Introduction to Logiweb

Klaus Grue

GRD-2005-05-26.UTC:06:49:32.964854

Contents

1 Introduction

Disclaimer: This document is under construction.

Logiweb [1] is a system for distribution of mathematical definitions, lemmas, and proofs. More precisely, the Logiweb standard comprises a format, a semantics, and a protocol for storage, interpretation, and transmission of formal mathematics.

A "clean" implementation of Logiweb comprises a Logiweb server, a Logiweb client, and an authoring tool. The Logiweb standard governs the server, the client, and the backend of the authoring tool.

Logiweb allows a user to submit Logiweb pages that contain formal formal mathematics. To submit a page, the user must first produce the page using the authoring tool and then make the page available on the ordinary World Wide Web (WWW) under some Hyper Text Transport Protocal Uniform Resource Locator (http URL). Then the authoring tool must notify the nearest Logiweb server about the submission.

Each Logiweb page has a Logiweb reference which essentially is a global hash key computed on the basis of the contents of the page. Logiweb pages are addressed using Logiweb references but retrieved from WWW using their http URL. The main task of the mesh of Logiweb servers is to translate Logiweb references to http URLs.

A single Logiweb server cannot do the translation from Logiweb reference to http URL alone. Rather, the Logiweb servers form a mesh that cooperate on indexing all Logiweb pages in the world.

1.1 Legal issues

Once a Logiweb page is submitted to Logiweb it is understood that anybody is allowed to make verbatim copies of the page and resubmit them to Logiweb. Resubmitted copies have the same Logiweb reference as the original but a different http URL. For this reason, each Logiweb page may have many http URLs but only one Logiweb reference.

To Logiweb, all copies of a Logiweb page are equal, and the original (i.e. the first copy submitted to Logiweb) has no special priviledges. A Logiweb page continues to exist as long as there is at least one copy of it on Logiweb. Hence, a Logiweb page may continue to exist even if the original is deleted.

Copyright notices on Logiweb pages may restrict human readers in what they do with the pages, such as modifying them or exporting them to other media than Logiweb. But copyright notices cannot prevent verbatim copying inside Logiweb: verbatim copying is what Logiweb does to submitted pages. To avoid verbatim copying of your page, don't submit it to Logiweb.

1.2 More legal issues

The word "Logiweb" is a good one and, hence, it is used as a trademark and a registered trademark for many different things. But when used for a format, a semantics, and a protocol for exchange of mathematics via the internet, "Logiweb" is a trademark of the author of the present paper. The purpose is to force incompatible versions of Logiweb to pick other names. Furthermore, the blue "Logiweb and stribes" logo is a trademark of the author.

1.3 References

The reference of the present page is

```
01
37 E5 C4 3D B7  CC 92 BF 23 49
70 40 5B C5 A3  0D AD A1 5B 46
F6 EF A9 B0 BF  AA 9B 08 06
```

when expressed in *mixed endian hexadecimal*. "Mixed endian" indicates that the references is written least significant byte first but most significant digit first. Or, stated otherwise, the bytes are written in network byte order but each byte is written with the most significant digit first as in dot-notation for ip-addresses. When written in mixed endian Kana, the reference of the present page reads

```
nani
nete kuti kata neki sete  kaka sinu seke nune tasi
tena tana tise kati sune  naki suki suni tise tatu
ketu kuke susi sena seke  susu sise nasa natu
```

In the Kana notation, "n", "t", "s", "k" represent 00, 01, 10, and 11, respectively. Also, "a", "i", "u", "e" represent 00, 01, 10, and 11, respectively. So "11000100 11000000" translates to "kata kana" and "00 01 10 11 11 10 01 00" translates to "nise kuta". The Kana format is useful if one has to pronounce a reference.

1.4 Bibliographies

Each Logiweb page contains a bibliography which is a list of references to other Logiweb pages. As an example the present page references a page named base.

As mentioned, the reference of a page depends on the contents of the page and one cannot change a page without affecting its reference. For that reason, an author of a Logiweb page can safely refer to another Logiweb page without fearing that that other page will change. Furthermore, if an author references a Logiweb page, then the author may copy and resubmit that page and in that way ensure that the referenced Logiweb page will not disappear from Logiweb.

