$$(a \wedge b)\Big|_{((a\ (p \wedge q))(b\ \ a))}$$

$$\equiv ((p \wedge q) \wedge a)$$

$$\cancel{(p \wedge \neg p)\Big|_{((p\ t)(p\ nil))}}$$

Conjecture
Theorem
Axiom
Corollary
Lemma

Sat.

$$(a \wedge b)\Big|_{((a\ t)(b\ t))}$$

Falsifiable

$$(a \wedge b)\Big|_{((a\ nil)(b\ t))}$$

<u>Instantiation</u>

$$(p \oplus q) \wedge \neg(p \oplus q)$$

$$\equiv \{ p \wedge \neg p \equiv nil \Big|_{((p\ (p \oplus q)))}$$

$$nil$$

<u>Conjecture</u>

$$(p \oplus q) \wedge \neg(p \oplus q) \equiv nil$$

$$\boxed{(\text{listp } z) \Rightarrow (\text{equal } (\text{len } (\text{cons } x\ z)) \ (\text{len } (\text{cons } y\ z)))}$$

$$P \Rightarrow Q$$

Proof

C.I. (listp z)

Prove $\Rightarrow$

Assume P

<u>Prove</u> Q

Use $\Rightarrow$

Show P is true

conclude Q

Modus Ponens

```
(defunc alen (x)
  :input-contract t
  :output-contract (natp (alen x))
  (if (atom x)
      0
      (+ 1 (alen (rest x)))))  ]] or } or whatever
                               ≡ magic parens
```

conjecture 1:

```
  0                 1
(alen x) = (alen (list x))
             Counter: x = 4
```

Conjecture 2

```
(alen (cons x z)) = (alen (cons y z))
```

Lemma 1

```
(alen (cons x z)) ≡ (+ 1 (alen z))
```

Lemma 1

LHS

```
(alen (cons x z))
≡ {Def alen |((x (cons x z)))}
(if (atom (cons x z))
    0
    (+ 1 (alen (rest (cons x z)))))
≡ {Def atom |((x (cons x z)))}
(if (not (consp (cons x z)))
    0
    (+ 1 (alen (rest (cons x z)))))
≡ {Def not |((a (consp (cons x z))))}
```

(if (if (consp (cons x z)) nil t) 0 (+1 (alen (rest (cons x z)))))

$\equiv$ { if axiom, consp axiom }

(if nil 0 (+1 (alen (rest (cons x z)))))

$\equiv$ { if axiom }

(+1 (alen (rest (cons x z))))

$\equiv$ { first-rest axioms }

$\boxed{\text{(+ 1 (alen z))}}$ $\Longleftarrow$ RHS

$\boxed{\text{LHS = RHS Q.E.D.}}$

RHS (alen (cons y z))

$\equiv$ {

LHS of conjecture

(alen (cons x z))

$\equiv$ { Lemma 1 $|_{((x\ x))}$ }

(+1 (alen z))

RHS

(alen (cons y z))

$\equiv$ { Lemma 1 $|_{((x\ y))}$ }

(+ 1 (alen z))

LHS = RHS Q.E.D.

Reflexivity

$X \equiv X$

Symmetry of Equality

$(a=b) \Rightarrow (b=a)$

Transitivity of Equality

$(a=b) \wedge (b=c) \Rightarrow (a=c)$

<u>if axioms</u>

$(x=nil) \Rightarrow (\text{if } x\ y\ z) \equiv z$

$(x \neq nil) \Rightarrow (\text{if } x\ y\ z) \equiv y$

t

<u>consp axiom</u>

$(\text{consp } (\text{cons } x\ y)) \equiv t$

<u>first-rest axioms</u>

$(\text{first } (\text{cons } x\ y)) \equiv x$

$(\text{rest } (\text{cons } x\ y)) \equiv y$

```
(defunc f (x1 x2 ... xn)
  :input-contract (and (R1 x1) (R2 x2) ... (Rn xn))
  :output-contract (R (f x1 x2 ... xn))
  <body>)
```

$IC \Rightarrow OC$ contract theorem eg $(\text{and } (R_1\, x_1) \ldots (R_n\, x_n)) \Rightarrow (R\, (f\, x_1 \ldots x_n))$

$IC \Rightarrow ((f\ x_1\ x_2 \ldots x_n) = \langle body \rangle)$

Equality Axiom Schema for Functions

$x_1 = y_1 \wedge x_2 = y_2 \ldots x_n = y_n \Rightarrow$

$(f\ x_1 \ldots x_n) = (f\ y_1 \ldots y_n)$

$(\text{equal } l1\ l2) \equiv (l1 = l2) \Longleftarrow$ Shortcut!

