Monads in ML

A monad in ML is given by type $\tau(\alpha)$ with a free type variable α together with expressions

$$\begin{aligned} \mathit{unit} &: \alpha \to \tau(\alpha) \\ \mathit{lift} &: (\alpha \to \tau(\beta)) \to \tau(\alpha) \to \tau(\beta) \end{aligned}$$

(writing $\tau(\beta)$ for the result of replacing α by β in τ) satisfying some equational properties [omitted].

PLC versus the Pure Type System $\boldsymbol{\lambda 2}$

PTS signature:

$$\mathbf{2} \triangleq (\mathcal{S}_2, \mathcal{A}_2, \mathcal{R}_2) \text{ where } \begin{cases} \mathcal{S}_2 & \triangleq & \{*, \square\} \\ \mathcal{A}_2 & \triangleq & \{(*, \square)\} \\ \mathcal{R}_2 & \triangleq & \{(*, *, *), (\square, *, *)\} \end{cases}$$

Translation of PLC types and terms to $\boldsymbol{\lambda 2}$ pseudo-terms:

$$\begin{aligned}
[\![\alpha]\!] &= \alpha \\
[\![\tau \to \tau']\!] &= \Pi x : [\![\tau]\!] \, ([\![\tau']\!]) \quad \text{(any } x \text{ not free in } \tau'\text{)} \\
[\![\forall \alpha \, (\tau)]\!] &= \Pi \alpha : * \, ([\![\tau']\!]) \\
[\![x]\!] &= x \\
[\![\lambda x : \tau \, (M)]\!] &= \lambda x : [\![\tau]\!] \, ([\![M]\!]) \\
[\![M \, M']\!] &= [\![M]\!] \, [\![M']\!] \\
[\![\Lambda \alpha \, (M)]\!] &= \lambda \alpha : * \, ([\![M]\!]) \\
[\![M \, \tau]\!] &= [\![M]\!] \, [\![\tau]\!]
\end{aligned}$$

System $\mathbf{F}_\omega$ as a Pure Type System: $\boldsymbol{\lambda\omega}$

PTS specification $\omega = (\mathcal{S}_\omega, \mathcal{A}_\omega, \mathcal{R}_\omega)$:

$$\begin{aligned}
\mathcal{S}_\omega &\triangleq \{*, \square\} \\
\mathcal{A} &\triangleq \{(*, \square)\} \\
\mathcal{R} &\triangleq \{(*, *, *), (\square, *, *), (\square, \square, \square)\}
\end{aligned}$$

"F_ω is the work horse of modern compilers"

(Greg Morrisett)

System $\mathbf{F}_\omega$ as a Pure Type System: $\lambda\omega$

PTS specification $\omega = (\mathcal{S}_\omega, \mathcal{A}_\omega, \mathcal{R}_\omega)$:

$$\mathcal{S}_\omega \triangleq \{*, \square\}$$
$$\mathcal{A} \triangleq \{(*, \square)\}$$
$$\mathcal{R} \triangleq \{(*, *, *), (\square, *, *), (\square, \square, \square)\}$$

As in $\lambda 2$, sort $*$ is a universe of types; but in $\lambda\omega$, the rule (**prod**) for $(\square, \square, \square)$ means that $\diamond \vdash t : \square$ holds for all the 'simple types' over the ground type $*$ – the ts generated by the grammar $t ::= * \mid t \to t$

$$(\text{prod}) \; \frac{\Gamma \vdash A : \square \qquad \Gamma, x : A \vdash B : \square}{\Gamma \vdash \Pi x : A \, (B) : \square} \quad \text{for } (\square, \square, \square)$$

System $\mathbf{F}_\omega$ as a Pure Type System: $\lambda\omega$

PTS specification $\omega = (\mathcal{S}_\omega, \mathcal{A}_\omega, \mathcal{R}_\omega)$:

$$
\begin{aligned}
\mathcal{S}_\omega &\triangleq \{*, \Box\} \\
\mathcal{A} &\triangleq \{(*, \Box)\} \\
\mathcal{R} &\triangleq \{(*, *, *), (\Box, *, *), (\Box, \Box, \Box)\}
\end{aligned}
$$

