Least Fixpoints Calculationally

Lambert Meertens*
Department of Algorithmics and Architecture, CWI, Amsterdam, and
Department of Computing Science, Utrecht University, The Netherlands
`http://www.cwi.nl/cwi/people/Lambert.Meertens.html`

Printed April 30, 1998

1 Functions

Function application. The application of function F to argument x may be denoted either as Fx or equivalently as $F.x$, where the purpose of the latter notation is to improve readability. In FGx, the implied parsing is that of $F(Gx)$.

Unless the contrary is stated, the function typing $F : A \to B$ states that F is total and has domain $\mathrm{dom}(F) = A$, so that, for $x \in A$, we have $Fx \in B$.

If $x \notin \mathrm{dom}(F)$, we say that Fx is *undefined*, or equivalently that Fx *does not exist*.

Two expressions are called *synonymous* if for all possible bindings of free variables they are either both undefined, or they are both defined and have the same value. Thus, if x is a free variable, Fx is synonymous with Gx iff $F = G$. Synonymy is an equivalence relation. It is also substitutive: if

*This work was performed while visiting Kestrel Institute, Palo Alto.

expressions E and E' are synonymous, then so are $C[E]$ and $C[E']$ for all contexts $C[\_]$.

Function restriction. If F is a function, and P a predicate such that $\mathrm{dom}(P) = \mathrm{dom}(F)$, the *function restriction* $(P : F)$ means: the function F restricted to arguments satisfying P. The parentheses are an obligatory part of the notation.

We have the following rules for composing a function and a function restriction:

$$
\begin{aligned}
G \circ (P : F) \quad &= \quad (P : G \circ F) \\
(P : F) \circ G \quad &= \quad (P \circ G : F \circ G)
\end{aligned}
$$

Function comprehension. The function $(P : F)$ can be written as a *function comprehension*, namely as $(z : Pz : Fz)$. A more traditional notation for the function $(n : n \in \mathbb{N} : 2^n)$ is the lambda form $\lambda\, n{\in}\mathbb{N}.2^n$. If there is no range restriction, it is assumed to be inferred from the context. Thus, if n may be assumed to be a natural-number dummy, we may write $(n :: 2^n)$. In $F = (x :: Fx)$, the implied range restriction is $x \in \mathrm{dom}(F)$. While function restrictions have a single colon, function comprehensions sport two colons.

We have the application rule:

$$
G\,(z : Pz : Fz) \;=\; (z : Pz : G.Fz)
$$

The introduction or elimination of a dummy variable in a calculation step will be done tacitly.

Lifting. A value $a \in A$ can be *lifted* to a constant function $a^{\mathsf{K}} : Z \to A$, polymorphic in the type variable Z, so $a^{\mathsf{K}}.x = a$ for any x. We have the following composition rules:

$$
\begin{aligned}
F \circ a^{\mathsf{K}} \quad &= \quad (Fa)^{\mathsf{K}} \\
a^{\mathsf{K}} \circ F \quad &= \quad a^{\mathsf{K}}
\end{aligned}
$$

A binary operator $\oplus : X \times Y \to A$ can be *lifted* in three ways: left, right and both, as follows. Let $x \in X, y \in Y, F : Z \to X, G : Z \to Y$. Then:

$$\begin{array}{lcl} x \stackrel{\bullet}{\oplus} G & = & (z :: \ x \ \oplus (Gz)) \ : \ Z \to A \\ F \stackrel{\bullet}{\oplus} y & = & (z :: (Fz) \oplus \ y \ ) \ : \ Z \to A \\ F \stackrel{\bullet}{\oplus} G & = & (z :: (Fz) \oplus (Gz)) \ : \ Z \to A \end{array}$$

So the dot indicates where the function argument goes: left, right, or both. Furthermore,

$$\begin{array}{lcl} F \stackrel{\bullet}{\oplus} y & = & F \stackrel{\bullet}{\oplus} y^{\mathsf{K}} \\ x \stackrel{\bullet}{\oplus} G & = & x^{\mathsf{K}} \stackrel{\bullet}{\oplus} G \\ (x \oplus y)^{\mathsf{K}} & = & x^{\mathsf{K}} \stackrel{\bullet}{\oplus} y^{\mathsf{K}} \end{array}$$

We have the following composition rules:

$$\begin{array}{rcl} H \circ F \stackrel{\bullet}{\oplus} G & = & (H \circ F) \ \stackrel{\bullet}{\oplus} \ (H \circ G) \\ F \stackrel{\bullet}{\oplus} G \circ H & = & (F \circ H) \ \stackrel{\bullet}{\oplus} \ (G \circ H) \end{array}$$

As is well known, algebraic properties of an operator $\oplus$, like associativity, symmetry and idempotence, are preserved by lifting it to $\stackrel{\bullet}{\oplus}$. If $\oplus$ is a relation (a binary predicate), properties like transitivity and reflexivity are likewise preserved.

Conditional. For $x, y \in A$, the function $x \diamond y : \{0, 1\} \to A$ is defined by:

$$\begin{array}{lcl} (x \diamond y)\, 0 & = & x \\ (x \diamond y)\, 1 & = & y \end{array}$$

We have: $F \circ x \diamond y = (Fx) \diamond (Fy)$.