Exportation

$(\text{listp } x) \wedge (\text{listp } y) \wedge (\text{listp } z) \cancel{\Rightarrow} \wedge$

$(\text{endp } x) \Rightarrow (\text{app } (\text{app } x\ y)\ z) = (\text{app } x\ \overbrace{(\text{app } y\ z)}^{\text{"y"}}))$

C1. $(\text{listp } x)$

C2. $(\text{listp } y)$

C3. $(\text{listp } z)$

C4 $(\text{endp } x)$

General Context

C5. $(x = \text{nil})$ {C1, C4, Def listp}

1. LHS = x
   RHS = x
   } LHS = RHS

2. LHS = RHS
   x = x

3. LHS = RHS.

(app (app x y) z)

≡ {Def of app, if axiom, C4}

(app $\boxed{y}$ z)

≡ {if axiom | ((x (endp x)) (y (app y z)) (z (cons (first x) (app (rest x) (app y z))))), C4}

(if (endp x)
    (app y z)
    (cons (first x) (app (rest x) (app y z))))

≡ {Def app | ((y (app y z)))}

(app x (app y z))

Theorem 4.1.

$$(\text{listp } y) \wedge (\text{listp } z) \Rightarrow (\text{app } (\text{cons } x\ y)\ z) = (\text{cons } x\ (\text{app } y\ z))$$

```
(defunc app (x y)
  :ic (and (listp x) (listp y))
  :oc (listp (app x y))
  (if (endp x)
      y
    (cons (first x) (app (rest x) y))))
```

Conjecture $\varphi_{assoc.app2}$

$$(\text{consp } x) \wedge \left[(\text{app } (\text{app } (\text{rest } x)\ y)\ z) = (\text{app } (\text{rest } x)\ (\text{app } y\ z)) \right]$$

Jerk move

$$\Rightarrow (\text{app } (\text{app } x\ y)\ z) = (\text{app } x\ (\text{app } y\ z))$$

$$(\text{listp } x) \wedge (\text{listp } y) \wedge (\text{listp } z) \wedge (\text{consp } x)$$
$$\wedge \underbrace{\left[(\text{listp } (\text{rest } x)) \wedge (\text{listp } y) \wedge (\text{listp } z) \Rightarrow (\text{app } (\text{app } (\text{rest } x)\ y)\ z) = (\text{app } (\text{rest } x)\ (\text{app } y\ z))\right]}_{\alpha}$$
$$\Rightarrow ((\text{app } (\text{app } x\ y)\ z) = (\text{app } x\ (\text{app } y\ z)))$$

$$A \Rightarrow ((B \Rightarrow E) \Rightarrow (C \Rightarrow D))$$
$$\equiv A \Rightarrow ((B \Rightarrow E) \wedge C \Rightarrow D)$$
$$\equiv A \wedge (B \Rightarrow E) \wedge C \Rightarrow D$$

C1. (listp x)
C2. (listp y)
C3. (listp z)
C4. (consp x)
C5 α

C6 (listp (rest x)) {C1, C4, Def listp}

C7. (app (app (rest x) y) z)
= (app (rest x) (app y z))
{C6, C2, C3, C5, MP}

Modus Ponens

$$[(A \Rightarrow B) \wedge A] \Rightarrow B$$

C8. (not (endp x))
{C4, C1, def endp}

(app (app x y) z)

≡ {Def app, if ~~axiom~~, C8}

Lazy

(app (cons (first x) (app (rest x) y)) z)

≡ {Theorem 4.1 | ((y (app (rest x) y))) }

(cons (first x) (app (app (rest x) y) z))

C7!

