An Introduction to Relational Formal Modelling

J.N. Oliveira

DI/UM, 2003-06

October 26, 2006

Functions are not enough

Partiality (in Haskell):

```
Mpi> (split head tail)(tail [1])

Program error: {head []}

Mpi> 2/0

Program error: {primDivDouble 2.0 0.0}
```

Functions such as `tail`, `/`, `head` (and many others!) are **partial**

Functions are not enough

VDM-SL notation:

```
vdm> p tl [ ]
l. 1, c. 4:
  Run-Time Error 77: The sequence was empty
vdm>  p 2/0
l. 1, c. 3:
  Run-Time Error 76: Division with zero
vdm>
```

Functions such as `tl`, `/`, `hd` (and many others!) are **partial**

Functions are not enough

Another example is

```
gets : set of nat -> nat * set of nat
gets(s) == let a in set s
           in mk_( a , s \ {a} ) ;
```

which is not only partial

```
vdm> p gets({})
/home/jno/work/x.vdm, l. 4, c. 25:
  Run-Time Error 53: The binding environment was empty
vdm>
```

but also **non-deterministic**:

$$gets\{a,b\} = \langle a, \{b\}\rangle \vee gets\{a,b\} = \langle b, \{a\}\rangle$$

Specifications as "properties"

- Specification of square root:

  $$(sqrt\ x)^2 = x$$

  that is

  $$sq \cdot sqrt = id$$

  ($= sqrt$ has left inverse sq)
- Specification of $sort$:

  $$l' = sort\, l \quad \Leftrightarrow \quad (IsOrdered\ l') \ \wedge \ IsPermutation(l', l)$$

Relational approach

Need to model

- total/**partial** functions
- **non-determinism**
- **properties**, datatype **invariants** and loop-invariants
- orders and **inductive** structures
- vagueness or **under-specification**...

$\Rightarrow$ adoption of binary relations, which have a long tradition in the...

Pre/post specification style

VDM-SL notation:

```
gets(s: set of nat) r: (nat * set of nat)
pre  card s > 0
post let mk_(n,s') = r
     in  n in set s  and  s' = s \ {n}  ;
```

Testing this in the interpreter:

```
>> p post_gets( {1,2} , mk_(2, {1}) )
true
>> p post_gets( {1,2} , mk_(2 , {1,2}) )
false
```

Pre/post specification style

Tolerance to incomplete design:

```
Sort(inp: seq of int) out: seq of int
post IsPermutation(out,inp) and IsOrdered(out);

IsPermutation: seq of int * seq of int -> bool
IsPermutation(l1,l2) == is not yet specified;

IsOrdered: seq of int -> bool
IsOrdered(l) == is not yet specified;
```

Of course ...

```
>> p post_Sort( [] , [] )
/Users/jno/work/x.vdm, l. 10, c. 3:
  Run-Time Error 233: Cannot evaluate 'not yet specified' functions
```

Pre/post specification layout

VDM-SL generic layout is

```
Spec(inp: A) out: B
pre Precond(inp)
post Postcond(out,inp);
```

where

```
Precond: A -> bool
Precond(a) == .....

Postcond: B * A -> bool
Postcond(b,a) == .....
```

This clearly leads to a **binary relation** approach, since $Postcond \in 2^{B \times A}$ is equivalent to $Postcond \subseteq B \times A$

From predicates to relations

- **Predicate logic** connectives such as eg. $\wedge$ are "overloaded" operators
- They can be regarded as models of a more structured logic — that of **binary relations**
- **Functions** generalize to **binary relations** in a very natural way.
- Predicates, sets, functions and relations can **all** be combined in a **single relational calculus**
- Usual infix notation, *e.g.* $a < b$, can be generalized to any relation R, *e.g.* $a \ R \ b$

Arrows "as" binary relations

- We will **"type"** relations in a way consistent with functions: $B \xleftarrow{R} A$ wherever $b\ R\ a$ is a statement involving $b \in B$ and $a \in A$.
- From now on, an arrow

$$B \xleftarrow{R} A$$

means a **binary relation** from A (source) to B (target) and write $b\ R\ a$ to denote that pair $\langle b, a \rangle$ is in R, eg.

$$\begin{array}{ccc} 1 & \leq & 2 \\ \texttt{John} & IsFatherOf & \texttt{Mary} \\ 3 & = (1+) & 2 \end{array}$$

Functions are relations

- Regard function $f : A \longrightarrow B$ as a binary relation relating a and b iff $b = f\ a$. So, $b\ f\ a$ literally means $b = f\ a$.
- Therefore, generalize

$$\boxed{B \xleftarrow{f} A \quad b = f\ a} \qquad \text{to} \qquad \boxed{B \xleftarrow{R} A \quad b\ R\ a}$$

- Extend **function composition** $f \cdot g$ to $R \cdot S$ in the obvious way

$$b(R \cdot S)c \quad \equiv \quad \langle \exists\ a\ ::\ b\ R\ a\ \wedge\ a\ S\ c \rangle$$

Check generalization

Back to functions,

$$
\begin{array}{lll}
b(f \cdot g)c & \equiv & \langle \exists\ a\ ::\ b\ f\ a\ \wedge\ a\ g\ c \rangle \\
& \equiv & \quad \{\ b\ f\ a \text{ means } b = f\ a\ \} \\
& & \langle \exists\ a\ ::\ b = f\ a\ \wedge\ a = g\ c \rangle \\
& \equiv & \quad \{\ \text{cancel } \exists \text{ by substitution of } a \text{ by } g\ c\ \} \\
& & b = f(g\ c)
\end{array}
$$

we recover what we had before.

Notation convention: function (relation) identifiers are always lowercase (uppercase) letters.

Relations generalize functions

- **Equality** on functions,

$$f = g \quad \equiv \quad \langle \forall\ a\ :\ a \in A :\ f\ a\ =_B\ g\ a \rangle$$

  generalizes to **ordering** relations:

$$R \subseteq S \quad \equiv \quad \langle \forall\ b, a\ ::\ bRa \Rightarrow bSa \rangle$$

- $R \subseteq S$ means that R is either **less defined** or **more deterministic** than S.
- As we shall see later on:

$$f \subseteq g \quad \equiv \quad f = g \quad \equiv \quad f \supseteq g$$

Converses

Every relation $B \xleftarrow{R} A$ has a **converse** $A \xleftarrow{R^\circ} B$ which is such that, for all a, b,

$$a(R^\circ)b \quad \equiv \quad b\ R\ a$$

Function converses f°, g° etc. always exist (as **relations**) enjoying the following (very useful!) property:

$$(f\ b)R(g\ a) \equiv b(f^\circ \cdot R \cdot g)a$$

Let us see examples of its use

From pointwise to PF-notation

Function

$$\begin{array}{rcl} fac\ 0 & = & 1 \\ fac(n+1) & = & (n+1) * fac\ n \end{array}$$

in pointfree notation:

$$fac \cdot [\underline{0}\ , suc] = * \cdot [suc\ , fac]$$

Property

$$\langle \forall\ n, m\ ::\ n \leq m \Rightarrow fac\ n \leq fac\ m \rangle$$

($\equiv$ fac is monotonic) in PF-notation?

