# PF-transform: using Galois connections to structure relational algebra 

J.N. Oliveira<br>Dept. Informática,<br>Universidade do Minho<br>Braga, Portugal

DI/UM, 2008 (updated: 2009-10, 2012, 2014)
$4 \square$ •

## Motivation

We motivate this subject by placing some very general questions:

- Why is programming "difficult"?
- Is there a generic skill, or competence, that one such acquire to become a "good programmer"?

Surely that of abstract modelling. But, still,

- What is it that makes abstract modelling a challanging task?
- Are there generic conceptual patterns that could be used to shorten the path from problems to models?


## Problems $=$ Easy + Hard

Superlatives in problem statements, eg.

- "... the smallest such number"
- "... the longest such list"
- "... the best approximation"
suggest two layers in specifications:
- the easy layer - broad class of solutions (eg. a prefix of a list)
- the difficult layer - requires one particular such solution regarded as optimal in some sense (eg. "longest prefix up to a given length").

Example - back to the primary school desk
The whole division algorithm

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

However

$$
\begin{array}{l|llll}
7 & 2 & & 2 \times 2+3=7 & \wedge \\
3 & 2 & 2 \neq 7 \div 2 \\
7 & 2 & & \\
5 & 1 & 2 \times 1+5=7 & \wedge & 1 \neq 7 \div 2
\end{array}
$$

That is: for some $r$,

$$
\begin{array}{l|l}
n & d \\
r & q
\end{array} \quad q=n \div d \equiv d \times q+r=n
$$

provided $q$ is the largest such $q(r$ smallest)

## Example - specifying $x \div y$

First version (literal):

$$
\begin{equation*}
x \div y=\langle\bigvee z:: z \times y \leq x\rangle \tag{203}
\end{equation*}
$$

Second version (involved):

$$
\begin{equation*}
z=x \div y \equiv\langle\exists r: 0 \leq r<y: x=z \times y+r\rangle \tag{204}
\end{equation*}
$$

Third version (clever!):

$$
\begin{equation*}
z \times y \leq x \equiv z \leq x \div y \quad(y>0) \tag{205}
\end{equation*}
$$

- a so-called Galois connection, as we shall soon see.


## Why $(205)$ is better than $(203,204)$

Equivalence (205),

$$
z \times y \leq x \equiv z \leq x \div y \quad(y>0)
$$

captures the requirements in an elegant way:

- It is a solution: $x \div y$ multiplied by $y$ approximates $x$

$$
(x \div y) \times y \leq x
$$

- let $z:=x \div y$ in (205) and simplify.
- It is the best solution because it provides the largest such number:

$$
z \times y \leq x \Rightarrow z \leq x \div y \quad(y>0)
$$

- the $\Rightarrow$ part of the $\equiv$ of (205).


## Reasoning

Equivalence (205)

$$
z \times y \leq x \equiv z \leq x \div y \quad(y>0)
$$

is not only simple to write but effective to reason about.
Let us see an example: we want to prove the following equality

$$
(n \div m) \div d=n \div(d \times m)
$$

What about

- using (203)? too many suprema!
- using (204)? too many existential quantifiers!
- using (205)? easy - see the next slide.


## Proving $(n \div m) \div d=n \div(d \times m)$

$$
\begin{aligned}
& q \leq(n \div m) \div d \\
& \equiv \quad\{(205)\} \\
& q \times d \leq n \div m \\
& \equiv \quad\{(205)\} \\
& (q \times d) \times m \leq n \\
& \equiv \quad\{\times \text { is associative }\} \\
& q \times(d \times m) \leq n \\
& \equiv \quad\{(205)\} \\
& q \leq n \div(d \times m) \\
& :: \quad\{\text { indirection (206) \}} \\
& (n \div m) \div d=n \div(d \times m)
\end{aligned}
$$

## (Generic) indirect equality

Note the use of the (generic) indirect equality rule

$$
\begin{equation*}
\langle\forall q:: q \leq x \equiv q \leq y\rangle \equiv(x=y) \tag{206}
\end{equation*}
$$

valid for any partial order $\leq$.
Exercise 95: Derive from (205) the two cancellation laws

$$
\begin{aligned}
q & \leq(q \times d) \div d \\
(n \div d) \times d & \leq n
\end{aligned}
$$

and reflexion law:

$$
\begin{equation*}
n \div d \geq 1 \equiv d \leq n \tag{27}
\end{equation*}
$$

## Galois connections

Equivalence (205) is an example of a Galois connection:

$$
\underbrace{z \times y}_{f z} \leq x \equiv z \leq \underbrace{x \div y}_{g x}
$$

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

( $f, g$ ) are said to be Galois connected iff, for all $a \in A$ and $b \in B .$.

## Galois adjoints

$$
\begin{equation*}
\underbrace{f}_{\text {er adjoint }} b \leq a \equiv b \sqsubseteq \underbrace{g}_{\text {upper adjoint }} a \tag{209}
\end{equation*}
$$

that is

$$
\begin{equation*}
f^{\circ} \cdot \leq=\sqsubseteq \cdot g \tag{210}
\end{equation*}
$$

Graphical interpretation of (210):

- $\sqsubseteq \cdot g$ is the "area" below function $g$ wrt. $\sqsubseteq$
- $f^{\circ} \cdot \leq$ is the "area" above function $f$ wrt. $\leq$
- $f$ and $g$ are such that these areas are the same.



## Still whole division

$f=(\times 2)$ is the lower adjoint of $g=(\div 2)$.

The area below $g=(\div 2)$ is the same as the area above $f=(\times 2)$.
$f=(\times 2)$ is not surjective.
$g=(\div 2)$ is not injective.


## Adjoints are "nearly" inverses

Easy to observe:

- $g(f y)=(y \times 2) \div 2=y-f$ is indeed a right inverse for $g$
- $f(g 5)=(5 \div 2) \times 2=2 \times 2=4 \leq 5-g$ is not a right inverse for $f$, but it provides an approximation.
In spite of this asymmetry, the connection enables us to reason about

$$
g=(\div y)
$$

— the "hard" operation - in terms of

$$
f=(\times y)
$$

- the "easy" operation. This is the main advantage of a Galois connection (GC).


## Notation

A GC can be expressed by point-wise equivalence (209)

$$
f x \leq y \equiv x \sqsubseteq g y
$$

or by the equivalent relational equality (210),

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

as we have seen.
Abbreviated notation

$$
\begin{equation*}
f \vdash g \tag{211}
\end{equation*}
$$

is used instead of (210) wherever the orders are implicit from the context.

