PF transform: where everything becomes a relation

J.N. Oliveira

Dept. Informática,
Universidade do Minho
Braga, Portugal

DI/UM, 2007 (last update: Oct-2014)

Motivation

So far, we have been using **predicate logic** in formalizing subtleties and complex aspects of real-life problems.

Question: Is this formalism the best for **formal modelling**?

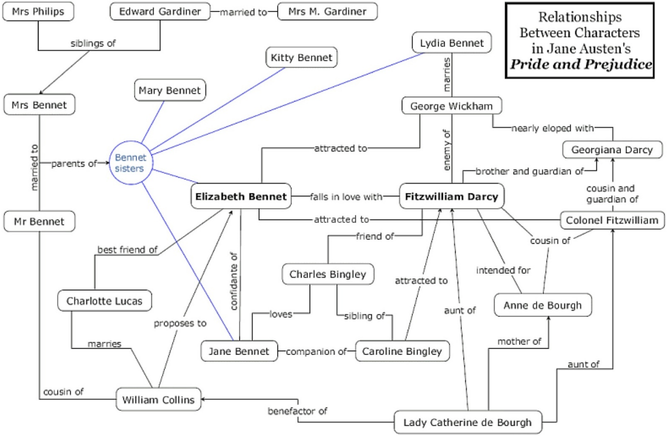
Historically, it was **not** the first to be proposed:

- Augustus de Morgan (1806-71) — recall *de Morgan* laws (12,13) — proposed a **Logic of Relations** as early as 1867.
- Predicate logic appeared later.

Perhaps de Morgan was right in the first place: in real life, "everything is a **relation**"...

Everything is a relation

... as diagram



shows. (Wikipedia: **Pride and Prejudice**, by Jane Austin, 1813.)

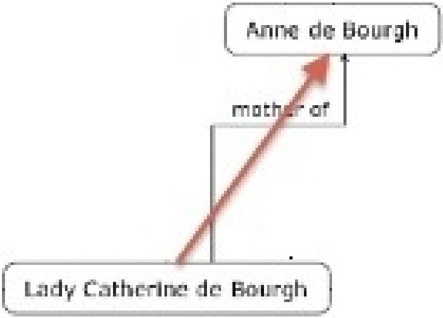
Arrow notation for relations

The picture is a collection of **relations** — vulg. a **semantic network** — elsewhere known as a (binary) **relational system**.

However, in spite of the use of **arrows** in the picture (aside) not many people would write

$$mother\_of : People \to People$$

as the **type** of **relation** $mother\_of$.



Pairs

Consider assertions

$$\begin{array}{rcl} 0 & \leq & \pi \\ \texttt{Catherine} & \mathit{isMotherOf} & \texttt{Anne} \\ 3 & =(1+) & 2 \end{array}$$

They are statements of fact concerning various kinds of object — real numbers, people, natural numbers, etc

They involve **two** such objects, that is, **pairs**

$$(0, \pi)$$
$$(\texttt{Catherine}, \texttt{Anne})$$
$$(3, 2)$$

respectively.

Sets of pairs

So, we might have written instead:

$$\begin{array}{rcl} (0, \pi) & \in & \leq \\ (\texttt{Catherine}, \texttt{Anne}) & \in & \mathit{isMotherOf} \\ (3, 2) & \in & (1+) \end{array}$$

What are $(\leq)$, $\mathit{isMotherOf}$, $(1+)$?

- they can be regarded as **sets of pairs**
- better, they should be regarded as **binary relations**.

Therefore,

- **orders** — eg. $(\leq)$ — are special cases of relations
- **functions** — eg. $\mathit{succ} \triangleq (1+)$ — are special cases of relations.

Binary Relations

Binary relations are typed:

> **Arrow notation.** *Arrow* $A \xrightarrow{R} B$ *denotes a binary relation from* A *(source) to* B *(target).*

A, B are types. Writing $B \xleftarrow{R} A$ means the same as $A \xrightarrow{R} B$.

> **Infix notation.** *The usual infix notation used in natural language — eg.* $\mathtt{Catherine}$ *isMotherOf* $\mathtt{Anne}$ *— and in maths — eg.* $0 \leq \pi$ *— extends to arbitrary* $B \xleftarrow{R} A$ *: we write*
>
> $$b\ R\ a$$
>
> *to denote that* $(b, a) \in R$.

Binary Relations

Binary relations are typed:

Arrow notation. *Arrow* $A \xrightarrow{R} B$ *denotes a binary relation from* A *(source) to* B *(target).*

A, B are types. Writing $B \xleftarrow{R} A$ means the same as $A \xrightarrow{R} B$.

Infix notation. *The usual infix notation used in natural language — eg.* $\mathtt{Catherine}\ \mathit{isMotherOf}\ \mathtt{Anne}$ *— and in maths — eg.* $0 \leq \pi$ *— extends to arbitrary* $B \xleftarrow{R} A$ *: we write*

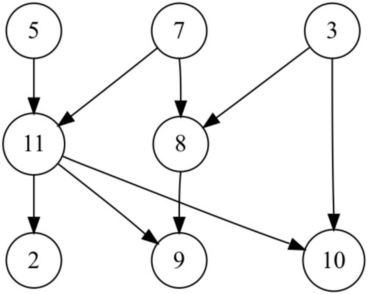
$$b\ R\ a$$

to denote that $(b, a) \in R$*.*

Binary relations are matrices

Binary relations can be regarded as Boolean **matrices**, eg.

Relation R:



Matrix M:

	1	2	3	4	5	6	7	8	9	10	11
1	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	1
3	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0
8	0	0	1	0	0	0	1	0	0	0	0
9	0	0	0	0	0	0	0	1	0	0	1
10	0	0	1	0	0	0	0	0	0	0	1
11	0	0	0	0	1	0	1	0	0	0	0

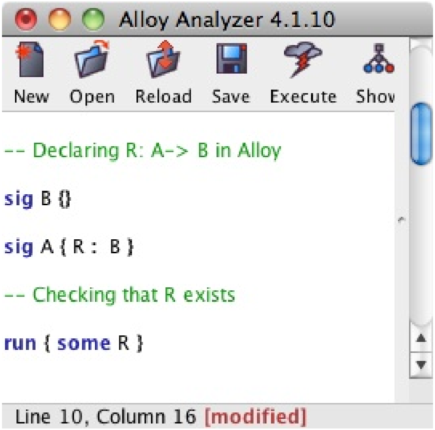
In this case $A = B = \{1..11\}$. Relations $A \xleftarrow{R} A$ over a single type are also referred to as (directed) **graphs**.

Alloy: where "everything is a relation"

Declaring binary relation $A \xrightarrow{R} B$ is **Alloy** (aside).

Alloy is a tool designed at MIT (`http://alloy.mit.edu/alloy`)

We shall be using **Alloy** later in this course.



Functions are relations

Lowercase letters (or identifiers starting by one such letter) will denote special relations known as **functions**, eg. f, g, $succ$, etc.