As in $\lambda 2$, sort $*$ is a universe of types; but in $\lambda\omega$, the rule (**prod**) for $(\Box, \Box, \Box)$ means that $\diamond \vdash t : \Box$ holds for all the 'simple types' over the ground type $*$ – the ts generated by the grammar $t ::= * \mid t \to t$

$$
(\text{prod}) \quad \frac{\Gamma \vdash A : \Box \qquad \Gamma, x : A \vdash B : \Box}{\Gamma \vdash \Pi x : A\,(B) : \Box} \quad \text{for } (\Box, \Box, \Box)
$$

$$
(A \to B \triangleq \Pi x : A\,(B) \text{ with } x \notin fv(B))
$$

System $\mathbf{F}_\omega$ as a Pure Type System: $\lambda\omega$

PTS specification $\omega = (\mathcal{S}_\omega, \mathcal{A}_\omega, \mathcal{R}_\omega)$:

$$
\begin{aligned}
\mathcal{S}_\omega &\triangleq \{*, \Box\} \\
\mathcal{A} &\triangleq \{(*, \Box)\} \\
\mathcal{R} &\triangleq \{(*,*,*), (\Box,*,*), (\Box,\Box,\Box)\}
\end{aligned}
$$

As in $\boldsymbol{\lambda 2}$, sort $*$ is a universe of types; but in $\boldsymbol{\lambda\omega}$, the rule (**prod**) for $(\Box,\Box,\Box)$ means that $\diamond \vdash t : \Box$ holds for all the 'simple types' over the ground type $*$ – the ts generated by the grammar $t ::= * \mid t \to t$

Hence rule (**prod**) for $(\Box,*,*)$ now gives many more legal pseudo-terms of type $*$ in $\boldsymbol{\lambda\omega}$ compared with $\boldsymbol{\lambda 2}$ (PLC), such as

$$
\left.
\begin{aligned}
&\diamond \vdash (\Pi T : * \to * \,(\Pi \alpha : * \,(\alpha \to T\,\alpha))) : * \\
&\diamond \vdash (\Pi T : * \to * \,(\Pi \alpha, \beta : * \,((\alpha \to T\,\beta) \to T\,\alpha \to T\,\beta))) : *
\end{aligned}
\right\}
$$

types for unit & lift operations, making T a <u>monad</u>

Examples of $\lambda\omega$ type constructions

- Monad transformer for state (using a type $\diamond \vdash S : *$ for states):

$$\mathtt{M} \triangleq \lambda T : * \to * \,(\lambda \alpha : * \,(S \to T(\mathtt{P}\,\alpha\, S)))$$

$$\diamond \vdash \mathtt{M} : (* \to *) \to * \to *$$

Examples of $\lambda\omega$ type constructions

- Product types (cf. the PLC representation of product types):

$$\mathtt{P} \triangleq \lambda\alpha,\beta : * \, (\Pi\gamma : * \, ((\alpha \to \beta \to \gamma) \to \gamma))$$

$$\diamond \vdash \mathtt{P} : * \to * \to *$$

- Monad transformer for state (using a type $\diamond \vdash S : *$ for states):

$$\mathtt{M} \triangleq \lambda T : * \to * \, (\lambda\alpha : * \, (S \to T(\mathtt{P}\,\alpha\,S)))$$

$$\diamond \vdash \mathtt{M} : (* \to *) \to * \to *$$

Examples of $\lambda\omega$ type constructions

- Product types (cf. the PLC representation of product types):

$$\mathtt{P} \triangleq \lambda\alpha, \beta : * \, (\Pi\gamma : * \, ((\alpha \to \beta \to \gamma) \to \gamma))$$

$$\diamond \vdash \mathtt{P} : * \to * \to *$$

$$\tau \times \tau' \triangleq \forall\gamma \, ((\tau \to \tau' \to \gamma) \to \gamma)$$

where $\gamma \notin ftv(\tau, \tau')$

(one definition per each choice of types τ & τ')

