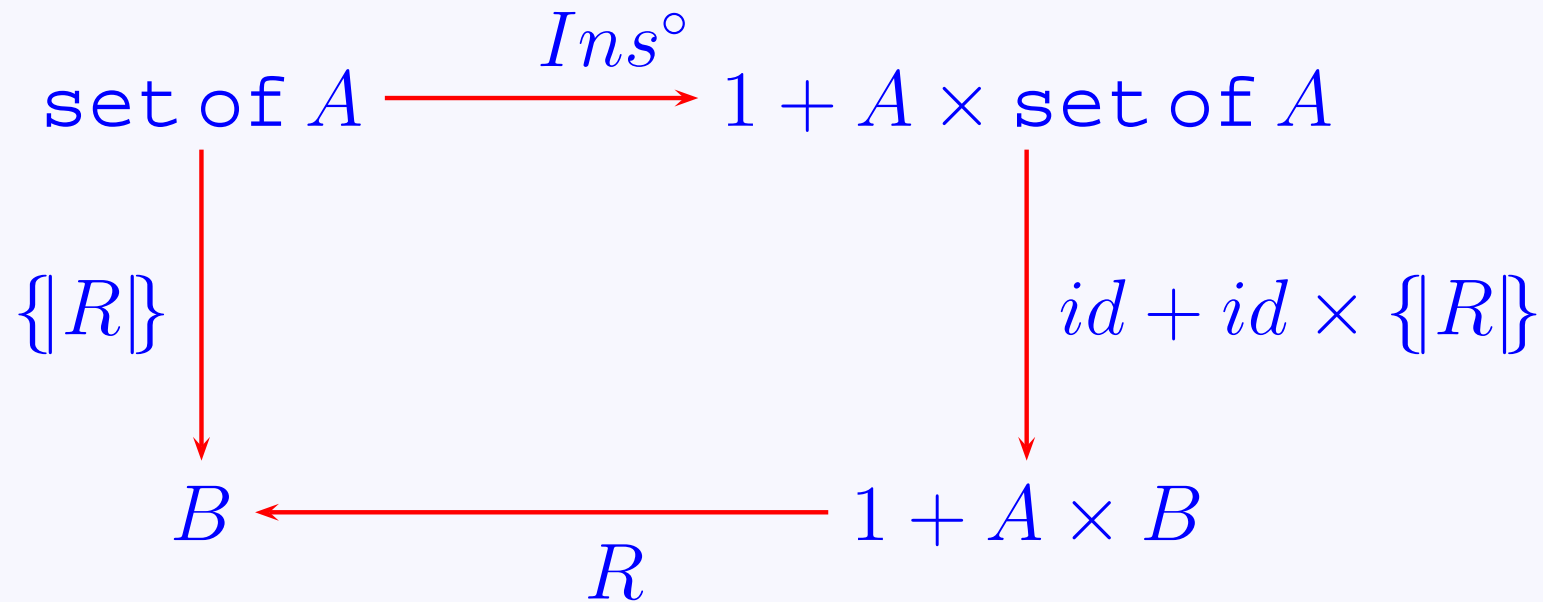
Hylo(ana)-fusion

$$\llbracket S, R \rrbracket \cdot V = \llbracket S, U \rrbracket \Leftarrow R \cdot V = FV \cdot U$$

Proof: $(\cdot V) = (/V)^b$. Then

$$\begin{aligned} & \llbracket S, R \rrbracket \cdot V = \llbracket S, U \rrbracket \\ \Leftarrow & \quad \{ \mu\text{-fusion theorem} \} \\ & (S \cdot F X \cdot R) \cdot V = S \cdot F (X \cdot V) \cdot U \\ \equiv & \quad \{ \text{associative } (\cdot) \text{ and relator } F \} \\ & S \cdot F X \cdot (R \cdot V) = S \cdot (F X) \cdot (F V) \cdot U \\ \Leftarrow & \quad \{ \text{Leibniz} \} \\ & R \cdot V = FV \cdot U \end{aligned}$$

Examples: VDM collective types



that is,

$$\{R\} = \llbracket R, \text{Ins}^\circ \rrbracket \quad \text{where} \quad \text{Ins} \stackrel{\text{def}}{=} [\underline{\emptyset}, \text{Puts}]$$

and ...