We regard **function** $f : A \longrightarrow B$ as the binary relation which relates b to a iff $b = f\ a$. So,

$$b\ f\ a \ \text{ literally means } \ b = f\ a \tag{52}$$

Therefore, we generalize

$$\boxed{\begin{array}{c} B \xleftarrow{f} A \\ b = f\ a \end{array}} \quad \text{to} \quad \boxed{\begin{array}{c} B \xleftarrow{R} A \\ b\ R\ a \end{array}}$$

Exercise

Taken from PROPOSITIONES AD ACUENDOS IUUENES ("Problems to Sharpen the Young"), by abbot Alcuin of York († 804):

> XVIII. PROPOSITIO DE HOMINE ET CAPRA ET LVPO.
> *Homo quidam debebat ultra fluuium transferre lupum, capram, et fasciculum cauli. Et non potuit aliam nauem inuenire, nisi quae duos tantum ex ipsis ferre ualebat. Praeceptum itaque ei fuerat, ut omnia haec ultra illaesa omnino transferret. Dicat, qui potest, quomodo eis illaesis transire potuit?*

Exercise

XVIII. FOX, GOOSE AND BAG OF BEANS PUZZLE. *A farmer goes to market and purchases a fox, a goose, and a bag of beans. On his way home, the farmer comes to a river bank and hires a boat. But in crossing the river by boat, the farmer could carry only himself and a single one of his purchases - the fox, the goose or the bag of beans. (If left alone, the fox would eat the goose, and the goose would eat the beans.) Can the farmer carry himself and his purchases to the far bank of the river, leaving each purchase intact?*

Identify the main **types** and **relations** involved in the puzzle and draw them in a diagram.

Propositio de homine et capra et lvpo

Data types:

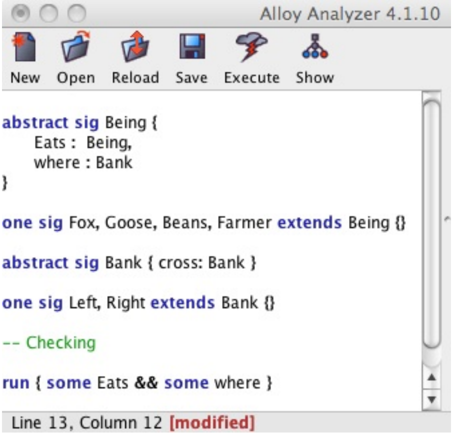
$$Being = \{Farmer, Fox, Goose, Beans\} \tag{53}$$
$$Bank = \{Left, Right\} \tag{54}$$

Relations:

$$\begin{tikzcd} Being \arrow[r, "Eats"] & Being \arrow[d, "where"'] & \\ & Bank \arrow[r, "cross"] & Bank \end{tikzcd} \tag{55}$$

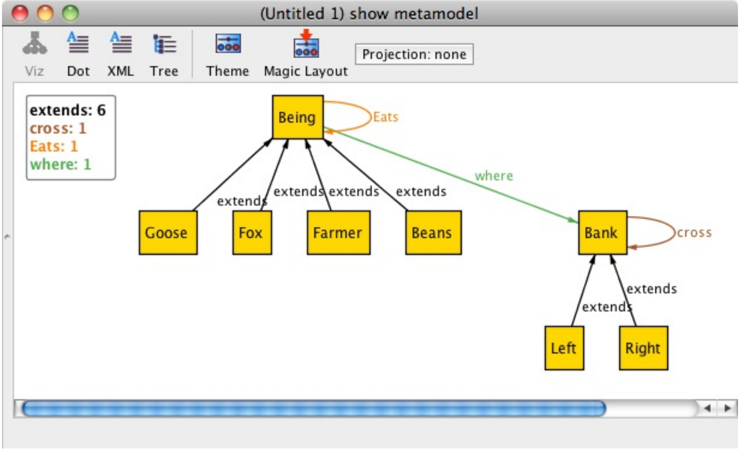
PROPOSITIO DE HOMINE ET CAPRA ET LVPO

Specification source written in Alloy:



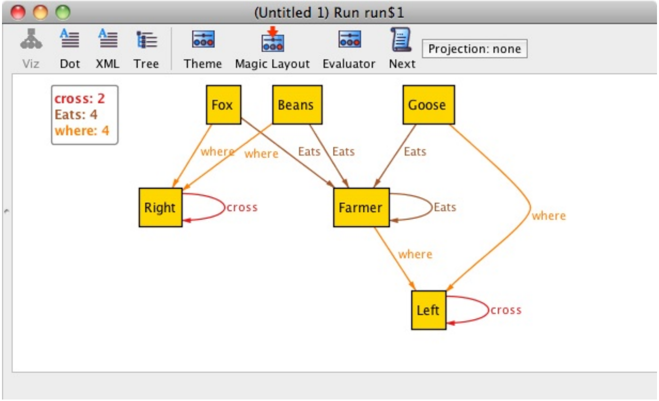
Propositio de homine et capra et lvpo

Diagram of specification (model) given by Alloy:



PROPOSITIO DE HOMINE ET CAPRA ET LVPO

Diagram of instance of the model given by Alloy:



Silly instance, why? — specification too **loose**...

Composition

Recall **function composition**

$$
\begin{tikzcd}
B & A \arrow[l, "f"'] & C \arrow[l, "g"'] \arrow[ll, bend left, "f \cdot g"]
\end{tikzcd}
\tag{56}
$$

$$b = f(g\ c)$$

We extend $f \cdot g$ to $R \cdot S$ in the obvious way:

$$b(R \cdot S)c \quad \equiv \quad \langle \exists\ a\ ::\ b\ R\ a\ \wedge\ a\ S\ c \rangle \tag{57}$$

Note how this rule *removes* $\exists$ when applied from right to left.

Notation $R \cdot S$ is said to be **point-free** (no variables, or points)

Check generalization

Back to functions, (57) becomes

$$
\begin{array}{ll}
b(f \cdot g)c & \equiv \quad \langle \exists\, a \;::\; b\, f\, a \wedge a\, g\, c \rangle \\
& \equiv \qquad \{\ a\ g\ c \text{ means } a = g\ c \text{ (52)} \ \} \\
& \quad\;\; \langle \exists\, a \;::\; b\, f\, a \wedge a = g\ c \rangle \\
& \equiv \qquad \{\ \exists\text{-trading} \ ;\ b\ f\ a \text{ means } b = f\ a \text{ (52)} \ \} \\
& \quad\;\; \langle \exists\, a \;:\; a = g\ c \;:\; b = f\ a \rangle \\
& \equiv \qquad \{\ \exists\text{-one point rule (15)} \ \} \\
& \quad\;\; b = f(g\ c)
\end{array}
$$

So, we easily recover what we had before (56).

