Elementary Order Theory

- A *partially ordered set* or *poset* is a pair $(P, \leq)$ where P is a set and $\leq$ is a *partial order* on P.
- $f\colon P \to P$ between posets is **monotone** just in case it preserves the order.
- If $S \subseteq P$ then we write $\bigwedge S$ for the **meet** or **infimum** of S; dually we write $\bigvee S$ for the **join** or **supremum** of S.
- P is called a **complete lattice** if the joins of all subsets S exist or (equivalently) the meets of all subsets exist.

Fixed Points

If $f\colon P \to P$ is an endofunction on a poset P, we call

- $x \in P$ a **fixed point** for f if $f(x) = x$;
- a **pre-fixed point** of f if $f(x) \leq x$; and
- a **post-fixed point** of f if $x \leq f(x)$.

If P is a complete lattice, and f is monotone, the least pre-fixed point, $\mu\, f$, and the greatest post-fixed point, $\nu\, f$, both exist, and are given by

$$\mu\, f \stackrel{\text{def}}{=} \bigwedge \{\, x \in P \mid f(x) \leq x \,\}$$
$$\nu\, f \stackrel{\text{def}}{=} \bigvee \{\, x \in P \mid x \leq f(x) \,\}$$

(Co)Inductively Defined Sets

- If X is any set, let $\mathbb{P}(X)$ be the *powerset* of X. Define a set of **rules** on X to be any subset $\mathcal{R}$ of the form

$$\mathcal{R} \subseteq \mathbb{P}(X) \times X$$

Define the **name** of $\mathcal{R}$ to be the function $\Phi_{\mathcal{R}} \colon \mathbb{P}(X) \to \mathbb{P}(X)$ given by setting

$$\Phi_{\mathcal{R}}(S) \stackrel{\text{def}}{=} \{\, x \in X \mid \exists (S', x) \in \mathcal{R} \text{ and } S' \subseteq S \,\}$$

- Given a set X, and a set of rules $\mathcal{R}$ on X, the **subset of** X **inductively defined** by $\mathcal{R}$ is $\mu\, \Phi_{\mathcal{R}}$, and the **subset of** X **coinductively defined** by $\mathcal{R}$ is $\nu\, \Phi_{\mathcal{R}}$.

Closed Sets

- A subset $S \subseteq X$ is **closed under the set of rules $\mathcal{R}$** if it is a pre-fixed point of $\Phi_{\mathcal{R}}$. This means

$$\{ x \in X \mid \exists (S', x) \in \mathcal{R} \text{ and } S' \subseteq S \} \subseteq S$$

- Note that S is closed just in case for each rule $(H, c) \in \mathcal{R}$,

$$H \subseteq S \Longrightarrow c \in S \qquad (*)$$

- We sometimes say that S is **closed under the rule** $R = (H, c)$ if $*$ holds for R. For each element $h \in H$, the assumption that $h \in S$ is called an **inductive hypothesis**.

Dense Sets

- A subset $S \subseteq X$ is **dense under the set of rules** $\mathcal{R}$ if it is a post-fixed point of $\Phi_{\mathcal{R}}$. This means

$$S \subseteq \{\ x \in X \ \mid\ \exists (S', x) \in \mathcal{R} \text{ and } S' \subseteq S\ \}$$

Rule Notation

We can write **finitary** rules like this

- a base rule $R = (\varnothing, c)$

$$\frac{}{c}\,R$$

- and a deductive rule $R = (H, c) = (\{\, h_1, \ldots, h_k \,\}, c)$

$$\frac{h_1 \quad h_2 \quad \ldots \quad h_k}{c}\,R$$

Examples of (Co)Inductively Defined Sets

- Consider $\mathcal{R} \subseteq \mathbb{P}(\mathbb{Z}) \times \mathbb{Z}$ given by $\frac{}{0}$ and $\frac{z}{z+1}$. Then $\mu\,\Phi_{\mathcal{R}} = \mathbb{N}$ and $\nu\,\Phi_{\mathcal{R}} = \mathbb{Z}$.

- Fix $R \subseteq A \times A$. Consider $\mathcal{R}$ given by $\frac{\{\,a'\,\}}{a}$ just in case $a\ R\ a'$. Then $\mu\,\Phi_{\mathcal{R}} = \varnothing$ and

$$\nu\,\Phi_{\mathcal{R}} = \{\, a \mid \exists \alpha \in A^{\omega}.\quad a\ R\ \alpha(0)\ R\ \alpha(1)\ R \ldots \,\}$$

Principles of (Co)Induction

- *Principle of Induction*

Suppose that $I \subseteq X$ is inductively defined by a set of rules $\mathcal{R}$, and that $S \subseteq I$. Then in order to verify that $S = I$ it is enough to show that S is closed under the rules.