Definition by characterization. A function F may be defined by characterization, for example in the form

$$y = Fx \quad \equiv \quad P(x, y)$$

This is an acceptable definition if P is such that

$$y = y' \quad \Leftarrow \quad P(x, y) \wedge P(x, y')$$

and then $x \in \mathrm{dom}(F)$ iff there exists y such that $P(x, y)$. In general, the characterization involves a proposition that has at most one solution in Fx.

2 Quantifications

Definition. A *quantifier* on some domain A is a possibly partial function $\bigoplus : (Z \to A) \to A$, polymorphic in the type variable Z, that satisfies the following three properties.

rearranging:	$\bigoplus F \;=\; \bigoplus(F \circ J)$ for all bijections J
1-pt rule:	$\bigoplus(\mathsf{id} \doteq x : F) \;=\; Fx$ for all $x \in \mathrm{dom}(F)$
range split:	$\bigoplus(P \mathbin{\dot\vee} Q : F) \;=\; \bigoplus(P : F) \oplus \bigoplus(Q : F) \;\Leftarrow\; P \nmid Q$

in which the binary operator $\oplus : A \to A \times A$, called the *binary version* of $\bigoplus$, is defined by

$$x \oplus y \;=\; \bigoplus(x \diamond y)$$

and $P \nmid Q$ means that the two predicates are mutually exclusive: no value satisfies both P and Q. If $\oplus$ is idempotent, this condition may be dropped.

An example of a quantifier is $\sum$. Its binary version is $+$. Another example is $\forall$, with binary version $\wedge$.

If $\oplus$ is the binary version of quantifier $\bigoplus$, we also say that $\bigoplus$ is a quantifier for $\oplus$. Only associative and symmetric operators can have quantifiers.

Definedness. The application of a quantifier to a function is called a *quantification*. The definedness of the quantifications involved in the rules above is to be understood as follows. In the rearranging rule, the two quanifications are either both defined or both undefined, so the two sides of the equation are synonymous. A quantification as in the 1-pt rule is always defined for $x \in \mathrm{dom}(F)$; again the two sides of the equation are synonymous. In the range-split rule, the left-hand quantification is defined if both right-hand quantifications are defined. If the left-hand quantification is defined and both P and Q are satisfiable — i.e. not constantly false — then both right-hand quantifications are defined. Together this ensures that at least all *non-empty finite* quantifications — i.e., in which the argument function has a non-empty and finite domain — are defined.

In the following it will be assumed, either explicitly or implicitly, that in each use of a quantification the argument is in the definedness domain of the quantifier — unless the contrary is stated.

Non-uniqueness. A given operator may have different quantifiers. For example, defining $\forall'$ by

$$\forall' F = \begin{cases} \forall F & \text{if } \mathrm{dom}(F) \text{ is finite} \\ \text{false} & \text{otherwise} \end{cases}$$

$\forall'$ is also a quantifier for $\wedge$. So the quantifier properties do not guarantee uniqueness. They do guarantee uniqueness for non-empty finite quantifications. Extra properties that may narrow the set of possible quantifiers $\bigoplus$ for an operator $\oplus$ are:

$$\begin{array}{ll} \text{empty quantification:} & \bigoplus(\text{false}^{\mathsf{K}} : \mathsf{id}) = e \\ & \qquad \text{if } e \text{ is a neutral element for } \oplus \\ \text{constant quantification:} & \bigoplus(P : x^{\mathsf{K}}) = x \quad \Leftarrow \quad x \oplus x = x \\ & \qquad \text{for satisfiable } P \end{array}$$

(Together these two imply, for a neutral element e for $\oplus$, that $\bigoplus(P : e^{\mathsf{K}}) = e$ for all P.) In general, though, quantifiers need an independent definition or characterization.

∀-rules. In addition to the general quantification properties introduced above, we use the following rules for ∀.

∀-specialization:	$\forall(P \circ F : Q \circ F) \;\Leftarrow\; \forall(P : Q)$
∀-swap:	$\forall(x : Px : \forall(y : Qy : R(x, y))) \;\equiv\;$ $\forall(y : Qy : \forall(x : Px : R(x, y)))$
∀-∧-distributivity:	$\forall(P : Q \mathbin{\dot\wedge} R) \;\equiv\; \forall(P : Q) \wedge \forall(P : R)$
∀-shunting:	$\forall(P : Q) \;\equiv\; \forall(Q \mathrel{\dot\Leftarrow} P)$
∀-true:	$\forall(P : \mathrm{true}^{\mathsf{K}})$
∀-reflexivity:	$\forall(P : P)$
∀-monotonicity:	$(\forall(P : Q) \Leftarrow \forall(P : R)) \;\Leftarrow\; \forall(R : Q)$
∀-range widening:	$(\forall(P : R) \Leftarrow \forall(Q : R)) \;\Leftarrow\; \forall(P : Q)$
∀-instantiation:	$(Qz \Leftarrow \forall(P : Q)) \;\Leftarrow\; Pz$

These rules are not independent. For example, the monotonicity and the range-widening rule are different presentations of basically the same rule and express the transitivity of $\dot\Leftarrow$; likewise, the ∀-instantiation rule is a 1-pt version of range widening. These rules are given separately and as they are because that is the form in which they are used in actual calculations. Appeals to ∀-true and ∀-monotonicity will be made tacitly.

Furthermore, we use the traditional mathematical proof technique of introducing a fresh variable z and deriving Qz from Pz to obtain a proof of $\forall(P : Q)$.