Inclusion generalizes equality

- **Equality** on functions

$$f = g \quad \equiv \quad \langle \forall\ a\ :\ a \in A :\ f\ a\ =_B\ g\ a \rangle \tag{58}$$

generalizes to **inclusion** on relations:

$$R \subseteq S \quad \equiv \quad \langle \forall\ b, a\ :\ b\ R\ a :\ b\ S\ a \rangle \tag{59}$$

(read $R \subseteq S$ as "R is at most S")

- For $R \subseteq S$ to hold both R and S need to be of the same **type**, say $B \xleftarrow{R,S} A$
- $R \subseteq S$ is a partial order (reflexive, transitive and antisymmetric); therefore:

$$R \subseteq S \ \wedge\ S \subseteq R \quad \equiv \quad R = S \tag{60}$$

Special relations

Every type $B \longleftarrow A$ has its

- **bottom** relation $B \xleftarrow{\perp} A$, which is such that, for all b, a,
  $b \perp a \equiv \textsc{False}$
- **topmost** relation $B \xleftarrow{\top} A$, which is such that, for all b, a,
  $b \top a \equiv \textsc{True}$

Every type $A \longleftarrow A$ has the

- **identity** relation $A \xleftarrow{id} A$ which is nothing but function

$$id\ a \triangleq a \tag{61}$$

Clearly, for every R,

$$\perp \subseteq R \subseteq \top \tag{62}$$

Exercises

Exercise 22: Let $s\ S\ n$ mean: *"student s is assigned number n"*. Using (57) and (59), check that assertion

$$S \cdot \geq \quad \subseteq \quad \top \cdot S \tag{63}$$

means that numbers are assigned sequentially.

□

Exercise 23: Resort to PF-transform rule (57) and to the Eindhoven quantifier calculus to show that

$$R \cdot id \;=\; R \;=\; id \cdot R \tag{64}$$

$$R \cdot \bot \;=\; \bot \;=\; \bot \cdot R \tag{65}$$

hold and that composition is associative:

$$R \cdot (S \cdot T) \quad = \quad (R \cdot S) \cdot T \tag{66}$$

□

Converses

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

$$a(R^\circ)b \quad \equiv \quad b\ R\ a \tag{67}$$

Note that converse commutes with composition

$$(R \cdot S)^\circ = S^\circ \cdot R^\circ \tag{68}$$

and with itself:

$$(R^\circ)^\circ = R \tag{69}$$

Converse captures the **passive voice**: *Catherine eats the apple* — $R = (eats)$ — the same as *the apple is eaten by Catherine* — $R^\circ = (is\ eaten\ by)$.

Function converses

Function converses f°, g° etc. always exist (as **relations**) and enjoy the following (very useful) PF-transform property:

$$(f\ b)R(g\ a) \ \equiv \ b(f^\circ \cdot R \cdot g)a \tag{70}$$

cf. diagram:

$$\begin{array}{ccc} C & \xleftarrow{R} & D \\ {\scriptstyle f}\uparrow & & \uparrow{\scriptstyle g} \\ B & \xleftarrow[f^\circ \cdot R \cdot g]{} & A \end{array}$$

Therefore (tell why):

$$b(f^\circ \cdot g)a \ \equiv \ f\ b = g\ a \tag{71}$$

Let us see an example of using these rules.

PF-transform at work

Transforming a well-known PW-formula:

$$
\begin{array}{ll}
& f \text{ is } \textbf{injective} \\
\equiv & \qquad \{ \text{ recall definition from discrete maths } \} \\
& \langle \forall\, y, x \ :\ (f\ y) = (f\ x) :\ y = x \rangle \\
\equiv & \qquad \{ \text{ (71) for } f = g \ \} \\
& \langle \forall\, y, x \ :\ y(f^{\circ} \cdot f)x :\ y = x \rangle \\
\equiv & \qquad \{ \text{ (70) for } R = f = g = id \ \} \\
& \langle \forall\, y, x \ :\ y(f^{\circ} \cdot f)x :\ y(id)x \rangle \\
\equiv & \qquad \{ \text{ go pointfree (59) i.e. drop } y, x \ \} \\
& f^{\circ} \cdot f \subseteq id
\end{array}
$$

The other way round

Now check what $id \subseteq f \cdot f^\circ$ means:

$$
\begin{array}{ll}
 & id \subseteq f \cdot f^\circ \\
\equiv & \quad \{ \text{ relational inclusion (59) } \} \\
 & \langle \forall\, y, x \;:\; y(id)x :\; y(f \cdot f^\circ)x \rangle \\
\equiv & \quad \{ \text{ identity relation ; composition (57) } \} \\
 & \langle \forall\, y, x \;:\; y = x :\; \langle \exists\, z \;::\; y\ f\ z \wedge z\ f^\circ x \rangle\rangle \\
\equiv & \quad \{ \text{ } \forall\text{-one point (14) ; converse (67) } \} \\
 & \langle \forall\, x \;::\; \langle \exists\, z \;::\; x\ f\ z \wedge x\ f\ z \rangle\rangle \\
\equiv & \quad \{ \text{ trivia ; function } f \text{ } \} \\
 & \langle \forall\, x \;::\; \langle \exists\, z \;::\; x = f\ z \rangle\rangle \\
\equiv & \quad \{ \text{ recalling definition from maths } \} \\
 & f \text{ is } \textbf{surjective}
\end{array}
$$

Why *id* (really) matters

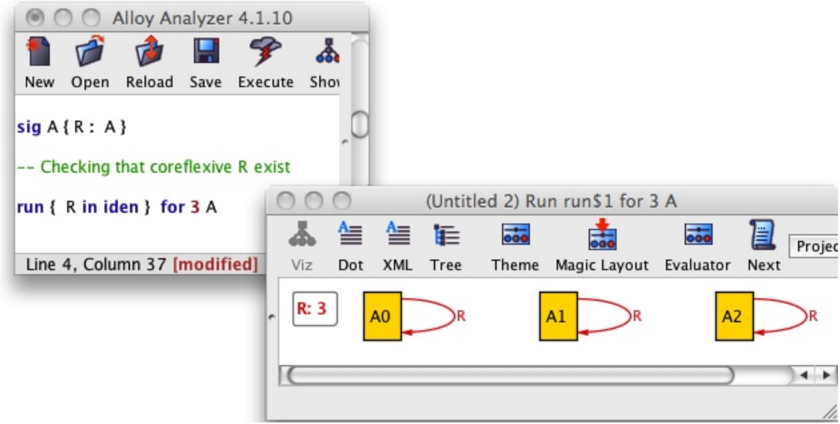
Terminology:

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

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

Kernel of R	**Image** of R
$A \xleftarrow{\ker R} A$ $\ker R \stackrel{\text{def}}{=} R^\circ \cdot R$	$B \xleftarrow{\operatorname{img} R} B$ $\operatorname{img} R \stackrel{\text{def}}{=} R \cdot R^\circ$

Alloy: checking for coreflexive relations



Example: kernels of functions

Meaning of $\ker f$:

$$
\begin{array}{cl}
 & a'(\ker f)a \\
\equiv & \{ \text{ substitution } \} \\
 & a'(f^\circ \cdot f)a \\
\equiv & \{ \text{ rule (71) } \} \\
 & f\ a' = f\ a
\end{array}
$$

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

Exercise 24: Let K be a nonempty data domain, $k \in K$ and $\underline{k}$ be the *"everywhere* k*"* function:

$$
\begin{array}{rcl}
\underline{k} & : & A \longrightarrow K \\
\underline{k}\,a & \triangleq & k
\end{array}
\tag{72}
$$

Compute which relations are defined by the following PF-expressions:

$$
\ker \underline{k} \quad , \quad \underline{b} \cdot \underline{c}^\circ \quad , \quad \operatorname{img} \underline{k}
\tag{73}
$$

