10 Techniques for recursion

This chapter provides techniques for proving properties of least fixed points of continuous functions. The characterisation of least fixed points as least prefixed points gives one method sometimes called Park induction. It is used to establish Bekic's Theorem, an important result giving different methods for obtaining least fixed points in products of cpo's. The general method of Scott's fixed-point induction is introduced along with the notion of inclusive property on which it depends; methods for the construction of inclusive properties are provided. A section gives examples of the use of well-founded induction extending our earlier work and, in particular, shows how to build-up well-founded relations. A general method called well-founded recursion is presented for defining functions on sets with a well-founded relation. The chapter concludes with a small but nontrivial exercise using several of the techniques to show the equality of two recursive functions on lists.

10.1 Bekić's Theorem

The Fixed-Point Theorem, Theorem 5.11, of Chapter 5 tells us that if D is a cpo with $\bot$ and $F : D \to D$ is continuous then $\mathit{fix}(F)$ is the least prefixed point of F. In other words,

$$F(d) \sqsubseteq d \Rightarrow \mathit{fix}(F) \sqsubseteq d \qquad \text{(fix1)}$$

for any $d \in D$. Of course, $\mathit{fix}(F)$ is a fixed point, i.e.

$$F(\mathit{fix}(F)) = \mathit{fix}(F) \qquad \text{(fix2)}$$

Facts (fix1) and (fix2) characterise $\mathit{fix}(F)$, and are useful in proving properties of fixed points generally.[1] The fact (fix1) states a principle of proof sometimes called Park induction, after David Park. We will use (fix1) and (fix2) to establish an interesting result due to Bekić. Essentially, Bekić's Theorem says how a simultaneous recursive definition can be replaced by recursive definitions of one coordinate at a time.

Theorem 10.1 *(Bekić)*
Let $F : D \times E \to D$ and $G : D \times E \to E$ be continuous functions where D and E are cpo's with bottom. The least fixed point of $\langle F, G\rangle : D \times E \to D \times E$ is the pair with coordinates

$$\begin{aligned}\hat{f} &= \mu f.\, F(f, \mu g.\, G(\mu f.\, F(f,g), g))\\ \hat{g} &= \mu g.\, G(\mu f.\, F(f,g), g)\end{aligned}$$

[1] In fact, because F is monotonic (fix2) could be replaced by $F(\mathit{fix}(F)) \sqsubseteq \mathit{fix}(F)$. Then by monotonicity, we obtain $F(F(\mathit{fix}(F))) \sqsubseteq F(\mathrm{fix}(F))$, *i.e.* $F(\mathrm{fix}(F))$ is a prefixed point. Now from (fix1) we get $\mathit{fix}(F) \sqsubseteq F(\mathit{fix}(F))$ which yields (fix2) .

Proof: We first show $(\hat{f},\hat{g})$ is a fixed point of $\langle F,G\rangle$. By definition

$$\hat{f} = \mu f.\ F(f,\hat{g}).$$

In other words $\hat{f}$ is the least fixed point of $\lambda f.\ F(f,\hat{g})$. Therefore $\hat{f} = F(\hat{f},\hat{g})$. Also, from the definition of $\hat{g}$,

$$\hat{g} = G(\mu f.\ F(f,\hat{g}),\hat{g}) = G(\hat{f},\hat{g}).$$

Thus $(\hat{f},\hat{g}) = \langle F,G\rangle(\hat{f},\hat{g})$ *i.e.* $(\hat{f},\hat{g})$ is a fixed point of $\langle F,G\rangle$.

Letting (f_0,g_0) be the least fixed point of $\langle F,G\rangle$ we must have

$$f_0 \sqsubseteq \hat{f} \text{ and } g_0 \sqsubseteq \hat{g}. \tag{1}$$

We require the converse orderings as well. As $f_0 = F(f_0,g_0)$,

$$\mu f.\ F(f,g_0) \sqsubseteq f_0.$$

By the monotonicity of G

$$G(\mu f.\ F(f,g_0),g_0) \sqsubseteq G(f_0,g_0) = g_0.$$

Therefore

$$\hat{g} \sqsubseteq g_0 \tag{2}$$

as $\hat{g}$ is the least prefixed point of $\lambda g.\ G(\mu f.\ F(f,g),g)$.

By the monotonicity of F,

$$F(f_0,\hat{g}) \sqsubseteq F(f_0,g_0) = f_0.$$

Therefore

$$\hat{f} \sqsubseteq f_0 \tag{3}$$

as $\hat{f}$ is the least prefixed point of $\lambda f.\ F(f,\hat{g})$.

Combining (1), (2), (3) we see $(\hat{f},\hat{g}) = (f_0,g_0)$, as required. □

The proof only relied on monotonicity and the properties of least fixed points expressed by (fix1) and (fix2) above. For this reason the same argument carries over to the situation of least fixed points of monotonic functions on lattices (see 5.5).

Bekić's Theorem gives an asymmetric form for the simultaneous least fixed point. We can deduce a symmetric form as a corollary: the simultaneous least fixed point is a pair

$$\begin{aligned}\hat{f} &= \mu f.\ F(f,\ \mu g.G(f,g))\\ \hat{g} &= \mu g.\ G(\mu f.F(f,g),\ g)\end{aligned}$$

To see this notice that the second equation is a direct consequence of Bekić's Theorem while the first follows by the symmetry there is between f and g.