The base page referenced by the present page contains loads of definitions of formal theories and many other things. The present page, which is under construction, states two small proofs that have been machine checked by Logiweb. Furthermore, the present page presents the same proofs in a style that will be supported in the near future.

2 Logiweb sequent calculus

2.1 Unification

2.1.1 Parameter terms

We shall refer to terms in which bound metavariables are replaced by cardinals as *parameter terms*. The function [parm(t, s, n)] converts an ordinary term [t] into a parameter term in which numbers are constructed from [n] using [b +2∗ n] iteratively. When calling [parm(t, s, n)], [s] should be [T].

[parm(t, s, n) ≐ n!
if t $\stackrel{\mathrm{r}}{=}$ ⌈∀x: y⌉ **then** ∀n: parm(t^2, (t^1 :: n) :: s, T +2∗ n) **else**
let m = **lookup**(t, s, T) **in**
if ¬m **then** m **else** t^{R} :: parm∗(t^t, s, n)][1]

[parm∗(t, s, n) ≐ s!n!If(t^a, T, parm(t^h, s, n) :: parm∗(t^t, s, n))][2]

2.1.2 Substitutions

We shall refer to an array of terms as a substitution. The following function instantiates a parameter term [t] using a substitution [s]:

[inst(t, s) ≐ If(t^c, s[t], t^{R} :: inst∗(t^t, s))][3]

[inst∗(t, s) ≐ s!If(t^a, T, inst(t^h, s) :: inst∗(t^t, s))][4]

2.1.3 Occurrence

[occur(t, u)] is true if the parameter [t] occurs in the parameter term [u]:

[occur(t, u) ≐ If(u^c, t ≈ u, occur∗(t, u^t))][5]

[occur∗(t, u) ≐ t!If(u^a, F, occur(t, u^h) ⩔ occur∗(t, u^t))][6]

---

[1] [parm(t, s, n) $\stackrel{\mathrm{pyk}}{=}$ “parameter term ∗ stack ∗ seed ∗ end parameter”]

[2] [parm∗(t, s, n) $\stackrel{\mathrm{pyk}}{=}$ “parameter term star ∗ stack ∗ seed ∗ end parameter”]

[3] [inst(t, s) $\stackrel{\mathrm{pyk}}{=}$ “instantiate ∗ with ∗ end instantiate”]

[4] [inst∗(t, s) $\stackrel{\mathrm{pyk}}{=}$ “instantiate star ∗ with ∗ end instantiate”]

[5] [occur(t, u) $\stackrel{\mathrm{pyk}}{=}$ “occur ∗ in ∗ end occur”]

[6] [occur∗(t, u) $\stackrel{\mathrm{pyk}}{=}$ “occur star ∗ in ∗ end occur”]

2.1.4 Unifications

We shall refer to the result of applying a substitution to a parameter term as an *instance* of the term. We shall refer to a common instance of two parameter terms as a *unification* of the terms. As an example, $[\mathcal{A} :: 2]$ and $[1 :: \mathcal{B}]$ (where $[\mathcal{A}]$ and $[\mathcal{B}]$ denote numbers) have exactly one unification, namely $[1 :: 2]$. We shall say that two terms are *compatible* if they have at least one unification and *incompatible* otherwise.

A substitution which yields the same result when applied to two terms $[\mathsf{u}]$ and $[\mathsf{v}]$ is said to *unify* the terms. As an example, the substitution which maps $[\mathcal{A}]$ to $[1]$ and $[\mathcal{B}]$ to $[2]$ unifies $[\mathcal{A} :: 2]$ and $[1 :: \mathcal{B}]$.

The *unification algorithm* presented in the following takes two terms as input and returns a unifying substitution if the terms are compatible. As an example, when applied to $[\mathcal{A} :: 2]$ and $[1 :: \mathcal{B}]$, the unification algorithm returns the substitution which maps $[\mathcal{A}]$ to $[1]$ and $[\mathcal{B}]$ to $[2]$.