Binary relation taxonomy

Topmost criteria:

$$\begin{tikzcd}[column sep=small] & & \text{binary relation} \arrow[dll, no head] \arrow[dl, no head] \arrow[dr, no head] \arrow[drr, no head] & & \\ \text{injective} & \text{entire} & & \text{simple} & \text{surjective} \end{tikzcd}$$

Definitions:

	Reflexive	*Coreflexive*
$\ker R$	entire R	injective R
$\operatorname{img} R$	surjective R	simple R

(74)

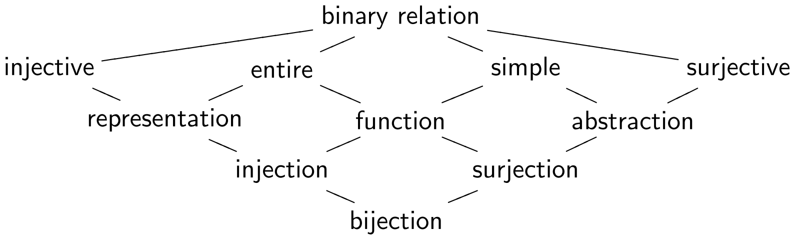
Facts:

$$\ker(R^\circ) = \operatorname{img} R \quad (75)$$

$$\operatorname{img}(R^\circ) = \ker R \quad (76)$$

Binary relation taxonomy

The whole picture:



(77)

Exercise 25: Resort to (75,76) and (74) to prove the following rules of thumb:

- converse of **injective** is **simple** (and vice-versa)
- converse of **entire** is **surjective** (and vice-versa)

□

Exercise

Exercise 26: Prove the following fact

$$\textit{A function } f \textit{ is a bijection } \textbf{iff} \textit{ its converse } f^\circ \textit{ is a function} \tag{78}$$

by completing:

$$
\begin{array}{cl}
 & f \text{ and } f^\circ \text{ are functions} \\
\equiv & \quad \{ \ \ldots \ \} \\
 & (id \subseteq \ker f \wedge \operatorname{img} f \subseteq id) \wedge (id \subseteq \ker (f^\circ) \wedge \operatorname{img} (f^\circ) \subseteq id) \\
\equiv & \quad \{ \ \ldots \ \} \\
 & \vdots \\
\equiv & \quad \{ \ \ldots \ \} \\
 & f \text{ is a bijection}
\end{array}
$$

□

PROPOSITIO DE HOMINE ET CAPRA ET LVPO

Exercise 27: Check which of the following properties,

simple , *entire* , *injective* ,
surjective , *reflexive* ,
coreflexive

hold for relation *Eats* (55) aside
("food chain" $Fox > Goose > Beans$).

	Fox	*Goose*	*Beans*
Fox	0	1	0
Goose	0	0	1
Beans	0	0	0

□

Exercise 28: Let relation $Bank \xrightarrow{cross} Bank$ (55) be defined by:

$$\begin{array}{ccc} Left & cross & Right \\ Right & cross & Left \end{array}$$

It therefore is a bijection. Why?

□

PROPOSITIO DE HOMINE ET CAPRA ET LVPO

Exercise 29: Relation $where : Being \to Bank$ should obey the following constraints:

- *everyone is somewhere in a bank*
- *no one can be in both banks at the same time.*

Encode such constraints in relational terms. Conclude that $where$ should be a **function**.

$\square$

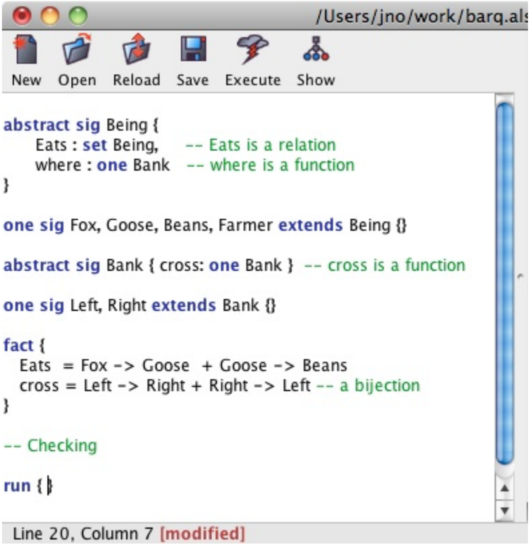
Exercise 30: There are only two constant functions in the type $Being \longrightarrow Bank$ of $where$. Identify them and explain the role they play in the puzzle.

$\square$

PROPOSITIO DE HOMINE ET CAPRA ET LVPO

Adding detail to the previous **Alloy** model (aside)

(More about Alloy syntax and semantics later.)



Functions in one slide

Recapitulating: a **function** f is a binary relation such that

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

NB: Following a widespread convention, functions will be denoted by lowercase characters (eg. f, g, ϕ) or identifiers starting with lowercase characters, and function application will be denoted by juxtaposition, eg. $f\ a$ instead of $f(a)$.

Functions, relationally

(The following properties of any function f are extremely useful.)

Shunting rules:

$$f \cdot R \subseteq S \quad \equiv \quad R \subseteq f^\circ \cdot S \tag{79}$$

$$R \cdot f^\circ \subseteq S \quad \equiv \quad R \subseteq S \cdot f \tag{80}$$

Equality rule:

$$f \subseteq g \equiv f = g \equiv f \supseteq g \tag{81}$$

Rule (81) follows from (79,80) by "cyclic inclusion" (next slide).

Proof of functional equality (81)

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

Thus $f = g \;\equiv\; f \subseteq g \wedge g \subseteq f \;\equiv\; f \subseteq g$ (same for $g \subseteq f$).

Exercises

Exercise 31: Infer $id \subseteq \ker f$ (f is total) and $\mathsf{img}\, f \subseteq id$ (f is simple) from any of the shunting rules (79) or (80).

□

Exercise 32: Check the meaning of shunting rules (79) and (80) by converting them to pointwise (Eindhoven) notation.

Show that they indeed hold by resorting to the rules of the Eindhoven calculus.

□

Exercises

Exercise 33: As generalization of exercise 22, draw the most general type diagram which accommodates relational assertion:

$$M \cdot R^\circ \quad \subseteq \quad \top \cdot M \tag{82}$$

□

Exercise 34: Type the following relational assertions

$$M \cdot N^\circ \quad \subseteq \quad \bot \tag{83}$$

$$M \cdot N^\circ \quad \subseteq \quad id \tag{84}$$

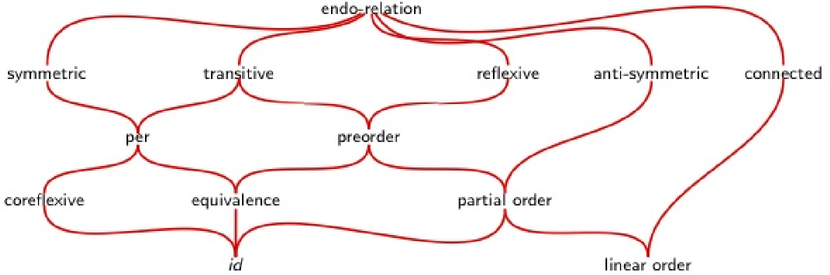
$$M^\circ \cdot \top \cdot N \quad \subseteq \quad > \tag{85}$$

and check their pointwise meaning.

