Minimal predicate logic in λP

Encode "⇒" and "∀", predicates, and sets

PAT: ·) If a term b inhabits a type B (i.e. $b : B$) where B is interpreted as a proposition, then we interpret b as a proof of B. b is called a "proof object"

·) If no inhabitants of a proposition B exists, then there is no proof of B, so B must be false

Summary: | Proposition B is inhabitated iff B is true
Proposition B is not inhabitated iff B is false |

How to encode basic entities of minimal predicate logic in λP

I Sets

Code sets as types, so $S : *$

Elements of sets are terms, so a is an element of S if $a : S$

Examples: $nat : *$, $nat \to nat : *$, $3 : nat$, $\lambda n : nat . n : nat \to nat$

II Propositions

Propositions are also coded as types, so $A : *$

According to PAT, a term p with $p : A$ codes a proof of A (in which case A is true)

If no proof of A exists, then A is considered false

III Predicates

A predicate P is a function from a set to the set of all propositions, so $P : S \to *$ is a predicate on the set S

For each $a : S$ we then have $Pa : *$

All of these Pa are propositions which are types, so Pa may be inhabitated:

(1) If Pa is inhabitated, so $A : Pa$ for some A, then P holds for a

(2) If Pa is not inhabitated, then P does not hold for a

IV Implication

Identify $A \Rightarrow B$ with $A \to B$ (short for $\Pi x : A . B$ if x does not occur free in B)

Justification: ·) $A \Rightarrow B$ is true

·) if A is true, then also B is true

·) if A is inhabitated, then also B is inhabitated

·) there is a function mapping inhabitants of A to inhabitants of B

·) there is an f with $f : A \to B$

·) $A \to B$ is inhabitated

Thus, the truth of $A \Rightarrow B$ is equivalent to the inhabitation of $A \to B$

Get the "⇒"-elimination and "⇒"-introduction rules for free!

(appl) $\dfrac{\Gamma \vdash M : A \to B \qquad \Gamma \vdash N : A}{\Gamma \vdash MN : B}$ ("⇒"-elim) $\dfrac{A \Rightarrow B \quad A}{B}$

(abst) $\dfrac{\Gamma, x : A \vdash M : B \qquad \Gamma \vdash A \to B : s}{\Gamma \vdash \lambda x : A . M : A \to B}$ ("⇒"-intro) | Assume: A |
⋮
B
$A \Rightarrow B$

$\forall$ Universal quantification

Identify $\forall x \in S: P(x)$ with $\Pi x : S . Px$

Justification:
- $\forall x \in S: P(x)$ is true
- for each x in the set S, the proposition $P(x)$ is true
- for each x in S, the type Px is inhabited
- there is a function mapping each x in S to an inhabitant of Px (such a function has type $\Pi x : S . Px$)
- there is an f with $f : \Pi x : S . Px$
- $\Pi x : S . Px$ is inhabited

So again, the truth of $\forall x \in S: P(x)$ is equivalent to the inhabitation of $\Pi x : S . Px$

Again get "$\forall$"-elimination and "$\forall$"-introduction for free!

(appl) $\dfrac{\Gamma \vdash M : \Pi x : A . B \qquad \Gamma \vdash N : A}{\Gamma \vdash MN : B[x := N]}$ ("$\forall$"-elim) $\dfrac{\forall x \in S : P(x) \qquad N \in S}{P(N)}$

Correspondence:
1) "$\forall$" is coded as "Π"
2) S corresponds to A
3) $P(x)$ corresponds to B and $P(N)$ to $B[x := N]$
4) In (appl) every judgement has a context, in ("$\forall$"-elim) the context is traditionally left implicit
5) In (appl) there are proof objects for the propositions $\Pi x : A . B$ and $B[x := N]$. If M is a proof of $\Pi x : A . B$ and N is of type A, then MN is a proof of $B[x := N]$

(abst) $\dfrac{\Gamma, x : A \vdash M : B \qquad \Gamma \vdash \Pi x : A . B : s}{\Gamma \vdash \lambda x : A . M : \Pi x : A . B}$ ("$\forall$"-intro)

$\boxed{\text{Let } x \in S}$
$\vdots$
$P(x)$
$\overline{\forall x \in S : P(x)}$