## Basic properties

For preorders in

the two cancellation laws hold:

$$
\begin{equation*}
(f \cdot g) a \leq a \quad \text { and } \quad b \sqsubseteq(g \cdot f) b \tag{213}
\end{equation*}
$$

- recall exercise 95 for the case of whole division.

Distribution laws

$$
\begin{align*}
f\left(b \sqcup b^{\prime}\right) & =(f b) \vee\left(f b^{\prime}\right)  \tag{214}\\
g\left(a \wedge a^{\prime}\right) & =(g a) \sqcap\left(g a^{\prime}\right) \tag{215}
\end{align*}
$$

## Basic properties

These hold wherever both preorder are lattices, that is, wherever suprema

$$
\begin{equation*}
b \sqcup b^{\prime} \sqsubseteq x \equiv b \sqsubseteq x \wedge b^{\prime} \sqsubseteq x \tag{216}
\end{equation*}
$$

and infima

$$
\begin{equation*}
x \sqsubseteq b \sqcap b^{\prime} \equiv x \sqsubseteq b \wedge x \sqsubseteq b^{\prime} \tag{217}
\end{equation*}
$$

exist. (Similarly for $A, \leq, \vee, \wedge$.)
Exercise 96: Resort to indirect equality to prove any of (214) or (215).

## Other properties

Conversely,

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

Moreover, if $(f, g)$ are Galois connected,

- $f$ and $g$ are monotonic
- $f(g)$ uniquely determines $g(f)$ - thus the $\__{-}^{b},{ }^{\sharp}$ notations
- $(g, f)$ are also Galois connected - just 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 | $f=g^{b}$ | $g=f^{\sharp}$ |
| Definition | $f b=\bigwedge\{a: b \sqsubseteq g \quad a\}$ | $g a=\bigsqcup\{b: f b \leq a\}$ |
| Cancellation | $f(g a) \leq a$ | $b \sqsubseteq g(f \quad b)$ |
| Distribution | $f\left(b \sqcup b^{\prime}\right)=\left(\begin{array}{ll}f \quad b\end{array}\right) \vee\left(f b^{\prime}\right)$ | $g\left(a^{\prime} \wedge a\right)=\left(\begin{array}{ll}g & \left.a^{\prime}\right) \sqcap(g a) \\ \hline \text { Monotonicity } & b \sqsubseteq b^{\prime} \Rightarrow f b \leq f b^{\prime} \\ a \leq a^{\prime} \Rightarrow g a \sqsubseteq g a^{\prime} \\ \hline\end{array}\right.$ |

Exercise 97: Derive from (209) that both $f$ and $g$ are monotonic.
$\square$

## Remark

Galois connections originate from the work of the French mathematician Evariste Galois (1811-1832). Their main advantages,
simple, generic and highly calculational
are welcome in proofs in computing, due to their size and complexity, recall E. Dijkstra:

$$
\text { elegant } \equiv \text { simple and }
$$

 remarkably effective.

In the sequel we will re-interpret the relational operators we've seen so far as Galois adjoints.

## Examples

Not only

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

but also the two shunting rules,

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

as well as converse,

$$
\underbrace{X^{\circ}}_{f X} \subseteq Y \equiv X \subseteq \underbrace{Y^{\circ}}_{g Y}
$$

and so and so forth - are adjoints of GCs: see the next slides.

## Converse

| $(f X) \subseteq Y \equiv X \subseteq(g Y)$ |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: |
| Description | $f=g^{b}$ | $g=f^{\sharp}$ | Obs. |  |
| converse | $(-)^{\circ}$ | $(-)^{\circ}$ | $b R^{\circ} a \equiv a R b$ |  |

Thus:

$$
\begin{aligned}
\text { Cancellation } & \left(R^{\circ}\right)^{\circ}=R \\
\text { Monotonicity } & R \subseteq S \equiv R^{\circ} \subseteq S^{\circ} \\
\text { Distributions } & (R \cap S)^{\circ}=R^{\circ} \cap S^{\circ},(R \cup S)^{\circ}=R^{\circ} \cup S^{\circ}
\end{aligned}
$$

Exercise 98: Why is it that converse-monotonicity can be strengthened to an equivalence?

## Example of calculation from the GC

Converse involution:

$$
\begin{equation*}
\left(R^{\circ}\right)^{\circ}=R \tag{218}
\end{equation*}
$$

Indirect proof of (218):

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

## Functions

| $(f X) \subseteq Y \equiv X \subseteq(g Y)$ |  |  |  |
| :---: | :---: | :---: | :---: |
| Description | $f=g^{b}$ | $g=f^{\sharp}$ | Obs. |
| shunting rule | $(h \cdot)$ | $\left(h^{\circ}\right)$ | NB: $h$ is a function |
| "converse" shunting rule | $\left(\cdot h^{\circ}\right)$ | $(\cdot h)$ | NB: $h$ is a function |

Consequences:
Functional equality:

$$
h \subseteq g \equiv h=k \quad \equiv h \supseteq k
$$

Functional division: $\quad R \cdot h=R / h^{\circ}$
Question: what does $R / S$ mean?

## Relational division

In the same way

$$
z \times y \leq x \equiv z \leq x \div y
$$

means that $x \div y$ is the largest number which multiplied by $y$ approximates $x$,

$$
\begin{equation*}
Z \cdot Y \subseteq X \equiv Z \subseteq X / Y \tag{219}
\end{equation*}
$$

means that $X / Y$ is the largest relation which pre-composed $Y$ approximates $X$.

What is the pointwise meaning of $X / Y$ ?

## We reason:

First, the types of

$$
Z \cdot Y \subseteq X \equiv Z \subseteq X / Y
$$

Next, the calculation:


$$
\begin{aligned}
& c(X / Y) a \\
\equiv & \left\{\text { introduce points } C \leftarrow^{\underline{c}} 1 \text { and } A \leftarrow^{\underline{a}} 1\right\} \\
& x\left(\underline{c}^{\circ} \cdot(X / Y) \cdot \underline{a}\right) x \\
\equiv & \{\text { one-point }(12)\} \\
& x^{\prime}=x \Rightarrow x^{\prime}\left(\underline{c}^{\circ} \cdot(X / Y) \cdot \underline{a}\right) x
\end{aligned}
$$

Proceed by going pointfree:

## We reason

$$
\begin{aligned}
& i d \subseteq \underline{c}^{\circ} \cdot(X / Y) \cdot \underline{a} \\
& \equiv \quad\{\text { shunting rules (Galois connections) }\} \\
& \underline{c} \cdot \underline{a}^{\circ} \subseteq X / Y \\
& \equiv \quad\{\text { rule (219) - Galois connection }\} \\
& \underline{c} \cdot \underline{a}^{\circ} \cdot Y \subseteq X \\
& \equiv \quad\{\text { now shunt } \underline{c} \text { back to the right }\} \\
& \underline{a}^{0} \cdot Y \subseteq \underline{c}^{\circ} \cdot X \\
& \equiv \quad\{\text { back to points via (47) }\} \\
& \langle\forall b: a Y b: c X b\rangle
\end{aligned}
$$

## Outcome

In summary:

$$
\begin{equation*}
c(X / Y) a \equiv\langle\forall b: a Y b: c X b\rangle \tag{200}
\end{equation*}
$$



Example:
a $Y b=$ passenger $a$ choses flight $b$
$c X b=$ company $c$ operates flight $b$
$c(X / Y) a=$ company $c$ is the only one trusted by passenger $a$, that is, a only flies $c$.

## Pointwise meaning in full

The full pointwise encoding of Galois connection

$$
Z \cdot Y \subseteq X \equiv Z \subseteq X / Y
$$

is:
$\langle\forall c, b:\langle\exists a: c Z a: a Y b\rangle: c X b\rangle \equiv\langle\forall c, a: c Z a:\langle\forall b: a Y b: c X b\rangle\rangle$
If we drop variables and regard the uppercase letters as denoting Boolean terms dealing without variable $c$, this becomes

$$
\langle\forall b:\langle\exists a: Z: Y\rangle: X\rangle \equiv\langle\forall a: Z:\langle\forall b: Y: X\rangle\rangle
$$

recognizable as the splitting rule (7) of the Eindhoven calculus.
Put in other words: existential quantification is lower adjoint of universal quantification.

## Exercises

Exercise 99: Prove the equalities

$$
\begin{align*}
X \cdot f & =X / f^{\circ}  \tag{221}\\
X / \perp & =T  \tag{222}\\
T / Y & =T \tag{223}
\end{align*}
$$

and check their pointwise meaning.
$\square$

Exercise 100: Define

$$
\begin{equation*}
X \backslash Y=\left(Y^{\circ} / X^{\circ}\right)^{\circ} \tag{224}
\end{equation*}
$$

and infer:

$$
\begin{align*}
a(R \backslash S) c & \equiv\langle\forall b: b R a: b S c\rangle  \tag{225}\\
R \cdot X \subseteq Y & \equiv X \subseteq R \backslash Y \tag{226}
\end{align*}
$$

## Relational division

| $(f X) \subseteq Y \equiv X \subseteq(g \quad Y)$ |  |  |  |
| :---: | :---: | :---: | :---: |
| Description | $f=g^{b}$ | $g=f^{\sharp}$ | Obs. |
| right-division | $(\cdot R)$ | $(/ R)$ | right-factor |
| left-division | $(R \cdot)$ | $(R \backslash)$ | left-factor |

that is,

$$
\begin{align*}
& X \cdot R \subseteq Y \equiv X \subseteq Y / R  \tag{227}\\
& R \cdot X \subseteq Y \equiv X \subseteq R \backslash Y \tag{228}
\end{align*}
$$

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

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

$(\backslash R)$ and $(/ R)$ are monotonic and distribute over $\cap$.

## Domain and range

| $(f X) \subseteq Y \equiv X \subseteq(g Y)$ |  |  |  |
| :---: | :---: | :---: | :---: |
| Description | $f=g^{b}$ | $g=f^{\sharp}$ | Obs. |
| domain | $\delta$ | $(T \cdot)$ | lower $\subseteq$ restricted to coreflexives |
| range | $\rho$ | $(\cdot T)$ | lower $\subseteq$ restricted to coreflexives |

Thus the universal properties of domain and range

$$
\begin{aligned}
\delta R \subseteq \Phi & \equiv R \subseteq \top \cdot \Phi \\
\rho R \subseteq \Phi & \equiv R \subseteq \Phi \cdot T
\end{aligned}
$$

- recall (126) and (127) - are Galois connections, and so

$$
\begin{aligned}
\delta(S \cup R) & =\delta S \cup \delta R \\
\top \cdot(\Phi \cap \Psi) & =\top \cdot \Phi \cap \top \cdot \psi
\end{aligned}
$$

hold — similarly for $\rho$ and $(\cdot T)$.

## Other operators

| $(f X) \subseteq Y \equiv X \subseteq(g Y)$ |  |  |  |
| :---: | :---: | :---: | :---: |
| Description | $f=g^{b}$ | $g=f^{\sharp}$ | Obs. |
| implication | $(R \cap)$ | $(R \Rightarrow)$ | $b(R \Rightarrow X) a \equiv b R a \Rightarrow b X a$ |
| difference | $(--R)$ | $(R \cup)$ |  |

Thus the universal properties of implication and difference,

$$
\begin{aligned}
R \cap X \subseteq Y & \equiv X \subseteq R \Rightarrow Y \\
X-R \subseteq Y & \equiv X \subseteq R \cup Y
\end{aligned}
$$

are GCs - etc, etc
Exercise 101: Show that $R \cap(R \Rightarrow Y) \subseteq Y$ ("modus ponens") holds and that $R-R=\perp-R=\perp$.

## Exercises

Exercise 102: Let $\mathcal{P} A=\{S: S \subseteq A\}$ and let $A<{ }^{\epsilon} \mathcal{P} A$ denote the membership relation $a \in S$, for any $a$ and $S$. What does the relation $\in \backslash \in$ mean?

Exercise 103: Show that the relation $\in \backslash \in$ of the previous exercise is reflexive and transitive.

Exercise 104: Prove that equality

$$
\begin{equation*}
(R \backslash S) \cdot f=R \backslash(S \cdot f) \tag{229}
\end{equation*}
$$

holds.
$\square$

## Exercises

Exercise 105: (a) Show that $R \subseteq \perp / S^{\circ} \equiv \delta R \cap \delta S=\perp$; (b) Then use indirect equality to infer the universal property of term $R \cap \perp / S^{\circ}$ - the largest sub-relation of $R$ whose domain is disjoint of that of $S$.