There is more than one substitution which unifies $[\mathcal{A} :: 2]$ and $[1 :: \mathcal{B}]$. As an example the substitution that maps $[\mathcal{A}]$ to $[1]$, $[\mathcal{B}]$ to $[2]$, and $[\mathcal{C}]$ to $[3]$ also unifies $[\mathcal{A} :: 2]$ and $[1 :: \mathcal{B}]$.

2.1.5 Circularity test

$[\mathrm{circular}(\mathsf{t} = \mathsf{u}, \mathsf{s})]$ is true if adding the equation $[\mathsf{t} = \mathsf{u}]$ to the unification $[\mathsf{s}]$ creates a circularity.

$[\mathrm{circular}(\mathsf{t} = \mathsf{u}, \mathsf{s}) \doteq \mathsf{s}!\mathrm{If}(\mathsf{u}^{\mathsf{c}}, \mathsf{t} \approx \mathsf{u} \mathbin{\ddot{\vee}} \mathrm{occur}(\mathsf{t}, \mathsf{s}[\mathsf{u}]), \mathrm{circular}^*(\mathsf{t} = \mathsf{u}^{\mathsf{t}}, \mathsf{s}))]$[7]

$[\mathrm{circular}^*(\mathsf{t} = \mathsf{u}, \mathsf{s}) \doteq \mathsf{t}!\mathsf{s}!\mathrm{If}(\mathsf{u}^{\mathsf{a}}, \mathsf{F}, \mathrm{circular}(\mathsf{t} = \mathsf{u}^{\mathsf{h}}, \mathsf{s}) \mathbin{\ddot{\vee}} \mathrm{circular}^*(\mathsf{t} = \mathsf{u}^{\mathsf{t}}, \mathsf{s}))]$[8]

2.1.6 Unification algorithm

$[\mathrm{unify}(\mathsf{t} = \mathsf{u}, \mathsf{s}) \doteq \mathsf{t}!\mathsf{u}!$
if s^{c} **then** s **else**
if t^{c} **then** $\mathrm{unify}_2(\mathsf{t} = \mathsf{u}, \mathsf{s})$ **else**
if u^{c} **then** $\mathrm{unify}_2(\mathsf{u} = \mathsf{t}, \mathsf{s})$ **else**
if $\mathsf{t} \overset{\mathsf{r}}{=} \mathsf{u}$ **then** $\mathrm{unify}^*(\mathsf{t}^{\mathsf{t}} = \mathsf{u}^{\mathsf{t}}, \mathsf{s})$ **else** $0]$[9]

$[\mathrm{unify}^*(\mathsf{t} = \mathsf{u}, \mathsf{s}) \doteq \mathsf{u}!\mathrm{If}(\mathsf{t}^{\mathsf{a}}, \mathsf{s}, \mathrm{unify}^*(\mathsf{t}^{\mathsf{t}} = \mathsf{u}^{\mathsf{t}}, \mathrm{unify}(\mathsf{t}^{\mathsf{h}} = \mathsf{u}^{\mathsf{h}}, \mathsf{s}))))]$[10]

$[\mathrm{unify}_2(\mathsf{t} = \mathsf{u}, \mathsf{s}) \doteq$
if $\mathsf{t} \approx \mathsf{u}$ **then** s **else**
let $\mathsf{t}' = \mathsf{s}[\mathsf{t}]$ **in**
if $\neg\mathsf{t}'$ **then** $\mathrm{unify}(\mathsf{t}' = \mathsf{u}, \mathsf{s})$ **else**
if $\mathrm{circular}(\mathsf{t} = \mathsf{u}, \mathsf{s})$ **then** 0 **else** $\mathsf{s}[\mathsf{t} \rightarrow \mathsf{u}]]$[11]

[7] $[\mathrm{circular}(\mathsf{t} = \mathsf{u}, \mathsf{s}) \overset{\mathrm{pyk}}{=}$ "circular $*$ term $*$ substitution $*$ end circular"]

[8] $[\mathrm{circular}^*(\mathsf{t} = \mathsf{u}, \mathsf{s}) \overset{\mathrm{pyk}}{=}$ "circular star $*$ term $*$ substitution $*$ end circular"]

[9] $[\mathrm{unify}(\mathsf{t} = \mathsf{u}, \mathsf{s}) \overset{\mathrm{pyk}}{=}$ "unify $*$ with $*$ substitution $*$ end unify"]