□

Relation taxonomy — orders

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



(Criteria definitions: next slide)

Orders and their taxonomy

Besides

reflexive: iff $id \subseteq R$ (86)

coreflexive: iff $R \subseteq id$ (87)

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

transitive: iff $R \cdot R \subseteq R$ (88)

anti-symmetric: iff $R \cap R^\circ \subseteq id$ (89)

symmetric: iff $R \subseteq R^\circ (\equiv R = R^\circ)$ (90)

connected: iff $R \cup R^\circ = \top$ (91)

Orders and their taxonomy

Therefore:

- **Preorders** are reflexive and transitive orders.
  Example: $(age\ y) \leq (age\ x)$
- **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: $elems\ y = elems\ x$ (**kernels** of functions are **equivalences**)
- **Pers** are partial equivalences
  Example: $y\ IsBrotherOf\ x$

Exercises

Exercise 35: Check which of the following properties,

transitive , *symmetric* , *anti-symmetric* , *connected*

hold for the relation *Eats* of exercise 27.

□

Exercise 36: Suppose that finite lists are represented by **simple** relations of type $A \xleftarrow{L} \mathbb{N}$, that is, as mappings from **indices** ($\mathbb{N}$) to list **elements** (A). Assuming that A is equipped with a **total order** $<_A$, show that assertion

$$L \cdot < \cdot L^{\circ} \subseteq <_A \qquad (92)$$

specifies that L is a strictly **ordered** list.

□

Exercises

Exercise 37: Expand all criteria in the previous slides to pointwise notation.

□

Exercise 38: A relation R is said to be *co-transitive* iff the following holds:

$$\langle \forall\, b, a \;:\; b\, R\, a : \langle \exists\, c \;:\; b\, R\, c : c\, R\, a \rangle\rangle \tag{93}$$

Compute the PF-transform of the formula above. Find a relation (eg. over numbers) which is co-transitive and another which is not.

□

Meet and join

Meet (intersection) and join (union) lift pointwise conjunction and disjunction, respectively,

$$b\,(R \cap S)\,a \;\equiv\; b\,R\,a \wedge b\,S\,a \tag{94}$$

$$b\,(R \cup S)\,a \;\equiv\; b\,R\,a \vee b\,S\,a \tag{95}$$

for R, S of the same type. Their meaning is captured by the following **universal** properties:

$$X \subseteq R \cap S \;\equiv\; X \subseteq R \;\wedge\; X \subseteq S \tag{96}$$

$$R \cup S \subseteq X \;\equiv\; R \subseteq X \;\wedge\; S \subseteq X \tag{97}$$

NB: recall the notions of **greatest lower bound** and **least upper bound**, respectively.

In summary

Type $B \longleftarrow A$ forms a lattice:

$$\begin{tikzcd}[column sep=small]
 & \top \arrow[d, dotted, no head] & \\
 & R \cup S \arrow[dl, dotted, no head] \arrow[dr, dotted, no head] & \\
R \arrow[dr, no head] & & S \arrow[dl, no head] \\
 & R \cap S \arrow[d, dotted, no head] & \\
 & \bot &
\end{tikzcd}$$

"top"

join, lub ("least upper bound")

meet, glb ("greatest lower bound")

"bottom"

PROPOSITIO DE HOMINE ET CAPRA ET LVPO

Back to our running example, we specify:

Being at the same bank:

$$SameBank \quad = \quad ker\, where$$

Risk of somebody eating somebody else:

$$CanEat \quad = \quad SameBank \cap Eats$$

"Starving" ensured by $Farmer$*'s presence at the same bank:*

$$CanEat \quad \subseteq \quad SameBank \cdot \underline{Farmer} \tag{98}$$

Propositio de homine et capra et lvpo

By (79), "starving" **invariant** (98) converts to:

$$\mathit{where} \cdot \mathit{CanEat} \quad \subseteq \quad \mathit{where} \cdot \underline{\mathit{Farmer}}$$

In this version, invariant (98) can be depicted as a diagram:

$$\begin{tikzcd} \mathit{Being} \arrow[d, "\mathit{where}"'] & & \mathit{Being} \arrow[ll, "\mathit{CanEat}"'] \arrow[d, "\underline{\mathit{Farmer}}"] \arrow[dll, phantom, "\subseteq"] \\ \mathit{Bank} & & \mathit{Being} \arrow[ll, "\mathit{where}"] \end{tikzcd} \tag{99}$$

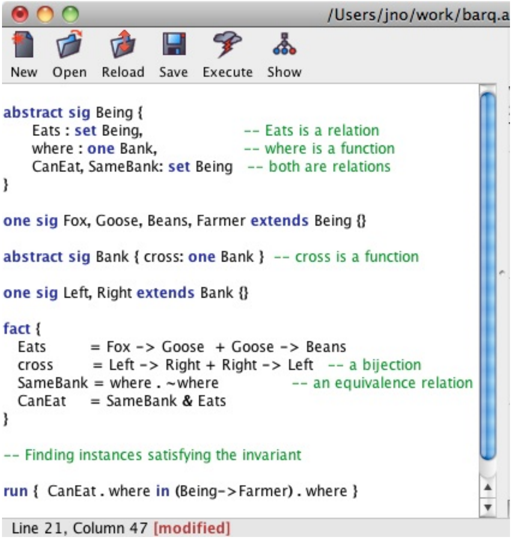
which "reads" in a nice way:

> *where* `(somebody)` *CanEat* `(somebody else) (that's)`
> *where* `(the)` *Farmer* `(is)`.

Propositio de homine et capra et lvpo

‘Starving’ invariant in **Alloy** (aside)

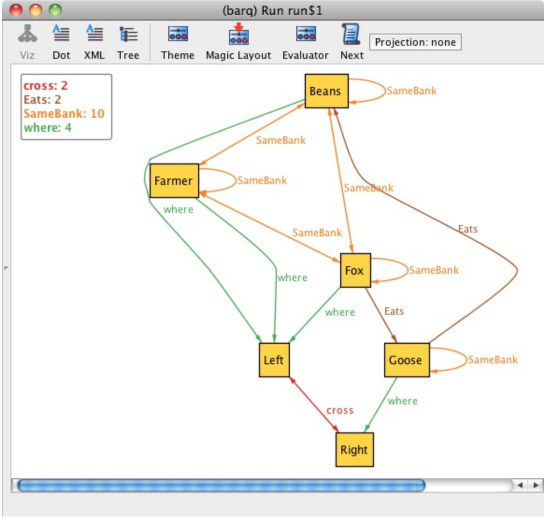
(Again we stress: missing details about Alloy syntax and semantics will be given later.)