Example: We refer to Section 9.8 where it is indicated how to extend **REC** to allow local declarations. Consider the term

$$\begin{array}{rcl} T & \equiv & \textbf{let rec } B \Leftarrow (\textbf{let rec } A \Leftarrow t \textbf{ in } u) \\ & & \textbf{in } (\textbf{let rec } A \Leftarrow t \textbf{ in } v). \end{array}$$

where A and B are assumed to be distinct function variables of arity 0. Let ρ, φ be arbitrary variable and function-variable environments. Abbreviate

$$F(f,g) = [\![t]\!]\varphi[f/A, g/B]\rho$$
$$G(f,g) = [\![u]\!]\varphi[f/A, g/B]\rho$$

From the semantics we see that

$$[\![T]\!]\varphi\rho = [\![v]\!]\varphi[\hat{f}/A, \hat{g}/B]\rho$$

where

$$\begin{array}{rcl} \hat{g} & = & \mu g.\ [\![\textbf{let rec } A \Leftarrow t \textbf{ in } u]\!]\varphi[g/B]\rho \\ & = & \mu g.\ [\![u]\!]\varphi[g/B, \mu f.\ [\![t]\!]\varphi[f/A, g/B]\rho/A]\rho \\ & = & \mu g.\ G(\mu f. F(f,g),\ g). \end{array}$$

and

$$\begin{array}{rl} \hat{f} & = \mu f.\ [\![t]\!]\varphi[f/A, \hat{g}/B]\rho \\ & = \mu f.\ F(f, \hat{g}). \end{array}$$

By Bekić's Theorem this means $(\hat{f}, \hat{g})$ is the (simultaneous) least fixed point of $\langle F, G\rangle$. consequently we could have achieved the same effect with a simultaneous declaration; we have

$$[\![T]\!] = [\![\textbf{let rec } A \Leftarrow t \textbf{ and } B \Leftarrow u \textbf{ in } v]\!].$$

The argument is essentially the same for function variables taking arguments by either call-by-name or call-by-value. Clearly Bekić's Theorem is crucial for establishing program equivalences between terms involving simultaneous declarations and others. □

Exercise 10.2 Generalise and state Bekić's Theorem for 3 equations. □

Exercise 10.3 Let D and E be cpo's with bottom. Prove that if $f : D \to E$ and $g : E \to D$ are continuous functions on cpo's D, E then

$$fix(g \circ f) = g(fix(f \circ g)).$$

(Hint: Use facts (fix1) and (fix2) above.) □

10.2 Fixed-point induction

Often a property can be shown to hold of a least fixed point by showing that it holds for each approximant by mathematical induction. This was the case, for example, in Chapter 5 where, in the proof of Theorem 5.7, stating the equivalence between operational and denotational semantics, the demonstration that

$$(\sigma, \sigma') \in \mathcal{C}[\![c]\!] \Rightarrow \langle c, \sigma \rangle \rightarrow \sigma',$$

for states σ, σ', in the case where the command c was a while-loop, was achieved by mathematical induction on the approximants of its denotation. In this case it was obvious that a property holding of all the approximants of a least fixed point implied that it held of their union, the fixed point itself. This need not be the case for arbitrary properties.

As its name suggests fixed-point induction, a proof principle due to Dana Scott, is useful for proving properties of least fixed points of continuous functions. Fixed-point induction is a proof principle which essentially replaces a mathematical induction along the approximants $F^n(\perp)$ of the least fixed point $\bigsqcup_n F^n(\perp)$ of a continuous function F. However, it is phrased in such a way as to avoid reasoning about the integers. It only applies to properties which are inclusive; a property being inclusive ensures that its holding of all approximants to a least fixed point implies that it holds of the fixed point itself.

Definition: Let D be a cpo. A subset P of D is *inclusive* iff for all ω-chains $d_0 \sqsubseteq d_1 \sqsubseteq \cdots \sqsubseteq d_n \sqsubseteq \cdots$ in D if $d_n \in P$ for all $n \in \omega$ then $\bigsqcup_{n\in\omega} d_n \in P$.

The significance of inclusive subsets derives from the principle of proof called *fixed-point induction.* It is given by the following proposition:

Proposition 10.4 *(Fixed-point induction—Scott)*
Let D be a cpo with bottom $\perp$, and $F : D \rightarrow D$ be continuous. Let P be an inclusive subset of D. If $\perp \in P$ and $\forall x \in D.\ x \in P \Rightarrow F(x) \in P$ then $\mathit{fix}(F) \in P$.

Proof: We have $\mathit{fix}(F) = \bigsqcup_n F^n(\perp)$. If P is an inclusive subset satisfying the condition above then $\perp \in P$ hence $F(\perp) \in P$, and inductively $F^n(\perp) \in P$. As we have seen, by induction, the approximants form an ω-chain

$$\perp \sqsubseteq F(\perp) \sqsubseteq \cdots \sqsubseteq F^n(\perp) \sqsubseteq \cdots$$

whence by the inclusiveness of P, we obtain $\mathit{fix}(F) \in P$. □

Exercise 10.5 What are the inclusive subsets of Ω? Recall Ω is the cpo consisting of:

$$0 \sqsubseteq 1 \sqsubseteq \cdots \sqsubseteq n \sqsubseteq \cdots \infty$$

□

Exercise 10.6 A *Scott-closed* subset of a cpo is the complement of a Scott-open subset (defined in Exercise 8.4). Show a Scott-closed subset is inclusive. Exhibit an inclusive subset of a cpo which is not Scott-closed. □

As a first, rather easy, application of fixed-point induction we show how it implies Park induction, discussed in the last section:

Proposition 10.7 *Let $F : D \to D$ be a continuous function on a cpo D with bottom. Let $d \in D$. If $F(d) \sqsubseteq d$ then $\mathit{fix}(F) \sqsubseteq d$.*

Proof: (via fixed-point induction)
Suppose $d \in D$ and $F(d) \sqsubseteq d$. The subset

$$P = \{x \in D \mid x \sqsubseteq d\}$$

is inclusive—if each element of an ω-chain $d_0 \sqsubseteq \cdots \sqsubseteq d_n \sqsubseteq \cdots$ is below d then certainly so is the least upper bound $\bigsqcup_n d_n$. Clearly $\bot \sqsubseteq d$, so $\bot \in P$. We now show $x \in P \Rightarrow F(x) \in P$. Suppose $x \in P$, *i.e.* $x \sqsubseteq d$. Then, because F is monotonic, $F(x) \sqsubseteq F(d) \sqsubseteq d$. So $F(x) \in P$. By fixed-point induction we conclude $\mathit{fix}(F) \in P$, *i.e.* $\mathit{fix}(F) \sqsubseteq d$, as required. □

Of course, this is a round-about way to show a fact we know from the Fixed-Point Theorem. It does however demonstrate that fixed-point induction is at least as strong as Park induction. In fact fixed-point induction enables us to deduce properties of least fixed points unobtainable solely by applying Park induction.