Examples of $\lambda\omega$ type constructions

- Product types (cf. the PLC representation of product types):

$$\mathtt{P} \triangleq \lambda\alpha,\beta : *\,(\Pi\gamma : *\,((\alpha \to \beta \to \gamma) \to \gamma))$$

$$\diamond \vdash \mathtt{P} : * \to * \to *$$

- Mon
  state

$$\exists\alpha(\tau) \triangleq \forall\beta\left((\forall\alpha(\tau \to \beta)) \to \beta \right)$$

where $\beta \notin ftv(\tau)$

- Existential types (cf. the PLC representation of existential types):

$$\exists \triangleq \lambda T : * \to *\,(\Pi\beta : *\,((\Pi\alpha : *\,(T\,\alpha \to \beta)) \to \beta))$$

$$\diamond \vdash \exists : (* \to *) \to *$$

Type-checking for $\mathbf{F}_\omega$ ($\lambda\omega$)

($\lambda\omega$)

Fact (Girard): System $\mathbf{F}_\omega$ is *strongly normalizing* in the sense that for any legal pseudo-term t, there is no infinite chain of beta-reductions $t \longrightarrow t_1 \longrightarrow t_2 \longrightarrow \cdots$.

Type-checking for $\mathbf{F}_\omega$ $(\lambda\omega)$

$(\lambda\omega)$

Fact (Girard): System $\mathbf{F}_\omega$ is *strongly normalizing* in the sense that for any legal pseudo-term t, there is no infinite chain of beta-reductions $t \to t_1 \to t_2 \to \cdots$.

As as corollary we have that the type-checking and typeability problems for $\mathbf{F}_\omega$ are decidable.

$(\lambda\omega)$

Propositions as Types

(sect. 6 of notes)

Curry-Howard correspondence

<u>Logic</u>	$\leftrightarrow$	<u>Type system</u>
propositions ϕ	$\leftrightarrow$	types τ
proofs p	$\leftrightarrow$	expressions M
'p is a proof of ϕ'	$\leftrightarrow$	'M is an expression of type τ'
simplification of proofs	$\leftrightarrow$	reduction of expressions

First arose for <u>constructive</u> logics

Constructive interpretation of logic

- **Implication:** a proof of $\varphi \to \psi$ is a construction that transforms proofs of φ into proofs of ψ.
- **Negation:** a proof of $\neg\varphi$ is a construction that from any (hypothetical) proof of φ produces a contradiction ($=$ proof of falsity $\perp$)
- **Disjunction:** a proof of $\varphi \vee \psi$ is an object that manifestly is either a proof of φ, or a proof of ψ.
- **For all:** a proof of $\forall x\,(\varphi(x))$ is a construction that transforms the objects a over which x ranges into proofs of $\varphi(a)$.
- **There exists:** a proof of $\exists\, x\,(\varphi(x))$ is given by a pair consisting of an object a and a proof of $\varphi(a)$.

Constructive interpretation of logic

- **Implication:** a proof of $\varphi \to \psi$ is a construction that transforms proofs of φ into proofs of ψ.
- **Negation:** a proof of $\neg\varphi$ is a construction that from any (hypothetical) proof of φ produces a contradiction (= proof of falsity $\perp$)
- **Disjunction:** a proof of $\varphi \vee \psi$ is an object that manifestly is either a proof of φ, or a proof of ψ.
- **For all:** a proof of $\forall x\,(\varphi(x))$ is a construction that transforms the objects a over which x ranges into proofs of $\varphi(a)$.
- **There exists:** a proof of $\exists\, x\,(\varphi(x))$ is given by a pair consisting of an object a and a proof of $\varphi(a)$.