Propositio de homine et capra et lvpo

Carefully observe instance of 'starving' invariant:

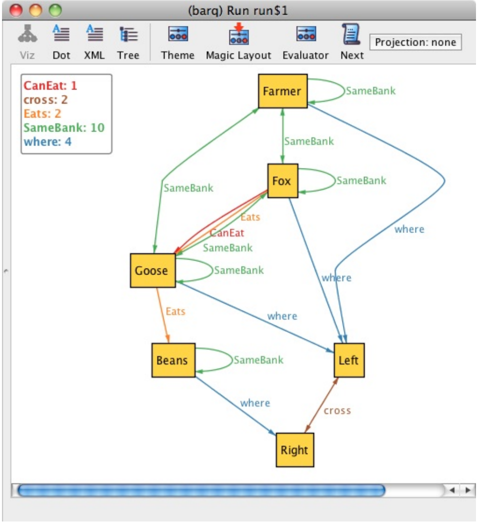
- *SameBank* is an **equivalence** — exactly the **kernel** of *where*
- *Eats* is simple but not transitive
- *cross* is a **bijection**
- *CanEat* is empty
- etc



Propositio de homine et capra et lvpo

In this other instance of 'starving' invariant:

- *CanEat* is **not** empty (*Fox* can eat *Goose*!)
- but *Farmer* is on the same bank :-)



Why is *SameBank* an equivalence?

Recall that $SameBank = \ker where$. Then:

Exercise 39: Knowing that property

$$f \cdot f^\circ \cdot f = f \tag{100}$$

holds for every function f, prove that $\ker f$ is an **equivalence** relation.
□

NB: Equivalence relations of this kind are captured in natural language by the textual pattern

$a(\ker f)b$ *the same as* *"a and b have the same f"*

which is very common in requirements.

Distributivity

As we will prove later, **composition** distributes over **union**

$$R \cdot (S \cup T) = (R \cdot S) \cup (R \cdot T) \qquad (101)$$
$$(S \cup T) \cdot R = (S \cdot R) \cup (T \cdot R) \qquad (102)$$

while distributivity over **intersection** is side-conditioned:

$$(S \cap Q) \cdot R = (S \cdot R) \cap (Q \cdot R) \;\Leftarrow\; \left\{ \begin{array}{c} Q \cdot \mathrm{img}\, R \subseteq Q \\ \vee \\ S \cdot \mathrm{img}\, R \subseteq S \end{array} \right. \qquad (103)$$

$$R \cdot (Q \cap S) = (R \cdot Q) \cap (R \cdot S) \;\Leftarrow\; \left\{ \begin{array}{c} (\ker R) \cdot Q \subseteq Q \\ \vee \\ (\ker R) \cdot S \subseteq S \end{array} \right. \qquad (104)$$

Monotonicity

All relational combinators seen so far are $\subseteq$-monotonic, namely:

$$R \subseteq S \;\Rightarrow\; R^\circ \subseteq S^\circ \tag{105}$$

$$R \subseteq S \wedge U \subseteq V \;\Rightarrow\; R \cdot U \subseteq S \cdot V \tag{106}$$

$$R \subseteq S \wedge U \subseteq V \;\Rightarrow\; R \cap U \subseteq S \cap V \tag{107}$$

$$R \subseteq S \wedge U \subseteq V \;\Rightarrow\; R \cup U \subseteq S \cup V \tag{108}$$

etc hold.

Exercise 40: Prove the following rules of thumb:

- **smaller** than injective (simple) is injective (simple)
- **larger** than entire (surjective) is entire (surjective)

□

Exercises (monotonicity)

Exercise 41: Check which of the following hold:

- If relations R and S are simple, then so is $R \cap S$
- If relations R and S are injective, then so is $R \cup S$
- If relations R and S are entire, then so is $R \cap S$

□

Exercise 42: Prove that relational **composition** preserves **all** relational classes in the taxonomy of (77).

□

Exercise 43: Show that the following conditional fusion law holds:

$$\langle R, S\rangle \cdot T = \langle R \cdot T, S \cdot T\rangle \quad \Leftarrow \quad R \cdot (\mathrm{img}\ T) \subseteq R \vee S \cdot (\mathrm{img}\ T) \subseteq S$$

□

Back to relational equality

Although **relational equality** could be established in a pointwise fashion, $R = S \equiv \langle \forall\ b, a\ ::\ b\ R\ a \equiv b\ S\ a \rangle$, we will prefer the **pointfree** style, in basically two variants:

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

$$R = S \quad \equiv \quad R \subseteq S \wedge S \subseteq R \tag{109}$$

- **Indirect equality** rules:

$$\begin{aligned} R = S \quad & \equiv \quad \langle \forall\ X\ ::\ (X \subseteq R \equiv X \subseteq S) \rangle & (110) \\ & \equiv \quad \langle \forall\ X\ ::\ (R \subseteq X \equiv S \subseteq X) \rangle & (111) \end{aligned}$$

Example of indirect proof

$$
\begin{array}{ll}
 & X \subseteq (R \cap S) \cap T \\
\equiv & \qquad \{ \ \cap\text{-universal (96) twice} \ \} \\
 & (X \subseteq R \wedge X \subseteq S) \wedge X \subseteq T \\
\equiv & \qquad \{ \ \wedge \text{ is associative} \ \} \\
 & X \subseteq R \wedge (X \subseteq S \wedge X \subseteq T) \\
\equiv & \qquad \{ \ \cap\text{-universal (96) twice} \ \} \\
 & X \subseteq R \cap (S \cap T) \\
:: & \qquad \{ \ \text{indirection (110)} \ \} \\
 & (R \cap S) \cap T = R \cap (S \cap T)
\end{array}
\tag{112}
$$

$\square$

All (data structures) in one (PF notation)

Pairing

$$\begin{tikzcd} A & A \times B \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & B \\ & C \arrow[ul, "R"] \arrow[u, "\langle R,S \rangle"] \arrow[ur, "S"'] & \end{tikzcd} \quad (113)$$

where $\pi_1(a,b) = a$, $\pi_2(a,b) = b$ and

$$\begin{array}{c|c} \psi & PF\ \psi \\ \hline\hline a\ R\ c \wedge b\ S\ c & (a,b)\langle R,S \rangle c \\ b\ R\ a \wedge d\ S\ c & (b,d)(R \times S)(a,c) \end{array} \quad (114)$$

Product: $R \times S = \langle R \cdot \pi_1, S \cdot \pi_2 \rangle$

Relational pairing example (in matrix layout)

Given

$$where^\circ \quad = \quad \begin{array}{r|cc} & Left & Right \\ \hline Fox & 1 & 0 \\ Goose & 0 & 1 \\ Beans & 0 & 1 \end{array} \qquad and \qquad cross \quad = \quad \begin{array}{r|cc} & Left & Right \\ \hline Left & 0 & 1 \\ Right & 1 & 0 \end{array}$$

pairing them up evaluates to:

$$\langle where^\circ, cross \rangle \quad = \quad \begin{array}{r|cc} & Left & Right \\ \hline (Fox, Left) & 0 & 0 \\ (Fox, Right) & 1 & 0 \\ (Goose, Left) & 0 & 1 \\ (Goose, Right) & 0 & 0 \\ (Beans, Left) & 0 & 1 \\ (Beans, Right) & 0 & 0 \end{array}$$

Exercises

Exercise 44: Show that

$$(b, c)\langle R, S\rangle a \equiv b\ R\ a \wedge c\ S\ a$$

PF-transforms to

$$\langle R, S\rangle = \pi_1^\circ \cdot R \cap \pi_2^\circ \cdot S \qquad (115)$$

□

Exercise 45: Infer universal property

$$\pi_1 \cdot X \subseteq R \ \wedge\ \pi_2 \cdot X \subseteq S \quad \equiv \quad X \subseteq \langle R, S\rangle \qquad (116)$$

from (115) via indirect equality (110). What can you say about (116) in case X, R and S are functions?