Properties in PF style

fac is monotonic:

$$\begin{array}{ll} & \langle \forall\ n, m\ ::\ n \leq m \Rightarrow fac\ n \leq fac\ m \rangle \\ \equiv & \quad \{ \text{ rule } (f\ b)R(g\ a) \equiv b(f^\circ \cdot R \cdot g)a \ \} \\ & \langle \forall\ n, m\ ::\ n \leq m \Rightarrow n(fac^\circ \cdot \leq \cdot fac)m \rangle \\ \equiv & \quad \{ \text{ ordering on relations (ie. dropping } n \text{ and } m) \ \} \\ & \leq \ \subseteq \ fac^\circ \cdot \leq \cdot fac \end{array}$$

From PF to pointwise notation

Role of id:

- Recall that every function f is a relation such that $b\ f\ a$ means $b = f\ a$.
- Therefore, function id is the **equality** relation: $b\ id\ a$ means $b = a$.

For example, what does

$$f^\circ \cdot f\ \subseteq\ id$$

mean? Let us introduce points and calculate:

From PF to pointwise notation

$$
\begin{array}{ll}
 & f^\circ \cdot f \subseteq id \\
\equiv & \quad \{ \text{ relational inclusion } \} \\
 & \langle \forall\ y, x\ ::\ y(f^\circ \cdot f)x \Rightarrow y(id)x \rangle \\
\equiv & \quad \{ \text{ introduce } id \text{ in antecedent ; } id \text{ is equality relation } \} \\
 & \langle \forall\ y, x\ ::\ y(f^\circ \cdot id \cdot f)x \Rightarrow y = x \rangle \\
\equiv & \quad \{ \text{ rule } (f\ b)R(g\ a) \equiv b(f^\circ \cdot R \cdot g)a \ \} \\
 & \langle \forall\ y, x\ ::\ (f\ y) = (f\ x) \Rightarrow y = x \rangle \\
\equiv & \quad \{ \text{ recall definition from discrete maths } \} \\
 & f \text{ is injective}
\end{array}
$$

Why id (really) matters

Terminology:

- Say R is reflexive iff $id \subseteq R$
  pointwise: $\langle \forall\ a\ ::\ a\ R\ a \rangle$ (check as homework);
- Say R is coreflexive iff $R \subseteq id$
  pointwise: $\langle \forall\ a\ ::\ b\ R\ a \Rightarrow b = a \rangle$ (check as homework).

Define, for $B \xleftarrow{R} A$:

Kernel of R	Image of R
$\ker R$	$\mathsf{img}\ R$
$A \longleftarrow A$	$B \longleftarrow B$
$\ker R \stackrel{\text{def}}{=} R^\circ \cdot R$	$\mathsf{img}\ R \stackrel{\text{def}}{=} R \cdot R^\circ$

Example: kernel of a function

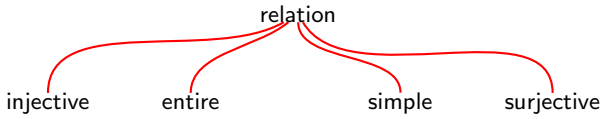
Let us go pointwise:

$$
\begin{array}{ll}
 & a'(\ker f)a \\
\equiv & \quad \{\ \text{substitution}\ \} \\
 & a'(f^\circ \cdot f)a \\
\equiv & \quad \{\ \text{rule}\ (f\ b)R(g\ a) \equiv b(f^\circ \cdot R \cdot g)a\ \} \\
 & (f\ a') = (fa)
\end{array}
$$

In words: $a'(\ker f)a$ means a' and a "have the same f"

Binary relation taxonomy

Topmost criteria:

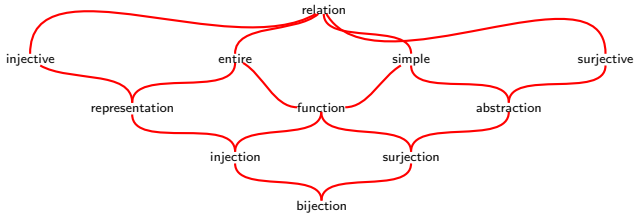


Definitions:

	Reflexive	Coreflexive
ker R	entire R	injective R
img R	surjective R	simple R

Binary relation taxonomy

The whole picture:



In general, "larger than entire means entire" and "smaller than simple means simple"

Functions in one slide

A function f is a binary relation such that

Pointwise	Pointfree	
"Left" Uniqueness		
$b\ f\ a\ \wedge\ b'\ f\ a \quad \Rightarrow \quad b = b'$	$\mathsf{img}\ f \quad \subseteq \quad id$	(f is simple)
Leibniz principle		
$a = a' \quad \Rightarrow \quad f\ a = f\ a'$	$id \quad \subseteq \quad \ker f$	(f is entire)

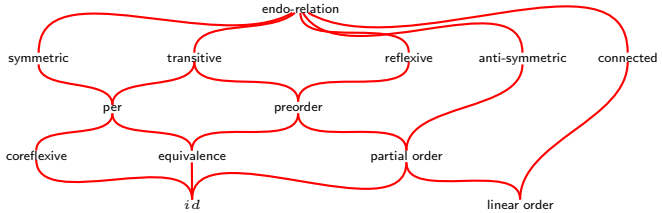
As we shall see, these two principles are captured by

$$f \cdot R \subseteq S \quad \equiv R \subseteq f^\circ \cdot S$$

$$R \cdot f^\circ \subseteq S \quad \equiv R \subseteq S \cdot f$$

Relation taxonomy — orders

Orders are endo-relations $A \xleftarrow{R} A$ classified as



(Criteria definitions: next slide)

Orders and their taxonomy

Besides

reflexive:	iff $id_A \subseteq R$
coreflexive:	iff $R \subseteq id_A$

an order (or endo-relation) $A \xleftarrow{R} A$ can be

transitive:	iff $R \cdot R \subseteq R$
anti-symmetric:	iff $R \cap R^\circ \subseteq id_A$
symmetric:	iff $R \subseteq R^\circ (\equiv R = R^\circ)$
connected:	iff $R \cup R^\circ = \top$

where $A \xleftarrow{\top} A$ is the largest relation of its type.