VDM-SL collective type set of A

```
Puts[@A] : @A * set of @A -> set of @A
Puts(e,s) == {e} union s
pre not e in set s ;
```

Pointfree version (for $R = [\underline{u}, f]$):

```
shylo[@A,@B] : (@A*@B -> @B) * @B -> set of @A -> @B
shylo(f,u)(s) ==
    if s={} then u
    else let a in set s,
           r = s \ {a}
           in f(a,shylo[@A,@B](f,u)(r));
```

VDM-SL collective type set of A

- For $\text{shylo}(f, u)$ to be a function the following must hold:

$$f(a, f(a', b)) = f(a', f(a, b))$$

- **Fusion** law

$$T \cdot \{R\} = \{S\} \Leftarrow T \cdot R = S \cdot (F T)$$

arises from $\text{hylo}(\text{cata})$ -fusion

- The **reflection** law holds:

$$\{Ins\} = id$$

Relational cata(ana)morphisms

Define

$$\begin{aligned} \langle R \rangle &= \llbracket R, in^\circ \rrbracket \\ \langle S \rangle &= \llbracket in, S \rrbracket \end{aligned}$$

where $\mu F \begin{array}{c} \xrightarrow{in^\circ} \\ \cong \\ \xleftarrow{in} \end{array} F \mu F$. For instance,

$$elems = \langle ins \rangle$$

Relational cata(ana)morphisms

From

$$\llbracket S, R \rrbracket^\circ = \llbracket R^\circ, S^\circ \rrbracket$$

infer

$$\begin{aligned} & \llbracket S \rrbracket \\ &= \llbracket in, S \rrbracket^{\circ\circ} \\ &= \llbracket S^\circ, in^\circ \rrbracket^\circ \\ &= (S^\circ)^\circ \end{aligned}$$

(=ana is the converse of the cata of the converse)

Inductive coreflexives

Recall

$$\llbracket R, S \rrbracket \subseteq id \iff R \cdot S \subseteq id$$

which entails

$$\langle R \rangle \subseteq id \iff R \subseteq in$$

that is,

$$\langle in \cdot S \rangle \subseteq id \iff S \subseteq id$$

Example (on finite lists):

$$IsOrdered \stackrel{\text{def}}{=} \langle in \cdot (id + ok) \rangle$$

Inductive coreflexives

where *ok* is the coreflexive induced by predicate

$ok(a, x) == \text{forall } b \text{ in set elems } x \ \& \ a \leq b$

This leads to

$$\begin{aligned} IsOrdered &= [nil, cons \cdot ok] \cdot \\ &\quad (id + id \times IsOrdered) \cdot \\ &\quad [nil, cons]^\circ \\ &= [nil, cons \cdot ok \cdot (id \times IsOrdered)] \cdot \\ &\quad [nil, cons]^\circ \end{aligned}$$

Inductive coreflexives

...and, finally, to

```
IsOrdered(l) ==  
  if l = []  
  then true  
  else (forall b in set elems tl l & hd l <= b) and  
        IsOrdered (tl l) ;
```

Exercise: calculate the above from $(in \cdot (id + ok))$

VDM-SL data type map A to B

$$\begin{array}{ccc}
 \text{map } A \text{ to } C & \xrightarrow{\text{Ins}^\circ} & 1 + (A \times C) \times \text{map } A \text{ to } C \\
 \downarrow \{\!\{R\}\!\} & & \downarrow id + id \times \langle id, \{\!\{R\}\!\} \rangle \\
 B & \xleftarrow{R} & 1 + (A \times C) \times B
 \end{array}$$

leading to the following pointwise syntax:

VDM-SL data type map A to B

```
mhylo[@A,@B,@C] : (@A*@C*@B -> @B) * @B ->
                  map @A to @C ->
                  @B

mhylo(f,u)(M) ==
  if M={|->} then u
  else let a in set dom M,
        c = M(a),
        R = {a} <-: M
  in f(c,mhylo[@A,@B,@C](f,u)(R));
```

Hylos as unique solutions

A standard result of the relational calculus establishes the following condition for

$$\mu X.(R \cdot F X \cdot S)$$

to be a **unique** solution:

- the “accessibility relation” of S is required to be **inductive** (cf. “well-founded” relations)
- This ensures **termination** insofar as the “size” of a sub-problem generated by S is strictly smaller than its source.
- One can perform **induction** over S .