The *Law of Excluded Middle* (LEM) $\boxed{\forall p\,(p \vee \neg p)}$ is a classical tautology (has truth-value `true`), but is rejected by constructivists.

Example of a non-constructive proof

Theorem. There exist two irrational numbers a and b such that b^a is rational.

Example of a non-constructive proof

Theorem. There exist two irrational numbers a and b such that b^a is rational.

Proof. Either $\sqrt{2}^{\sqrt{2}}$ is rational, or it is not (LEM!).

Example of a non-constructive proof

Theorem. There exist two irrational numbers a and b such that b^a is rational.

Proof. Either $\sqrt{2}^{\sqrt{2}}$ is rational, or it is not (LEM!).

If it is, we can take $a = b = \sqrt{2}$, since $\sqrt{2}$ is irrational by a well-known theorem attributed to Euclid.

Example of a non-constructive proof

Theorem. There exist two irrational numbers a and b such that b^a is rational.

Proof. Either $\sqrt{2}^{\sqrt{2}}$ is rational, or it is not (LEM!).

If it is, we can take $a = b = \sqrt{2}$, since $\sqrt{2}$ is irrational by a well-known theorem attributed to Euclid.

If it is not, we can take $a = \sqrt{2}$ and $b = \sqrt{2}^{\sqrt{2}}$, since then $b^a = (\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = \sqrt{2}^{\sqrt{2}.\sqrt{2}} = \sqrt{2}^2 = 2$.

QED

Example of a constructive proof

Theorem. There exist two irrational numbers a and b such that b^a is rational.

Proof. $\sqrt{2}$ is irrational by a well-known constructive proof attributed to Euclid.

$2\log_2 3$ is irrational, by an easy constructive proof (exercise).

(If $2\log_2 3 = m/n$, then

$$3^{2n} = 2^{2n\log_2 3} = 2^m \quad \times\!\!\times)$$

Example of a constructive proof

Theorem. There exist two irrational numbers a and b such that b^a is rational.

Proof. $\sqrt{2}$ is irrational by a well-known constructive proof attributed to Euclid.

$2\log_2 3$ is irrational, by an easy constructive proof (exercise).

So we can take $a = 2\log_2 3$ and $b = \sqrt{2}$, for which we have that $b^a = (\sqrt{2})^{2\log_2 3} = (\sqrt{2}^2)^{\log_2 3} = 2^{\log_2 3} = 3$ is rational.

QED

Curry-Howard correspondence

? $\longleftrightarrow$ PLC

Logic	$\leftrightarrow$	Type system
propositions ϕ	$\leftrightarrow$	types τ
proofs p	$\leftrightarrow$	expressions M
'p is a proof of ϕ'	$\leftrightarrow$	'M is an expression of type τ'
simplification of proofs	$\leftrightarrow$	reduction of expressions

E.g.

2IPC $\leftrightarrow$ PLC

Second-order intuitionistic propositional calculus (2IPC)

2IPC propositions: $\boxed{\phi ::= p \mid \phi \to \phi \mid \forall p\,(\phi)}$ where p ranges over an infinite set of propositional variables.

2IPC sequents: $\boxed{\Phi \vdash \phi}$ where Φ is a finite multiset (= unordered list) of 2IPC propositions and ϕ is a 2IPC proposition.

Second-order intuitionistic propositional calculus (2IPC)

2IPC propositions: $\boxed{\phi ::= p \mid \phi \to \phi \mid \forall p\,(\phi)}$ where p ranges over an infinite set of propositional variables.

2IPC sequents: $\boxed{\Phi \vdash \phi}$ where Φ is a finite multiset (= unordered list) of 2IPC propositions and ϕ is a 2IPC proposition.