Orders and their taxonomy

Therefore:

- **Preorders** are reflexive and transitive orders.
  Example: $y\ IsAtMostAsOldAs\ x$
- **Partial** orders are anti-symmetric preorders
  Example: $y \subseteq x$
- **Linear** orders are connected partial orders
  Example: $y \leq x$
- **Equivalences** are symmetric preorders
  Example: $y\ IsPermutation\ x$
- **Pers** are partial equivalences
  Example: $y\ IsBrotherOf\ x$

Predicates & sets made PF

Strategy: identify every

- binary **predicate** $B \times A \xrightarrow{p} 2$ with binary relation $B \xleftarrow{\lceil p \rceil} A$

  such that $b \lceil p \rceil a \equiv p(b, a)$.
- unary **predicate** $A \xrightarrow{q} 2$ with coreflexive $A \xleftarrow{\lceil q \rceil} A$ such that $a \lceil q \rceil a' \equiv a = a' \ \wedge \ (q\ a)$.
- **set** $S \subseteq A$ with $\lceil \lambda a.a \in S \rceil$. So,

$$b \lceil S \rceil a \equiv b = a \ \wedge \ a \in S$$

Last but not least

- Intersection (meet):

$$b(R \cap S)a \quad \equiv \quad (b\ R\ a)\ \wedge\ (b\ S\ a)$$

- Union (join):

$$b(R \cup S)a \quad \equiv \quad (b\ R\ a) \vee (b\ S\ a)$$

- Bottom relation $B \xleftarrow{\bot} A$ — the smallest relation of its type:
  $\langle \forall\ b, a\ ::\ b \bot a \equiv false \rangle$
- Top relation $B \xleftarrow{\top} A$ — the largest relation of its type:
  $\langle \forall\ b, a\ ::\ b \top a \equiv true \rangle$

Of course, for all $B \xleftarrow{R} A$ one has $\bot \subseteq \quad R \quad \subseteq \top$

Last but not least

Coreflexives:

- Example: $\lceil fac\ n \leq 1 \rceil = \begin{array}{c} 0 \xleftarrow{} 0 \\ 1 \xleftarrow{} 1 \end{array}$
- Useful properties:
  - Coreflexives are **symmetric** and **transitive**:

$$R = R^{\circ} = R \cdot R = R \cap id$$

  - **Meet** of two coreflexives is composition:

$$R \cap S = R \cdot S$$

Last but not least

Domain:

$$\delta\, R \quad = \quad \mathsf{ker}\, R \cap id$$

Range:

$$\rho\, R \quad = \quad \mathsf{img}\, R \cap id$$

Facts:

$$\delta\, R \quad = \quad \rho\, R^{\circ}$$

$$\delta\,(R \cdot S) \quad = \quad \delta\,(\delta\, R \cdot S) \qquad , \qquad \rho\,(R \cdot S) \quad = \quad \rho\,(R \cdot \rho\, S)$$

$$R \quad = \quad R \cdot (\delta\, R) \;\; , \;\; R \quad = \quad (\rho\, R) \cdot R$$

PF-meaning of VDM-SL specs

Given $bool \xleftarrow{prec} A$ and $bool \xleftarrow{postc} B \times A$, VDM spec

```
Spec(a: A) r: B
pre prec(a)
post postc(r,a);
```

means binary relation $B \xleftarrow{Spec} A$ defined by

$$Spec \quad \stackrel{\mathrm{def}}{=} \quad \lceil postc \rceil \cdot \lceil prec \rceil$$

where $A \xleftarrow{\lceil prec \rceil} A$ is coreflexive and $B \xleftarrow{Post\, =\, \lceil postc \rceil} A$ is of the same type as $Spec$.

Example: VDM-SL $Sqrt$ spec

```
Sqrt(x: real) r: real
pre true
post sq(r) = x ;
```

means

$$
\begin{array}{rcl}
 & Sqrt & \stackrel{\text{def}}{=} \lceil sq\ r = x \rceil \cdot id \\
\equiv & & \{\ (f\ b)R(g\ a) \equiv b(f^\circ \cdot R \cdot g)a\ ;\ \text{natural-}id\ \} \\
 & Sqrt & \stackrel{\text{def}}{=} \lceil r(sq^\circ \cdot id \cdot id)x \rceil \\
\equiv & & \{\ \text{natural-}id\ \text{(twice)}\ \} \\
 & Sqrt & \stackrel{\text{def}}{=} sq^\circ
\end{array}
$$

Pre/post-consistency

In

$$Spec \stackrel{\text{def}}{=} \lceil postc \rceil \cdot \lceil prec \rceil$$

it is common practice to ensure that the pre-condition be at most the domain of the post-condition:

$$\lceil prec \rceil \subseteq \delta \lceil postc \rceil$$

Example:

```
Inv(x: real) r: real
pre x > 0
post r = 1 / x ;
```

Pre/post-consistency

By rule $(f\ b)R(g\ a) \equiv b(f^\circ \cdot R \cdot g)a$ we easily obtain $\lceil Inv \rceil = (1/)$, which is undefined for $x = 0$. Consistency is ensured since $x > 0 \Rightarrow x \neq 0$.
Counter example:

```
Get(x: seq of int) r: int
pre len x = card elems x
post r = hd x ;
```

(no repeats property does not ensure list non-emptiness).

Invariants

Note that the same problem would be found in

```
Get(x: NoRepeatsList) r: int
post r = hd x ;
```

where datatype $NoRepeatsList$ is defined with an invariant

```
types
NoRepeatsList = seq of int
inv x == len x = card elems x;
```

Data type invariants

Arising from natural phenomena:

```
dateOk : nat1 * nat1 * nat1 -> bool
dateOk(y,m,d) ==
     if m in set {1,3,5,7,8,10,12} then
          d <= 31 and
          (not (y=1582 and m=10) or (d<5) or (14<d))
     else if m in set {4,6,9,11} then d <= 30
     else if m=2 and leapYear(y) then d <= 29
     else if m=2 and not leapYear(y) then d <= 28
     else false;
```

where

Data type invariants

```
leapYear : nat1 -> bool
leapYear(y) ==
     0 = rem(y, if 1700 <= y and rem(y,100)=0
                then 400 else 4);
```

Business rules are also invariants, cf eg.

```
Account :: personal: Data
           avgbalance: real
           currentbalance: real
  inv a == a.currentbalance > - a.avgbalance/3 ;
```

Invariants can thus be as arbitrary as human inventiveness...