[10] $[\mathrm{unify}^*(\mathsf{t} = \mathsf{u}, \mathsf{s}) \overset{\mathrm{pyk}}{=}$ "unify star $*$ with $*$ substitution $*$ end unify"]

[11] $[\mathrm{unify}_2(\mathsf{t} = \mathsf{u}, \mathsf{s}) \overset{\mathrm{pyk}}{=}$ "unify two $*$ with $*$ substitution $*$ end unify"]

2.2 Proof generation

2.2.1 Example lemmas

As examples of sequent proofs, we prove two lemmas in the example theory $[\mathrm{T_E}]$:

$[\mathrm{T_E}$ **lemma** Reflexivity: $\forall\mathcal{A}\colon \mathcal{A} = \mathcal{A}]$[12]

[**Proof of** Reflexivity: $\lceil \mathrm{T_E} \vdash \forall\mathcal{A}\colon ($
$\mathrm{HeadPair}^{\mathrm{I}\triangleright *\triangleright} \mathbin{@} \mathcal{A} \mathbin{@} \mathcal{A}$;
$(\mathrm{Transitivity}^{\mathrm{I}\triangleright *\triangleright} \mathbin{@} (\mathcal{A} :: \mathcal{A})^{\mathrm{h}} \mathbin{@} \mathcal{A} \mathbin{@} \mathcal{A})^{\triangleright\triangleright})\rceil]$

$[\mathrm{T_E}$ **lemma** Commutativity: $\forall\mathcal{A}\colon\forall\mathcal{B}\colon \mathcal{A} = \mathcal{B} \vdash \mathcal{B} = \mathcal{A}]$[13]

[**Proof of** Commutativity: $\lceil \mathrm{T_E} \vdash \forall\mathcal{A}\colon\forall\mathcal{B}\colon \mathcal{A} = \mathcal{B} \vdash ($
$\mathrm{Reflexivity}^{\mathrm{I}\triangleright *\triangleright} \mathbin{@} \mathcal{A}$;
$(\mathrm{Transitivity}^{\mathrm{I}\triangleright *\triangleright} \mathbin{@} \mathcal{A} \mathbin{@} \mathcal{B} \mathbin{@} \mathcal{A})^{\triangleright\triangleright})\rceil]$

As can be seen on the diagnose hook of the present page, the proofs above are correct according the the machine check made by the verifier.

2.2.2 Medium level proofs

In the following we define "proof tactics" and a "proof expander" which can translate medium level proofs like the one below to Logiweb sequent calculus proofs like the ones above. Actually, the proof below will translate to the proof of reflexivity above.

$[\mathrm{T_E}$ **lemma** $\mathrm{Reflexivity_1}\colon \forall\mathcal{A}\colon \mathcal{A} = \mathcal{A}]$[14]

$\mathrm{T_E}$ **proof of** $\mathrm{Reflexivity_1}$:

L01:	Arbitrary $\gg$	$\mathcal{A}$	;
L02:	HeadPair $\gg$	$(\mathcal{A} :: \mathcal{A})^{\mathrm{h}} = \mathcal{A}$	;
L03:	Transitivity $\triangleright$ L02 $\triangleright$ L02 $\gg$	$\mathcal{A} = \mathcal{A}$	$\square$

2.2.3 Proof tactics

Counted in sequent operators, proofs typically comprise a small amount of original thought and a lot of trivial derivations. Frequently, trivial derivations can be generated by computer programs. Such programs are typically called *tactics* or *proof tactics* [2].

To support proof tactics, we introduce a *tactic aspect* for defining them and a *proof expander* for evaluating them. The proof evaluates proof tactics

[12] [Reflexivity $\overset{\text{pyk}}{=}$ "reflexivity lemma"]

[13] [Commutativity $\overset{\text{pyk}}{=}$ "commutativity lemma"]

[14] [$\mathrm{Reflexivity_1}$ $\overset{\text{pyk}}{=}$ "reflexivity lemma one"]

and generates sequent proofs, which the proof evaluator may then evaluate to sequents.