3 Indirect (in)equality

Let $(A, \sqsupseteq)$ be a poset. A useful rule for deriving inequations of the form $x \sqsupseteq y$ is:

indirect inequality: $x \sqsupseteq y \;\equiv\; \forall(y \mathrel{\dot\sqsupseteq} \mathsf{id} : x \mathrel{\dot\sqsupseteq} \mathsf{id})$

or, equivalently, using $\forall$-shunting and introducing a dummy:

indirect inequality: $x \sqsupseteq y \quad \equiv \quad \forall(z :: x \sqsupseteq z \Leftarrow y \sqsupseteq z)$

The rule is useful in particular when y has a form for which $y \sqsupseteq z$ is readily rewritten. In using the rule to prove an inequation $x \sqsupseteq y$, we usually jump ahead and derive $x \sqsupseteq z$ from $y \sqsupseteq z$ for "arbitrary" z without bothering to introduce z.

Proof. By transitivity of $\sqsupseteq$ the right-hand side follows from the left-hand side. For the other way around,

$$
\begin{array}{ll}
 & x \sqsupseteq y \\
\equiv & \qquad \{\sqsupseteq \text{ is reflexive, propositional calculus}\} \\
 & x \sqsupseteq y \Leftarrow y \sqsupseteq y \\
\Leftarrow & \qquad \{\forall\text{-instantiation}\} \\
 & \forall(z :: x \sqsupseteq z \Leftarrow y \sqsupseteq z)
\end{array}
$$

End of proof.

Using the rule twice, once for $x \sqsupseteq y$ and once for $y \sqsupseteq x$, we obtain the following rule:

indirect equality: $x = y \quad \equiv \quad \forall(z :: x \sqsupseteq z \equiv y \sqsupseteq z)$

Dual rules. By duality we also have:

indirect inequality: $y \sqsupseteq x \quad \equiv \quad \forall(z :: z \sqsupseteq x \Leftarrow z \sqsupseteq y)$

indirect equality: $y = x \quad \equiv \quad \forall(z :: z \sqsupseteq x \equiv z \sqsupseteq y)$

4 Infima

$\sqcap$-characterization. Let $(A, \sqsupseteq)$ be a poset. Define $\sqcap$ on A by

$\sqcap$-characterization: $\sqcap F \sqsupseteq z \quad \equiv \quad \forall(F \mathrel{\dot{\sqsupseteq}} z)$ for all $z \in A$

whenever this has a solution in $\sqcap F$; for other F, the expression $\sqcap F$ is not defined.

The function $\sqcap$ is uniquely determined by the $\sqcap$-characterization rule. This is shown as follows: Suppose there is another function $\sqcap'$ satisfying $\sqcap' F \sqsupseteq z \equiv \forall(F \mathrel{\dot{\sqsupseteq}} z)$ for all $z \in A$ whenever this has a solution. Then

$$
\begin{array}{ll}
 & \sqcap' F \; = \; \sqcap F \\
\equiv & \qquad \{\text{indirect equality}\} \\
 & \forall(z :: \sqcap' F \sqsupseteq z \;\equiv\; \sqcap F \sqsupseteq z) \\
\equiv & \qquad \{\text{property of } \sqcap', \sqcap\text{-characterization}\} \\
 & \forall(z :: \forall(F \mathrel{\dot{\sqsupseteq}} z) \;\equiv\; \forall(F \mathrel{\dot{\sqsupseteq}} z)) \\
\equiv & \qquad \{\equiv \text{ is reflexive}\} \\
 & \text{true}
\end{array}
$$

$\sqcap$ is a quantifier. We show that $\sqcap$ is a quantifier by establishing the three quantifier properties. For rearranging, under the assumption that J is a bijection:

$$
\begin{array}{ll}
 & \sqcap F \; = \; \sqcap(F \circ J) \\
\equiv & \qquad \{\text{indirect equality}\} \\
 & \sqcap F \sqsupseteq z \;\equiv\; \sqcap(F \circ J) \sqsupseteq z \\
\equiv & \qquad \{\sqcap\text{-characterization}\} \\
 & \forall(F \mathrel{\dot{\sqsupseteq}} z) \;\equiv\; \forall((F \circ J) \mathrel{\dot{\sqsupseteq}} z) \\
\equiv & \qquad \{\text{rearranging}\} \\
 & \text{true}
\end{array}
$$

For the 1-pt rule, under the assumption that $x \in \mathrm{dom}(F)$:

$$
\begin{array}{ll}
 & \sqcap(\mathsf{id} \mathrel{\dot{=}} x : F) \; = \; Fx \\
\equiv & \qquad \{\text{indirect equality}\} \\
 & \sqcap(\mathsf{id} \mathrel{\dot{=}} x : F) \sqsupseteq z \;\equiv\; Fx \sqsupseteq z \\
\equiv & \qquad \{\sqcap\text{-characterization}\} \\
 & \forall(\mathsf{id} \mathrel{\dot{=}} x : F \mathrel{\dot{\sqsupseteq}} z) \;\equiv\; Fx \sqsupseteq z \\
\equiv & \qquad \{\text{1-pt rule}\}
\end{array}
$$

true

Before proceeding, we prove an important rule:

$\sqcap$-characterization: $x \sqcap y \sqsupseteq z \;\equiv\; x \sqsupseteq z \wedge y \sqsupseteq z$

in which $\sqcap$ denotes the binary version of $\bigsqcap$.