Exercise 106: The relational overriding combinator,

$$
\begin{equation*}
R \dagger S=S \cup R \cap \perp / S^{\circ} \tag{230}
\end{equation*}
$$

means the relation which contains the whole of $S$ and that part of $R$ where $S$ is undefined - read $R \dagger S$ as " $R$ overridden by $S$ ". (a) Show that $\perp \dagger S=S$ and that $R \dagger \perp=R$; (b) Infer the universal property:

$$
\begin{equation*}
X \subseteq R \dagger S \equiv X-S \subseteq R \wedge \delta(X-S) \cdot \delta S=\perp \tag{231}
\end{equation*}
$$

## Binary adjoints

Recall the universal property of $\cup(65), R \cup S \subseteq X \equiv R \subseteq X \wedge S \subseteq X$, which can be written thus

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

or even as

$$
\cup(R, S) \subseteq X \equiv(R, S)(\subseteq \times \subseteq)(\Delta X)
$$

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


Similarly, the universal property of $\cap$ (64) can be captured by

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

## A glimpse of GC (generic) algebra

Assume $f \vdash g$ and $f^{\prime} \vdash g^{\prime}$ hold in:

Functors (preorders)

## Identity

$$
i d \vdash i d
$$

Splitting (lattices)
Composition

$$
\left\langle f, f^{\prime}\right\rangle \vdash \sqcap \cdot\left(g \times g^{\prime}\right)
$$

$$
f \cdot f^{\prime} \vdash g^{\prime} \cdot g
$$

Converse (symmetry)

$$
\begin{equation*}
f \vdash g \equiv g \vdash f \tag{232}
\end{equation*}
$$

$$
F f \vdash F g
$$

In particular, for $f, f^{\prime}:=i d$, $g, g^{\prime}:=i d:$

for $\triangle x=(x, x)$.

## Application I - Hoare Logic

## Handling Hoare triples in relation algebra

As application of the above we show next how to handle Hoare triples such as

$$
\begin{equation*}
\{p\} P\{q\} \tag{233}
\end{equation*}
$$

in relation algebra. First we spell out the meaning of (233):

$$
\begin{equation*}
\left\langle\forall s: p s:\left\langle\forall s^{\prime}: s \xrightarrow{P} s^{\prime}: q s^{\prime}\right\rangle\right\rangle \tag{234}
\end{equation*}
$$

that is:
if program $P$ is in state $s$ satisfying condition $p$, and it moves to state $s^{\prime}$, then $s^{\prime}$ satisfies $q$.

In other words:
Condition $p$ holding before $P$ executes is sufficient for condition $q$ to hold after $P$ executes.

## Handling Hoare triples in relation algebra

Let $\llbracket P \rrbracket$ denote the state transition relation of $P$, that is $s^{\prime} \llbracket P \rrbracket s$ means the same as $s \xrightarrow{P} s^{\prime}$.

Then (234) re-writes as follows:

$$
\left.\begin{array}{rl} 
& \left\langle\forall s: p s:\left\langle\forall s^{\prime}: s^{\prime} \llbracket P \rrbracket s: q s^{\prime}\right\rangle\right\rangle \\
\equiv & \quad\{\text { coreflexives }\} \\
& \left\langle\forall s: s \Phi_{p} s:\left\langle\forall s^{\prime}: s^{\prime} \llbracket P \rrbracket s: s^{\prime} \Phi_{q} s^{\prime}\right\rangle\right\rangle \\
\equiv & \quad\{\top ; \text { coreflexives }\}
\end{array}\right\} \begin{array}{ll}
\left\langle\forall s, s^{\prime \prime}: s \Phi_{p} s^{\prime \prime}:\left\langle\forall s^{\prime}: s^{\prime} \llbracket P \rrbracket s: s^{\prime}\left(\Phi_{q} \cdot \top\right) s^{\prime \prime}\right\rangle\right\rangle \\
\equiv & \quad\{\text { recall (225) and remove variables }\}
\end{array} \quad \begin{aligned}
& \quad \Phi_{p} \subseteq \llbracket P \rrbracket \backslash\left(\Phi_{q} \cdot \top\right)
\end{aligned}
$$

## Handling Hoare triples in relation algebra

Finally:

$$
\left.\begin{array}{cc} 
& \Phi_{p} \subseteq \llbracket P \rrbracket \backslash\left(\Phi_{q} \cdot \top\right) \\
\equiv & \{\text { GC of division (228) }\}
\end{array}\right\} \begin{gathered}
\llbracket P \rrbracket \cdot \Phi_{p} \subseteq \Phi_{q} \cdot \top \\
\equiv \\
\left\{\begin{array}{c}
(118)
\end{array}\right\} \\
\\
\llbracket P \rrbracket \cdot \Phi_{p} \subseteq \Phi_{q} \cdot \llbracket P \rrbracket
\end{gathered}
$$

Comparing this with the meaning of contract $\Phi_{q} \stackrel{f}{\longleftarrow} \Phi_{p}-$ recall (143) - we realize that they are the same in case $\llbracket P \rrbracket$ is a function - $P$ deterministic and wholly defined.

## Hoare triples are contracts

In summary:
The meaning of Hoare triple $\{p\} P\{q\}$ is the contract

$$
\begin{equation*}
\llbracket P \rrbracket \cdot \Phi_{p} \subseteq \Phi_{q} \cdot \llbracket P \rrbracket \tag{235}
\end{equation*}
$$

where $\llbracket P \rrbracket$ denotes the state transition semantics of $P$.

We will write

$$
\Phi_{p} \xrightarrow{P} \Phi_{q}
$$

to mean (235) which, as seen above, is the same as

$$
\begin{equation*}
\llbracket P \rrbracket \cdot \Phi_{p} \subseteq \Phi_{q} \cdot \top \tag{236}
\end{equation*}
$$

## Hoare triples are GCs

In turn, (236) is equivalent to

$$
\Phi_{p} \subseteq \llbracket P \rrbracket \backslash\left(\Phi_{q} \cdot \top\right) \cap i d
$$

Thanks to GC (127), (236) is also equivalent to

$$
\rho\left(\llbracket P \rrbracket \cdot \Phi_{p}\right) \subseteq \Phi_{q}
$$

Thus we have the following Galois connection for Hoare triples, where $P, \Phi$ and $\Psi$ abbreviate $\llbracket P \rrbracket, \Phi_{p}$ and $\Phi_{q}$, respectively:

$$
\begin{equation*}
\underbrace{\rho(P \cdot \Phi)}_{f \Phi} \subseteq \Psi \equiv \Phi \subseteq \underbrace{P \backslash(\Psi \cdot \top) \cap i d}_{g \psi} \tag{237}
\end{equation*}
$$

Adjoints $f$ and $g$ are known as predicate transformers.