Invariants entail proof obligations

Let

- ϕ be the invariant associated to datatype A and ψ be the one associated to datatype B.
- f be a function of type $B \overset{f}{\longleftarrow} A$

Clearly,

- **Guarantee** — it is the onus of the designer of f to ensure that every output of f will satisfy ψ
- **Rely** — designer of f can rely on ϕ holding for every input

Formally:

$$\langle \forall\ a\ ::\ \phi\ a \Rightarrow \psi(f\ a) \rangle$$

Data type invariants

PF-transform of **proof obligation**, where $\Phi = \lceil \phi \rceil$ and $\Psi = \lceil \psi \rceil$:

$$f \cdot \Phi \subseteq \Psi \cdot f$$

cf.

$$\begin{array}{ll} & f \cdot \Phi \subseteq \Psi \cdot f \\ \equiv & \quad \{ \text{ shunting rule } \} \\ & \Phi \subseteq f^\circ \cdot \Psi \cdot f \\ \equiv & \quad \{ \text{ introduce variables } \} \\ & \langle \forall\ a, a' \ ::\ a\ \Phi\ a' \Rightarrow (f\ a) \Psi (f\ a') \rangle \\ \equiv & \quad \{ \text{ coreflexives } (a = a') \ \} \\ & \langle \forall\ a \ ::\ a\ \Phi\ a \Rightarrow (f\ a) \Psi (f\ a) \rangle \end{array}$$

Data type invariants

$$\begin{array}{ll} \equiv & \quad \{ \text{ back to predicates } \} \\ & \langle \forall\ a \ ::\ \phi\ a \Rightarrow \psi (f\ a) \rangle \end{array}$$

Another way to write it:

$$\begin{array}{ll} & f \cdot \Phi \subseteq \Psi \cdot f \\ \equiv & \quad \{ \text{ shunting } \} \\ & f \cdot \Phi \cdot f^\circ \subseteq \Psi \\ \equiv & \quad \{ \text{ coreflexives } \} \\ & f \cdot \Phi \cdot \Phi^\circ \cdot f^\circ \subseteq \Psi \end{array} \qquad \begin{array}{ll} \equiv & \quad \{ \text{ image definition } \} \\ & \mathsf{img}\,(f \cdot \Phi) \subseteq \Psi \\ \equiv & \quad \{ f \cdot \Phi \text{ is simple } \} \\ & \rho\,(f \cdot \Phi) \subseteq \Psi \end{array}$$

Data type invariants

In general — let $Spec = (pre, post)$ be a pre/post-condition pair in VDM-SL:

```
Spec(a: A) r: A
pre ... a ...
post ... r ... a ... ;
```

Let inv be an invariant property associated with type A:

```
A = .....
inv a == ... ;
```

Data type invariants

Invariant preservation proof obligation:

$$\langle \forall\ r, a\ ::\ post(r,a)\ \wedge\ pre\ a\ \wedge\ inv\ a \Rightarrow inv\ r \rangle \tag{1}$$

which PF-transforms to

$$\rho\,(Spec \cdot Inv) \subseteq Inv \tag{2}$$

for $Spec = \lceil post \rceil \cdot \lceil pre \rceil$ and $Inv = \lceil inv \rceil$.

- Question: How do we reason about invariants?

We first need to know how to calculate with PF-expressions.

Basic Relation Calculus

Monotonicity: All operations are monotonic, eg.

$$\frac{R \subseteq S \quad T \subseteq U}{(R \cdot T) \subseteq (S \cdot U)} \qquad \frac{R \subseteq S}{R^\circ \subseteq S^\circ}$$

Composition:

- Composition is associative: $R \cdot (S \cdot T) = (R \cdot S) \cdot T$
- Identity: $R \cdot id = id \cdot R = R$
- Empty relation: $R \cdot \bot = \bot \cdot R = \bot$

Basic Relation Calculus

Relational Equality:

- **Pointwise** equality:

$$R = S \quad \equiv \quad \langle \forall\ b, a\ ::\ bRa \equiv bSa \rangle$$

- **Pointfree** equality:
  - **Cyclic inclusion** ("ping-pong") rule:

$$R = S \quad \equiv \quad R \subseteq S \wedge S \subseteq R$$

  - **Indirect equality** rules:

$$\begin{array}{lll} R = S & \equiv & \langle \forall\ X\ ::\ (X \subseteq R \equiv X \subseteq S) \rangle \\ & \equiv & \langle \forall\ X\ ::\ (R \subseteq X \equiv S \subseteq X) \rangle \end{array}$$

Basic Relation Calculus

Converse:

$$\begin{array}{ll} {}^\circ\text{-universal:} & X^\circ \subseteq Y \quad \equiv \quad X \subseteq Y^\circ \\ {}^\circ\text{-monotonicity:} & R \subseteq S \equiv R^\circ \subseteq S^\circ \end{array}$$

Then:

$$\begin{array}{ll} \text{Involution :} & (R^\circ)^\circ = R \\ \text{Contravariance :} & (R \cdot S)^\circ = S^\circ \cdot R^\circ \end{array}$$

These can be proved from ${}^\circ$**-universal** by (elegant) indirect proofs (example in next slide):

Basic Relation Calculus

Indirect proof of involution

$$\begin{array}{ll} & (R^\circ)^\circ \subseteq Y \\ \equiv & \quad \{\ {}^\circ\text{-universal}\quad X^\circ \subseteq Y \quad \equiv \quad X \subseteq Y^\circ \quad \text{for } X := R^\circ\ \} \\ & R^\circ \subseteq Y^\circ \\ \equiv & \quad \{\ {}^\circ\text{-monotonicity}\ \} \\ & R \subseteq Y \\ :: & \quad \{\ \text{indirection}\ \} \\ & (R^\circ)^\circ = R \end{array}$$

Meet, join and converse

$\cap$-universal

$$X \subseteq (R \cap S) \quad \equiv \quad (X \subseteq R) \wedge (X \subseteq S)$$

$\cup$-universal

$$R \cup S \subseteq X \quad \equiv \quad (R \subseteq X) \wedge (S \subseteq X)$$

Converse distributes over $\cap$ and $\cup$ (see eg. calculation of next slide):

$$(R \cap S)^\circ = R^\circ \cap S^\circ$$

Another indirect proof

$$\begin{array}{ll}
 & X \subseteq R^\circ \cap S^\circ \\
\equiv & \quad \{ \cap\text{-universal} \} \\
 & (X \subseteq R^\circ) \wedge (X \subseteq S^\circ) \\
\equiv & \quad \{ \text{monotonicity and involution} \} \\
 & (X^\circ \subseteq R) \wedge (X^\circ \subseteq S) \\
\equiv & \quad \{ \cap\text{-universal} \} \\
 & X^\circ \subseteq R \cap S \\
\equiv & \quad \{ \text{monotonicity and involution} \} \\
 & X \subseteq (R \cap S)^\circ \\
:: & \quad \{ \text{indirection} \} \\
 & R^\circ \cap S^\circ = (R \cap S)^\circ
\end{array}$$

Reasoning about functions

Shunting rules:

$$\begin{array}{lll} f \cdot R \subseteq S & \equiv & R \subseteq f^\circ \cdot S \\ R \cdot f^\circ \subseteq S & \equiv & R \subseteq S \cdot f \end{array}$$

Equality:

$$f \subseteq g \equiv f = g \equiv f \supseteq g$$

"Ping-pong" proof of the equality rule follows.

