1 Syntax

The minimal syntax, may extend someday.

```
letter      ::=  a..z | A..Z
ident       ::=  letter {letter}
term        ::=  forall binder {binder}, term
             |   fun {binder} => term
             |   fix fix_body
             |   let ident {binder} :term := term in term
             |   term -> term
             |   term arg {arg}
             |   match term with
                   { | equation}
                 end
             |   sort
             |   (term)
arg         ::=  term
binder      ::=  name : term
name        ::=  _
             |   ident
sort        ::=  Prop | Set | Type
fix_body    ::=  ident {binder} :term := term
equation    ::=  pattern => term
pattern     ::=  ident {name}
```

```
sentence    ::=  axiom
             |   definition
             |   inductive
             |   fixpoint
             |   assertion proof
axiom       ::=  Axiom ident : term .
definition  ::=  Definition ident {binder} : term := term .
inductive   ::=  Inductive ident {binder} : term :=
                   { | ident {binder} : term} .
assertion   ::=  Theorem ident {binder} : term .
proof       ::=  Proof . {tactic .} Qed .
```

<table>
<tr><td>tactic</td><td>::=</td><td>applying</td></tr>
<tr><td></td><td>|</td><td>context_managing</td></tr>
<tr><td></td><td>|</td><td>case_analyzing</td></tr>
<tr><td></td><td>|</td><td>rewriting</td></tr>
<tr><td></td><td>|</td><td>computing</td></tr>
<tr><td></td><td>|</td><td>equality</td></tr>
<tr><td>applying</td><td>::=</td><td>exact term</td></tr>
<tr><td></td><td>|</td><td>apply term [in ident]</td></tr>
<tr><td>context_managing</td><td>::=</td><td>intro [ident]</td></tr>
<tr><td></td><td>|</td><td>intros</td></tr>
<tr><td>case_analyzing</td><td>::=</td><td>destruct term</td></tr>
<tr><td></td><td>|</td><td>induction term</td></tr>
<tr><td>rewriting</td><td>::=</td><td>rewrite [<- | ->] term [in term]</td></tr>
<tr><td>computing</td><td>::=</td><td>simpl</td></tr>
<tr><td>equality</td><td>::=</td><td>reflexivity</td></tr>
<tr><td></td><td>|</td><td>symmetry</td></tr>
</table>

<table>
<tr><td>helper</td><td>::=</td><td>printing</td></tr>
<tr><td></td><td>|</td><td>proof_handling</td></tr>
<tr><td>printing</td><td>::=</td><td>Print ident .</td></tr>
<tr><td></td><td>|</td><td>Check term .</td></tr>
<tr><td>proof_handling</td><td>::=</td><td>Undo .</td></tr>
<tr><td></td><td>|</td><td>Restart .</td></tr>
<tr><td></td><td>|</td><td>Admitted .</td></tr>
<tr><td></td><td>|</td><td>Abort .</td></tr>
</table>

2 Calculus

2.1 Term

1. $\texttt{Set}, \texttt{Prop}$ are terms.
2. Variables $\texttt{x}, \texttt{y}$, etc., are terms.
3. Constants $\texttt{c}, \texttt{d}$, etc., are terms.
4. If $\texttt{x}$ is a variable and $\texttt{T}, \texttt{U}$ are terms, then $\forall \texttt{x} : \texttt{T}, \texttt{U}$ is a term.
5. If $\texttt{x}$ is a variable and $\texttt{T}, \texttt{u}$ are terms, then $\lambda \texttt{x} : \texttt{T}.\ \texttt{u}$ is a term.
6. If $\texttt{x}$ and $\texttt{u}$ are terms, then $(\texttt{t}\ \texttt{u})$ is a term.
7. If $\texttt{x}$ is a variable and $\texttt{t}, \texttt{T}, \texttt{u}$ are terms, then $\texttt{let}\ \ \texttt{x} := \texttt{t} : \texttt{T}\ \ \texttt{in}\ \ \texttt{u}$ is a term.

2.2 Typing Rule

2.2.1 Notation

- $\mathcal{S}$: $\{\texttt{Prop}, \texttt{Set}\}$.
- E : global environment.
- Γ : local context.
- $\texttt{u}\{\texttt{x}/\texttt{t}\}$: substitute free occurrence of variable $\texttt{x}$ to term $\texttt{t}$ in term $\texttt{u}$.
- $\mathcal{WF}(E)[\Gamma]$: E is well-formed and Γ is valid in E.

2.2.2 Typing Rules

$$\mathcal{WF}([])[] \tag{T-Empty}$$

$$\frac{E[\Gamma] \vdash \texttt{T} : \texttt{s} \qquad \texttt{s} \in \mathcal{S} \qquad \texttt{x} \notin \Gamma}{\mathcal{WF}(E)[\Gamma :: (\texttt{x} : \texttt{T})]} \tag{T-Local-Ax}$$

$$\frac{E[] \vdash \texttt{t} : \texttt{T} \qquad \texttt{c} \notin E}{\mathcal{WF}(E : \texttt{c} := \texttt{t} : \texttt{T})} \tag{T-Local-Def}$$

$$\frac{\mathcal{WF}(E)[\Gamma] \qquad (\texttt{x} : \texttt{T}) \in \Gamma}{E[\Gamma] \vdash \texttt{x} : \texttt{T}} \tag{T-Var1}$$

$$\frac{\mathcal{WF}(E)[\Gamma] \qquad (\texttt{x} := \texttt{t} : \texttt{T}) \in \Gamma}{E[\Gamma] \vdash \texttt{x} : \texttt{T}} \tag{T-Var2}$$

$$\frac{\mathcal{WF}(E)[\Gamma] \qquad (\texttt{c} : \texttt{T}) \in E}{E[\Gamma] \vdash \texttt{c} : \texttt{T}} \tag{T-Const1}$$