## Hoare triples are GCs

The usual notation for $g \Psi$ is $P \emptyset \psi$ - the weakest (liberal) pre-condition (WP) for $\Psi$ to hold on the outputs of $P$.

Dually, $f \Phi=\rho(P \cdot \Phi)$ is known as the strongest post-condition (SP) holding on all outputs of $P$ restricted by $\Phi$ on the input.

These concepts are independent of their use in Hoare logic. In general, given a binary relation $B \Vdash^{R} A$ and coreflexives $A<{ }^{\Phi} A$ and $B<{ }^{\Psi} B$, we define

$$
\begin{align*}
\Phi \xrightarrow{R} \psi & \equiv R \cdot \Phi \subseteq \psi \cdot R  \tag{238}\\
& \equiv \Phi \subseteq R \emptyset \psi \tag{239}
\end{align*}
$$

which extends functional contracts to arbitrary relations.

## Exercises

Exercise 107: Prove

$$
\begin{equation*}
i d \leftarrow^{R} \Phi \quad \equiv \quad \text { TRUE } \equiv \Phi \leftarrow^{R} \perp \tag{240}
\end{equation*}
$$

Exercise 108: Prove the special cases:

- WP of a function $f$ :

$$
\begin{equation*}
f \bullet \Phi_{q}=\lambda a \cdot q(f a) \tag{241}
\end{equation*}
$$

- SP of a function $f$ :

$$
\begin{equation*}
\rho\left(f \cdot \Phi_{p}\right)=\lambda b \cdot b \in\{f a \mid p a\} \tag{242}
\end{equation*}
$$

NB: recall that (241) has been used several times earlier on in contract calculation.

## Exercises

Exercise 109: The formal meaning of (imperative) code sequential composition is

$$
\llbracket \mathrm{P} ; \mathrm{Q} \rrbracket=\llbracket \mathrm{Q} \rrbracket \cdot \llbracket \mathrm{P} \rrbracket
$$

Show that the following rule of the Hoare logic of programs,

$$
\frac{\{p\} \mathrm{P}\{q\},\{q\} \mathrm{Q}\{s\}}{\{p\} P ; Q\{s\}}
$$

is an instance of the following relational typing rule:

$$
\begin{equation*}
\Psi<{ }_{<}^{R \cdot S} \Phi \quad \Leftarrow \quad \Psi<\frac{R}{} \uparrow \wedge \Upsilon<S \tag{243}
\end{equation*}
$$

## Exercises

Exercise 110: Prove the "trading rule":

$$
\begin{equation*}
\Upsilon<{ }^{R} \phi \cdot \psi \equiv \Upsilon \digamma^{R \cdot \phi} \psi \tag{244}
\end{equation*}
$$

Exercise 111: Re-write the following "contract splitting" rule,

$$
\begin{equation*}
\Psi_{1} \cdot \Psi_{2} \leftarrow^{R} \Phi \equiv \Psi_{1} \leftarrow^{R} \Phi \wedge \Psi_{2}{ }^{R} \Phi \tag{245}
\end{equation*}
$$

in Hoare logic. Then prove (245).
$\square$

## WP calculus

Facts (237) and (239) show that whatever one can do in Hoare logic can be done with Dijkstra's WPs.

Let us show an example by converting (245) to WP-calculus:

$$
\begin{array}{cc} 
& \Upsilon \cdot \Psi<^{R} \Phi \equiv \Upsilon<^{R} \Phi \Phi \wedge \Psi<^{R} \Phi \\
\equiv & \{\text { WPs }(239) \text { three times }\} \\
& \Phi \subseteq R \emptyset(\Upsilon \cdot \Psi) \equiv \Phi \subseteq R \emptyset \Upsilon \wedge \Phi \subseteq R \emptyset \Psi \\
\equiv & \{\text { coreflexives }(112) ; \text { meet-universal }(64)\} \\
& \langle\forall \Phi:: \Phi \subseteq R \emptyset(\Upsilon \cdot \Psi) \equiv \Phi \subseteq(R \emptyset \Upsilon) \cap(R \emptyset \Psi)\rangle \\
\equiv & \{\text { meet of correflexives; indirect equality }(69)\} \\
& R \emptyset(\Upsilon \cdot \Psi)=(R \emptyset \Upsilon) \cdot(R \emptyset \Psi)
\end{array}
$$

## WP calculus

A more interesting example is the transformation of the WP-rule for sequential composition

$$
\begin{equation*}
(S \cdot R) \bullet \Phi=R \bullet(S \bullet \Phi) \tag{246}
\end{equation*}
$$

into a contract:

$$
\begin{array}{ll} 
& R \emptyset(S \emptyset \phi)=(S \cdot R) \emptyset \phi \\
\equiv & \{\text { indirect equality }(69)\} \\
& \psi \subseteq R \emptyset(S \emptyset \phi) \equiv \psi \subseteq(S \cdot R) \downarrow \phi \\
\equiv & \{(239) \text { twice }\} \\
& (S \emptyset \phi)<R<\psi \equiv \phi \stackrel{(S \cdot R)}{\leftarrow} \psi \tag{247}
\end{array}
$$

The outcome, still involving the operator, is an advantageous replacement for (243), since it is an equivalence.

## Exercises

Exercise 112: Show that $\rho R \longleftarrow^{R} \delta R$ holds. However, WP $R \emptyset(\rho R)=i d$ rather than $\delta R$. Explain why.

Exercise 113: Show that $\rho R<^{R} \delta R$ holds. However, WP $R \emptyset(\rho R)=i d$ rather than $\delta R$. Explain why.
$\square$

Exercise 114: The two "shunting" rules for $S$ a simple relation,

$$
\begin{align*}
S \cdot R \subseteq Q & \equiv(\delta S) \cdot R \subseteq S^{\circ} \cdot Q  \tag{248}\\
R \cdot S^{\circ} \subseteq Q & \equiv R \cdot \delta S \subseteq Q \cdot S \tag{249}
\end{align*}
$$

are "almost" Galois connections. (a) Derive the following variants concerning coreflexives,

$$
\begin{aligned}
& R \cdot \Phi \subseteq S \equiv R \cdot \Phi \subseteq S \cdot \Phi \\
& \Phi \cdot R \subseteq S \equiv \Phi \cdot R \subseteq \Phi \cdot S
\end{aligned}
$$

## Application II Optimization calculus

## Programming is optimization

Abstract models are derived from requirements by ignoring unnecessary detail.

This often results in models whose operations are vague or non-deterministic.

Such operations, often recorded as pre/post condition pairs, are binary relations.

As computers cannot handle vagueness, deriving code for such operations calls for determinization - some way to convert such relations into functions.