A predicate $Q(x_1, \ldots, x_k)$ with free variables $x_1, \ldots, x_k$, ranging over a cpo's $D_1, \ldots, D_k$ respectively, determines a subset of $D_1 \times \cdots \times D_k$, *viz.*the set

$$P = \{(x_1, \ldots, x_k) \in D_1 \times \cdots \times D_k \mid Q(x_1, \ldots, x_k)\},$$

and we will say the predicate $Q(x_1, \ldots, x_k)$ is inclusive if its extension as a subset of the cpo $D_1 \times \cdots \times D_k$ is inclusive. As with other induction principles, we shall generally use predicates, rather than their extensions as sets, in carrying out a fixed-point induction. Then fixed-point induction amounts to the following statement:

Let $F : D_1 \times \cdots \times D_k \to D_1 \times \cdots \times D_k$ be a continuous function on a product cpo $D_1 \times \cdots \times D_k$ with bottom element $(\bot_1, \ldots, \bot_k)$. Assuming $Q(x_1, \ldots, x_k)$ is an inclusive predicate on $D_1 \times \cdots \times D_k$,

$$\begin{aligned}&\text{if } Q(\bot_1, \ldots, \bot_k) \text{ and}\\ &\forall x_1 \in D_1, \cdots, x_k \in D_k.\ Q(x_1, \ldots, x_k) \Rightarrow Q(F(x_1, \ldots, x_k))\\ &\text{then } Q(\mathit{fix}(F)).\end{aligned}$$

Fortunately we will be able to ensure that a good many sets and predicates are inclusive because they are built-up in a certain way:

Basic relations: Let D be a cpo. The binary relations

$$\{(x, y) \in D \times D \mid x \sqsubseteq y\} \text{ and } \{(x, y) \in D \times D \mid x = y\}$$

are inclusive subsets of $D \times D$ (Why?). It follows that the predicates

$$x \sqsubseteq y, \qquad x = y$$

are inclusive.

Inverse image and substitution: Let $f : D \to E$ be a continuous function between cpo's D and E. Suppose P is an inclusive subset of E. Then the inverse image

$$f^{-1}P = \{x \in D \mid f(x) \in P\}$$

is an inclusive subset of D.

This has the consequence that inclusive predicates are closed under the substitution of terms for their variables, provided the terms substituted are continuous in their variables. Let $Q(y_1, \ldots, y_l)$ be an inclusive predicate of $E_1 \times \cdots \times E_l$. In other words,

$$P =_{def} \{(y_1, \ldots, y_l) \in E_1 \times \cdots \times E_l \mid Q(y_1, \ldots, y_l)\}$$

is an inclusive subset of $E_1 \times \cdots \times E_l$. Suppose $e_1, \ldots, e_l$ are expressions for elements of $E_1, \ldots, E_l$, respectively, continuous in their variables $x_1, \ldots, x_k$ ranging, in order, over $D_1, \ldots, D_k$—taking them to be expressions in our metalanguage of Section 8.4 would ensure this. Then, defining f to be

$$\lambda x_1, \ldots, x_k.(e_1, \ldots, e_l),$$

ensures f is a continuous function. Thus $f^{-1}P$ is an inclusive subset of $D_1 \times \cdots \times D_k$. But this simply means

$$\{(x_1, \ldots, x_k) \in D_1 \times \cdots \times D_k \mid Q(e_1, \ldots, e_l)\}$$

is an inclusive subset, and thus that $Q(e_1, \ldots, e_l)$ is an inclusive predicate of $D_1 \times \cdots \times D_k$.

For instance, taking $f = \lambda x \in D.\ (x, c)$ we see if $R(x, y)$ is an inclusive predicate of $D \times E$ then the predicate $Q(x) \iff_{def} R(x, c)$, obtained by fixing y to a constant c, is an inclusive predicate of D. Fixing one or several arguments of an inclusive predicate yields an inclusive predicate.

Exercise 10.8 Show that if $Q(x)$ is an inclusive predicate of a cpo D then

$$R(x, y) \iff_{def} Q(x)$$

is an inclusive predicate of $D \times E$, where the extra variable y ranges over the cpo E. (Thus we can "pad-out" inclusive predicates with extra variables. Hint: projection function.) $\square$

Logical operations: Let D be a cpo. The subsets D and $\emptyset$ are inclusive. Consequently the predicates "true" and "false", with extensions D and $\emptyset$ respectively, are inclusive. Let $P \subseteq D$ and $Q \subseteq D$ be inclusive subsets of D. Then

$$P \cup Q \text{ and } P \cap Q$$

are inclusive subsets. In terms of predicates, if $P(x_1, \ldots, x_k)$ and $Q(x_1, \ldots, x_k)$ are inclusive predicates then so are

$$P(x_1, \ldots, x_k) \text{ or } Q(x_1, \ldots, x_k), \qquad P(x_1, \ldots, x_k)\ \&\ Q(x_1, \ldots, x_k)$$