Correspondence:
1) The second premiss in (abst) does not occur in ("$\forall$"-intro), makes sure that $\Pi x : A . B$ is well-formed which is taken for granted in ("$\forall$"-intro)
2) "$\forall$" is coded as "Π"
3) Again in ("$\forall$"-intro) the context is implicit, only the context extension $x \in S$ in the flag is given explicitly
4) S corresponds to A, $P(x)$ to B
5) In (appl) there are proof objects for B and $\Pi x : A . B$

Summary:

Minimal predicate logic	λP
S is a set A is a proposition	$S : *$ $A : *$
$a \in S$ p proves A	$a : S$ $p : A$
P is a predicate on S	$P : S \to *$
$A \Rightarrow B$ $\forall x \in S : P(x)$	$A \to B \; (= \Pi x : A . B)$ $\Pi x : S . Px$
("$\Rightarrow$"-elim) ("$\Rightarrow$"-intro) ("$\forall$"-elim) ("$\forall$"-intro)	(appl), (abst) (appl), (abst)

Note: We have no negation, conjunction, disjunction or existential quantification in minimal predicate logic. Need to combine several systems!

PAT interpretation of previous example

$\Gamma \equiv A:*,\ P:A\to *$

(12) $\Gamma \vdash \Pi x:A.Px : *$

If A is a set and P is a predicate on A, then $\forall x\in A: P(x)$ is a proposition

(13) $\Gamma \vdash \Pi x:A.Px \to Px : *$

In the same setting, $\forall x\in A: P(x) \Rightarrow P(x)$ is a proposition

(14) $\Gamma \vdash \lambda x:A.\lambda y:Px.y : \Pi x:A.Px\to Px$

In the same setting, there is an inhabitant $\lambda x:A.\lambda y:Px.y$ of the proposition $\forall x\in A\ Px \Rightarrow Px$, i.e. $\forall x\in A: P(x)\Rightarrow P(x)$ is a logical tautology and $\lambda x:A.\lambda y:Px.y$ is a coded version of the proof.

Note that $\forall x\in A: P(x)$ is not a tautology and, indeed, no inhabitant of $\Pi x:A.Px$ can be found.

<u>Example of a logical derivation in λP</u>

$\forall x\in S: \forall y\in S: Q(x,y) \Rightarrow \forall u\in S: Q(u,u)$ where S is a set and Q is a binary predicate on S

Natural deduction:

(a) $\boxed{\text{Assume: } \forall x\in S: \forall y\in S: Q(x,y)}$

(b) $\boxed{\text{Let } u\in S}$

(1) $\forall y\in S: Q(u,y)$ ("$\forall$"-elim) on (a), (b)

(2) $Q(u,u)$ ("$\forall$"-elim) on (1), (b)

(3) $\forall u\in S: Q(u,u)$ ("$\forall$"-intro) on (2)

(4) $\forall x\in S: \forall y\in S: Q(x,y) \Rightarrow \forall u\in S: Q(u,u)$ ("$\Rightarrow$"-intro) on (3)

λP: I

(a) $\boxed{S:*}$

(b) $\boxed{Q:S\to S\to *}$

⋮

(n) $?: \Pi x:S.\Pi y:S.Qxy \to \Pi u:S.Quu$

II

(a) $\boxed{S:*}$

(b) $\boxed{Q:S\to S\to *}$

(c) $\boxed{z:(\Pi x:S.\Pi y:S.Qxy)}$

⋮

(m) $? : \Pi u:S.Quu$

(n) $\ldots : \Pi x:S.\Pi y:S.Qxy \to \Pi u:S.Quu$

III

(a) $\boxed{S:*}$

(b) $\boxed{Q:S\to S\to *}$

(c) $\boxed{z:\Pi x:S.\Pi y:S.Qxy}$

(d) $\boxed{u:S}$

⋮

(e) $? : Quu$

(m) $\ldots : \Pi u:S.Quu$

(n) $\ldots : \Pi x:S.\Pi y:S.Qxy \to \Pi u:S.Quu$

IV

(a) $\boxed{S:*}$

(b) $\boxed{Q:S\to S\to *}$

(c) $\boxed{z:\Pi x:S.\Pi y:S.Qxy}$

(d) $\boxed{u:S}$

(1) $zu : \Pi y:S.Quy$ (appl)

(2) $zuu : Quu$ (appl)

(3) $\lambda u:S.zuu : \Pi u:S.Quu$ (abst)

(4) $\lambda z:(\Pi x:S.\Pi y:S.Qxy).\lambda u:S.zuu : \Pi x:S.\Pi y:S.Qxy \to \Pi u:S.Quu$ (abst)