Proof of functional equality

$$
\begin{array}{ll}
 & f \subseteq g \\
\equiv & \quad \{ \text{ identity } \} \\
 & f \cdot id \subseteq g \\
\equiv & \quad \{ \text{ shunting on } f \ \} \\
 & id \subseteq f^\circ \cdot g \\
\equiv & \quad \{ \text{ shunting on } g \ \} \\
 & id \cdot g^\circ \subseteq f^\circ \\
\equiv & \quad \{ \text{ converses; identity } \} \\
 & g \subseteq f
\end{array}
$$

Adding structure to the calculus

Note a recurrent **pattern** in several laws above:

$$
\begin{array}{lcl}
\underbrace{X^\circ}_{f\ X} \subseteq Y & \equiv & X \subseteq \underbrace{Y^\circ}_{g\ Y} \\
\underbrace{(h\cdot)X}_{f\ X} \subseteq Y & \equiv & X \subseteq \underbrace{(h^\circ\cdot)Y}_{g\ Y} \\
\underbrace{X(\cdot h^\circ)}_{f\ X} \subseteq Y & \equiv & X \subseteq \underbrace{Y(\cdot h)}_{g\ Y}
\end{array}
$$

as well as in

$$
\underbrace{(d\times)q}_{f\ q} \leq n \quad \equiv \quad q \leq \underbrace{n(/d)}_{g\ n}
$$

Back to the primary school desk

The **integral division** algorithm

$$\begin{array}{r|l} 7 & 2 \\ \hline 1 & 3 \end{array} \qquad 2 \times 3 + 1 = 7 \quad \text{, "ie."} \qquad 3 = 7/2$$

However

$$\begin{array}{r|l} 7 & 2 \\ \hline 3 & 2 \end{array} \qquad 2 \times 2 + 3 = 7 \quad \wedge \quad 2 \neq 7/2$$

$$\begin{array}{r|l} 7 & 2 \\ \hline 5 & 1 \end{array} \qquad 2 \times 1 + 5 = 7 \quad \wedge \quad 1 \neq 7/2$$

Quotient is a supremum

$$\begin{array}{r|l} n & d \\ \hline r & q \end{array} \qquad d \times q + r = n \equiv q = n/d$$

provided q is the largest such q (r smallest)

$$\begin{aligned} n/d &= \bigvee \{ q \mid \exists r \,.\, d \times q + r = n \} \\ &= \bigvee \{ q \mid d \times q \leq n \} \end{aligned}$$

Maths teachers tell: it takes a while before children master the "$\bigvee$ semantics"!
What about you? Can you easily reason about n/d in this format?
Try and prove $(n/m)/d = n/(d \times m)$.

"Universal" property instead

Alternative:

$$\begin{array}{r|l} n & d \\ \hline r & q \end{array} \qquad q \times d \leq n \equiv q \leq n/d$$

"universal" property of integral division

Reasoning:

$$\begin{array}{ll} & q \leq (n/m)/d \\ \equiv & \quad \{ \text{ "universal" property } \} \\ & q \times d \leq n/m \end{array}$$

Reasoning continued

$$
\begin{array}{ll}
\equiv & \{ \text{ "universal" property again } \} \\
& (q \times d) \times m \leq n \\
\equiv & \{ \times \text{ is associative } \} \\
& q \times (d \times m) \leq n \\
\equiv & \{ \text{ "universal" property again } \} \\
& q \leq n/(d \times m)
\end{array}
$$

Indirect equality

So we have

$$q \leq (n/m)/d \quad \equiv \quad q \leq n/(d \times m)$$

that is,

$$(n/m)/d \quad = \quad n/(d \times m)$$

by the **indirect equality** rule:

$$(q \leq x \equiv q \leq y) \equiv (x = y)$$

Also easy to check

Cancellation law:

$$
\begin{array}{ll}
& (n/d) \times d \leq n \\
\equiv & \{ \text{ universal property } \} \\
& n/d \leq n/d \\
\equiv & \{ \text{ reflexive } \leq \ \} \\
& true
\end{array}
$$

"Reflection":

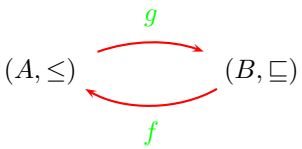
$$
\begin{array}{ll}
& 1 \times d \leq n \equiv 1 \leq n/d \\
\equiv & \{ \ 1 \text{ is the unit of } \times \ \} \\
& d \leq n \equiv n/d \geq 1
\end{array}
$$

Galois connections

n/d is a Galois connection:

$$\begin{array}{r|l} n & d \\ \hline r & q \end{array} \qquad \underbrace{q \times d}_{f\ q} \leq n \equiv q \leq \underbrace{n/d}_{g\ n}$$

In general, for **preorders** $(A, \leq)$ and $(B, \sqsubseteq)$ and



(f, g) are Galois connected iff...

Galois adjoints

$$\underbrace{f}_{\text{lower adjoint}}\ b \leq a \quad \equiv \quad b \sqsubseteq \underbrace{g}_{\text{upper adjoint}}\ a$$

that is

$$f^{\circ} \cdot \leq \quad = \quad \sqsubseteq \cdot g$$

Remarks:

- Galois (connected) adjoints enjoy a number of interesting generic properties
- Very elegant — **calculational** — way of performing inequational reasoning (including logical deduction)

Basic properties

Cancellation:

$$(f \cdot g)a \leq a \quad \text{and} \quad b \sqsubseteq (g \cdot f)b$$

Distribution (in case of lattice structures):

$$\begin{aligned} f(a \sqcup a') &= (f\ a) \vee (f\ a') \\ g(b \wedge b') &= (g\ b) \sqcap (g\ b') \end{aligned}$$

Conversely,

- If f distributes over $\sqcup$ then it has an upper adjoint g ($f^{\#}$)
- If g distributes over $\wedge$ then it has a lower adjoint f ($g^{\flat}$)

Other properties

If (f, g) are Galois connected,

- f (g) **uniquely** determines g (f) — thus the $\_^{\flat}$, $\_^{\sharp}$ notations
- f and g are **monotonic**
- (g, f) are also Galois connected — **reverse** the orderings
- $f = f \cdot g \cdot f$ and $g = g \cdot f \cdot g$

etc

Summary

$(f\ b) \leq a \equiv b \sqsubseteq (g\ a)$		
Description	$f = g^{\flat}$	$g = f^{\sharp}$
Definition	$f\ b = \bigwedge\{a \mid b \sqsubseteq g\ a\}$	$g\ a = \bigsqcup\{b \mid f\ b \leq a\}$
Cancellation	$f(g\ a) \leq a$	$b \sqsubseteq g(f\ b)$
Distribution	$f(b \sqcup b') = (f\ b) \vee (f\ b')$	$g(a' \sqcap a) = (g\ a') \sqcap (g\ a)$
Monotonicity	$b \sqsubseteq b' \Rightarrow f\ b \leq f\ b'$	$a \leq a' \Rightarrow g\ a \sqsubseteq g\ a'$