- *Principle of Coinduction*

Suppose that $C \subseteq X$ is coinductively defined by a set of rules $\mathcal{R}$. Then in order to verify that $x \in C$ it is enough to find a set S which is dense under the rules and for which $x \in S$.

Theorem

Given a finitary set of rules $\mathcal{R}$, a **deduction** for $x \in X$ is a finitely branching tree with root x such that for each node $c \in X$, if H is the (possibly empty) finite set of children of c, then (H, c) must be a rule in $\mathcal{R}$.

Theorem: Let $I \subseteq X$ be inductively defined by a set of rules $\mathcal{R}$. Then

$$I = \{\, x \mid \text{there exists a deduction of } x \,\},$$

that is for any element $x \in X$, we have

$$\boxed{x \in I \text{ if and only if there exists a deduction of } x.}$$

Part II: (Co)Induction in Program Semantics

- *Inductively* define some programs $\underline{4} + \underline{3}$, $\lambda x.x * \underline{7}$ and $\mathsf{rec}\, f.\lambda n.\mathsf{if}\ n = \underline{1}\ \mathsf{then}\ \underline{1}\ \mathsf{else}\ n * (f\,(n - \underline{1}))$.

- *Inductively* define *evaluation of programs* $P \Downarrow V$ such as $(\lambda x.x * \underline{7})\,(\underline{4} + \underline{3}) \Downarrow \underline{49}$.

- *Inductively* define *program transitions* $P \rightsquigarrow P'$ such as $(\lambda x.x * \underline{7})\,(\underline{4} + \underline{3}) \rightsquigarrow (\underline{4} + \underline{3}) * \underline{7} \rightsquigarrow \underline{7} * \underline{7} \rightsquigarrow \underline{49}$

- *Coinductively* define *divergence* $P \Uparrow$, such as $\mathsf{rec}\, x.x \Uparrow$, where this means $P \rightsquigarrow P_1 \rightsquigarrow P_2 \rightsquigarrow P_3 \rightsquigarrow \dots$

- *Coinductively* define *program equivalence* $P \sim P'$ such as $\lambda x.x * \underline{1} * \underline{7} \sim \lambda x.x * \underline{7}$

Rules for Inductively Generating the Set $\mathcal{T}$

$$\frac{}{x}\ [x \in Var] \ \text{VAR} \qquad \frac{}{\underline{c}}\ [c \in \mathbb{Z} \cup \mathbb{B}] \ \text{CONST}$$

$$\frac{M \quad N}{M \ op \ N}\ [op \in Opr] \ \text{OP} \qquad \frac{M \quad N \quad L}{\mathsf{if}\ M\ \mathsf{then}\ N\ \mathsf{else}\ L} \ \text{COND}$$

$$\frac{M}{\lambda x.M} \ \text{ABS} \qquad \frac{M \quad N}{M\ N} \ \text{AP} \qquad \frac{M}{\mathsf{rec}\, x.M} \ \text{REC} \qquad \frac{}{\mathsf{nil}} \ \text{NIL}$$

$$\frac{M}{\mathsf{hd}(M)} \ \text{HD} \qquad \frac{M}{\mathsf{tl}(M)} \ \text{TL} \qquad \frac{M \quad N}{M : N} \ \text{CONS} \qquad \frac{M}{\mathsf{elist}(M)} \ \text{ELIST}$$

Simple Programs

- If P and P' are two programs, write $P \rightsquigarrow P'$ to mean P "computes in one step" to P'

We write $M[N/v]$ to mean "M where v is replaced by N"

- For example,

$$(\underline{2} + \underline{5}) + \underline{1} \rightsquigarrow \underline{7} + \underline{1} \rightsquigarrow \underline{8} \qquad \text{and} \qquad (x + y)[\underline{4}/y] = x + \underline{4}.$$

- The term $\lambda x.M$ is code for the function which maps x to M. If $f \stackrel{\text{def}}{=} \lambda x.M$, then $f\,a \rightsquigarrow M[a/x]$. Thus $\lambda x.x + \underline{2}$ is the function which "adds 2", and for example $(\lambda x.x + \underline{2})\,\underline{4} \rightsquigarrow \underline{4} + \underline{2}$.

Simple Recursion

$\mathsf{rec}\, x.M$ denotes a solution to the equation $x = M$.