$\Phi \vdash \phi$ is *provable* if it is in the set of sequents inductively generated by:

$$(\mathbf{Id})\ \Phi \vdash \phi \quad \text{if } \phi \in \Phi$$

$$(\to\mathbf{I})\ \frac{\Phi, \phi \vdash \phi'}{\Phi \vdash \phi \to \phi'} \qquad (\to\mathbf{E})\ \frac{\Phi \vdash \phi \to \phi' \qquad \Phi \vdash \phi}{\Gamma \vdash \phi'}$$

$$(\forall\mathbf{I})\ \frac{\Phi \vdash \phi}{\Phi \vdash \forall p\,(\phi)} \ \text{if } p \notin fv(\Phi) \qquad (\forall\mathbf{E})\ \frac{\Phi \vdash \forall p\,(\phi)}{\Phi \vdash \phi[\phi'/p]}$$

Logical operations definable in 2IPC

- *Truth* $\top \triangleq \forall p\,(p \to p)$
- *Falsity* $\bot \triangleq \forall p\,(p)$

Logical operations definable in 2IPC

- *Truth* $\top \triangleq \forall p\,(p \to p)$
- *Falsity* $\bot \triangleq \forall p\,(p)$
- *Conjunction* $\phi \wedge \psi \triangleq \forall p\,((\phi \to \psi \to p) \to p)$
  (where $p \notin fv(\phi, \psi)$)

Logical operations definable in 2IPC

- *Truth* $\top \triangleq \forall p\,(p \to p)$
- *Falsity* $\bot \triangleq \forall p\,(p)$
- *Conjunction* $\phi \land \psi \triangleq \forall p\,((\phi \to \psi \to p) \to p)$ (where $p \notin fv(\phi, \psi)$)
- *Disjunction* $\phi \lor \psi \triangleq \forall p\,((\phi \to p) \to (\psi \to p) \to p)$ (where $p \notin fv(\phi, \psi)$)

Logical operations definable in 2IPC

- *Truth* $\top \triangleq \forall p\,(p \to p)$
- *Falsity* $\bot \triangleq \forall p\,(p)$
- *Conjunction* $\phi \wedge \psi \triangleq \forall p\,((\phi \to \psi \to p) \to p)$ (where $p \notin fv(\phi, \psi)$)
- *Disjunction* $\phi \vee \psi \triangleq \forall p\,((\phi \to p) \to (\psi \to p) \to p)$ (where $p \notin fv(\phi, \psi)$)
- *Negation* $\neg\phi \triangleq \phi \to \bot$
- *Bi-implication* $\phi \leftrightarrow \psi \triangleq (\phi \to \psi) \wedge (\psi \to \phi)$

Logical operations definable in 2IPC

- *Truth* $\top \triangleq \forall p\,(p \to p)$
- *Falsity* $\bot \triangleq \forall p\,(p)$
- *Conjunction* $\phi \wedge \psi \triangleq \forall p\,((\phi \to \psi \to p) \to p)$ (where $p \notin fv(\phi, \psi)$)
- *Disjunction* $\phi \vee \psi \triangleq \forall p\,((\phi \to p) \to (\psi \to p) \to p)$ (where $p \notin fv(\phi, \psi)$)
- *Negation* $\neg\phi \triangleq \phi \to \bot$
- *Bi-implication* $\phi \leftrightarrow \psi \triangleq (\phi \to \psi) \wedge (\psi \to \phi)$
- *Existential quantification* $\exists p\,(\phi) \triangleq \forall q\,(\forall p\,(\phi \to q) \to q)$ (where $q \notin fv(\phi, p)$)

A 2IPC proof

Writing $p \wedge q$ as an abbreviation for $\forall r\,((p \to q \to r) \to r)$, the sequent

$$\{\} \vdash \forall p\,(\forall q\,((p \wedge q) \to p))$$

is provable in 2IPC:

A 2IPC proof

Writing $p \wedge q$ as an abbreviation for $\forall r\,((p \to q \to r) \to r)$, the sequent

$$\{\} \vdash \forall p\,(\forall q\,((p \wedge q) \to p))$$

is provable in 2IPC:

$$
\begin{prooftree}
\AxiomC{}
\LeftLabel{(Id)}
\UnaryInfC{$\{p \wedge q, p, q\} \vdash p$}
\LeftLabel{($\to$I)}
\UnaryInfC{$\{p \wedge q, p\} \vdash q \to p$}
\LeftLabel{($\to$I)}
\UnaryInfC{$\{p \wedge q\} \vdash p \to q \to p$}
\AxiomC{}
\LeftLabel{(Id)}
\UnaryInfC{$\{p \wedge q\} \vdash \forall r\,((p \to q \to r) \to r)$}
\LeftLabel{($\forall$E)}
\UnaryInfC{$\{p \wedge q\} \vdash (p \to q \to q) \to q$}
\LeftLabel{($\to$E)}
\BinaryInfC{$\{p \wedge q\} \vdash p$}
\LeftLabel{($\to$I)}
\UnaryInfC{$\{\} \vdash (p \wedge q) \to p$}
\LeftLabel{($\forall$I)}
\UnaryInfC{$\{\} \vdash \forall q\,((p \wedge q) \to p)$}
\LeftLabel{($\forall$I)}
\UnaryInfC{$\{\} \vdash \forall p\,(\forall q\,((p \wedge q) \to p))$}
\end{prooftree}
$$

p p

TYPO!

Curry-Howard correspondence

2IPC		PLC
Logic	$\leftrightarrow$	Type system

Curry-Howard correspondence

2IPC		PLC
Logic	$\leftrightarrow$	Type system
propositions ϕ	$\leftrightarrow$	types τ

Curry-Howard correspondence

2IPC PLC

Logic	$\leftrightarrow$	Type system
propositions ϕ	$\leftrightarrow$	types τ
proofs p	$\leftrightarrow$	expressions M
'p is a proof of ϕ'	$\leftrightarrow$	'M is an expression of type τ'

Mapping 2IPC proofs to PLC expressions

$$(\mathbf{Id})\ \Phi, \phi \vdash \phi \quad \mapsto \quad (\mathbf{id})\ \overline{x} : \Phi, x : \phi \vdash x : \phi$$

$$(\to\mathbf{I})\ \frac{\Phi, \phi \vdash \phi'}{\Phi \vdash \phi \to \phi'} \quad \mapsto \quad (\mathbf{fn})\ \frac{\overline{x} : \Phi, x : \phi \vdash M : \phi'}{\overline{x} : \Phi \vdash \lambda x : \phi\,(M) : \phi \to \phi'}$$

$$(\to\mathbf{E})\ \frac{\Phi \vdash \phi \to \phi' \qquad \Phi \vdash \phi}{\Phi \vdash \phi'} \quad \mapsto \quad (\mathbf{app})\ \frac{\overline{x} : \Phi \vdash M_1 : \phi \to \phi' \qquad \overline{x} : \Phi \vdash M_2 : \phi}{\overline{x} : \Phi \vdash M_1\, M_2 : \phi'}$$

$$(\forall\mathbf{I})\ \frac{\Phi \vdash \phi}{\Phi \vdash \forall p\,(\phi)} \quad \mapsto \quad (\mathbf{gen})\ \frac{\overline{x} : \Phi \vdash M : \phi}{\overline{x} : \Phi \vdash \Lambda p\,(M) : \forall p\,(\phi)}$$

$$(\forall\mathbf{E})\ \frac{\Phi \vdash \forall p\,(\phi)}{\Phi \vdash \phi[\phi'/p]} \quad \mapsto \quad (\mathbf{spec})\ \frac{\overline{x} : \Phi \vdash M : \forall p\,(\phi)}{\overline{x} : \Phi \vdash M\,\phi' : \phi[\phi'/p]}$$