This process is known as model refinement, and it is performed in a stepwise manner; however, how does one control it? What is the guiding principle (if any)?

## Programming is optimization

Recall (203), one of the definitions given for whole division:

$$
x \div y=\langle\bigvee z:: z \times y \leq x\rangle
$$

Given some $y$, term $z \times y \leq x$ denotes a binary relation with input $x$ and output $z$. But not every output $z$ is acceptable - (203) tells that one wants the largest such $z$.

So there is an ordering $(\leq)$ on the outputs $\left(\mathbb{N}_{0}\right)$ telling what the optimization principle should be: largest wrt. $\mathbb{N}_{0} \leq \mathbb{N}_{0}$.

Whole division is (perhaps) the first optimization problem one solves at school; programmers do it all the time, most often unconsciously!

## Programming is optimization

Another example is provided by the Galois connection which specifies the take function available in Haskell, for instance:

$$
\begin{equation*}
\text { length } y s \leq n \wedge y s \preceq x s \quad \equiv \quad y s \preceq \text { take } n x s \tag{250}
\end{equation*}
$$

Here the ordering on outputs is the prefix relation $(\preceq)$ on lists.
For each $n$, term length $y s \leq n \wedge y s \preceq x s$ tells which outputs ys are candidates for take $n$ xs.

But only one of these is acceptable - the longest such prefix, which is optimal with respect to the prefix ordering.

## Exercise

Exercise 115: Before implementing take one can start proving properties about this function solely relying on (250):

- Show that

$$
\text { take (length } x s \text { ) } x s=x s
$$

holds.

- Resort to indirect equality over $\preceq$ in proving

$$
\text { take } n(\text { take } m x s)=\text { take }(\min n m) x s
$$

where min, the minimum of two natural numbers, is given by the obvious Galois connection.

## Optimization in an abstract setting

Let us once again go back to (203) and spell out the meaning of its supremum:

$$
\begin{aligned}
z(\div y) x & \equiv z \times y \leq x \wedge\left\langle\forall z^{\prime}: z^{\prime} \times y \leq x: z \geq z^{\prime}\right\rangle \\
\equiv & \{\text { define } z R x=z \times y \leq x\}
\end{aligned} \quad \underbrace{z \times y \leq x}_{z R x} \wedge \underbrace{\langle\forall z^{\prime}: \underbrace{z^{\prime} \times y \leq x}_{x R^{\circ} z^{\prime}}: z \geq z^{\prime}\rangle}_{z\left(\geq / R^{\circ}\right) x} .
$$

Im summary:

$$
\begin{equation*}
(\div y)=R \cap \geq / R^{\circ} \text { where } R=(\times y)^{\circ} \leq \quad \geq / R^{\circ} \tag{251}
\end{equation*}
$$

## Optimization in an abstract setting

Generalization: given any relation $B \stackrel{R}{\longleftarrow} A$ and an optimization criterion $B<{ }^{S} B$ on its outputs,

define a new relational combinator $R \upharpoonright S$ (read: $R$ optimized by $S$, or $R$ "shrunk" by $S$ ) as follows:

$$
\begin{equation*}
R \upharpoonright S=\underbrace{R}_{\text {easy }} \cap \underbrace{S / R^{\circ}}_{\text {hard }} \tag{252}
\end{equation*}
$$

The "hard" term specifies the optimization taking place.

## Optimization in an abstract setting

By standard application of indirect equality to (252) one obtains the universal property of the "shrinking" operator:

$$
\begin{equation*}
X \subseteq R \upharpoonright S \equiv X \subseteq R \wedge X \cdot R^{\circ} \subseteq S \tag{253}
\end{equation*}
$$

This ensures $R \upharpoonright S$ as the largest sub-relation $X$ of $R$ such that, for all $b^{\prime}, b \in B$, if there exists $a \in A$ such that $b^{\prime} X a \wedge b R a$, then $b^{\prime} S b$ holds (" $b^{\prime}$ better than $b^{\prime \prime}$ ).

(253) can be regarded as a GC between the set of all subrelations of $R$ and the set of optimization criteria on its outputs.

## Optimization calculus

Both the definition of $R \upharpoonright S$ and its universal property (253) provide a rich setting for exploiting generic properties of optimization in this abstract setting.

Below we give a brief account of such algebra, as obtained using relational calculus.

The interested reader is referred to the works by Mu and Oliveira (2012) and Oliveira and Ferreira (2012) for a more complete account of optimization by shrinking, with applicatons to software design.

## Basic properties of $R \upharpoonright S$

Chaotic optimization:

$$
\begin{equation*}
R \upharpoonright T=R \tag{254}
\end{equation*}
$$

Impossible optimization:

$$
\begin{equation*}
R \upharpoonright \perp=\perp \tag{255}
\end{equation*}
$$

"Brute force" determinization:

$$
\begin{equation*}
R \upharpoonright i d=\text { largest deterministic fragment of } R \tag{256}
\end{equation*}
$$

Thus $R \upharpoonright i d$ is the part of $R$ which cannot be further refined.

Exercise 116: Prove the two first equalities above.

## Basic properties of $R \upharpoonright S$

$R \upharpoonright i d$ is the extreme case of the fact which follows:

$$
\begin{equation*}
R \upharpoonright S \text { is simple } \Leftarrow S \text { is anti-symmetric } \tag{257}
\end{equation*}
$$

Thus anti-symmetric criteria always lead to determinism, possibly at the sacrifice of totality. Clearly: for $R$ simple,

$$
\begin{equation*}
R \upharpoonright S=R \quad \equiv \quad \operatorname{img} R \subseteq S \tag{258}
\end{equation*}
$$

Thus (functions)

$$
\begin{equation*}
f \upharpoonright S=f \quad \Leftarrow \quad S \text { is reflexive } \tag{259}
\end{equation*}
$$

## Basic properties of $R \upharpoonright S$

Pre-condition fusion:

$$
\begin{equation*}
(R \upharpoonright S) \cdot \Phi=(R \cdot \Phi) \upharpoonright S \tag{260}
\end{equation*}
$$

Two function fusion rules

$$
\begin{align*}
& (R \upharpoonright S) \cdot f=(R \cdot f) \upharpoonright S  \tag{261}\\
& (f \cdot R) \upharpoonright S=f \cdot\left(R \upharpoonright S_{f}\right) \tag{262}
\end{align*}
$$

where $S_{f}$ abbreviates $f^{\circ} \cdot S \cdot f$.
Exercise 117: Show that, for $S$ a preorder, $S_{f}$ above is also a preorder.