Write $R \stackrel{\text{def}}{=} \mathsf{rec}\, x.M$. The program R "computes in one step" to $M[R/x]$. Thus if we take $M \stackrel{\text{def}}{=} \underline{0} : x$, then

$$R \rightsquigarrow (\underline{0} : x)[R/x] \equiv \underline{0} : R \rightsquigarrow \underline{0} : (\underline{0} : R) \rightsquigarrow \ldots$$

and so R is a *program which recursively evaluates to an infinite list of zeros*. We call each step in the computation of R an **unfolding**.

Subterms

- A **subterm** S of a term M is any subtree of the finite tree denoted by M. We write $S \lhd M$. A variable x **occurs** in M if $x \lhd M$. There may be many occurrences. We say N lies in the **scope** of λy or $\mathsf{rec}\, y$ in any subterm of the form $\lambda y.N$ or $\mathsf{rec}\, y.N$ respectively.

- $u + \underline{2}$ is the scope of λu in $\lambda x.(\lambda u.u + \underline{2})\, z$. Note that

$$\lambda u.u + \underline{2} \lhd \lambda x.(\lambda u.u + \underline{2})\, z.$$

If $N \stackrel{\text{def}}{=} \lambda x.xxy\underline{x}zx$ then the underlined x is the *fourth* occurrence of x in N.

Free and Bound Variables

- Each *occurrence* of x in M is either free or bound. We say that an occurrence of x is **bound** in M if the occurrence of x in M is in a subterm of the form $\lambda x.N$ or $\mathsf{rec}\, x.N$

- If there is an occurrence of x in such N then we say that occurrence of x has been **captured** by (the scope of) λx or $\mathsf{rec}\, x$ to mean that the occurrence of x is bound by the respective λx or $\mathsf{rec}\, x$.

- An occurrence of x in M is **free** iff the occurrence of x is not bound. $\mathit{fvar}(M\ op\ N) \stackrel{\text{def}}{=} \mathit{fvar}(M) \cup \mathit{fvar}(N)$ etc etc

Explaining Substitution

- We substitute a term N for free occurrences of x in M by replacing each free x with N. For example,

$$(\mathsf{if}\ x\ \mathsf{then}\ \underline{4}\ \mathsf{else}\ \underline{5})[\ \underline{1} = \underline{2}\ /x]$$

denotes the term $\mathsf{if}\ \underline{1} = \underline{2}\ \mathsf{then}\ \underline{4}\ \mathsf{else}\ \underline{5}$.

- Suppose that $M \stackrel{\text{def}}{=} \lambda x.L$. Given any term N, then in fact $M\,N \rightsquigarrow L[N/x]$. Thus if L is y, then $M\,N \rightsquigarrow y[N/x] \equiv y$. So if $M \stackrel{\text{def}}{=} \lambda x.y$, M is "the function with constant value y". $M[x/y]$ ought to be "the function with constant value x". But $M[x/y] \equiv \lambda x.x$, the identity!

Defining Substitution

Given M and N, and a variable x, we define $M[N/x]$, *by recursion on the finite tree structure of* M:

- $y[N/x] \stackrel{\text{def}}{=} y$ where $x \neq y$ (if $M \equiv y$);
- $(L\ op\ L')[N/x] \stackrel{\text{def}}{=} L[N/x]\ op\ L'[N/x]$ (if $M \equiv L\ op\ L'$);
- $(\lambda x.L)[N/x] \stackrel{\text{def}}{=} \lambda x.L$;
- $(\lambda y.L)[N/x] \stackrel{\text{def}}{=} \lambda y.L[N/x]$ if $x \neq y$ and $x \notin fvar(L)$ or $y \notin fvar(N)$;
- $(\lambda y.L)[N/x] \stackrel{\text{def}}{=} \lambda z.L[z/y][N/x]$ if $x \neq y$ and $x \in fvar(L)$ and $y \in fvar(N)$, where z is the first variable in Var where $z \notin var(N) \cup var(L)$.

Alpha Equivalence

- We wish to consider two terms differing only in their bound variables as being "equal":

$$\lambda u.(u + \mathsf{rec}\, z.xz) = \lambda w.(w + \mathsf{rec}\, v.xv)$$

- We inductively define an equivalence relation $\sim_\alpha$, on the set $\mathcal{T}$ of terms, by rules such as

$$\frac{}{\lambda v.M \ \sim_\alpha \ \lambda v'.M[v'/v]}\,(\dagger) \qquad \frac{M \ \sim_\alpha \ M' \quad N \ \sim_\alpha \ N'}{M\,N \ \sim_\alpha \ M'\,N'}$$

Defining Expressions

- We define the set $\mathcal{E}$ of **expressions** to be the set of α-equivalence classes of terms:

$$\mathcal{E} \quad \stackrel{\text{def}}{=} \quad \mathcal{T}/\sim_\alpha = \{\, \overline{M} \mid M \in \mathcal{T} \,\}.$$

- We have

$$\begin{aligned} \overline{\lambda u.u + x} &= \{\, M \mid \lambda u.u + x \sim_\alpha M \,\} \\ &= \{\, \lambda u.u + x, \lambda z.z + x, .. \} \\ &= \overline{\lambda z.z + x} \end{aligned}$$

Check this!! Rule (†) gives us $\lambda u.u + x \sim_\alpha \lambda z.z + x$ taking M to be $u + x$, v to be u and v' to be z.