Accessibility and membership

Accessibility relation for $F A \xleftarrow{S} A$:

$$A \xleftarrow{\prec_S} A$$

$$\prec_S \stackrel{\text{def}}{=} \in_F \cdot S$$

where $A \xleftarrow{\in_F} F A$ extends $A \xleftarrow{\in} \mathcal{P} A$ inductively over polynomial functors, as follows:

$$\begin{aligned} \in_{\mathcal{P}} &\stackrel{\text{def}}{=} \in \\ \in_C &\stackrel{\text{def}}{=} \perp \\ \in_{\lambda X.X} &\stackrel{\text{def}}{=} id \\ \in_{F \times G} &\stackrel{\text{def}}{=} (\in_F \cdot \pi_1) \cup (\in_G \cdot \pi_2) \\ \in_{F \cup G} &\stackrel{\text{def}}{=} [\in_F, \in_G] \end{aligned}$$

Example

Let $F X = 1 + B \times X$. Then,

$$\begin{aligned}
 & \in_{1+B \times X} \\
 = & \quad \{ \in \text{ for coproduct bifunctor } \} \\
 & [\in_1, \in_{B \times X}] \\
 = & \quad \{ \in \text{ for constant and product (bi)functors } \} \\
 & [\perp, (\in_B \cdot \pi_1) \cup (\in_{\lambda X.X} \cdot \pi_2)] \\
 = & \quad \{ \in \text{ for constant and identity functor } \} \\
 & [\perp, (\perp \cdot \pi_1) \cup (id \cdot \pi_2)] \\
 = & \quad \{ \perp \text{ and } [R, S] = (R \cdot i_1^\circ) \cup (S \cdot i_2^\circ) \} \\
 & \pi_2 \cdot i_2^\circ
 \end{aligned}$$

Example (pointwise)

Then,

$$\begin{aligned}\prec_S &= \in_{1+B \times X} \cdot S \\ &= (\pi_2 \cdot i_2^\circ) \cdot S \\ &= \pi_2 \cdot (i_2^\circ \cdot S)\end{aligned}$$

meaning

$$a' \prec_S a \equiv a' = \pi_2 x \wedge (i_2 x)Sa$$

For example, for $S = [nil, cons]^\circ$ on finite sequences, we get

Accessibility on finite sequences

$$\begin{aligned} & \pi_2 \cdot (i_2^\circ \cdot [nil, cons]^\circ) \\ = & \pi_2 \cdot ([nil, cons] \cdot i_2)^\circ \\ = & \pi_2 \cdot cons^\circ \end{aligned}$$

and therefore

$$\begin{aligned} a' \prec_{[nil, cons]^\circ} a & \equiv a' = \pi_2(b, a') \wedge a = cons(b, a') \\ \equiv a' \prec_{[nil, cons]^\circ} cons(b, a') \\ \equiv \mathbf{tl} a \prec_{[nil, cons]^\circ} a \end{aligned}$$

Hylo factorization (2)

For such inductive S , we can factor $\llbracket R, S \rrbracket$ in two components

$$\begin{array}{ccc} & \xrightarrow{\quad in \quad} & \\ \mu F & \cong & F(\mu F) \\ & \xleftarrow{\quad in^\circ \quad} & \end{array}$$