The proof of the 2IPC sequent

$$\{\} \vdash \forall p\,(\forall q\,((p \wedge q) \to p))$$

given before is transformed by the mapping of 2IPC proofs to PLC expressions to

$$\{\} \vdash \Lambda p, q\,(\lambda z : p \wedge q\,(z\,p\,(\lambda x : p, y : q\,(x)))) \\ : \forall p\,(\forall q\,((p \wedge q) \to p))$$

with typing derivation:

$$
\text{(gen)}\ \dfrac{\text{(gen)}\ \dfrac{\text{(fn)}\ \dfrac{\text{(app)}\ \dfrac{\text{(fn)}\ \dfrac{\text{(fn)}\ \dfrac{\text{(id)}\ \dfrac{}{\{z : p \wedge q, x : p, y : q\} \vdash x : p}}{\{z : p \wedge q, x : p\} \vdash \lambda y : q\,(x) : q \to p}}{\{z : p \wedge q\} \vdash \lambda x : p, y : q\,(x) : p \to q \to p} \qquad \text{(spec)}\ \dfrac{\text{(id)}\ \dfrac{}{\{z : p \wedge q\} \vdash z : \forall r\,((p \to q \to r) \to r)}}{\{z : p \wedge q\} \vdash z\,p : (p \to q \to p) \to p}}{\{z : p \wedge q\} \vdash z\,p\,(\lambda x : p, y : q\,(x)) : p}}{\{\} \vdash \lambda z : p \wedge q\,(z\,p\,(\lambda x : p, y : q\,(x))) : (p \wedge q) \to p}}{\{\} \vdash \Lambda q\,(\lambda z : p \wedge q\,(z\,p\,(\lambda x : p, y : q\,(x)))) : \forall q\,((p \wedge q) \to p)}}{\{\} \vdash \Lambda p, q\,(\lambda z : p \wedge q\,(z\,p\,(\lambda x : p, y : q\,(x)))) : \forall p, q\,((p \wedge q) \to p)}
$$

Curry-Howard correspondence

Logic	$\leftrightarrow$	Type system
propositions ϕ	$\leftrightarrow$	types τ
proofs p	$\leftrightarrow$	expressions M
'p is a proof of ϕ'	$\leftrightarrow$	'M is an expression of type τ'
simplification of proofs	$\leftrightarrow$	reduction of expressions

Proof simplification $\leftrightarrow$ Expression reduction

$$
\dfrac{\dfrac{\dfrac{\vdots}{\Phi, \phi \vdash \psi}}{\Phi \vdash \phi \to \psi}\,(\to\mathbf{I}) \qquad \dfrac{\vdots}{\Phi \vdash \phi}}{\Phi \vdash \psi}\,(\to\mathbf{E})
\quad \mapsto \quad
\dfrac{\dfrac{\dfrac{\vdots}{\overline{x} : \Phi, x : \phi \vdash M : \psi}}{\overline{x} : \Phi \vdash \lambda x : \phi\,(M) : \phi \to \psi} \qquad \dfrac{\vdots}{\overline{x} : \Phi \vdash N : \phi}}{\overline{x} : \Phi \vdash (\lambda x : \phi\,(M))\,N : \psi}
$$

Proof simplification $\leftrightarrow$ Expression reduction

$$
\dfrac{\dfrac{\dfrac{\vdots}{\Phi, \phi \vdash \psi}}{\Phi \vdash \phi \to \psi}\,(\to\mathrm{I}) \qquad \dfrac{\vdots}{\Phi \vdash \phi}}{\Phi \vdash \psi}\,(\to\mathrm{E})
\quad\mapsto\quad
\dfrac{\dfrac{\dfrac{\vdots}{\overline{x} : \Phi, x : \phi \vdash M : \psi}}{\overline{x} : \Phi \vdash \lambda x : \phi\,(M) : \phi \to \psi} \qquad \dfrac{\vdots}{\overline{x} : \Phi \vdash N : \phi}}{\overline{x} : \Phi \vdash (\lambda x : \phi\,(M))\,N : \psi}
$$

$$
\Big\downarrow \ \text{beta-reduce expression}
$$

$$
\dfrac{\dfrac{\vdots}{\overline{x} : \Phi, x : \phi \vdash M : \psi} \qquad \dfrac{\vdots}{\overline{x} : \Phi \vdash N : \phi}}{\overline{x} : \Phi \vdash M[N/x] : \psi}\,(\mathbf{subst})
$$

The rule $(\mathbf{subst})$ for PLC is *admissible*: if its hypotheses are valid PLC typing judgements, then so is its conclusion.