The proof expander is itself a tactic since it generates proofs. The proof expander is a value defined tactic like the rule lemma tactic defined previously. But the proof expander is a particularly general tactic which brings life to tactics that are defined using the tactic aspect.

We make the [<tactic>] aspect self-evaluating and use [tactic] to denote it:

$$[\text{<tactic>} \doteq \lceil \text{<tactic>} \rceil]^{15}$$

$$[\text{tactic} \overset{\text{msg}}{=} \text{<tactic>}]^{16}$$

For convenience, we define a construct for making tactic definitions:

$$[[\mathsf{x} \overset{\text{tactic}}{=} \mathsf{y}] \doteq [(\mathsf{x})^{\mathbf{p}} \overset{\text{tactic}}{\rightarrow} \mathsf{y}]]^{17}$$

2.2.4 The proof expander

The proof expander $[\mathcal{P}(\mathsf{t},\mathsf{s},\mathsf{c})]$ proof expands the term [t] for the *proof state* [s] and the cache [c] and returns the result as a semitagged map. The untagged version $[\mathcal{U}(\mathcal{P}(\mathsf{t},\mathsf{s},\mathsf{c}))]$ is the expansion itself.

The proof expander $[\mathcal{P}(\mathsf{t},\mathsf{s},\mathsf{c})]$ differs from the macro expander $[\mathcal{M}(\mathsf{t},\mathsf{s},\mathsf{c})]$ in that it has no special treatment for page symbols and in that it uses the [tactic] aspect instead of the [macro] aspect.

When proofs are expanded, they are first macro expanded and then proof expanded. Macro expansion is done iteratively until the codex reaches a fixed point whereas proof expansion is done only once. Furthermore, the result of macro expansion is kept in the codex whereas the result of proof expansion is discarded as soon as a proof is checked. Hence, proof expansion is a momentary burden to the computers memory whereas macro expansion is chronic.

The definition of $[\mathcal{P}(\mathsf{t},\mathsf{s},\mathsf{c})]$ is almost identical to that of $[\mathcal{M}(\mathsf{t},\mathsf{s},\mathsf{c})]$. But it is easier to express since we can use macros like the 'let' macro when defining proof expansion. For obvious reasons, macros cannot be used when defining the notion of macro expansion. The definition of $[\mathcal{P}(\mathsf{t},\mathsf{s},\mathsf{c})]$ reads:

$$[\mathcal{P}(\mathsf{t},\mathsf{s},\mathsf{c}) \doteq \mathsf{s}!$$
$$\textbf{let } \mathsf{d} = \textbf{aspect}(\text{<tactic>},\mathsf{t},\mathsf{c}) \textbf{ in}$$
$$\textbf{if } \mathsf{d} \textbf{ then } \mathsf{t}^{\mathrm{h}} :: \mathcal{P}^*(\mathsf{t}^{\mathrm{t}},\mathsf{s},\mathsf{c}) \textbf{ else}$$
$$\mathcal{U}^{\mathrm{M}}(\mathcal{E}(\mathsf{d}^3,\mathsf{T},\mathsf{c}) \text{ ‘ } \mathsf{t} \text{ ‘ } \mathsf{s} \text{ ‘ } \mathsf{c})]^{18}$$

$$[\mathcal{P}^*(\mathsf{t},\mathsf{s},\mathsf{c}) \doteq \mathsf{s}!\mathsf{c}!\mathrm{If}(\mathsf{t},\mathsf{T},\mathcal{P}(\mathsf{t}^{\mathrm{h}},\mathsf{s},\mathsf{c}) :: \mathcal{P}^*(\mathsf{t}^{\mathrm{t}},\mathsf{s},\mathsf{c}))]^{19}$$

---

[15] $[\text{<tactic>} \overset{\text{pyk}}{=} \text{“the tactic aspect”}]$

[16] $[\text{tactic} \overset{\text{pyk}}{=} \text{“tactic”}]$

[17] $[[\mathsf{x} \overset{\text{tactic}}{=} \mathsf{y}] \overset{\text{pyk}}{=} \text{“tactic define} * \text{as} * \text{end define”}]$

[18] $[\mathcal{P}(\mathsf{t},\mathsf{s},\mathsf{c}) \overset{\text{pyk}}{=} \text{“proof expand} * \text{state} * \text{cache} * \text{end expand”}]$

[19] $[\mathcal{P}^*(\mathsf{t},\mathsf{s},\mathsf{c}) \overset{\text{pyk}}{=} \text{“proof expand list} * \text{state} * \text{cache} * \text{end expand”}]$

2.2.5 The initial proof state

Proof states have exactly the same format as macro states.