If P_i, $i \in I$, is an indexed family of inclusive subsets of D then $\bigcap_{i \in I} P_i$ is an inclusive subset of D. Consequently, if $P(x_1, \ldots, x_k)$ is an inclusive predicate of $D_1 \times \cdots \times D_k$ then $\forall x_i \in D_i.\ P(x_1, \ldots, x_k)$, with $1 \le i \le k$, is an inclusive predicate of D. This is because the corresponding subset

$$\{(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_k) \in D_1 \times \cdots D_{i-1} \times D_{i+1} \times \cdots \times D_k \mid \forall x_i \in D_i.\ P(x_1, \ldots, x_k)\}$$

equals the intersection,

$$\bigcap_{d \in D_i} \{(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_k) \in D_1 \times \cdots D_{i-1} \times D_{i+1} \times \cdots \times D_k \mid P(x_1, \ldots, x_{i-1}, d, x_{i+1}, \ldots, x_k)\}$$

of inclusive subsets—each predicate $P(x_1, \ldots, x_{i-1}, d, x_{i+1}, \ldots, x_k)$, for $d \in D_i$, is inclusive because it is obtained by fixing one argument.

However, note that infinite unions of inclusive subsets need not be inclusive, and accordingly, that inclusive predicates are not generally closed under $\exists$-quantification.

Exercise 10.9
(i)Provide a counter example which justifies the latter claim.
(ii) Show that the direct image fP of an inclusive subset $P \subseteq D$, under a continuous function $f : D \to E$ between cpo's, need not be an inclusive subset of E.
(iii) Also, provide examples of inclusive subsets $P \subseteq D \times E$ and $Q \subseteq E \times F$ such that their relation composition

$$Q \circ P =_{def} \{(d, f) \mid \exists e \in E.\ (d, e) \in P \& (e, f) \in Q\}$$

is not inclusive.
(Hint for (iii): Take D to be the singleton cpo $\{\top\}$, E to be the discrete cpo of nonnegative integers ω and F to be the cpo Ω consisting of an ω-chain together with its least upper bound ∞.) □

Athough the direct image of an inclusive subset under a general continuous function need not be inclusive, direct images under *order-monics* necessarily preserve inclusiveness. Let D, E be cpo's. A continuous function $f : D \to E$ is an *order-monic* iff

$$f(d) \sqsubseteq f(d') \Rightarrow d \sqsubseteq d'$$

for all $d, d' \in D$. Examples of order-monics include the "lifting" function $\lfloor - \rfloor$ and injections in_i associated with a sum. It is easy to see that if P is an inclusive subset of D then so is its direct image fP when f is an order-monic. This means that if $Q(x)$ is an inclusive predicate of D then

$$\exists x \in D.\ y = f(x) \ \&\ Q(x),$$

with free variable $y \in E$, is an inclusive predicate of E.

Now we can consider inclusive subsets and predicates associated with particular cpo's and constructions on them:

Discrete cpo's: Any subset of a discrete cpo, and so any predicate on a discrete cpo, is inclusive.

Products: Suppose $P_i \subseteq D_i$ are inclusive subsets for $1 \leq i \leq k$. Then

$$P_1 \times \cdots \times P_k = \{(x_1, \ldots, x_k) \mid x_1 \in P_1 \ \&\ \cdots \ \&\ x_k \in P_k\}$$

is an inclusive subset of the product $D_1 \times \cdots \times D_k$. This follows from our earlier results, by noting

$$P_1 \times \cdots \times P_k = \pi_1^{-1} P_1 \cap \cdots \cap \pi_k^{-1} P_k.$$

Each inverse image $\pi_i^{-1}P_i$ is inclusive, for $i = 1, \ldots, k$, and therefore so too is their intersection.

Warning: Let $D_1, \ldots, D_k$ be cpo's. It is tempting to believe that a predicate $P(x_1, \ldots, x_k)$, where $x_1 \in D_1, \cdots, x_k \in D_k$, is an inclusive predicate of the product $D_1 \times \cdots \times D_k$ if it is an inclusive predicate in each argument separately. This is not the case however. More precisely, say $P(x_1, \ldots, x_k)$ is *inclusive in each argument separately*, if for each $i = 1, \ldots, k$, the predicate $P(d_1, \ldots, d_{i-1}, x_i, d_{i+1}, \ldots, d_k)$, got by fixing all but the ith argument, is an inclusive predicate of D_i. Certainly if $P(x_1, \ldots, x_k)$ is inclusive then it is inclusive in each argument separately—we can substitute constants for variables and preserve inclusiveness from the discussion above. The converse does not hold however. The fact that $P(x_1, \ldots, x_k)$ is inclusive in each argument separately does *not* imply that it is an inclusive predicate of $D_1 \times \cdots \times D_k$.

Exercise 10.10 Let Ω be the cpo consisting of ω together with ∞ ordered:

$$0 \sqsubseteq 1 \sqsubseteq \cdots \sqsubseteq n \sqsubseteq \cdots \sqsubseteq \infty$$

By considering the predicate

$$P(x, y) \iff_{def} (x = y \ \& \ x \neq \infty)$$

show that a predicate being inclusive in each argument separately does not imply that it is inclusive. □

Function space: Let D and E be cpo's. Suppose $P \subseteq D$, and $Q \subseteq E$ is an inclusive subset. Then

$$P \to Q =_{def} \{f \in [D \to E] \mid \forall x \in P.\ f(x) \in Q\}$$

is an inclusive subset of the function space $[D \to E]$ (Why?). Consequently, the predicate $\forall x \in D. P(x) \Rightarrow Q(f(x))$, with free variable $f \in [D \to E]$, is inclusive when $P(x)$ is a predicate of D and $Q(y)$ is an inclusive predicate of E.