≡ {C7}

(cons (first x) (app (rest x) (app y z)))

"y"

≡ {Def app, if ~~axiom~~, C8}

(app x (app y z))

QED.

Assoc of App

(listp x) ∧ (listp y) ∧ (listp z)

⇒ [(app (app x y) z)

= (app x (app y z))]

$\wedge$ (endp x) $\wedge$ (consp x) $\wedge$ [——] $\Rightarrow$ [(app (app x y) z) = (app x (app y z))]

(listp x) $\wedge$ (listp y) $\wedge$ (listp z)

C1. (listp x)
C2. (listp y)
C3. (listp z)
C4. (endp x)
C5. (consp x)
C6. [——]

- - - - - - -

C7. (not (consp x)) {Def endp, C4}
C8. nil {C7, C5, PL}

nil $\Rightarrow$ X $\equiv$ true

QED

ϕ_{in_app}:

$$(not\ (in\ a\ (app\ x\ y)) \Rightarrow (not\ (in\ a\ x))$$

$$\Downarrow$$

$$(listp\ x) \wedge (listp\ y) \cancel{\Rightarrow} \wedge (in\ a\ x) \Rightarrow (in\ a\ (app\ x\ y))$$

```
(defun in (e x)
  :ic (listp x)
  :oc (booleanp (in e x))
  (if (endp x)
      nil
      (or (equal e (first x))
          (in e (rest x)))))
```

$$\phi_{in\_app2}: [((endp\ x) \vee (\neg (endp\ x) \wedge (a = (first\ x)))$$
$$\vee (\neg (endp\ x) \wedge (a \neq (first\ x)) \wedge \phi_{in\_app}|_{((x\ (rest\ x)))}]$$

$$\wedge (listp\ x) \wedge (listp\ y) \wedge (in\ a\ x) \Rightarrow (in\ a\ (app\ x\ y))$$

Inductive Assumption

Case 1: endp x

(endp X) ∧ (listp X) ∧ (listp y) ∧ (in a X) ⇒ (in a (app X y))

C1. (endp X)

C2. (listp X)

C3. (listp y)

C4. (in a X)

- - - - - - - - - -

C5. (not (in a X)) {C1, Def in , if axiom}

C6. nil {C5, C4, PL}

nil ⇒ X ≡ true

¬(endp X) ∧ (listp X) ∧ (listp y) ∧ (a = (first X))

∧ (in a X) ⇒ (in a (app X y))

C1. (listp X)

C2. (listp y)

C3. ¬(endp X)

C4. (a = (first X))

C5. (in a X)

(in a (app x y))

≡ {Def of app, C3}

"x"

(in a (cons (first x) (app (rest x) y)))

≡ {Def in, Def endp, cons axiom, if axiom}

(or (equal a (first (cons (first x) (app (rest x) y))))
    (in a (rest (cons (first x) (app (rest x) y))))).

≡ {first-rest axioms}

(or (equal a (first x)) (in a (app (rest x) y)))

≡ {C4, PL}

true

Case 3:

$$(\text{listp } x) \wedge (\text{listp } y) \wedge \neg(\text{endp } x) \wedge (a \neq (\text{first } x)) \wedge \underbrace{[(\text{listp}(\text{rest } x)) \wedge (\text{listp } y) \wedge (\text{in } a (\text{rest } x)) \Rightarrow (\text{in } a (\text{app}(\text{rest } x)\ y))]}_{\alpha} \wedge (\text{in } a\ x)$$

$$\Rightarrow (\text{in } a (\text{app } x\ y))$$

$$((A \Rightarrow B) \wedge A) \Rightarrow B$$

C1. $(\text{listp } x)$

C2. $(\text{listp } y)$

C3. $\neg(\text{endp } x)$

C4. $(a \neq (\text{first } x))$

C5. α

C6. $(\text{in } a\ x)$

- - - - - - - - - -

C7. $(\text{listp}(\text{rest } x))$ {C1, C3, Def listp, Def endp}

C8. $(\text{in } a (\text{rest } x))$ {C6, Def in, C3, PL, C4}

C9. $(\text{in } a (\text{app}(\text{rest } x)\ y))$ {C5, C8, C7, C2, MP}