Proof simplification $\leftrightarrow$ Expression reduction

$$
(\to\mathbf{E})\;\dfrac{(\to\mathbf{I})\;\dfrac{\dfrac{\vdots}{\Phi, \phi \vdash \psi}}{\Phi \vdash \phi \to \psi} \qquad \dfrac{\vdots}{\Phi \vdash \phi}}{\Phi \vdash \psi}
\quad\mapsto\quad
\dfrac{\dfrac{\dfrac{\vdots}{\overline{x} : \Phi, x : \phi \vdash M : \psi}}{\overline{x} : \Phi \vdash \lambda x : \phi\,(M) : \phi \to \psi} \qquad \dfrac{\vdots}{\overline{x} : \Phi \vdash N : \phi}}{\overline{x} : \Phi \vdash (\lambda x : \phi\,(M))\,N : \psi}
$$

$$\Big\downarrow \text{ beta-reduce expression}$$

$$
(\mathbf{cut})\;\dfrac{\dfrac{\vdots}{\Phi, \phi \vdash \psi} \qquad \dfrac{\vdots}{\Phi \vdash \phi}}{\Phi \vdash \psi}
\quad\longleftarrow\!\!\!\!\!\!\mid\quad
\dfrac{\dfrac{\vdots}{\overline{x} : \Phi, x : \phi \vdash M : \psi} \qquad \dfrac{\vdots}{\overline{x} : \Phi \vdash N : \phi}}{\overline{x} : \Phi \vdash M[N/x] : \psi}\;(\mathbf{subst})
$$

The rule $(\mathbf{subst})$ for PLC is *admissible*: if its hypotheses are valid PLC typing judgements, then so is its conclusion.

Proof simplification $\leftrightarrow$ Expression reduction

$$\begin{array}{ccc}
(\to\mathbf{I})\atop(\to\mathbf{E})\;\dfrac{\dfrac{\dfrac{\vdots}{\Phi,\phi\vdash\psi}}{\Phi\vdash\phi\to\psi}\qquad\dfrac{\vdots}{\Phi\vdash\phi}}{\Phi\vdash\psi} & \mapsto & \dfrac{\dfrac{\dfrac{\vdots}{\overline{x}:\Phi,x:\phi\vdash M:\psi}}{\overline{x}:\Phi\vdash\lambda x:\phi\,(M):\phi\to\psi}\qquad\dfrac{\vdots}{\overline{x}:\Phi\vdash N:\phi}}{\overline{x}:\Phi\vdash(\lambda x:\phi\,(M))\,N:\psi} \\
\text{simplify proof}\Big\downarrow & & \Big\downarrow\text{beta-reduce expression} \\
(\mathbf{cut})\;\dfrac{\dfrac{\vdots}{\Phi,\phi\vdash\psi}\qquad\dfrac{\vdots}{\Phi\vdash\phi}}{\Phi\vdash\psi} & \longleftarrow\!\shortmid & \dfrac{\dfrac{\vdots}{\overline{x}:\Phi,x:\phi\vdash M:\psi}\qquad\dfrac{\vdots}{\overline{x}:\Phi\vdash N:\phi}}{\overline{x}:\Phi\vdash M[N/x]:\psi}\;(\mathbf{subst})
\end{array}$$

The rule $(\mathbf{subst})$ for PLC is *admissible*: if its hypotheses are valid PLC typing judgements, then so is its conclusion.

Hence, the rule $(\mathbf{cut})$ is admissible for 2IPC.