$$
\begin{array}{ll}
 & x \sqcap y \sqsupseteq z \\
\equiv & \qquad \{\text{definition of } \sqcap\} \\
 & \bigsqcap(x \diamond y) \sqsupseteq z \\
\equiv & \qquad \{\bigsqcap\text{-characterization}\} \\
 & \forall((x \sqsupseteq z) \diamond (y \sqsupseteq z)) \\
\equiv & \qquad \{\wedge \text{ is the binary version of } \forall\} \\
 & x \sqsupseteq z \wedge y \sqsupseteq z
\end{array}
$$

For range split, we calculate now:

$$
\begin{array}{ll}
 & \bigsqcap(P \dot{\vee} Q : F) \;=\; \bigsqcap(P : F) \sqcap \bigsqcap(Q : F) \\
\equiv & \qquad \{\text{indirect equality}\} \\
 & \bigsqcap(P \dot{\vee} Q : F) \sqsupseteq z \;\equiv\; (\bigsqcap(P : F) \sqcap \bigsqcap(Q : F)) \sqsupseteq z \\
\equiv & \qquad \{\sqcap\text{-characterization}\} \\
 & \bigsqcap(P \dot{\vee} Q : F) \sqsupseteq z \;\equiv\; \bigsqcap(P : F) \sqsupseteq z \wedge \bigsqcap(Q : F) \sqsupseteq z \\
\equiv & \qquad \{\bigsqcap\text{-characterization}\} \\
 & \forall(P \dot{\vee} Q : F \mathrel{\dot{\sqsupseteq}} z) \;\equiv\; \forall(P : F \mathrel{\dot{\sqsupseteq}} z) \wedge \forall(Q : F \mathrel{\dot{\sqsupseteq}} z) \\
\equiv & \qquad \{\text{range split}\} \\
 & \text{true}
\end{array}
$$

Note that we did not need the assumption that $P \nmid Q$. In fact, it is obvious from $\sqcap$-characterization that $\sqcap$ is idempotent.

5 Properties of $\bigsqcap$

$\bigsqcap$-range widening and $\bigsqcap$-instantiation. Since the truth values form a lattice, all rules for $\bigsqcap$ and $\sqcap$ are valid under the replacements

$$\begin{array}{ccc} \bigsqcap & \rightsquigarrow & \forall \\ \sqcap & \rightsquigarrow & \wedge \\ \sqsupseteq & \rightsquigarrow & \Leftarrow \end{array}$$

We now derive two general $\bigsqcap$-rules that are generalizations of $\forall$-rules we saw before:

$$\begin{array}{ll} \bigsqcap\text{-range widening:} & \bigsqcap(P:F) \sqsupseteq \bigsqcap(Q:F) \quad \Leftarrow \quad \forall(P:Q) \\ \bigsqcap\text{-instantiation:} & Fx \sqsupseteq \bigsqcap(P:F) \quad \Leftarrow \quad Px \end{array}$$

The rules are given in this form because it is the form that we tend to use in actual calculations.

$\bigsqcap$-range widening is established by indirect inequality:

$$\begin{array}{ll} & \bigsqcap(P:F) \sqsupseteq z \;\Leftarrow\; \bigsqcap(Q:F) \sqsupseteq z \\ \equiv & \qquad \{\bigsqcap\text{-characterization}\} \\ & \forall(P : F \mathrel{\dot{\sqsupseteq}} z) \;\Leftarrow\; \forall(Q : F \mathrel{\dot{\sqsupseteq}} z) \\ \Leftarrow & \qquad \{\forall\text{-range widening}\} \\ & \forall(P:Q) \end{array}$$

For $\bigsqcap$-instantiation we now have:

$$\begin{array}{ll} & Fx \\ = & \qquad \{\text{1-pt rule}\} \\ & \bigsqcap(\mathsf{id} \mathrel{\dot{=}} x : F) \\ \sqsupseteq & \qquad \{\bullet\ Px,\ \bigsqcap\text{-range widening}\} \\ & \bigsqcap(P:F) \end{array}$$

6 The least x satisfying P

Let P be a predicate whose domain is the carrier A of some partial order $(A, \sqsupseteq)$. Then λP denotes the least value x in A — if such a value exists — such that x satisfies P. Otherwise, λP is undefined. There is not necessarily a least x in A satisfying P. There may be no x at all such that Px holds; or if there is, there may always be a smaller one, or the minimal solutions may be incomparable.

λ-characterization. $x = \lambda P$ if the following two properties hold:

satisfaction: Px

limitation: $\forall(P : \mathsf{id} \mathrel{\dot{\sqsupseteq}} x)$

(If we replace "$\mathsf{id} \mathrel{\dot{\sqsupseteq}} x$" by "$x \mathrel{\dot{\not\sqsupseteq}} \mathsf{id}$", we get *minimality*, which is not sufficient to characterize the solution, if any.)

We show by mutual inequation that these properties together imply $x = \sqcap(P : \mathsf{id})$, thereby establishing the uniqueness of λP:

$$\begin{array}{ll} & x \sqsupseteq \sqcap(P : \mathsf{id}) \\ \Leftarrow & \quad \{\sqcap\text{-instantiation}\} \\ & Px \end{array}$$

and

$$\begin{array}{ll} & \sqcap(P : \mathsf{id}) \sqsupseteq x \\ \equiv & \quad \{\sqcap\text{-characterization}\} \\ & \forall(P : \mathsf{id} \mathrel{\dot{\sqsupseteq}} x) \end{array}$$

So each of the two properties supplies one inequation. The last part also shows that the equation $\sqcap(P : \mathsf{id}) \sqsupseteq x$ already establishes the property called "limitation". The full equation $x = \sqcap(P : \mathsf{id})$ is obviously not strong enough to establish the other property, x satisfies P. Still, the expression $\sqcap(P : \mathsf{id})$ — assuming that the quantifier application is defined — defines "something", only it is not necessarily λP.