## Basic properties of $R \upharpoonright S$

Union:

$$
\begin{equation*}
(R \cup S) \upharpoonright Q=(R \upharpoonright Q) \cap Q / S^{\circ} \cup(S \upharpoonright Q) \cap Q / R^{\circ} \tag{263}
\end{equation*}
$$

This has a number of corollaries, namely a conditional rule,

$$
\begin{equation*}
(p \rightarrow R, T) \upharpoonright S=p \rightarrow(R \upharpoonright S),(p \upharpoonright S) \tag{264}
\end{equation*}
$$

the distribution over alternatives (77),

$$
\begin{equation*}
[R, S] \upharpoonright U=[R \upharpoonright U, S \upharpoonright U] \tag{265}
\end{equation*}
$$

and the "function competition" rule:

$$
\begin{equation*}
(f \cup g) \upharpoonright S=(f \cap S \cdot g) \cup(g \cap S \cdot f) \tag{266}
\end{equation*}
$$

since $S / g^{\circ}=S \cdot g$.

## "Function competition" rule

With points:

$$
y((f \cup g) \upharpoonright S) x \equiv\left\{\begin{aligned}
& y=f x \wedge(f x) S(g x) \\
& \vee \\
& y=g \times \wedge(g x) S(f x)
\end{aligned}\right.
$$

that is: $f$ (resp. $g$ ) "wins" wherever it is better than $g$ (resp. $f$ ) wrt. S. For instance,

$$
a b s=(i d \cup \operatorname{sim}) \upharpoonright \geq
$$

for $\operatorname{sim} x=-x, c f$.

$$
\begin{aligned}
y=a b s x & \equiv y=x \wedge x \geq-x \vee y=-x \wedge-x \geq x \\
& \equiv y=x \wedge x \geq 0 \vee y=-x \wedge 0 \geq x
\end{aligned}
$$

## $R \upharpoonright S$ on data

Combinator $R \upharpoonright S$ also makes sense when $R$ and $S$ are finite, relational data structures (eg. tables in a database).

Example of $R \upharpoonright S$ in data-processing: given
$\left(\begin{array}{c|c|c}\text { Examiner } & \text { Mark } & \text { Student } \\ \hline \text { Smith } & 10 & \text { John } \\ \text { Smith } & 11 & \text { Mary } \\ \text { Smith } & 15 & \text { Arthur } \\ \text { Wood } & 12 & \text { John } \\ \text { Wood } & 11 & \text { Mary } \\ \text { Wood } & 15 & \text { Arthur }\end{array}\right)$
and wishing to "choose the best mark", project over Mark, Student and optimize over the $\geq$ ordering on Mark (next slide):

## $R \upharpoonright S$ on data

$$
\left(\begin{array}{c|c}
\text { Mark } & \text { Student } \\
\hline 10 & \text { John } \\
11 & \text { Mary } \\
12 & \text { John } \\
15 & \text { Arthur }
\end{array}\right) \upharpoonright \geq=\begin{array}{c|c}
\text { Mark } & \text { Student } \\
\hline 11 & \text { Mary } \\
12 & \text { John } \\
15 & \text { Arthur }
\end{array}
$$

Relational shrinking can also be used for induction-free reasoning about sequences (lists), welcome in Alloy where no explicit recursion is available.

Example of $R \upharpoonright S$ in list-processing: given a sequence $A \longleftarrow S$

$$
A \underbrace{\text { nub } S} \mathbb{N} \triangleq\left(S^{\circ} \upharpoonright \leq\right)^{\circ}
$$

removes all duplicates while keeping the first instances. (Data in $\mathbb{N}$ could be regarded as "time stamps".)

## Galois connections (211) as optimization problems

$$
\left.\begin{array}{ll} 
& f^{\circ} \cdot(\leq)=(\sqsubseteq) \cdot g \\
\equiv & \{\text { ping-pong }\}
\end{array}\right\}
$$

## Galois connections as optimization problems

Comments:

- Given the two orderings $(\leq)$ and $(\sqsupseteq)$ and the "easy adjoint" $f$, implementing the "hard adjoint" amounts to solving the inequation (267) for $g$.
- We have already seen an instance of this result in (251), for whole division.

Question:
Implementations are usually recursive. Where in (267) is the "guideline" for introducing recursion in the calculations ?

Since $g \subseteq\left(f^{\circ} \cdot(\leq)\right) \upharpoonright(\sqsupseteq)$ expresses an optimization by $(\sqsupseteq)$, it is this ordering which controls the implementation process. How?

## Exercises

Assume a generic Galois connection $f^{\circ} \cdot \leq=\sqsubseteq \cdot g$ in the following exercises.

Exercise 118: Show that $f$ monotonicity, $x \sqsubseteq y \Rightarrow f x \leq f y$, can be written point-free as

$$
\begin{equation*}
(\sqsubseteq) \cdot f^{\circ} \subseteq f^{\circ} \cdot(\leq), \tag{268}
\end{equation*}
$$

Exercise 119: Show that, once (268) is assumed, the following equivalence holds:

$$
\begin{equation*}
g \subseteq f^{\circ} \cdot(\leq) \equiv(\sqsubseteq) \cdot g \subseteq f^{\circ} \cdot(\leq) \tag{269}
\end{equation*}
$$

Suggestion: do a "ping-pong" proof.
$\square$

# Application III Optimization versus induction 

## Optimizing over inductive relations

As shown in (Bird and de Moor, 1997) and (Mu and Oliveira, 2012), most often the orderings involved in program optimization are inductive relations.

- Inductive orderings lead to recursive programs
- "Greedy algorithms" and "dynamic programming" studied in this way in the Algebra of Programming book (Bird and de Moor, 1997).
- Complexity of the approach puts many readers off (need for always transposing relations to powerset functions; ...)
What's new in (Mu and Oliveira, 2012):
$R \upharpoonright S$ algebra greatly simplifies and generalizes the calculation of programs from such specifications. (Notably, there is no need for power transpose.)


## Folds (k $k \tau \alpha$ )

In general, for F a polynomial functor (relator) and initial $\mu \mathrm{F}<\stackrel{\text { in }}{ } \mathrm{F}(\mu \mathrm{F})$,

there is a unique solution to equation $X=R \cdot \mathrm{~F} X \cdot i n^{\circ}$ - thus universal property:

$$
\begin{equation*}
X=(|R|) \equiv X \cdot i n=R \cdot \mathrm{~F} X \tag{270}
\end{equation*}
$$

(Read $(|R|)$ as "fold $R$ " or " $\kappa \alpha \tau \alpha R^{\prime \prime}$.)