$$\frac{\mathcal{WF}(E)[\Gamma] \qquad (\texttt{c} := \texttt{t} : \texttt{T}) \in E}{E[\Gamma] \vdash \texttt{c} : \texttt{T}} \tag{T-Const2}$$

$$\frac{E[\Gamma] \vdash \texttt{T} : \texttt{s} \qquad \texttt{s} \in \mathcal{S} \qquad E[\Gamma :: (\texttt{x} : \texttt{T})] \vdash \texttt{U} : \texttt{Prop}}{E[\Gamma] \vdash \forall \texttt{x} : \texttt{T}, \texttt{U} : \texttt{Prop}} \tag{T-Prod-Prop}$$

$$\frac{E[\Gamma] \vdash \texttt{T} : \texttt{s} \qquad \texttt{s} \in \mathcal{S} \qquad E[\Gamma :: (\texttt{x} : \texttt{T})] \vdash \texttt{U} : \texttt{Set}}{E[\Gamma] \vdash \forall \texttt{x} : \texttt{T}, \texttt{U} : \texttt{Set}} \tag{T-Prod-Set}$$

$$\frac{E[\Gamma] \vdash \forall \texttt{x} : \texttt{T}, \texttt{U} : \texttt{s} \qquad E[\Gamma :: (\texttt{x} : \texttt{T})] \vdash \texttt{t} : \texttt{U}}{E[\Gamma] \vdash \lambda \texttt{x} : \texttt{T}.\ \texttt{t} : \forall \texttt{x} : \texttt{T}, \texttt{U}} \tag{T-Abs}$$

$$\frac{E[\Gamma] \vdash \forall \mathtt{x} : \mathtt{U}, \mathtt{T} \qquad E[\Gamma] \vdash \mathtt{u} : \mathtt{U}}{E[\Gamma] \vdash (\mathtt{t}\ \mathtt{u}) : \mathtt{T}\{\mathtt{x}/\mathtt{u}\}} \tag{T-App}$$

$$\frac{E[\Gamma] \vdash \mathtt{t} : \mathtt{T} \qquad E[\Gamma :: (\mathtt{x} := \mathtt{t} : \mathtt{T})] \vdash \mathtt{u} : \mathtt{U}}{E[\Gamma] \vdash \mathtt{let}\ \ \mathtt{x} := \mathtt{t} : \mathtt{T}\ \ \mathtt{in}\ \ \mathtt{u} : \mathtt{U}\{\mathtt{x}/\mathtt{t}\}} \tag{T-Let}$$

2.3 Conversion Rule

2.3.1 Notation

- $E[\Gamma] \vdash \mathtt{t} \rhd \mathtt{u}$: $\mathtt{t}$ reduces to $\mathtt{u}$ in E, Γ with one of the $\beta, \iota, \delta, \zeta$ reductions.
- $E[\Gamma] \vdash \mathtt{t} \overset{*}{\rhd} \mathtt{u}$: $E[\Gamma] \vdash \mathtt{t} \rhd \cdots \rhd \mathtt{u}$.
- $\mathtt{u} \equiv \mathtt{v}$: $\mathtt{u}$ and $\mathtt{v}$ are identical.

2.3.2 Conversion Rules

$$\frac{E[\Gamma] \vdash (\lambda \mathtt{x} : \mathtt{T}.\ \mathtt{t})\ u}{\mathtt{t}\{\mathtt{x}/\mathtt{u}\}} \tag{β-Conv}$$

$$\frac{\mathtt{case}((\mathtt{c_p}\ \mathtt{q_1}\ \cdots\ \mathtt{q_r}\ \mathtt{a_1}\ \cdots\ \mathtt{a_m}), \mathtt{P}, \mathtt{f_1} | \cdots | \mathtt{f_n})}{f_i\ a_1\ \cdots\ a_m} \tag{ι-Conv}$$

$$\frac{E[\Gamma] \vdash \mathtt{x} \qquad (\mathtt{x} := \mathtt{t} : \mathtt{T}) \in \Gamma}{\mathtt{t}} \tag{δ-Conv1}$$

$$\frac{E[\Gamma] \vdash \mathtt{c} \qquad (\mathtt{x} := \mathtt{t} : \mathtt{T}) \in E}{\mathtt{t}} \tag{δ-Conv2}$$

$$\frac{E[\Gamma] \vdash \mathtt{let}\ \ \mathtt{x} := \mathtt{u}\ \ \mathtt{in}\ \ \mathtt{t}}{\mathtt{t}\{\mathtt{x}/\mathtt{u}\}} \tag{ζ-Conv}$$

$$\frac{E[\Gamma] \vdash \mathtt{t} : \forall \mathtt{x} : \mathtt{T}, \mathtt{U} \qquad \mathtt{x} \text{ fresh in } \mathtt{t}}{\lambda \mathtt{x} : \mathtt{T}.\ (\mathtt{t}\ \mathtt{x})} \tag{η-Exp}$$

Definition 1 (Convertibility)**.** $\mathtt{t_1}$ *and* $\mathtt{t_2}$ *are convertible iff there exists* $\mathtt{u_1}$ *and* $\mathtt{u_2}$ *such that* $E[\Gamma] \vdash \mathtt{t_1} \overset{*}{\rhd} \mathtt{u_1}$ *and* $E[\Gamma] \vdash \mathtt{t_2} \overset{*}{\rhd} \mathtt{u_2}$ *and either* $u_1 \equiv u_2$ *or they are convertible up to* η*-expansion.*

2.4 Inductive Definition

References

[1] The Coq Development Team. *The Coq Proof Assistant Reference Manual*, 8.7.2 edition, February 2018.