Terms in Context

It is convenient to keep track of the free variables in an expression. We define $\Gamma \vdash M$ where Γ is a set of variables, M is a term, and the free variables of M all appear in Γ. An example is

$$\{ x, y, z \} \vdash x + y$$

We shall inductively define a relation $\vdash$ between finite sets of variables and terms by rules such as

$$\frac{\Gamma \vdash M \quad \Gamma \vdash N \quad \Gamma \vdash L}{\Gamma \vdash \mathsf{if}\ M\ \mathsf{then}\ N\ \mathsf{else}\ L} \qquad \frac{\Gamma, x \vdash M}{\Gamma \vdash \mathsf{rec}\, x.M}$$

We say M is **closed** if $\varnothing \vdash M$

Programs, Values and Evaluation

A **program** is a closed expression. A value is a program that is as "fully evaluated as possible". We give rules which tell us to which values programs evaluate. For example, $(\lambda x.x + \underline{2})\underline{3}$ evaluates to $\underline{5}$:

$$(\lambda x.x + \underline{2})\underline{3} \quad \Downarrow \quad \underline{5}$$

Note that $\underline{5}$ is a value!! Functions $\lambda x.M$ will also be regarded as values. Lists of the form $P : Q$, where P and Q are programs, are also values. The head or tail of a list will only be evaluated *if* "extracted" by a hd or tl function. So $(\underline{3} + \underline{4}) : \mathsf{nil}$ is a value; it does not evaluate to $\underline{7} : \mathsf{nil}$.

Defining Program Evaluation

- A **value** V is any program given by the grammar

$$V ::= \underline{c} \mid \lambda x.M \mid \mathsf{nil} \mid M : M$$

where M ranges over expressions. The set of programs is denoted by $\mathcal{P}$, and values by $\mathcal{V}$.

- We define a binary relation, with relationships denoted by $P \Downarrow V$, as an inductively defined set given by rules such as

$$\frac{}{V \Downarrow V} \quad \frac{P \Downarrow \underline{m} \quad Q \Downarrow \underline{n}}{P \; op \; Q \Downarrow \underline{m \; op \; n}} \quad \frac{P \Downarrow \lambda x.M \quad M[Q/x] \Downarrow V}{P \, Q \Downarrow V}$$

Program Evaluation is Deterministic

The relation $\Downarrow$ is **deterministic**: For any P, V and V', if $P \Downarrow V$ and $P \Downarrow V'$, then $V = V'$.

Proof: We show that the set

$$S \stackrel{\text{def}}{=} \{\ (P, V) \in \Downarrow \ \mid\ \forall V'\ (P \Downarrow V' \Longrightarrow V = V')\ \}$$

is closed under the rules generating $\Downarrow$. Then $S \subset \Downarrow$ is all of $\Downarrow$, and we are done.

Program Transitions

- When a program P does evaluate to a value V, how can we calculate V? We define a new relation $P \rightsquigarrow Q$. The intuitive idea is that if P and Q are related by $\rightsquigarrow$, then P "computes in one step" to Q. For example,

$$(\lambda x.x + \underline{2})\underline{3} \rightsquigarrow \underline{3} + \underline{2} \qquad \text{and} \qquad \underline{3} + \underline{2} \rightsquigarrow \underline{5}.$$

- We define a **transition relation** between programs. It takes the form $P \rightsquigarrow Q$ and is inductively defined by rules such as

$$\frac{P \rightsquigarrow P'}{P \; op \; Q \rightsquigarrow P' \; op \; Q} \qquad \frac{Q \rightsquigarrow Q'}{\underline{n} \; op \; Q \rightsquigarrow \underline{n} \; op \; Q'} \qquad \frac{}{\underline{n} \; op \; \underline{m} \rightsquigarrow \underline{n \; op \; m}}$$