Converse

$(f\ X) \subseteq Y \equiv X \subseteq (g\ Y)$			
Description	$f = g^\flat$	$g = f^\sharp$	**Obs.**
converse	$(\_)^\circ$	$(\_)^\circ$	$bR^\circ a \equiv aRb$

Thus:

Cancellation	$(R^\circ)^\circ = R$
Monotonicity	$R \subseteq S \equiv R^\circ \subseteq S^\circ$
Distributions	$(R \cap S)^\circ = R^\circ \cap S^\circ, (R \cup S)^\circ = R^\circ \cup S^\circ$

Functions

$(f\ X) \subseteq Y \equiv X \subseteq (g\ Y)$			
Description	$f = g^\flat$	$g = f^\sharp$	**Obs.**
shunting rule	$(h\cdot)$	$(h^\circ\cdot)$	NB: h is a function
"converse" shunting rule	$(\cdot h^\circ)$	$(\cdot h)$	NB: h is a function

Consequences:

Functional equality:	$h \subseteq g \equiv \quad h = k \quad \equiv h \supseteq k$
Functional division:	$h^\circ \cdot R = h \setminus R$

Question: what does $h \setminus R$ mean?

Relational division

$(f\ X) \subseteq Y \equiv X \subseteq (g\ Y)$			
Description	$f = g^\flat$	$g = f^\sharp$	**Obs.**
left-division	$(R\cdot)$	$(R \setminus\ )$	left-factor
right-division	$(\cdot R)$	$(\ /\ R)$	right-factor

Immediate: $(R\cdot)$ and $(\cdot R)$ distribute over union:

$$\begin{array}{rcl} R \cdot (S \cup T) & = & (R \cdot S) \cup (R \cdot T) \\ (S \cup T) \cdot R & = & (S \cdot R) \cup (T \cdot R) \end{array}$$

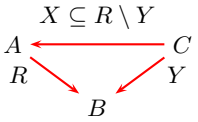
Some intuition about relational division operators follows.

Relational division

The **relational division** operators are upper-adjoints:

$$R \cdot X \subseteq Y \equiv X \subseteq R \setminus Y$$
$$X \cdot R \subseteq Y \equiv X \subseteq Y / R$$

Left division abstracts a (pointwise) universal quantification

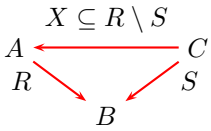


$$a(R \setminus Y)c \quad \equiv \quad \langle \forall\ b\ ::\ bRa \Rightarrow bYc \rangle$$

An example follows.

Example

Recall data division in the **relational model**:



$$a(R \setminus S)c \equiv (\forall b.bRa \Rightarrow bSc)$$

$b\ R\ a$ = flight b carries passenger a

$b\ S\ c$ = flight b belongs to air-company c

$a\ (R \setminus S)\ c$ = passenger a is faithful to company c, that is, (s)he only flies company c.

Right division

By taking converses we arrive at $S \,/\, R = (R^\circ \setminus S^\circ)^\circ$:

$$
\begin{array}{ll}
 & X \subseteq S \,/\, R \\
\equiv & \quad \{ \text{ Galois connection } ((\cdot R), (/R)) \quad \} \\
 & X \cdot R \subseteq S \\
\equiv & \quad \{ \text{ converses } \quad \} \\
 & R^\circ \cdot X^\circ \subseteq S^\circ \\
\equiv & \quad \{ \text{ Galois connection } ((R\cdot), (R\backslash)) \quad \} \\
 & X^\circ \subseteq R^\circ \setminus S^\circ \\
\equiv & \quad \{ \text{ converses } \} \\
 & X \subseteq (R^\circ \setminus S^\circ)^\circ
\end{array}
$$

ie. Galois connection

$$X \cdot R \subseteq S \quad \equiv \quad X \subseteq S \,/\, R$$

Meet

$\cap$-universal

$$X \subseteq (R \cap S) \quad \equiv \quad (X \subseteq R) \wedge (X \subseteq S)$$

is a Galois connection

$$(\Delta, \cap)$$

where $\Delta\ X = (X, X)$, cf.

$$(X, X)(\subseteq \times \subseteq)(R, S) \quad \equiv \quad X \subseteq \cap(R, S)$$

So $\cap = \Delta^\sharp$ distributes over itself, etc

Properties of $\cap$

From $\cap$-universal infer:

- **$\cap$-cancellation** $(X := R \cap S)$

$$R \cap S \subseteq R \quad \wedge \quad R \cap S \subseteq S$$

- **$\cap$-abbreviation** $(X := R)$

$$R \subseteq S \quad \equiv \quad R = R \cap S$$

- **$\cap$-idempotency** $(S := R)$

$$R \cap R \quad = \quad R$$

More properties of $\cap$

$\cap$ is **commutative**:

$$R \cap S \quad = \quad S \cap R$$

$\cap$ is **associative**:

$$R \cap (S \cap T) \quad = \quad (R \cap S) \cap T$$

$\cap$-fusion:

$$\begin{array}{lcl} T \cdot (R \cap S) & \subseteq & (T \cdot R) \cap (T \cdot S) \\ (R \cap S) \cdot T & \subseteq & (R \cdot T) \cap (S \cdot T) \end{array}$$

Meet and join

$(f\ X) \leq Y \equiv X \sqsubseteq (g\ Y)$			
Description	$f = g^\flat$	$g = f^\sharp$	**Obs.**
meet	Δ	$\cap$	$\leq$ is $(\subseteq \times \subseteq)$
join	$\cup$	Δ	$\sqsubseteq$ is $(\subseteq \times \subseteq)$

Join:

$$\cup(R,S) \subseteq Y \equiv (R,S)(\subseteq \times \subseteq)(Y,Y)$$

that is,

$$R \cup S \subseteq Y \equiv R \subseteq Y \ \wedge \ S \subseteq Y$$

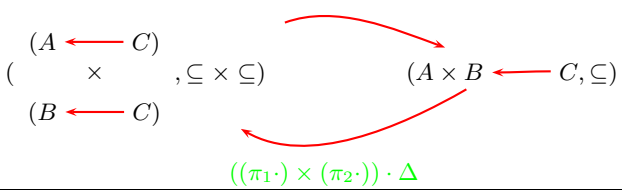
Relational split

Functions:

$$x = \langle f, g\rangle \quad \equiv \quad \pi_1 \cdot x = f \quad \wedge \quad \pi_2 \cdot x = g$$

Relations:

$$X \subseteq \langle R, S\rangle \quad \equiv \quad \pi_1 \cdot X \subseteq R \quad \wedge \quad \pi_2 \cdot X \subseteq S$$