Lifting: Let P be an inclusive subset of a cpo D. Because the function $\lfloor - \rfloor$ is an order-monic, the direct image $\lfloor P \rfloor = \{\lfloor d \rfloor \mid d \in P\}$ is an inclusive subset of $D_\perp$. If $Q(x)$ is an inclusive predicate of D then

$$\exists x \in D.\ y = \lfloor x \rfloor \ \& \ Q(x),$$

with free variable $y \in D_\perp$, is an inclusive predicate of $D_\perp$.

Sum: Let P_i be an inclusive subset of the cpo D_i for $i = 1, \ldots, k$. Then

$$P_1 + \cdots + P_k = in_1 P_1 \cup \cdots \cup in_k P_k$$

is an inclusive subset of the sum $D_1 + \cdots + D_k$. This follows because each injection is an order-monic so each $in_i P_i$ is inclusive, and the finite union of inclusive sets is inclusive. Expressing the same fact using predicates we obtain that the predicate

$$(\exists x_1 \in D_1.\ y = in_1(x_1)\ \&\ Q_1(x_1)) \text{ or} \cdots \text{or } (\exists x_k \in D_k.\ y = in_k(x_k)\ \&\ Q_k(x_k)),$$

with free variable $y \in D_1 + \cdots + D_k$, is an inclusive predicate of the sum if each $Q_i(x_i)$ is an inclusive predicate of the component D_i.

The methods described above form the basis of a a language of inclusive predicates. Provided we build up predicates from basic inclusive predicates using the methods admitted above then they are guaranteed to be inclusive. For example, any predicate built-up as a universal quantification over several variables of conjunctions and disjunctions of basic predicates of the form $e_1 \sqsubseteq e_2$ for terms e_1, e_2 in our metalanguage will be inclusive.

Proposition 10.11 *Any predicate of the form*

$$\forall x_1, \ldots, x_n.\ P$$

is inclusive where $x_1, \ldots, x_n$ are variables ranging over specific cpo's, and P is built up by conjunctions and disjunctions of basic predicates of the form $e_0 \sqsubseteq e_1$ or $e_0 = e_1$, where e_0 and e_1 are expressions in the metalanguage of expressions from Section 8.4.

Unfortunately, such syntactic means fail to generate all the predicates needed in proofs and the manufacture of suitable inclusive predicates can become extremely difficult when reasoning about recursively defined domains.

Example: Let $\mathbf{T}_\perp$ be the usual complete partial order of truth values $\{\mathbf{true}, \mathbf{false}\}_\perp$. Abbreviate $\lfloor\mathbf{true}\rfloor$ to *tt* and $\lfloor\mathbf{false}\rfloor$ to *ff*. Let $p : D \to \mathbf{T}_\perp$ and $h : D \to D$ be continuous with h strict (*i.e.* $h(\perp) = \perp$). Let $f : D \times D \to D$ be the least continuous function such that

$$f(x, y) = p(x) \to y \mid h(f(h(x), y))$$

for all $x, y \in D$. We prove

(i) $h(b \to d|e) = b \to h(d)|h(e)$ for all $b \in \mathbf{T}_\perp$ and $d, e \in D$, and
(ii) $h(f(x, y)) = f(x, h(y))$ for all $x, y \in D$.

Part (i) follows easily by considering the three possible values $\perp, tt, ff$ for $b \in \mathbf{T}_\perp$.

If $b = \perp$	then	$h(b \to d\|e) = h(\perp) = \perp = b \to h(d)\|h(e)$
If $b = tt$	then	$h(b \to d\|e) = h(d) = b \to h(d)\|h(e)$
If $b = ff$	then	$h(b \to d\|e) = h(e) = b \to h(d)\|h(e)$

Hence the required equation holds for all possible values of the boolean b.

Part (ii) follows by fixed-point induction. An appropriate predicate is

$$P(g) \Leftrightarrow_{def} \forall x, y \in D.\ h(g(x,y)) = g(x, h(y))$$

The predicate $P(g)$ is inclusive because it can be built-up by the methods described earlier. Because h is strict we see that $P(\perp)$ is true. To apply fixed-point induction we require further that

$$P(g) \Rightarrow P(F(g))$$

where $F(g) = \lambda x, y.\ p(x) \rightarrow y \mid (h(g(h(x), y))$.

Assume $P(g)$. Let $x, y \in D$. Then

$$\begin{array}{rcll} h((F(g))(x,y)) & = & h(p(x) \rightarrow y \mid h(g(h(x), y))) & \\ & = & p(x) \rightarrow h(y) \mid h^2(g(h(x), y)), & \text{by (i)} \\ & = & p(x) \rightarrow h(y) \mid h(g(h(x), h(y))), & \text{by the assumption } P(g) \\ & = & (F(g))(x, h(y)) & \end{array}$$

Thus $P(F(g))$. Hence $P(g) \Rightarrow p(F(g))$.

By fixed-point induction, we deduce $P(\mathit{fix}(F))$ *i.e.* $P(f)$ *i.e.* $\forall x, y \in D.\ h(f(x,y)) = f(x, h(y))$ as required. □