Relating Evaluations and Reductions

We need to relate $\Downarrow$ and $\leadsto$. Consider

$$\begin{aligned} \mathsf{hd}((\lambda x.x + \underline{2})\,\underline{3} : \mathsf{nil}) &\leadsto (\lambda x.x + \underline{2})\,\underline{3} \\ &\leadsto \underline{3} + \underline{2} \\ &\leadsto \underline{5} \end{aligned}$$

and $\mathsf{hd}((\lambda x.x + \underline{2})\,\underline{3} : \mathsf{nil}) \Downarrow \underline{5}$ This suggests that $\Downarrow$ might be the transitive closure of $\leadsto$. In fact $\Downarrow$ is (more-or-less) the reflexive transitive closure $\leadsto^*$. We have

Theorem: For every program P and value V, we have

$$P \Downarrow V \iff P \leadsto^* V.$$

Convergence and Divergence

- P is **terminal** if there is no P' where $P \rightsquigarrow P'$.

- P has a **finite transition sequence** if there is a (unique) transition sequence of the form

$$P \equiv P_0 \rightsquigarrow P_1 \rightsquigarrow P_2 \rightsquigarrow \ldots \rightsquigarrow P_m$$

with P_m terminal. Such P are called **convergent**. If m does not exist, P is **divergent** or **loops**. We write $P \rightsquigarrow^{\omega}$.

- It is easy to see from the definition of $\rightsquigarrow$ that a value V is terminal, and hence convergent.

Charaterizing Divergence Coinductively

We can give a coinductive definition of divergence based solely on the evaluation relation. We coinductively define a subset of $\mathcal{P}$, denoted by $\Uparrow$, by rules such as

$$\frac{P \Uparrow}{P \ op \ Q \Uparrow} \qquad \frac{Q \Uparrow}{P \ op \ Q \Uparrow} \, P \Downarrow \underline{n}$$

$$\frac{P \Uparrow}{P\, Q \Uparrow} \qquad \frac{M[Q/x] \Uparrow}{P\, Q \Uparrow} \, P \Downarrow \lambda x.M \qquad \frac{M[\mathsf{rec}\, x.M/x] \Uparrow}{\mathsf{rec}\, x.M \Uparrow}$$

$$\frac{P \Uparrow}{\mathsf{hd}(P) \Uparrow} \qquad \frac{H \Uparrow}{\mathsf{hd}(P) \Uparrow} \, P \Downarrow H : T \qquad \frac{P \Uparrow}{\mathsf{elist}(P) \Uparrow}$$

Proving that $\Uparrow$ is $\rightsquigarrow^\omega$

Theorem: A program P diverges just in case $P \Uparrow$.

Proof: Set $\mathcal{D} \stackrel{\text{def}}{=} \{ D \in \mathcal{P} \mid D \rightsquigarrow^\omega \}$. Then we need $\mathcal{D} = \nu\, \Phi_{\Uparrow}$, where $\Phi_{\Uparrow} \colon \mathbb{P}(\mathcal{P}) \to \mathbb{P}(\mathcal{P})$ is the name of the set of rules $\mathcal{R}_{\Uparrow}$. We verify $\mathcal{D}$ is a greatest $\mathcal{R}_{\Uparrow}$ dense set, and is thus $\nu\, \Phi_{\Uparrow}$.

We first check that it is $\mathcal{R}_{\Uparrow}$ dense, that is, $\mathcal{D} \subseteq \Phi_{\Uparrow}(\mathcal{D})$. We write down a description of the set $\Phi_{\Uparrow}(\mathcal{D})$ using the definition of $\Phi_{\Uparrow}$. Let X range over $\Phi_{\Uparrow}(\mathcal{D})$; and C_V is any program for which $C_V \Downarrow V$. Then each such X takes the form

$$
\begin{array}{rcll}
X & \equiv & D \; op \; Q & \\
 & | & C_{\underline{n}} \; op \; D & \\
 & | & D\, Q & \\
 & | & C_{\lambda x.M}\, Q & \text{provided that } M[Q/x] \in \mathcal{D} \\
 & | & \mathsf{rec}\, x.M & \text{provided that } M[\mathsf{rec}\, x.M/x] \in \mathcal{D}
\end{array}
$$

We show $X \in \mathcal{D}$ implies $X \in \Phi_{\Uparrow}(\mathcal{D})$ by a structural case analysis of X. First note that X can't be either x or $\underline{c}$.

Case X is $P \; op \; Q$ If $P \in \mathcal{D}$, then $X \in \Phi_{\Uparrow}(\mathcal{D})$. If not, P converges to a terminal, say T, and $X \rightsquigarrow^* T \; op \; Q$. Hence as X diverges, T must be $\underline{m}$ so P has the form $C_{\underline{m}}$, and thus $X \rightsquigarrow^* \underline{m} \; op \; Q \rightsquigarrow^\omega$. It follows that $Q \rightsquigarrow^\omega$.