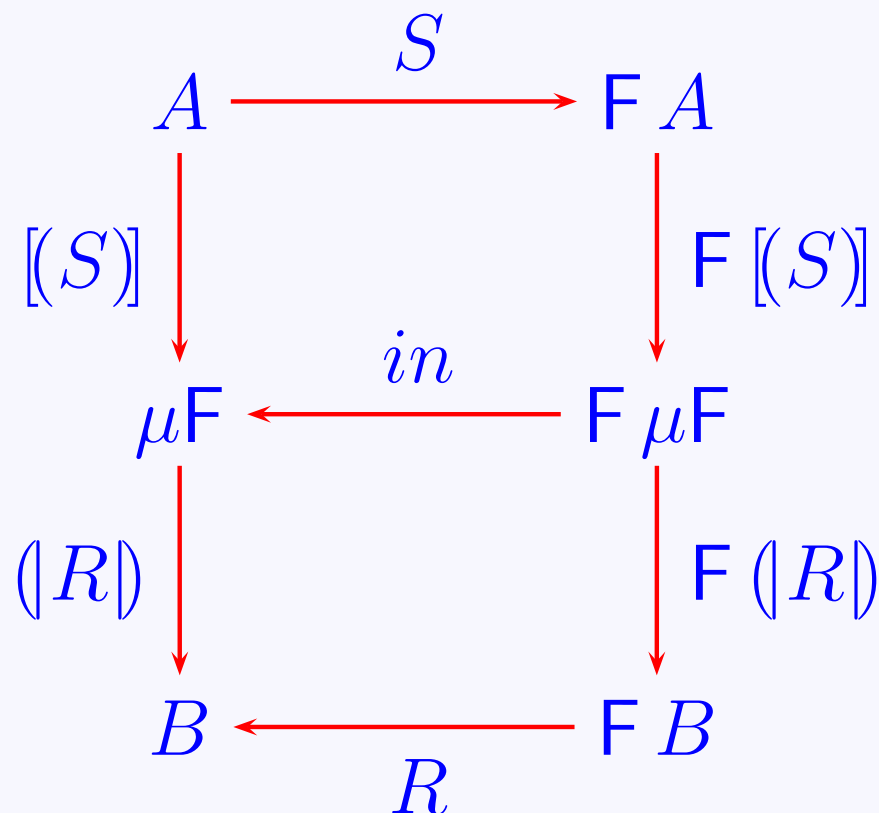
$$\begin{array}{ccc} A & \xrightarrow{\quad S \quad} & F A \\ \downarrow X_1 & & \downarrow F X_1 \\ \mu F & \xleftarrow{\quad in \quad} & F \mu F \\ \downarrow X_2 & & \downarrow F X_2 \\ B & \xleftarrow{\quad R \quad} & F B \end{array}$$

$$X_1 = in \cdot F X_1 \cdot S$$

Hylo-factorization Theorem

Using $(_)$, $[(_)]$ notation:

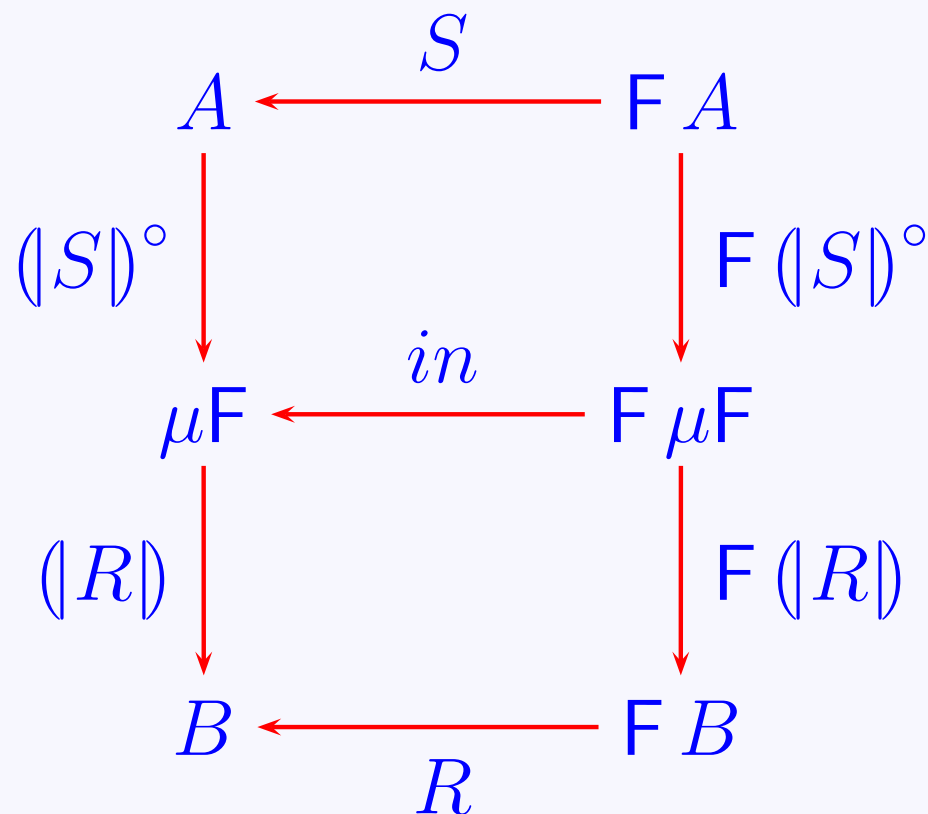
$$\mu X.(R \cdot F X \cdot S) = (_R) \cdot [(S)]$$