Properties

$\langle -, - \rangle$ is an **upper-adjoint**, so it distributes over meet

$$\begin{aligned} \langle R, S \cap T\rangle &= \langle R, S\rangle \cap \langle R, T\rangle \\ \langle S \cap T, R\rangle &= \langle S, R\rangle \cap \langle T, R\rangle \end{aligned}$$

etc. Moreover:

$$\langle R, S\rangle = (\pi_1^\circ \cdot R) \cap (\pi_2^\circ \cdot S) \qquad (3)$$

Why? Again Galois at work:

Calculation

$$
\begin{array}{lcl}
X \subseteq \langle R, S\rangle & \equiv & \pi_1 \cdot X \subseteq R \ \wedge \ \pi_2 \cdot X \subseteq S \\
& \equiv & \quad \{ \text{ Galois connected } ((f\cdot), (f^\circ\cdot)) \ \} \\
& & X \subseteq \pi_1^\circ \cdot R \ \wedge \ X \subseteq \pi_2^\circ \cdot S \\
& \equiv & \quad \{ \text{ Galois connected } (\cap^\flat, \cap) \ \} \\
& & X \subseteq (\pi_1^\circ \cdot R \cap \pi_2^\circ \cdot S) \\
& :: & \quad \{ \text{ indirect equality } \} \\
& & \langle R, S\rangle = \pi_1^\circ \cdot R \cap \pi_2^\circ \cdot S
\end{array}
$$

Pointwise $\langle R, S\rangle$

$$
\begin{array}{lcl}
(a,b)\langle R, S\rangle c & \equiv & (a,b)(\pi_1^\circ \cdot R \cap \pi_2^\circ \cdot S)c \\
& \equiv & \quad \{ \text{ pointwise } \cap \ \} \\
& & (a,b)(\pi_1^\circ \cdot R)c \ \wedge \ (a,b)(\pi_2^\circ \cdot S)c \\
& = & \quad \{ \text{ rule } (f\ b)Ra \equiv b(f^\circ \cdot R)a \ \} \\
& & \pi_1(a,b)Rc \ \wedge \ \pi_2(a,b)Sc \\
& = & \quad \{ \text{ projections } \} \\
& & aRc \ \wedge \ bSc
\end{array}
$$

Relational either

Functions:

$$[f\ , g] = x \quad \equiv \quad f = x \cdot i_1 \ \wedge \ g = x \cdot i_2$$

Relations:

$$[R\ , S] \subseteq X \quad \equiv \quad R \subseteq X \cdot i_1 \ \wedge \ S \subseteq X \cdot i_2 \tag{4}$$

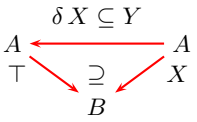
Thus $[\_\ , \_]$ is a **lower-adjoint**, it distributes over $\cup$, etc. Moreover,

$$[R\ , S] \quad = \quad (R \cdot i_1^\circ) \cup (S \cdot i_2^\circ) \tag{5}$$

Domain and range

$(f\ X) \subseteq Y \equiv X \subseteq (g\ Y)$			
Description	$f = g^\flat$	$g = f^\sharp$	**Obs.**
domain	δ	$(\top\cdot)$	lower $\subseteq$ restricted to coreflexives
range	ρ	$(\cdot\top)$	lower $\subseteq$ restricted to coreflexives

cf.



$$\delta\ X \subseteq Y \equiv X \subseteq \top \cdot Y$$

Domain and split

The following fact holds:

$$\langle R, S\rangle^\circ \cdot \langle X, Y\rangle \quad = \quad (R^\circ \cdot X) \cap (S^\circ \cdot Y)$$

Corollary:

$$\delta\ R \quad = \quad \ker \langle id, R\rangle$$

Another consequence of the fact above:

$$\ker R \subseteq \ker (S \cdot R) \quad \Leftarrow \quad S \text{ entire}$$

Corollary:

$$\ker R \quad \subseteq \quad \ker (f \cdot R)$$

Modular law

Dedekind's rule (also known as the **modular law**):

$$(R \cdot S) \cap T \quad \subseteq \quad R \cdot (S \cap (R^\circ \cdot T))$$

Dually (apply converses and rename):

$$(R \cdot S) \cap T \quad \subseteq \quad (R \cap (T \cdot S^\circ)) \cdot S$$

Symmetrical equivalent statement:

$$(R \cdot S) \cap T \quad \subseteq \quad (R \cap (T \cdot S^\circ)) \cdot (S \cap (R^\circ \cdot T))$$

= "weak right-distribution of meet over composition".

Comprehending relations

For each $B \xleftarrow{R} A$ define its **graph** or comprehension by

$$\mathcal{G}\ R = \{(b, a) \mid bRa\}$$

Clearly, $R = \lceil \mathcal{G}\ R \rceil$ and so we often abbreviate $\mathcal{G}\ \mathcal{R}$ to R.
The graph of every **coreflexive** S is made simpler for obvious reasons:

$$\mathcal{G}\ S = \{a \mid aSa\}$$

Finite relations

R is said to be finite wherever $\mathcal{G}\ \mathcal{R}$ is a finite set.

- Finite relations, which can be enumerated, browsed and stored in a computer, are the subject of **relational database** design.
- Every finite, **simple** relation expresses a **functional dependency**.
- The graphs of finite and simple relations are called **mappings** in VDM-SL terminology.
- We will use Greek literals (σ, τ etc) to denote (finite) mappings

VDM-SL mapping notation

- Datatype: $\texttt{map}\ A\ \texttt{to}\ B$
- Pointwise VDM-SL concrete syntax

$$\{a \mapsto b \mid b\,\sigma\,a\}$$

  replaces $\{(b,a) \mid b\,\sigma\,a\}$.

- In VDM-SL notation, $b\,\sigma\,a$ is furthermore rephrased as $a \in \delta\,\sigma \ \wedge\ b = \sigma(a)$ — cf. $\sigma = \sigma \cdot \delta\,\sigma$ — that is, we have

$$\sigma \quad = \quad \{a \mapsto \sigma(a) \mid a \in \delta\,\sigma\}$$

Relational projection

Given a binary relation R and suitably typed functions f and g,

- the g, f-projection of R is defined as binary relation

$$\pi_{g,f} R \quad \stackrel{\text{def}}{=} \quad g \cdot R \cdot f^{\circ} \tag{6}$$

- wherever R is simple and $g \cdot R \cdot f^{\circ}$ is also simple, we write $f \rightharpoonup g$ instead of $\pi_{g,f} R$. So,

$$f \rightharpoonup g \stackrel{\text{def}}{=} (g\cdot) \cdot (\cdot f^{\circ})$$

- $(f \rightharpoonup g)R$ is always simple when f is injective.