The situation can be summed up as follows. Abbreviating $\pi = \lambda P$ and $\hat{\pi} = \bigsqcap(P : \mathsf{id})$:

$$\pi \text{ exists } \wedge \ \hat{\pi} \text{ exists } \wedge \ \pi = \hat{\pi}$$

$$\Leftarrow$$

$$\hat{\pi} \text{ exists } \wedge \ P\hat{\pi}$$

$$\Leftarrow$$

$$\pi \text{ exists}$$

By elementary propositional calculus we obtain as a corollary the λ-$\bigsqcap$-synonymy rule:

$$\pi \text{ is synonymous with } \hat{\pi} \quad \equiv \quad (P\hat{\pi} \ \Leftarrow \ \hat{\pi} \text{ exists})$$

$\bigsqcap$-closed predicates. We would like to have additional conditions on P that allow us to conclude to Px from $x = \bigsqcap(P : \mathsf{id})$, thereby establishing that λP and $\bigsqcap(P : \mathsf{id})$ are synonymous. Here is one. Define a predicate P to be $\bigoplus$-*closed* whenever for any restriction $(Q : P)$ of P we have: $P \bigoplus(Q : \mathsf{id}) \ \Leftarrow \ \forall(Q : P)$. Then

$$P \bigoplus(P : \mathsf{id}) \quad \Leftarrow \quad P \text{ is } \bigoplus\text{-closed}$$

Proof.

$$\begin{array}{ll} & P \bigoplus(P : \mathsf{id}) \\ \Leftarrow & \qquad \{\bullet \ P \text{ is } \bigoplus\text{-closed}\} \\ & \forall(P : P) \\ \equiv & \qquad \{\forall\text{-reflexivity}\} \\ & \text{true} \end{array}$$

End of proof.

So if P is $\bigsqcap$-closed, λP is synonymous with $\bigsqcap(P : \mathsf{id})$.

$\sqcap$-completeness implies $\sqcup$-completeness. A poset $(A, \sqsupseteq)$ is called $\sqcap$-*complete* if for all functions F whose target is A the quantification $\sqcap F$ is defined.

We define $\sqcup$ as the dual of $\sqcap$. More precisely, $\sqcup$ is defined on a poset $(A, \sqsupseteq)$ by:

$$\sqcup\text{-characterization:}\ \ z \sqsupseteq \sqcup F \quad\equiv\quad \forall(z \mathrel{\dot{\sqsupseteq}} F) \ \text{ for all } z \in A$$

Putting $x = \sqcup F$ and $Pz = \forall(z \mathrel{\dot{\sqsupseteq}} F)$, we rewrite this as:

$$z \sqsupseteq x \quad\equiv\quad Pz \ \text{ for all } z \in A$$

We show that this implies that $x = \lambda P$. To establish satisfaction is easy:

$$\begin{array}{ll} & Px \\ \equiv & \qquad \{\sqcup\text{-characterization}\} \\ & x \sqsupseteq x \\ \equiv & \qquad \{\sqsupseteq \text{ is reflexive}\} \\ & \text{true} \end{array}$$

To show limitation:

$$\begin{array}{ll} & \forall(P : \mathsf{id} \mathrel{\dot{\sqsupseteq}} x) \\ \equiv & \qquad \{\text{definition of } P\} \\ & \forall(P : P) \\ \equiv & \qquad \{\forall\text{-reflexivity}\} \\ & \text{true} \end{array}$$

We saw that λP — when defined — equals $\sqcap(P : \mathsf{id})$. So if $\sqcup F$ exists, it can be defined in terms of $\sqcap$, namely by:

$$\sqcup F \quad=\quad \sqcap(z : \forall(z \mathrel{\dot{\sqsupseteq}} F) : z)$$

We prove that $\sqcup F$, thus defined, satisfies $\sqcup$-characterization, thereby establishing its existence under the assumption of $\sqcap$-completeness. We use x and P as before to keep the formulas simple, and establish $z \sqsupseteq x \equiv Pz$. The proof proceeds by mutual implication.

Proof.

$$
\begin{array}{ll}
 & z \sqsupseteq x \\
\equiv & \qquad \{\text{definition of } x\} \\
 & z \sqsupseteq \bigsqcap(P : \mathsf{id}) \\
\Leftarrow & \qquad \{\bigsqcap\text{-instantiation}\} \\
 & Pz \\
\equiv & \qquad \{\text{definition of } P\} \\
 & \forall(z \mathrel{\dot{\sqsupseteq}} F) \\
\equiv & \qquad \{\text{non-restriction}\} \\
 & \forall(\text{true}^{\mathsf{K}} : z \mathrel{\dot{\sqsupseteq}} F) \\
\Leftarrow & \qquad \{\bigsqcap\text{-range widening (see below)}\} \\
 & \forall(x \mathrel{\dot{\sqsupseteq}} F : z \mathrel{\dot{\sqsupseteq}} F) \\
\equiv & \qquad \{\text{composition rules}\} \\
 & \forall(x \mathrel{\dot{\sqsupseteq}} \mathsf{id} \circ F : z \mathrel{\dot{\sqsupseteq}} \mathsf{id} \circ F) \\
\Leftarrow & \qquad \{\forall\text{-specialization}\} \\
 & \forall(x \mathrel{\dot{\sqsupseteq}} \mathsf{id} : z \mathrel{\dot{\sqsupseteq}} \mathsf{id}) \\
\equiv & \qquad \{\text{indirect inequality}\} \\
 & z \sqsupseteq x
\end{array}
$$