Case X is $P\,Q$ If $P \in \mathcal{D}$, then $X \in \Phi_{\Uparrow}(\mathcal{D})$. If not, P converges to a terminal, say T, and $X \rightsquigarrow^* T\,Q$. Hence as X diverges, T must be $\lambda x.M$ so P is of the form $C_{\lambda x.M}$. Now note

$$X \rightsquigarrow^* (\lambda x.M)\,Q \rightsquigarrow M[Q/x] \rightsquigarrow^{\omega}$$

because X diverges. Thus $M[Q/x] \in \mathcal{D}$.

We leave the remaining cases as an exercise. So $\mathcal{D}$ is $\mathcal{R}_{\Uparrow}$ dense.

Let $\mathcal{S} \subseteq \Phi_{\Uparrow}(\mathcal{S})$. We show that $\mathcal{D}$ is greatest among all such $\mathcal{R}_{\Uparrow}$ dense sets. To do this, we first prove that

$$\forall S \in \mathcal{E}. \quad S \in \mathcal{S} \Longrightarrow \exists S' \in \mathcal{S}. S \rightsquigarrow S' \qquad (\dagger)$$

by the Principle of Induction for $\mathcal{T}$.

($\mathit{Closure\ under}$ VAR): Of course $x \notin \mathcal{S} \subseteq \mathcal{P}$.

($\mathit{Closure\ under}$ OP): Suppose $P\ op\ Q \in \mathcal{S}$. Note that as $\mathcal{S} \subseteq \Phi_{\Uparrow}(\mathcal{S})$, then either $P \in \mathcal{S}$, or $P \equiv C_{\underline{m}}$ and $Q \in \mathcal{S}$. In the former case, by the induction hypothesis $P \rightsquigarrow P'$ for some P', and hence $P\ op\ Q \rightsquigarrow P'\ op\ Q$. Else if $P \rightsquigarrow P'$ for some P' we are similarly done, and otherwise P must be $\underline{m}$, in which case $P\ op\ Q \rightsquigarrow P\ op\ Q'$ by the induction hypothesis, for some Q'.

We omit the remaining cases (exercise!).

It is quite easy to conclude from †, determinism, and the definition of $\mathcal{D}$, that $\mathcal{S} \subseteq \mathcal{D}$. Thus, as $\nu\,\Phi_{\Uparrow} \subseteq \Phi_{\Uparrow}(\nu\,\Phi_{\Uparrow})$ we have $\nu\,\Phi_{\Uparrow} \subseteq \mathcal{D}$, and as $\mathcal{D}$ is indeed $\mathcal{R}_{\Uparrow}$ dense, equality follows.

Part 3: (Co)Algebras and (Co)Induction

- Define algebras and coalgebras.
- Give an example.
- Illustrate the isomorphism theorems for coalgebras.
- Illustrate categorical induction and coinduction.

Defining Algebras and Coalgebras

- Let $F: \mathcal{C} \to \mathcal{C}$ be an endofunctor. An **algebra** for the functor F is specified by a pair (A, α^A) where A is an object of $\mathcal{C}$ and $\alpha^A: FA \to A$ is a morphism.

- We define the **category of F-algebras**, denoted by $\mathcal{C}^F$, to have objects the algebras of F, and a morphism $f: (A, \alpha^A) \to (B, \alpha^B)$ is a morphism $f: A \to B$ in $\mathcal{C}$ for which the diagram

$$\begin{CD} FA @>{Ff}>> FB \\ @V{\alpha^A}VV @VV{\alpha^B}V \\ A @>>{f}> B \end{CD}$$

commutes in $\mathcal{C}$.

- Dually, a **coalgebra** for F is a pair (A, α^A) where $\alpha^A: A \to FA$ is a morphism in $\mathcal{C}$.
- The **category of F-coalgebras** $\mathcal{C}_F$ is defined similarly to the category of algebras.
- An **initial** F-algebra (I, α^I) is an initial object in $\mathcal{C}^F$
- If (A, α^A) is an F-algebra, we write $\overline{\alpha^A}: (I, \alpha^I) \to (A, \alpha^A)$ for the unique mediating morphism.

Further Notation

Given families of morphisms $(f_i: A \to B_i \mid i \in I)$ and $(g_i: C_i \to D \mid i \in I)$, then

$$\langle f_i \mid i \in I \rangle: A \to \Pi_{i \in I} B_i \quad \text{and} \quad [g_i \mid i \in I]: \Sigma_{i \in I} C_i \to D$$

denote pairing and copairing. Projections are denoted by $\pi_j: \Pi_{i \in I} B_i \to B_j$ and insertions by $ins_j: C_j \to \Sigma_{i \in I} C_i$.