□

Modeling a toy IS

Data model of a toy **loan library** captured by diagram

$$
\begin{tikzcd}
ISBN \arrow[d, "M"'] \arrow[dr, phantom, "\supseteq"] & ISBN \times UID \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, "R"] \arrow[dr, phantom, "\subseteq"] & UID \arrow[d, "N"] \\
Title \times Publisher \arrow[r, "\top"'] & Date & Name \times Address \times Phone \arrow[l, "\top"]
\end{tikzcd}
\tag{117}
$$

where

- M — records **books** on loan, identified by $ISBN$;
- N — records library **users** (identified by user id's in UID);

(both simple) and

- R — records **loan** dates.

Modeling a toy IS

The two squares in the diagram impose bounds on R:

- Non-existing **books** cannot be loaned (left square);
- Only known **users** can take books home (right square).

(**NB:** in the database terminology these are known as **integrity constraints**.)

Exercise 46: Add variables to both squares in (117) so that the same conditions are expressed pointwise. Then show that the conjunction of the two squares means the same as assertion

$$R^\circ \quad \subseteq \quad \langle M^\circ \cdot \top, N^\circ \cdot \top \rangle \tag{118}$$

and draw this in a diagram.

□

Modeling a toy IS

Exercise 47: Consider implementing M, R and N as **files** in a relational **database**. For this, think of **operations** on the database such as, for example, that which records new loans (K):

$$borrow(K,(M,R,N)) \quad \triangleq \quad (M, R \cup K, N) \tag{119}$$

It can be checked that the **pre-condition**

$$\text{pre-}borrow(K,(M,R,N)) \quad \triangleq \quad R \cdot K^{\circ} \subseteq id$$

is necessary for maintaining (117) (why?) but it is not enough. Calculate — for a rectangle in (117) of your choice — the corresponding clause to be added to pre-*borrow*.

□

Modeling a toy IS

Exercise 48: The operations which **buy** new books

$$buy(X, (M, R, N)) \quad \triangleq \quad (M \cup X, R, N) \tag{120}$$

and **register** new users

$$register(Y, (M, R, N)) \quad \triangleq \quad (M, R, N \cup Y) \tag{121}$$

don't need any **pre-conditions**. Why? (Hint: compute their WP.)
□

NB: see annex on proofs by $\subseteq$-monotonicity for a strategy generalizing the exercise above.

Exercises

Exercise 49: Unconditional distribution laws

$$
\begin{aligned}
(P \cap Q) \cdot S &= (P \cdot S) \cap (Q \cdot S) \\
R \cdot (P \cap Q) &= (R \cdot P) \cap (R \cdot Q)
\end{aligned}
$$

will hold provide one of R or S is simple and the other injective. Tell which (justifying).

□

Exercise 50: Derive from

$$
\langle R, S \rangle^\circ \cdot \langle X, Y \rangle = (R^\circ \cdot X) \cap (S^\circ \cdot Y) \tag{122}
$$

the following properties:

$$
\ker \langle R, S \rangle = \ker R \cap \ker S \tag{123}
$$

$\langle R, id \rangle$ is always **injective**, for whatever R

□

Exercises

Exercise 51: Recalling (78), prove that

$$swap \triangleq \langle \pi_2, \pi_1 \rangle \tag{124}$$

is a bijection. (Assume property $(R \cap S)^\circ = R^\circ \cap S^\circ$.)

□

Exercise 52: Let $\leq$ be a **preorder** and f be a function taking values on the carrier set of $\leq$.

1. Define the pointwise version of relation $\sqsubseteq \;\triangleq\; f^\circ \cdot \leq \cdot f$
2. Show that $\sqsubseteq$ is a **preorder.**
3. Show that $\sqsubseteq$ is not (in general) a total order even in the case $\leq$ is so.

□

Lexicographic orderings

Let $R \Rightarrow S$ be the relational operator

$$b(R \Rightarrow S)a \quad \equiv \quad (b\ R\ a) \Rightarrow (b\ S\ a) \tag{125}$$

It can be shown that universal property

$$R \cap X \subseteq Y \quad \equiv \quad X \subseteq (R \Rightarrow Y) \tag{126}$$

holds.

We define the **lexicographic chaining** of two relations R and S as follows:

$$R\,;S \quad \triangleq \quad R \cap (R^\circ \Rightarrow S) \tag{127}$$

Exercise

Exercise 53: Let students in a course have two numeric marks,

$$\mathbb{N} \xleftarrow{mark1} Student \xrightarrow{mark2} \mathbb{N}$$

and define the **preorders**:

$$\begin{array}{lcl} \leq_{mark1} & \triangleq & mark1^\circ \cdot \leq \cdot\, mark1 \\ \leq_{mark2} & \triangleq & mark2^\circ \cdot \leq \cdot\, mark2 \end{array}$$

Spell out in pointwise notation the meaning of lexicographic ordering

$$\leq_{mark1} ; \leq_{mark2}$$

$\square$

Exercises

Exercise 54: (a) From (126) infer:

$$\bot \Rightarrow R \quad = \quad \top \tag{128}$$

$$R \Rightarrow \top \quad = \quad \top \tag{129}$$

(b) via indirect equality over (127) show that

$$\top \,;\, S \quad = \quad S \tag{130}$$

holds for any S and that, for R symmetric, we have:

$$R \,;\, R \quad = \quad R \tag{131}$$

$\square$

Last but not least: sums

Example (Haskell):

```
data X = Boo Bool | Err String
```

PF-transforms to

$$\begin{tikzcd} Bool \arrow[r, "i_1"] \arrow[dr, "Boo"'] & Bool + String \arrow[d, "{[Boo\ ,Err]}"] & String \arrow[l, "i_2"'] \arrow[dl, "Err"] \\ & X & \end{tikzcd} \tag{132}$$

where

$[R\ , S] \ = \ (R \cdot i_1^\circ) \cup (S \cdot i_2^\circ)$ cf.

$$\begin{tikzcd} A \arrow[r, "i_1"] \arrow[dr, "R"'] & A + B \arrow[d, "{[R\ ,S]}"] & B \arrow[l, "i_2"'] \arrow[dl, "S"] \\ & C & \end{tikzcd}$$

Dually: $R + S = [i_1 \cdot R\ , i_2 \cdot S]$

Sums

From $[R\ ,S]\ =\ (R\cdot i_1^\circ)\cup(S\cdot i_2^\circ)$ above one easily infers, by indirect equality,

$$[R\ ,S]\subseteq X \quad\equiv\quad R\subseteq X\cdot i_1\ \wedge\ S\subseteq X\cdot i_2$$

(check this).