Exercise 10.12 Define $h : \mathbf{N} \rightarrow \mathbf{N}_\perp$ recursively by

$$h(x) = h(x) +_\perp \lfloor 1 \rfloor$$

Show $h = \perp$, the always-$\perp$ function, using fixed-point induction. □

Exercise 10.13 Let D be a cpo with bottom. Let $p : D \rightarrow \mathbf{T}_\perp$ be continuous and strict (*i.e.* $p(\perp) = \perp$) and $h : D \rightarrow D$ be continuous. Let $f : D \rightarrow D$ to be the least continuous function which satisfies

$$f(x) = p(x) \rightarrow x \mid f(f(h(x)))$$

for all $x \in D$. Prove

$$\forall x \in D.\ f(f(x)) = f(x).$$

(Hint:Take as induction hypothesis the predicate

$$P(g) \iff {}_{def} \forall x \in D.\ f(g(x)) = g(x).)$$

□

Exercise 10.14 Let $h, k : D \to D$ be continuous functions on a cpo D with bottom, with h strict. Let $p : D \to \mathbf{T}_\perp$ be a continuous function. Let f, g be the least continuous functions $D \times D \to D$ satisfying

$$\begin{aligned} f(x,y) &= p(x) \to y \mid h(f(k(x),y)) \\ g(x,y) &= p(x) \to y \mid g(k(x),h(y)) \end{aligned}$$

for all $x, y \in D$. Using fixed-point induction show $f = g$.
(Hint: Regard the solutions as simultaneous fixed points and take the inclusive predicate to be

$$P(f,g) \iff {}_{def}\forall x,y.\ [f(x,y) = g(x,y)\ \&\ g(x,h(y)) = h(g(x,y))].)$$

□

It is probably helpful to conclude this section with a general remark on the use of fixed-point induction. Faced with a problem of proving a property holds of a least fixed point it is often not the case that an inclusive property appropriate to fixed point induction suggests itself readily. Like induction hypotheses, or invariants of programs, spotting a suitable inclusive property frequently requires fairly deep insight. The process of obtaining a suitable inclusive property can often make carrying out the actual proof a routine matter. It can sometimes be helpful to start by exploring the first few approximants to a least fixed point, with the hope of seeing a pattern which can be turned into an induction hypothesis. The proof can then be continued by mathematical induction on approximants (provided the property holding of each approximant implies it holds of the least fixed point), or, often more cleanly, by fixed-point induction (provided the property is inclusive).

10.3 Well-founded induction

Fixed-point induction is inadequate for certain kinds of reasoning. For example, suppose we want to show a recursively defined function on the integers always terminates on integer inputs. We cannot expect to prove this directly using fixed-point induction. To do so would involve there being an inclusive predicate P which expressed termination and yet was true of $\perp$, the completely undefined function. An extra proof principle is needed which can make use of the way data used in a computation is inductively defined. An appropriate principle is that of well-founded induction. Recall from Chapter 3 that a well-founded relation on a set A is a binary relation $\prec$ which does not have any infinite descending chains. Remember the principle of well-founded induction says:

Let $\prec$ be a well founded relation on a set A. Let P be a property. Then $\forall a \in A.\ P(a)$ iff

$$\forall a \in A.\ ([\forall b \prec a.\ P(b)] \Rightarrow P(a)).$$

Applying the principle often depends on a judicious choice of well-founded relation. We have already made use of well-founded relations like that of proper subexpression on syntactic sets, or $<$ on natural numbers. Here some well-known ways to construct further well-founded relations are given. Note that we use $x \preceq y$ to mean ($x \prec y$ or $x = y$).

Product: If $\prec_1$ is well-founded on A_1 and $\prec_2$ is well-founded on A_2 then taking

$$(a_1, a_2) \preceq (a_1', a_2') \Leftrightarrow_{def} a_1 \preceq_1 a_1' \text{ and } a_2 \preceq_2 a_2'$$

determines a well-founded relation $\prec = (\preceq \setminus 1_{A_1 \times A_2})$ in $A_1 \times A_2$. However product relations are not as generally applicable as those produced by lexicographic orderings.

Lexicographic products: Let $\prec_1$ be well-founded on A_1 and $\prec_2$ be well-founded on A_2. Define

$$(a_1, a_2) \prec_{lex} (a_1', a_2') \text{ iff } a_1 \prec_1 a_1' \text{ or } (a_1 = a_1' \ \&\ a_2 \prec_2 a_2')$$

Inverse image: Let $f : A \to B$ be a function and $\prec_B$ a well-founded relation on B. Then $\prec_A$ is well-founded on A where

$$a \prec_A a' \Leftrightarrow_{def} f(a) \prec_B f(a')$$

for $a, a' \in A$.

Exercise 10.15 Let $\prec$ be a well-founded relation on a set X such that $\preceq$ is a total order. Show it need not necessarily satisfy

$$\{x \in X \mid x \prec y\}$$

is finite for all $y \in X$.
(A total order is a partial order $\leq$ such that $x \leq y$ or $y \leq x$ for all its elements x, y.)
(Hint: Consider the lexicographic product of $<$ and $<$ on $\omega \times \omega$.) □

Exercise 10.16 Show the product, lexicographic product and inverse image constructions do produce well-founded relations from well-founded relations. □

Example: A famous example is Ackermann's function which can be defined in **REC** by the declaration:

$$\begin{aligned} A(x,y) = \ & \textbf{if } x \textbf{ then } \ y+1 \textbf{ else} \\ & \textbf{if } y \textbf{ then } \ A(x-1,1) \textbf{ else} \\ & \qquad\qquad\ A(x-1, A(x,y-1)) \end{aligned}$$

Under the denotational semantics for call-by-value, this declares A to have denotation the least function a in $[\mathbf{N}^2 \to \mathbf{N}_\perp]$ such that

$$a(m,n) = \begin{cases} \lfloor n+1 \rfloor & \text{if } \ m = 0 \\ a(m-1,1) & \text{if } \ m \neq 0, n = 0 \\ \mathit{let}\ l \Leftarrow a(m,n-1).\ a(m-1,l) & \text{otherwise} \end{cases}$$

for all $m,n \in \mathbf{N}$. The fact that Ackermann's function $a(m,n)$ terminates on all integers $m,n \geq 0$ is shown by well-founded induction on (m,n) ordered lexicographically. □