For the range-widening step we have the following proof obligation: $\forall(\text{true}^{\mathsf{K}} : x \mathrel{\dot{\sqsupseteq}} F)$ or, equivalently, $\forall(x \mathrel{\dot{\sqsupseteq}} F)$. For this we argue:

$$
\begin{array}{ll}
 & \forall(x \mathrel{\dot{\sqsupseteq}} F) \\
\equiv & \qquad \{\text{definition of } x\} \\
 & \forall(\bigsqcap(P : \mathsf{id}) \mathrel{\dot{\sqsupseteq}} F) \\
\equiv & \qquad \{\bigsqcap\text{-characterization}\} \\
 & \forall(y :: \forall(z : Pz : z \sqsupseteq Fy)) \\
\equiv & \qquad \{\forall\text{-swap}\} \\
 & \forall(z : Pz : \forall(y :: z \sqsupseteq Fy)) \\
\equiv & \qquad \{\text{definition of } P\} \\
 & \forall(z : Pz : Pz) \\
\equiv & \qquad \{\forall\text{-reflexivity}\} \\
 & \text{true}
\end{array}
$$

End of proof.

7 Adjoints

Let $(A, \sqsupseteq)$ and $(B, \sqsupseteq)$ be two posets, and let $F : A \to B$. Function $F^\flat : B \to A$ is called a *lower adjoint* of F if, for all $x \in A$ and $y \in B$:

$$x \sqsupseteq F^\flat y \quad \equiv \quad Fx \sqsupseteq y$$

Dually, function $F^\sharp : B \to A$ is called an *upper adjoint* of F if for all $x \in A$ and $y \in B$:

$$F^\sharp y \sqsupseteq x \quad \equiv \quad y \sqsupseteq Fx$$

Clearly, $G = F^\flat \equiv F = G^\sharp$.

Adjoints imply monotonicity. If a function has a lower adjoint, it is unique, justifying the $\_^\flat$ notation. To see this, assume that both G and H are lower adjoints of F. We show by indirect equality that then $Gy = Hy$ for all $y \in B$:

$$\begin{array}{ll} & z \sqsupseteq Gy \;\equiv\; z \sqsupseteq Hy \\ \equiv & \qquad \{\text{adjoints}\} \\ & Fz \sqsupseteq y \;\equiv\; Fz \sqsupseteq y \\ \equiv & \qquad \{\equiv \text{ is reflexive}\} \\ & \text{true} \end{array}$$

Dually, upper adjoints are unique.

Further, if F has a lower adjoint, F is monotonic.

Proof.

$$\begin{array}{ll} & Fx \sqsupseteq Fy \\ \Leftarrow & \qquad \{\text{transitivity of } \sqsupseteq\} \end{array}$$

$$
\begin{array}{ll}
 & Fx \sqsupseteq Fy \;\Leftarrow\; Fy \sqsupseteq Fy \\
\equiv & \qquad \{\bullet\ F \text{ has a lower adjoint}\} \\
 & x \sqsupseteq F^{\flat}.Fy \;\Leftarrow\; y \sqsupseteq F^{\flat}.Fy \\
\Leftarrow & \qquad \{\text{transitivity of } \sqsupseteq\} \\
 & x \sqsupseteq y
\end{array}
$$

End of proof.

Dually, functions having an upper adjoint are monotonic. Since lower adjoints have upper adjoints and *vice versa*, adjoints are monotonic as well.

$F^{\flat}y$ expressed as λP. We show that $F^{\flat}y = \lambda(F \stackrel{\bullet}{\sqsupseteq} y)$ for monotonic F, or, more precisely,

$$
\begin{array}{l}
F : A \to B \text{ has a lower adjoint iff} \\
F \text{ is monotonic and } \lambda(F \stackrel{\bullet}{\sqsupseteq} y) \text{ exists for all } y \in B
\end{array}
$$

and then $F^{\flat}y = \lambda(F \stackrel{\bullet}{\sqsupseteq} y)$.

Proof. For brevity, put, with implicit argument y, $P = F \stackrel{\bullet}{\sqsupseteq} y$.

(If part) Assume that λP is defined for all $y \in B$, and let function $F^{\flat}$ be defined by $F^{\flat}y = \lambda P$. First we derive the auxiliary result that $F \circ F^{\flat}$ is augmenting, or, $F.F^{\flat}y \sqsupseteq y$:

$$
\begin{array}{ll}
 & F.F^{\flat}y \sqsupseteq y \\
\equiv & \qquad \{\text{definition of } P\} \\
 & P.F^{\flat}y \\
\equiv & \qquad \{\text{definition of } F^{\flat}\} \\
 & P.\lambda P \\
\equiv & \qquad \{\bullet\ \lambda P \text{ is defined, satisfaction}\} \\
 & \text{true}
\end{array}
$$

Using this, we show that for monotonic F, the function $F^{\flat}$ satisfies the lower-adjoint characterization. The equivalence is shown by mutual implication.