If $f\colon A \to B$ is a morphism in a category with pullbacks, the **kernel** is given by the pullback

$$\begin{array}{ccc} K_f & \xrightarrow{\pi_1} & A \\ {\scriptstyle \pi_2}\downarrow & & \downarrow{\scriptstyle f} \\ A & \xrightarrow[f]{} & B \end{array}$$

K_{id_A} is denoted by Eq_A and the subobject $\langle \pi, \pi \rangle\colon Eq_A \to A \times A$ is called the **equality relation** on A.

An Example ($1 + (A \times -)\colon \mathcal{S}et \to \mathcal{S}et$)

For $k \geq 1$ define A^k to be the collection of k-tuples of elements of A. If $l = (a_1, \ldots, a_k) \in A^k$, then we shall regard l as a partial function $\{ 1, \ldots, k \} \rightharpoonup A$. If $a \in A$ and $l \in A^k$, then define $al \in A^{k+1}$ by $al(1) \stackrel{\text{def}}{=} a$ and $al(r) \stackrel{\text{def}}{=} l(r-1)$ for $r \geq 2$.

The functor $1 + (A \times -)$ has an initial algebra (L, α^L), where we shall set $L \stackrel{\text{def}}{=} \{\, \mathsf{nil} \,\} \cup (\bigcup_{1 \leq k < \omega} A^k)$, and $\alpha^L \colon 1 + (A \times L) \to L$ is defined by

$$
\begin{aligned}
\alpha^L(ins_L(*)) &\stackrel{\text{def}}{=} \mathsf{nil} \\
\alpha^L(ins_R(a, nil)) &\stackrel{\text{def}}{=} a \\
\alpha^L(ins_R(a, l)) &\stackrel{\text{def}}{=} al
\end{aligned}
$$

The functor $1 + (A \times -)$ has a final coalgebra (L, α^L), where $L \stackrel{\text{def}}{=} \{\, \mathsf{nil} \,\} \cup (\bigcup_{1 \le k \le \omega} A^k)$, and $\alpha^L \colon L \to 1 + (A \times L)$ is defined by

$$\alpha^L(\mathsf{nil}) \stackrel{\text{def}}{=} ins_L(*)$$

$$\alpha^L(l) \stackrel{\text{def}}{=} \text{case } l \text{ of}$$

$$l \in A^1 \mapsto ins_R(l(1), \mathsf{nil})$$

$$l \in \bigcup_{2 \le k \le \omega} A^k \mapsto ins_R(l(1), \lambda r. l(r+1))$$

Defining Bisimulations

We define $\mathbb{P}\colon \mathcal{S}et \to \mathcal{S}et$ on objects by $S \mapsto \mathbb{P}\,S$, and on $f\colon S \to T$ by defining $\mathbb{P}\,f\colon \mathbb{P}\,S \to \mathbb{P}\,T$ to be the function

$$\mathbb{P}\,f(X) \stackrel{\mathrm{def}}{=} \{\ fx \ \mid\ x \in X\ \}$$

for each subset X of S. We call a morphism of the form $h\colon (S, \alpha^S) \longrightarrow (T, \alpha^T)$ in $\mathcal{S}et_{\mathbb{P}}$ a **homomorphism**.

- If (S, α^S) is a $\mathbb{P}$-coalgebra, then we shall write $s \overset{\alpha^S}{\rightsquigarrow} s'$ to mean $s' \in \alpha^S(s)$ for any $s, s' \in S$.

- **Lemma** $h\colon (S, \alpha^S) \longrightarrow (T, \alpha^T)$ is a homomorphism just in case for any $s, s' \in S$ and $t \in T$,

  - $s \overset{\alpha^S}{\rightsquigarrow} s' \Longrightarrow hs \overset{\alpha^T}{\rightsquigarrow} hs'$; and
  - $hs \overset{\alpha^T}{\rightsquigarrow} t \Longrightarrow \exists s'.t = hs'$ and $s \overset{\alpha^S}{\rightsquigarrow} s'$.

- A **bisimulation** R on a coalgebra (S, α^S) is an equivalence relation on S for which there is a coalgebra (R, α^R) such that $\pi_1, \pi_2\colon R \to S$ give rise to homomorphisms

$$\pi_1, \pi_2\colon (R, \alpha^R) \longrightarrow (S, \alpha^S)$$

Lemma An equivalence relation R on S is a bisimulation on (S, α^S) just in case for all $s, s', t \in S$, if $s \overset{\alpha^S}{\leadsto} s'$ and $s \; R \; t$, then there exists $t' \in S$ for which $t \overset{\alpha^S}{\leadsto} t'$ and $s' \; R \; t'$.