Exercise 10.17 Prove Ackermann's function $a(m,n)$ terminates on all integers $m,n \geq 0$ by well-founded induction by taking as induction hypothesis

$$P(m,n) \Leftrightarrow_{def} (a(m,n) \neq \perp \text{ and } a(m,n) \geq 0)$$

for $m,n \geq 0$. □

Exercise 10.18 The 91 function of McCarthy is defined to be the least function in $[\mathbf{N} \to \mathbf{N}_\perp]$ such that

$$f(x) = \mathit{cond}(x > 100,\ \lfloor x-10 \rfloor,\ \mathit{let}\ y \Leftarrow f(x+11).\ f(y)).$$

(This uses the conditional of 8.3.5)
Show this implies

$$f(x) = \mathit{cond}(x > 100,\ \lfloor x-10 \rfloor,\ \lfloor 91 \rfloor)$$

for all nonnegative integers x. Use well-founded induction on ω with relation

$$n \prec m \Leftrightarrow m < n \leq 101,$$

for $n,m \in \omega$. First show $\prec$ is a well-founded relation. □

10.4 Well-founded recursion

In Chapter 3 we noticed that both definition by induction and structural induction allow a form of recursive definition, that the length of an arithmetic expression can, for instance, be defined recursively in terms of the lengths of its strict subexpressions; how the length function acts on a particular argument, like $(a_1 + a_2)$ is specified in terms of how the length function acts on strictly smaller arguments, like a_1 and a_2. In a similar way we are entitled to define functions on an arbitrary well-founded set. Suppose B is a set with a well-founded relation $\prec$. Definition by well-founded induction, called well-founded recursion, allows the definition of a function f from B by specifying its value $f(b)$ at an arbitrary b in B in terms of $f(b')$ for $b' \prec b$. We need a little notation to state and justify the general method precisely. Each element b in B has a set of predecessors

$$\prec^{-1} \{b\} = \{b' \in B \mid b' \prec b\}.$$

For any $B' \subseteq B$, a function $f : B \to C$ restricts to a function $f \restriction B' : B' \to C$ by taking

$$f \restriction B' = \{(b, f(b)) \mid b \in B'\}.$$

Definition by well-founded recursion is justified by the following theorem:

Theorem 10.19 *(Well-founded recursion)*
Let $\prec$ be a well-founded relation on a set B. Suppose $F(b,h) \in C$, for all $b \in B$ and functions $h : \prec^{-1} \{b\} \to C$. There is a unique function $f : B \to C$ such that

$$\forall b \in B.\ f(b) = F(b, f \restriction \prec^{-1} \{b\}). \qquad (*)$$

Proof: The proof has two parts. We first show a uniqueness property:

$$\begin{aligned}&\forall y \prec^* x.\ f(y) = F(y, f \restriction \prec^{-1} \{y\})\ \&\ g(y) = F(y, g \restriction \prec^{-1} \{y\})\\ &\quad \Rightarrow f(x) = g(x),\end{aligned}$$

for any $x \in B$. This uniqueness property $P(x)$ is proved to hold for all $x \in B$ by well-founded induction on $\prec$: For $x \in B$, assume $P(z)$ for every $z \prec x$. We require $P(x)$. To this end suppose

$$f(y) = F(y, f \restriction \prec^{-1} \{y\})\ \&\ g(y) = F(y, g \restriction \prec^{-1} \{y\})$$

for all $y \prec^* x$. If $z \prec x$, then as $P(z)$ we obtain

$$f(z) = g(z).$$

Hence

$$f \restriction \prec^{-1}\{x\} = g \restriction \prec^{-1}\{x\}.$$

It now follows that

$$f(x) = F(x, f \restriction \prec^{-1} \{x\}) = F(x, g \restriction \prec^{-1} \{x\}) = g(x).$$

Thus $P(x)$.

It follows that there can be at most one function f satisfying $(*)$. We now show that there exists such a function. We build the function by unioning together a set of functions $f_x : {\prec^*}^{-1}\{x\} \to C$, for $x \in B$. To show suitable functions exist we prove the following property $Q(x)$ holds for all $x \in B$ by well-founded induction on $\prec$:

$$\exists f_x : {\prec^*}^{-1}\{x\} \to C. \\ \forall y \prec^* x.\ f_x(y) = F(y, f_x \restriction \prec^{-1} \{y\}).$$

Let $x \in B$. Suppose $\forall z \prec x.\ Q(z)$. Then we claim

$$h = \bigcup \{f_z \mid z \prec x\}$$

is a function. Certainly it is a relation giving at least one value for every argument $z \prec x$. The only difficulty is in checking the functions f_z agree on values assigned to common arguments y. But they must—otherwise we would violate the uniqueness property proved above. Taking

$$f_x = h \cup \{(x, F(x, h))\}$$

gives a function $f_x : {\prec^*}^{-1}\{x\} \to C$ such that

$$\forall y \prec^* x.\ f_x(y) = F(y, f_x \restriction \prec^{-1} \{y\}).$$

This completes the well-founded induction, yielding $\forall x \in B.\ Q(x)$.

Now we take $f = \bigcup_{x \in B} f_x$. By the uniqueness property, this yields $f : B \to C$, and moreover f is the unique function satisfying $(*)$. □

Well-founded recursion and induction constitute a general method often appropriate when functions are intended to be total. For example, it immediately follows from the recursion theorem that that there is a unique total function on the nonnegative integers such that

$$ack(m, n) = \begin{cases} n + 1 & \text{if } m = 0 \\ ack(m - 1, 1) & \text{if } m \neq 0, n = 0 \\ ack(m - 1, ack(m, n - 1)) & \text{otherwise} \end{cases}$$

for all $m, n \geq 0$; observe that the value of ack at the pair (m, n) is defined in terms of its values at the lexicographically smaller pairs $(m - 1, 1)$ and $(m, n - 1)$. In fact, a great many recursive programs are written so that some measure within a well-founded set decreases as they are evaluated. For such programs often the machinery of least fixed points can be replaced by well-founded recursion and induction.