## Relational folds

It is very easy to show that

$$
\begin{equation*}
(|i n|)=i d \tag{271}
\end{equation*}
$$

holds - just make $X=i d$ in (270) and solve for $R$ (this is known as the reflexion property).

Example: in $=[$ nil, cons $]$ for lists. Reflexion (271) means that the function $f=([$ nil, cons $])$ is bound to be the identity, cf.

$$
\begin{aligned}
& f[]=[] \\
& f(\operatorname{cons}(a, x))=\operatorname{cons}(a, f x)
\end{aligned}
$$

Now suppose we have $R=[$ nil, cons $\cup$ nil $]$ in (270). What is the meaning of $([$ nil, cons $\cup$ nil $])$ ?

## Relational folds

Unfolding $X=([$ nil, cons $\cup$ nil $])$ we get

$$
X \cdot[\text { nil }, \text { cons }]=[\text { nil }, \text { cons } \cup \text { nil }] \cdot(i d+i d \times X)
$$

that is, $X \cdot$ nil $=$ nil and $X \cdot$ cons $=($ cons $\cup$ nil $) \cdot(i d \times X)$.
Introducing variables in $X \cdot$ nil $=$ nil we get $y X[] \equiv y=[]$ since nil $\quad=[]$. That is, [] X [] $\equiv$ True. Doing the same for the other clause we get:

$$
y X(a: x) \equiv y=[] \vee\left\langle\exists x^{\prime}: x^{\prime} X x: y=a: x^{\prime}\right\rangle
$$

Thus $([$ nil, cons $\cup$ nil $])$ is the prefix relation:

$$
(\preceq)=([\text { nil }, \text { cons } \cup \text { nil }])
$$

## The "Greedy" theorem

$$
\begin{equation*}
(R \upharpoonright S) \subseteq(R \mid) \upharpoonright S \Leftarrow S^{\circ}<^{R} \mathrm{~F} S^{\circ} \tag{272}
\end{equation*}
$$

for $S$ transitive. (NB: $R \leftarrow^{X} S$ means $X \cdot S \subseteq R \cdot X$ ) In a diagram, where the side condition is depicted in dashed arrows:


Proof: see (Mu and Oliveira, 2012).

## Example of greedy programming

The msp problem ("maximum sum prefix"), whose spec

$$
\begin{aligned}
& m s p::[\operatorname{Int}] \leftarrow[\operatorname{lnt}] \\
& y \text { msp } x=y \text { is a prefix of } x \text { that yields the maximum } \\
& \text { sum }
\end{aligned}
$$

translates into $(\preceq=([$ nil , cons $\cup$ nil $\rrbracket)$ is the prefix ordering $)$

$$
y m s p x \quad \Rightarrow \quad y \preceq x \wedge\langle\forall z: z \preceq x: \operatorname{sum} y \geq \operatorname{sum} z\rangle
$$

which in turn PF-transforms into

$$
m s p \quad \subseteq \quad \preceq \upharpoonright \geq_{\text {sum }}
$$

(NB: not a GC, it is nevertheless a good example to understand greedy programming.)

## Example of greedy programming

We calculate:

$$
\begin{aligned}
& m s p \subseteq \preceq \upharpoonright \geq_{\text {sum }} \\
& \equiv \quad\{\text { definition of prefix ordering }\} \\
& \text { msp } \subseteq([\text { nil, cons } \cup \text { nil }]) \upharpoonright \geq_{\text {sum }} \\
& \Leftarrow \quad\{\text { greedy theorem (272) \} } \\
& m s p \subseteq\left([\text { nil }, \text { cons } \cup \text { nil }] \upharpoonright \geq_{\text {sum }} \rrbracket\right) \\
& \equiv \quad\{\text { junc-rule (265) ; determinism of nil }\} \\
& \left.m s p \subseteq \text { 【nil },(\text { cons } \cup n i l) \upharpoonright \geq_{\text {sum }} \rrbracket\right) \\
& \equiv \quad\{\text { function competition rule (266) \}} \\
& m s p \subseteq\left(\left[\text { nil },\left(\text { cons } \cap \geq_{\text {sum }} \cdot n i l\right) \cup\left(n i l \cap \geq_{\text {sum }} \cdot \text { cons }\right)\right]\right)
\end{aligned}
$$

(Side condition ignored for brevity.)

## Example of greedy programming

Let $R$ abbreviate the inductive step

$$
\left(\text { nil } \cap \geq_{\text {sum }} \cdot \text { cons }\right) \cup\left(\text { cons } \cap \geq_{\text {sum }} \cdot \text { nil }\right)
$$

Then $y R(a: x)$ means

$$
y=[] \wedge 0 \geq a+\operatorname{sum} x \vee y=a: x \wedge a+\operatorname{sum} x \geq 0
$$

The case $a+\operatorname{sum} x=0$ is ambiguous, in the sense that the algorithm may either stop yielding $y=[$ ] or yield $y=a: x$, where $x$ is the outcome of the recursive step.

As we still have non-determinism, we need to further shrink what we started from,

$$
\begin{equation*}
m s p=\left(\preceq \upharpoonright \geq_{\text {sum }}\right) \upharpoonright \preceq \tag{273}
\end{equation*}
$$

to obtain the function which yields the shortest such prefix.

## Example of greedy programming

Putting everything together, the overall outcome will be, in Haskell syntax:

$$
\begin{aligned}
\operatorname{msp}[]= & {[] } \\
\operatorname{msp}(\mathrm{a}: \mathrm{s})= & \text { let } \mathrm{x}=\operatorname{msp} \mathrm{s} \\
& \text { in if sum } \mathrm{x}>-\mathrm{a} \text { then } \mathrm{a}: \mathrm{x} \text { else [] }
\end{aligned}
$$

See more theorems and examples in (Mu and Oliveira, 2012) covering also optimizations which lead to hylomorphisms and anamorphisms.

It turns out that whole division $(x \div y)$, take etc end up being anamorphisms.
R. Bird and O. de Moor. Algebra of Programming. Series in Computer Science. Prentice-Hall International, 1997.
S.-C. Mu and J.N. Oliveira. Programming from Galois connections. Journal of Log. Algebraic Programming, 81(6):680-704, 2012. doi: 10.1016/j.jlap.2012.05.003. .
J.N. Oliveira and M.A. Ferreira. Alloy meets the algebra of programming: a case study, 2012. To appear in IEEE Transactions on Software Engineering.