Hylo-factorization Theorem

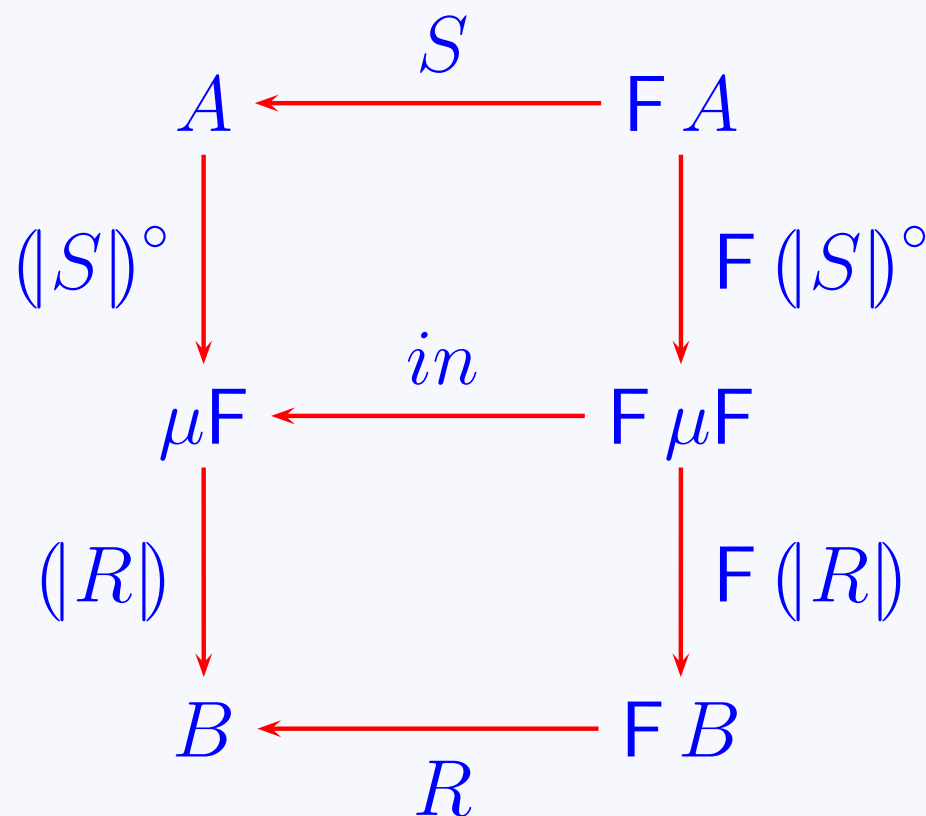
Taking converses:

$$\mu X.(R \cdot F X \cdot S^\circ) = (\lceil R \rceil) \cdot (\lceil S \rceil)^\circ$$



Hylo-factorization Theorem

Entire /simple factorization if both R and S° are
entire /simple ($=S$ surjective /injective)



Virtual data-structuring

- Particular choice of F for sub-problem organization induces **intermediate** type μF .
This is made explicit by hylo-factorization.
- **Intermediate** data-structure saves the outcome of a “one go” **divide** step $(S)^\circ$ and passes it on to the **conquer** step (R) for processing.
- In general, people “**fuse**” things very early in design, thus **virtualizing** this structure.
- Factorization helps in spec **understanding** and **classification**.

Final note on inductive relation \prec

Is such that the validity of a predicate ϕ can be proved by **structural induction** over it:

$$(\forall a. \phi a) \Leftarrow \underbrace{(\forall a. \phi a \Leftarrow (\forall c \prec a. \phi c))}_{\text{induction step}}$$

which corresponds to pointfree definition

$$\prec \setminus R \subseteq R \Rightarrow \top \subseteq R$$

where R generalizes ϕ such that $\phi a = aRb$, for some fixed b .