Mappings in specifications

Example: VDM-SL $Sort$ spec (sorting) in VDM-SL notation:

```
Sort(l: seq of int) r: seq of int
post IsOrdered(r) and IsPermutation(r,l);
```

where

```
IsPermutation: seq of int * seq of int -> bool
IsPermutation(l1,l2) ==
  forall e in set (elems l1 union elems l2) &
    card {i | i in set inds l1 & l1(i) = e} =
    card {i | i in set inds l2 & l2(i) = e};
```

Example: VDM-SL $Sort$ spec

. . . abbreviates to

$$\begin{aligned} Sort &\stackrel{\text{def}}{=} \lceil IsOrdered \rceil \cdot IsPermutation \\ &= \lceil IsOrdered \rceil \cdot (\ker seq2bag) \end{aligned}$$

assuming

```
seq2bag: seq of int -> map int to nat1
seq2bag(l) ==
  { e |-> card { i | i in set inds l & l(i) = e } |
      e in set elems l };
```

`IsPermutation` is an **equivalence** because ker f always is reflexive, symmetric and transitive.

Relational semantics of VDM-SL

From the VDM-SL on-line manual:

Operator	Name	Semantics description
`s <: m`	Domain restrict to	Creates the map consisting of the elements in `m` whose key is in `s`. `s` need not be a subset of `dom m`.

Formal **semantics**:

$$\lceil s \mathbin{<:} m \rceil = \lceil m \rceil \cdot \lceil s \rceil$$

where $\lceil s \rceil$ is coreflexive and $\lceil m \rceil$ is simple.

Relational semantics of VDM-SL

From the VDM-SL on-line manual:

Operator	Name	Semantics description
`m1 ++ m2`	Override	overrides and merges `m1` with `m2`, i.e. it is like a merge except that `m1` and `m2` need not be compatible; any common elements are as by `m2` (so `m2` overrides `m1`.)

Formal **semantics**:

$$\lceil m_1 \mathbin{+\!\!+} m_2 \rceil \quad = \quad \lceil m_2 \rceil \rightarrow \lceil m_2 \rceil \ , \ \lceil m_1 \rceil$$

cf. relational **McCarthy** conditional:

Relational McCarthy conditional

It is defined by

$$R \rightarrow S \ , \ T \quad \stackrel{\text{def}}{=} \quad (S \cdot \delta \, R) \cup T \cdot (id - \delta \, R)$$

where

$(f \ X) \subseteq Y \equiv X \subseteq (g \ Y)$			
Description	$f = g^\flat$	$g = f^\sharp$	**Obs.**
difference	$(\_ - R)$	$(R \cup \ )$	

that is,

$$\begin{array}{rcl} X - R \subseteq Y & \equiv & X \subseteq R \cup Y \\ X - R & = & \bigcap \{Y \mid X \subseteq R \cup Y\} \end{array}$$

Reasoning about VDM-SL

We want to prove VDM-SL properties such as

$$\begin{aligned} X \mathbin{<:} (Y \mathbin{<:} \sigma) &= (X \cap Y) \mathbin{<:} \sigma \\ \{\} \mathbin{<:} \sigma &= \{\mapsto\} \\ X \mathbin{<:} (\sigma_1 \mathbin{++} \sigma_2) &= (X \mathbin{<:} \sigma_1) \mathbin{++} (X \mathbin{<:} \sigma_2) \end{aligned}$$

First, recall properties of coreflexives:

- Coreflexives are **symmetric** and **transitive**:

$$R = R^\circ = R \cdot R = R \cap id$$

- **Meet** of two coreflexives is composition:

$$R \cap S = R \cdot S$$

Example of proof

$$\begin{aligned} & \lceil X \mathbin{<:} (Y \mathbin{<:} \sigma) \rceil \\ = & \quad \{ \text{ relational meaning of } <: \ \} \\ & \lceil Y \mathbin{<:} \sigma \rceil \cdot \lceil X \rceil \\ = & \quad \{ \text{ relational meaning of } <: \ \} \\ & (\lceil \sigma \rceil \cdot \lceil Y \rceil) \cdot \lceil X \rceil \\ = & \quad \{ \text{ associativity of } \cdot \text{ and coreflexives } \} \\ & \lceil \sigma \rceil \cdot (\lceil X \rceil \cdot \lceil Y \rceil) \\ = & \quad \{ \text{ meet of two coreflexives is composition } \} \\ & \lceil \sigma \rceil \cdot (\lceil X \rceil \cap \lceil Y \rceil) \end{aligned}$$

Proof continued

$$
\begin{array}{ll}
 & \lceil\sigma\rceil \cdot (\lceil X\rceil \cap \lceil Y\rceil) \\
= & \quad \{ \text{ meaning of set intersection } \} \\
 & \lceil\sigma\rceil \cdot \lceil X \cap Y\rceil \\
= & \quad \{ \text{ relational meaning of <: } \} \\
 & \lceil (X \cap Y) <: \sigma\rceil
\end{array}
$$

Another proof

$$
\begin{array}{ll}
 & \lceil X <: (\sigma_1 \mathbin{++} \sigma_2)\rceil \\
= & \quad \{ \text{ relational meaning of <: and ++ } \} \\
 & (\lceil\sigma_2\rceil \to \lceil\sigma_2\rceil \ , \ \lceil\sigma_1\rceil) \cdot \lceil X\rceil \\
= & \quad \{ \text{ McCarthy fusion law } \} \\
 & \lceil\sigma_2\rceil \cdot \lceil X\rceil \to \lceil\sigma_2\rceil \cdot \lceil X\rceil \ , \ \lceil\sigma_1\rceil \cdot \lceil X\rceil \\
= & \quad \{ \text{ relational meaning of <: } \} \\
 & \lceil X <: \sigma_2\rceil \to \lceil X <: \sigma_2\rceil \ , \ \lceil X <: \sigma_1\rceil \\
= & \quad \{ \text{ relational meaning of ++ } \} \\
 & \lceil (X <: \sigma_1) \mathbin{++} (X <: \sigma_2)\rceil
\end{array}
$$

Etc.

Home work: define the relational semantics of *e.g.*.

Operator	Name	Semantics description
`m <-: s`	Domain restricted by	Creates the map consisting of the elements in `m` whose key is not in `s`. `s` need not be a subset of `dom m`.

and prove similar properties.

Override pointwise

Since

$$\delta\,(\sigma_1 \mathbin{+\!\!+} \sigma_2) = \delta\,\sigma_1 \cup \delta\,\sigma_2$$

we have, after expansion of the relational definition:

```
s1 ++ s2 ==
   { k |-> if k in set dom s2
           then s2(k)
           else s1(k)
        | k in set dom s1 union dom s2 }
```

Performing the above proof over this definition would have been far less compact.

Inductive override

Another version of map override:

```
s1 ++ s2 ==
   if s1 = {|->}
   then s2
   else let k in set dom s1
        in  { k |-> if k in set dom s2
              then s2(k)
              else s1(k) } munion { k } <-: s1 ++ s2
```

How do we arrive at this recursive scheme?

See next set of slides.