10.5 An exercise

We round off this chapter with an exercise showing that two recursive functions on lists are equal. The solution of this single problem brings together many of the techniques for reasoning about recursive definitions. We have tended to concentrate on arithmetical and boolean operations. Here we look instead at operations on finite lists of integers. An *integer-list* is typically of the form

$$[m_1; m_2; \ldots; m_k]$$

consisting of k elements from $\mathbf{N}$. The *empty list* is also a list which will be written as:

$$[\,]$$

There are two basic operations for constructing lists. One is the constant operation taking the empty tuple of arguments () to the empty list [ ]. The other is generally called *cons* and prefixes an integer m to the front of a list l, the result of which is written as:

$$m :: l$$

Thus, for example,

$$1 :: [2; 3; 4] = [1; 2; 3; 4].$$

The set of integer-lists forms a discrete cpo which we will call *List*. It is built up as the sum of two discrete cpo's

$$List = in_1\{()\} \cup in_2(\mathbf{N} \times List) = \{()\} + (\mathbf{N} \times List)$$

with respect to the injection functions which act so:

$$\begin{aligned} in_1() &= [\,] \quad \text{and} \\ in_2(m, l) &= m :: l. \end{aligned}$$

That lists can be regarded as a sum in this way reflects the fact that the discrete cpo of integer-lists is isomorphic to that of all tuples of integers including the ().

The sum is accompanied by a cases construction

$$\begin{aligned} case\ l\ of\ & [\,].\ e_1 \mid \\ & x :: l'.\ e_2. \end{aligned}$$

Its use is illustrated in a recursive definition of a function

$$append : List \times List \to (List)_\perp$$

which performs the operation of appending two lists:

$$\begin{aligned} append = \mu\alpha.\ \lambda l, ls \in List.\ & \\ & case\ l\ of\ [\,].\ \lfloor ls \rfloor \mid \\ & \qquad x :: l'.\ (let\ r \Leftarrow \alpha(l', ls).\ \lfloor x :: r \rfloor). \end{aligned}$$

The function append is the least α function in the cpo $[List \times List \to (List)_\perp]$ which satisfies

$$\begin{aligned} \alpha([\,], ls) &= \lfloor ls \rfloor \\ \alpha(x :: l', ls) &= (let\ r \Leftarrow \alpha(l', ls).\ \lfloor x :: r \rfloor). \end{aligned}$$

An induction on the size of list in the first argument ensures that *append* is always total. Relating lists by $l' \prec l$ iff the list l' is strictly smaller than the list l, we might instead define a slightly different append operation on lists $@ : List \times List \to List$ by well-founded recursion. By the well-founded recursion, Theorem 10.19, @ is the unique (total) function such that

$$\begin{aligned} l@ls = case\ l\ of\ [\,].\ & ls \mid \\ & x :: l'.\ \ x :: (l'@ls) \end{aligned}$$

for all $l, ls \in List$. The two functions can be proved to be related by

$$append(l, ls) = \lfloor l@ls \rfloor,$$

for all lists l, ls, by well-founded induction.

Now we can state the problem:

Exercise 10.20 Assume functions on integers $s : \mathbf{N} \times \mathbf{N} \to \mathbf{N}$ and $r : \mathbf{N} \times \mathbf{N} \to List$. Let f be the least function in $[List \times \mathbf{N} \to \mathbf{N}_\perp]$ satisfying

$$\begin{aligned} f([\,], y) &= \lfloor y \rfloor \\ f(x :: xs, y) &= f(r(x,y)@xs, s(x,y)). \end{aligned}$$

Let g be the least function in $[List \times \mathbf{N} \to \mathbf{N}_\perp]$ satisfying

$$\begin{aligned} g([\,], y) &= \lfloor y \rfloor \\ g(x :: xs, y) &= let\ v \Leftarrow g(r(x,y), s(x,y)).\ g(xs, v). \end{aligned}$$

Prove $f = g$.
Hints: First show g satisfies

$$g(l@xs, y) = let\ v \Leftarrow g(l, y).\ g(xs, v)$$

by induction on the size of list l. Deduce $f \sqsubseteq g$. Now show f satisfies

$$(\mathit{let}\ u \Leftarrow f(l,y).\ f(xs,u)) \sqsubseteq f(l@xs,y)$$

by fixed-point induction—take as inclusive predicate

$$P(F) \iff_{def} [\forall xs,l,y.\ (\mathit{let}\ u \Leftarrow F(l,y).\ f(xs,u)) \sqsubseteq f(l@xs,y)].$$

Deduce $g \sqsubseteq f$. □

10.6 Further reading

The presentation of this chapter has been influenced by [80], [59], and [89]. In particular, Manna's book [59] is a rich source of exercises in fixed point and well-founded induction (though unfortunately the latter principle is called "structural induction" there). I am grateful to Larry Paulson for the problem on lists. The reader is warned that the terminology for the concept of "inclusive" property and predicate is not universal. The term "inclusive" here is inherited from Gordon Plotkin's lecture notes [80]. Others use "admissible" but there are other names too. The issue of terminology is complicated by option of developing domain theory around directed sets rather than ω-chains—within the wide class of ω-algebraic cpo's this yields an equivalent notion, although it does lean on the terminology used. Other references are [13], [58] and [21] (though the latter wrongly assumes a predicate on a product cpo is inclusive if inclusive in each argument separately). Enderton's book [39] contains a detailed treatment of well-founded recursion (look up references to "recursion" in the index of [39], and bear in mind his proofs are with respect to a "well ordering," a *transitive* well-founded relation.)