It turns out that inclusion can be strengthened to equality, and therefore **relational coproducts** have exactly the same properties as functional ones, stemming from the universal property:

$$[R\ ,S]=X \quad\equiv\quad R=X\cdot i_1\ \wedge\ S=X\cdot i_2 \tag{133}$$

Thus $[i_1\ ,i_2]=id$ — solve (133) for R and S when $X=id$, etc etc.

Divide and conquer

The property for sums (coproducts) corresponding to (122) for products is:

$$[R\ ,S] \cdot [T\ ,U]^{\circ} \quad = \quad (R \cdot T^{\circ}) \cup (S \cdot U^{\circ}) \tag{134}$$

NB: This *divide-and-conquer* rule is essential to **parallelizing** relation composition by **block** decomposition.

Exercise 55: Show that:

$$\mathrm{img}\,[R\ ,S] \quad = \quad \mathrm{img}\,R \cup \mathrm{img}\,S \tag{135}$$

$$\mathrm{img}\,i_1 \cup \mathrm{img}\,i_2 \quad = \quad id \tag{136}$$

□

$+$ meets $\times$

Exercise 56: Start by proving the **fusion** law

$$\langle R, S\rangle \cdot f \;=\; \langle R \cdot f, S \cdot f\rangle \tag{137}$$

where f is a function. Then, relying on both (133) and (137) infer the **exchange law**,

$$[\langle R, S\rangle \;, \langle T, V\rangle] \;=\; \langle [R \;, T], [S \;, V]\rangle \tag{138}$$

holding for all relations as in diagram

$$\begin{tikzcd}
A \arrow[r, "i_1"] \arrow[d, "R"'] \arrow[drr, "S"' near end] & A + B & B \arrow[l, "i_2"'] \arrow[d, "V"] \arrow[dll, "T" near start] \\
C & C \times D \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] & D
\end{tikzcd}$$

□

Annex

Rules of the PF-transform seen so far:

ϕ	$PF\ \phi$
$\langle \exists\ a\ ::\ b\ R\ a \wedge a\ S\ c \rangle$	$b(R \cdot S)c$
$\langle \forall\ a, b\ ::\ b\ R\ a \Rightarrow b\ S\ a \rangle$	$R \subseteq S$
$\langle \forall\ a\ ::\ a\ R\ a \rangle$	$id \subseteq R$
$(f\ b)\ R\ (g\ a)$	$b(f^\circ \cdot R \cdot g)a$
$b\ R\ a \wedge b\ S\ a$	$b\ (R \cap S)\ a$
$b\ R\ a \vee b\ S\ a$	$b\ (R \cup S)\ a$
$b\ R\ a \wedge c\ S\ a$	$(b, c)\langle R, S \rangle a$
$x = i_1\ a \wedge c\ R\ a\ \vee\ x = i_2\ b \wedge c\ S\ b$	$c[R\ , S]x$
$b\ R\ a \wedge d\ S\ c$	$(b, d)(R \times S)(a, c)$
TRUE	$b \top a$
FALSE	$b \perp a$

Annex — proofs by $\subseteq$-transitivity

Wanting to prove $R \subseteq S$, the following rules may help in doing so by relying on a "mid-point" M (analogy with interval arithmetics):

- Rule A: **lowering the upper side**

$$
\begin{array}{ll}
 & R \subseteq S \\
\Leftarrow & \quad \{ \ M \subseteq S \text{ is known ; transitivity of } \subseteq \ \} \\
 & R \subseteq M
\end{array}
$$

  and then proceed with $R \subseteq M$.

- Rule B: **raising the lower side**

$$
\begin{array}{ll}
 & R \subseteq S \\
\Leftarrow & \quad \{ \ R \subseteq M \text{ is known; transitivity of } \subseteq \ \} \\
 & M \subseteq S
\end{array}
$$

  and then proceed with $M \subseteq S$.

Example

Proof of shunting rule (79):

$$
\begin{array}{ll}
 & R \subseteq f^\circ \cdot S \\
\Leftarrow & \quad \{ \ id \subseteq f^\circ \cdot f \ ; \text{raising the lower-side} \ \} \\
 & f^\circ \cdot f \cdot R \subseteq f^\circ \cdot S \\
\Leftarrow & \quad \{ \ \text{monotonicity of } (f^\circ \cdot) \ \} \\
 & f \cdot R \subseteq S \\
\Leftarrow & \quad \{ \ f \cdot f^\circ \subseteq id \ ; \text{lowering the upper-side} \ \} \\
 & f \cdot R \subseteq f \cdot f^\circ \cdot S \\
\Leftarrow & \quad \{ \ \text{monotonicity of } (f \cdot) \ \} \\
 & R \subseteq f^\circ \cdot S
\end{array}
$$

Thus the equivalence in (79) is established by circular implication.

Annex

Conversion of *simplicity* ('left uniqueness') of functions,

$$\mathsf{img}\, f \quad \subseteq \quad id \tag{139}$$

— recall slide 36 — into pointwise notation (Eindhoven quantifier notation). We calculate:

$$
\begin{array}{cl}
 & \mathsf{img}\, f \quad \subseteq \quad id \\
\equiv & \qquad \{ \ (59) \text{ etc } \ \} \\
 & \langle \forall\ b, a \ :\ b(f \cdot f^{\circ})a :\ b\ id\ a \rangle \\
\equiv & \qquad \{ \ \text{composition } (57)\ ;\ \text{converse } (67)\ ;\ id\ a = a \ \} \\
 & \langle \forall\ b, a \ :\ \langle \exists\ c \ :\ b\ f\ c :\ a\ f\ c \rangle :\ b = a \rangle
\end{array}
$$

Annex

$$\equiv \quad \{ \text{ prepare for splitting (22) via nesting (16) } \}$$

$$\langle \forall\, b, a \;:\; \textsc{True} \wedge \langle \exists\, c \;:\; b\ f\ c :\; a\ f\ c \rangle :\; b = a \rangle$$

$$\equiv \quad \{ \text{ nesting (16) } \}$$

$$\langle \forall\, b \;:\; \textsc{True} :\; \langle \forall\, a \;:\; \langle \exists\, c \;:\; b\ f\ c :\; a\ f\ c \rangle :\; b = a \rangle\rangle$$

$$\equiv \quad \{ \text{ splitting (22) } \}$$

$$\langle \forall\, b \;:\; \textsc{True} :\; \langle \forall\, c \;:\; b\ f\ c :\; \langle \forall\, a \;:\; a\ f\ c :\; b = a \rangle\rangle\rangle$$

$$\equiv \quad \{ \text{ (un)nesting (16) } \}$$

$$\langle \forall\, b, c \;:\; b\ f\ c :\; \langle \forall\, a \;:\; a\ f\ c :\; b = a \rangle\rangle$$

$$\equiv \quad \{ \text{ (un)nesting (16) } \}$$

$$\langle \forall\, b, c, a \;:\; b\ f\ c \wedge a\ f\ c :\; b = a \rangle$$

Final exercises

Exercise 57: Prove the **union simplicity** rule:

$$M \cup N \text{ is simple} \quad \equiv \quad M,\ N \text{ are simple and } M \cdot N^\circ \subseteq id \qquad (140)$$

□