Quotients by Bisimulations

Given a bisimulation R on a coalgebra (S, α^S), we can endow S/R with a coalgebra structure by defining

$$S/R \xrightarrow{\alpha^{S/R}} \mathbb{P}(S/R)$$

$$[s] \longmapsto (\mathbb{P}\, q \circ \alpha^S)s$$

Note that this is well-defined precisely because R is a bisimulation. It follows that the quotient function is indeed a homomorphism $q\colon (S, \alpha^S) \to (S/R, \alpha^{S/R})$.

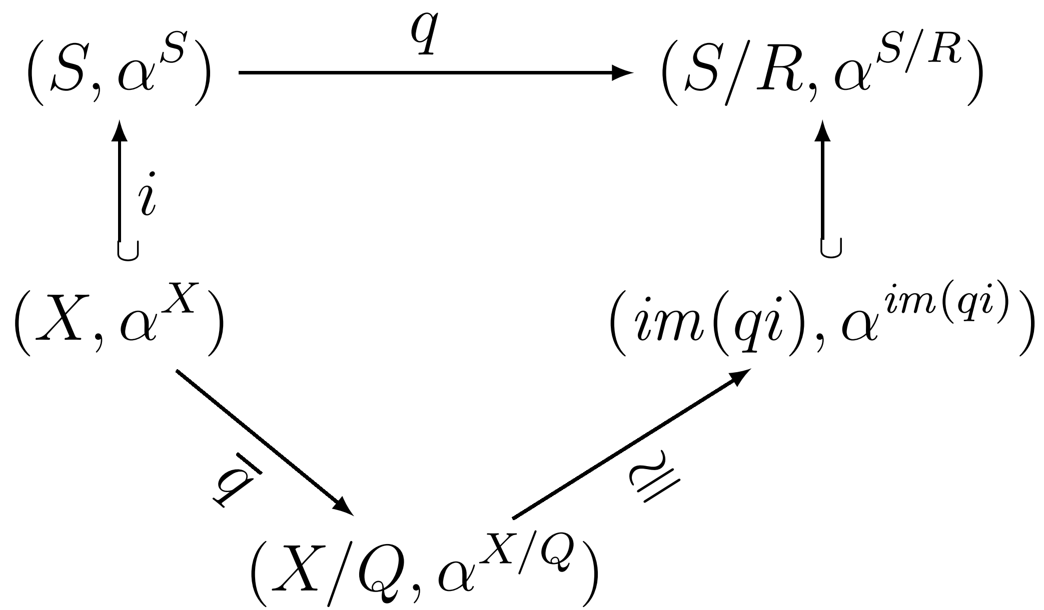
Isomorphism Theorems

Theorem Let $h\colon (S, \alpha^S) \to (T, \alpha^T)$ be a homomorphism, and let K_h be the kernel of h. Then there is a diagram of the form

$$\begin{array}{ccc}
(S, \alpha^S) & \xrightarrow{\quad h \quad} & (T, \alpha^T) \\
q \Big\downarrow & & \Big\uparrow \iota \\
(S/K_h, \alpha^{S/K_h}) & \underset{\psi}{\overset{\phi}{\rightleftarrows}}\ \cong & (im(h), \alpha^{im(h)})
\end{array}$$

for which $h = \iota \circ \phi \circ q$.

Theorem Let (S, α^S) be a coalgebra, (X, α^X) a **subcoalgebra** of (S, α^S), and R a bisimulation on (S, α^S). Set $Q \stackrel{\text{def}}{=} R \cap X^2$, $\overline{X} \stackrel{\text{def}}{=} (\pi_1 \circ \pi_2^{-1})X$, and $Q' \stackrel{\text{def}}{=} R \cap \overline{X}^2$. Then there is a diagram of the form



A Principle of Induction

Let $\alpha^I\colon FI \to I$ be an initial algebra. If $R \rightarrowtail I \times I$ is a binary relation on I, then to show that Eq_I is a subobject of R, it is sufficient to prove that R admits a congruence $\gamma^R\colon FR \to R$ on $\alpha^I\colon FI \to I$.

A Principle of Coinduction

If $\iota: R \rightarrowtail C \times C$ is a binary relation on C, to show R is a subobject of Eq_C, it is sufficient to prove that R admits a bisimulation $\gamma^R: R \to FR$. In order to prove that the global elements $x, x': 1 \to C$ are equal, it is sufficient to prove that $\langle x, x' \rangle: 1 \to C \times C$ factors through $\iota: R \rightarrowtail C \times C$.