$$
\begin{array}{ll}
& x \sqsupseteq F^\flat y \\
\equiv & \qquad \{\text{definition of } F^\flat\} \\
& x \sqsupseteq \lambda P \\
\equiv & \qquad \{\bullet\ \lambda P \text{ is defined, so } \lambda P = \sqcap(P : \mathsf{id})\} \\
& x \sqsupseteq \sqcap(P : \mathsf{id}) \\
\Leftarrow & \qquad \{\sqcap\text{-instantiation}\} \\
& Px \\
\equiv & \qquad \{\text{definition of } P\} \\
& Fx \sqsupseteq y \\
\Leftarrow & \qquad \{F \circ F^\flat \text{ is augmenting, } \sqsupseteq \text{ is transitive}\} \\
& Fx \sqsupseteq F.F^\flat y \\
\Leftarrow & \qquad \{\bullet\ F \text{ is monotonic}\} \\
& x \sqsupseteq F^\flat y
\end{array}
$$

(Only-if part) Assume that F has a lower adjoint $F^\flat$. It was shown before that then F is monotonic. We first establish the existence of $\sqcap(P : \mathsf{id})$ by showing that, taking $\sqcap(P : \mathsf{id})$ to be $F^\flat y$, $\sqcap$-characterization is satisfied:

$$
\begin{array}{ll}
& \sqcap(P : \mathsf{id}) \sqsupseteq x \\
\equiv & \qquad \{\sqcap(P : \mathsf{id}) = F^\flat y\} \\
& F^\flat y \sqsupseteq x \\
\equiv & \qquad \{\text{indirect inequality}\} \\
& \forall(\mathsf{id} \mathrel{\dot\sqsupseteq} F^\flat y : \mathsf{id} \mathrel{\dot\sqsupseteq} x) \\
\equiv & \qquad \{F^\flat \text{ is lower adjoint}\} \\
& \forall(F \mathrel{\dot\sqsupseteq} y : \mathsf{id} \mathrel{\dot\sqsupseteq} x) \\
\equiv & \qquad \{\text{definition of } P\} \\
& \forall(P : \mathsf{id} \mathrel{\dot\sqsupseteq} x)
\end{array}
$$

To establish now the existence of λP, all that is left to do is to show that P is $\sqcap$-closed:

$$
\begin{array}{ll}
& P.\sqcap(Q : \mathsf{id}) \\
\equiv & \qquad \{\text{definition of } P\} \\
& F.\sqcap(Q : \mathsf{id}) \sqsupseteq y \\
\equiv & \qquad \{\bullet\ F^\flat \text{ is lower adjoint}\}
\end{array}
$$

$$
\begin{array}{ll}
& \sqcap(Q : \mathsf{id}) \;\sqsupseteq\; F^\flat y \\
\equiv & \qquad \{\sqcap\text{-characterization}\} \\
& \forall(Q : \mathsf{id} \mathrel{\dot{\sqsupseteq}} F^\flat y) \\
\equiv & \qquad \{\bullet\ F^\flat \text{ is lower adjoint}\} \\
& \forall(Q : F \mathrel{\dot{\sqsupseteq}} y) \\
\equiv & \qquad \{\text{definition of } P\} \\
& \forall(Q : P)
\end{array}
$$

End of proof.

8 Least fixpoints

The least fixpoint μF of a *monotonic* function $F:\ A \to A$ on a poset $(A, \sqsupseteq)$, if it exists, is given by $\mu F = \lambda(\mathsf{id} \doteq F)$. A definition that is more convenient in proofs, is its expression as the quantification:

$$\mu F \quad = \quad \sqcap(\mathsf{id} \mathrel{\dot{\sqsupseteq}} F : \mathsf{id})$$

We will show that the two definitions are synonymous. In fact, introducing the abbreviations

$$
\begin{array}{lcl}
\varphi & = & \lambda(\mathsf{id} \doteq F) \\
\hat{\varphi} & = & \sqcap(\mathsf{id} \doteq F : \mathsf{id}) \\
\pi & = & \lambda(\mathsf{id} \mathrel{\dot{\sqsupseteq}} F) \\
\hat{\pi} & = & \sqcap(\mathsf{id} \mathrel{\dot{\sqsupseteq}} F : \mathsf{id})
\end{array}
$$

we show that all four are synonymous.

Proof. Introduce the abbreviation $P = \mathsf{id} \mathrel{\dot{\sqsupseteq}} F$.

First we show the synonymy of π and $\hat{\pi}$, which, by the λ-$\sqcap$-synonymy rule, amounts to showing that $\hat{\pi}$, if defined, satisfies P. As we saw, it suffices to show that P is $\sqcap$-closed, or, $P.\sqcap(Q : \mathsf{id}) \;\Leftarrow\; \forall(Q : P)$ for all Q. This is established as follows:

$$
\begin{array}{ll}
 & P.\bigsqcap(Q : \mathsf{id}) \\
\equiv & \qquad \{\text{definition of } P\} \\
 & \bigsqcap(Q : \mathsf{id}) \;\sqsupseteq\; F.\bigsqcap(Q : \mathsf{id}) \\
\equiv & \qquad \{\bigsqcap\text{-characterization}\} \\
 & \forall(Q : \mathsf{id} \mathrel{\dot{\sqsupseteq}} F.\bigsqcap(Q : \mathsf{id})) \\
\equiv & \qquad \{\text{lifting}\} \\
 & \forall(Q : \mathsf{id} \mathrel{\dot{\sqsupseteq}} F \circ (\bigsqcap(Q : \mathsf{id}))^{\mathsf{K}}) \\
\Leftarrow & \qquad \{\dot{\sqsupseteq} \text{ is transitive}\} \\
 & \forall(Q : \mathsf{id} \mathrel{\dot{\sqsupseteq}} F \mathrel{\dot{\wedge}} F \mathrel{\dot{\sqsupseteq}} F \circ (\bigsqcap(Q : \mathsf{id}))^{\mathsf{K}}) \\
\equiv & \qquad \{\text{definition of } P\} \\
 & \forall(Q : P \mathrel{\dot{\wedge}} F \mathrel{\dot{\sqsupseteq}} F \circ (\bigsqcap(Q : \mathsf{id}))^{\mathsf{K}}) \\
\Leftarrow & \qquad \{F \text{ is monotonic}\} \\
 & \forall(Q : P \;\dot{\wedge}\; \mathsf{id} \mathrel{\dot{\sqsupseteq}} (\bigsqcap(Q : \mathsf{id}))^{\mathsf{K}}) \\
\equiv & \qquad \{\forall\text{-}\wedge\text{-distributivity}\} \\
 & \forall(Q : P) \;\wedge\; \forall(Q : \mathsf{id} \mathrel{\dot{\sqsupseteq}} (\bigsqcap(Q : \mathsf{id}))^{\mathsf{K}}) \\
\equiv & \qquad \{\text{see below}\} \\
 & \forall(Q : P)
\end{array}
$$

For the last step we must show the validity of $\forall(Q : \mathsf{id} \mathrel{\dot{\sqsupseteq}} (\bigsqcap(Q : \mathsf{id}))^{\mathsf{K}})$:

$$
\begin{array}{ll}
 & \forall(Q : \mathsf{id} \mathrel{\dot{\sqsupseteq}} (\bigsqcap(Q : \mathsf{id}))^{\mathsf{K}}) \\
\equiv & \qquad \{\text{shunting}\} \\
 & \forall(z :: z \sqsupseteq \bigsqcap(Q : \mathsf{id}) \;\Leftarrow\; Qz) \\
\equiv & \qquad \{\bigsqcap\text{-instantiation}\} \\
 & \text{true}
\end{array}
$$

Next, we show by mutual inequation that $\hat{\varphi}$ and $\hat{\pi}$ are synonymous. First,

$$
\begin{array}{ll}
 & \hat{\varphi} \;\sqsupseteq\; \hat{\pi} \\
\equiv & \qquad \{\text{definition of } \hat{\varphi} \text{ and } \hat{\pi}\} \\
 & \bigsqcap(\mathsf{id} \doteq F : \mathsf{id}) \;\sqsupseteq\; \bigsqcap(\mathsf{id} \mathrel{\dot{\sqsupseteq}} F : \mathsf{id}) \\
\Leftarrow & \qquad \{\bigsqcap\text{-range widening}\} \\
 & \forall(\mathsf{id} \doteq F : \mathsf{id} \mathrel{\dot{\sqsupseteq}} F) \\
\equiv & \qquad \{\sqsupseteq \text{ is reflexive}\}
\end{array}
$$

true

Next,

$$
\begin{array}{ll}
 & \hat{\pi} \sqsupseteq \hat{\varphi} \\
\equiv & \qquad \{\text{definition of } \hat{\varphi}\} \\
 & \hat{\pi} \sqsupseteq \bigsqcap(\mathsf{id} \doteq F : \mathsf{id}) \\
\Leftarrow & \qquad \{\bigsqcap\text{-instantiation}\} \\
 & \hat{\pi} = F\hat{\pi} \\
\equiv & \qquad \{\text{proved above}\} \\
 & \text{true}
\end{array}
$$

We have now established the synonymy of $\hat{\varphi}$, π and $\hat{\pi}$.

Finally, we show that φ and $\hat{\varphi}$ are synonymous, thus establishing the synonymy of all four. By the λ-$\bigsqcap$-synonymy rule this amounts to showing, under the assumption that $\hat{\varphi}$ exists, that $\hat{\varphi}$ a fixpoint of F. Because of the synonymy of $\hat{\pi}$ and $\hat{\varphi}$, this is equivalent to showing that $\hat{\pi}$, if defined, is a fixpoint of F. We have already proved that $\hat{\pi}$ satisfies P, which, by the definition of P, means that $\hat{\pi} \sqsupseteq F\hat{\pi}$, so the only thing that remains to be shown is the validity of the inequation $F\hat{\pi} \sqsupseteq \hat{\pi}$:

$$
\begin{array}{ll}
 & F\hat{\pi} \sqsupseteq \hat{\pi} \\
\equiv & \qquad \{\text{definition of } \hat{\pi}\} \\
 & F\hat{\pi} \sqsupseteq \bigsqcap(\mathsf{id} \mathrel{\dot{\sqsupseteq}} F : \mathsf{id}) \\
\Leftarrow & \qquad \{\bigsqcap\text{-instantiation}\} \\
 & F\hat{\pi} \sqsupseteq F.F\hat{\pi} \\
\Leftarrow & \qquad \{F \text{ is monotonic}\} \\
 & \hat{\pi} \sqsupseteq F\hat{\pi} \\
\equiv & \qquad \{\text{just shown}\} \\
 & \text{true}
\end{array}
$$

End of proof.