The *initial proof state* $[\mathsf{p}_0]$ is useful for passing as the second parameter to $[\mathcal{P}(\mathsf{t},\mathsf{s},\mathsf{c})]$. $[\mathsf{p}_0]$ is a pair whose head is the proof expander itself and whose tail is left blank. The definition of $[\mathsf{p}_0]$ reads:

$$[\mathsf{p}_0 \overset{\text{val}}{\to} \mathcal{M}(\lambda\mathsf{t}.\lambda\mathsf{s}.\lambda\mathsf{c}.\mathcal{P}(\mathsf{t},\mathsf{s},\mathsf{c})) :: \mathsf{T}]^{20}$$

The function $[\tilde{\mathcal{M}}(\mathsf{t},\mathsf{s},\mathsf{c})]$ defined previously macro expands the term $[\mathsf{t}]$ using the macro expander embedded in the macro state $[\mathsf{s}]$ using the cache $[\mathsf{c}]$. Since proof states have exactly the same syntax and semantics as macro states, we shall take the liberty to use $[\tilde{\mathcal{M}}(\mathsf{t},\mathsf{s},\mathsf{c})]$ for proof states also.

2.2.6 The conclusion tactic

The *conclusion tactic* $[\mathsf{x} \gg \mathsf{y}]$ constructs a proof of $[\mathsf{y}]$ from the partial proof $[\mathsf{x}]$. The conclusion tactic does the following to the proof $[\mathsf{x}]$:

- Whenever $[\mathsf{x}]$ references a lemma $[\mathsf{l}]$, the conclusion tactic replaces $[\mathsf{l}]$ by $[\mathsf{l}^{\mathrm{I}\triangleright*\triangleright}]$.
- Whenever a subproof $[\mathsf{u}]$ of $[\mathsf{x}]$ proves $[\forall \mathsf{v}\colon \mathsf{w}]$, the conclusion tactic replaces $[\mathsf{u}]$ by $[\mathsf{u} \mathbin{@} \mathsf{a}]$ where the tactic uses unification to guess $[\mathsf{a}]$.
- Whenever a subproof $[\mathsf{u}]$ of $[\mathsf{x}]$ proves $[\mathsf{v} \Vdash \mathsf{w}]$, the conclusion tactic replaces $[\mathsf{u}]$ by $[\mathsf{u}^{\mathrm{V}}]$.

$$[\mathsf{x} \gg \mathsf{y} \overset{\text{tactic}}{=} \lambda\mathsf{t}.\lambda\mathsf{s}.\lambda\mathsf{c}.\text{conclude}_1(\mathsf{t},\mathsf{c})]^{21}$$

$$[\text{conclude}_1(\mathsf{t},\mathsf{c}) \doteq$$
$$\textbf{let } \mathsf{r} = \text{conclude}_2(\mathsf{t}^1,\mathsf{t}^2,\mathsf{c}) \textbf{ in}$$
$$\textbf{if } \mathsf{r}^{\mathrm{c}} \textbf{ then } \text{error}(\text{“Unification failed”},\mathsf{t}) \textbf{ else } \mathsf{r}]^{22}$$

$$[\text{conclude}_2(\mathsf{a},\mathsf{t},\mathsf{c}) \doteq \mathsf{t}!$$
$$\textbf{if } \mathsf{a} \overset{\mathrm{r}}{=} \lceil \mathsf{x} \triangleright \mathsf{y} \rceil \textbf{ then } \text{conclude}_2(\mathsf{a}^1, \mathsf{a}\text{-color}(\mathsf{t} \triangleright \mathsf{a}^2),\mathsf{c}) \textbf{ else}$$
$$\textbf{if } \mathsf{a} \overset{\mathrm{r}}{=} \lceil \mathsf{x} \mathbin{\triangleright\!\!\triangleright} \mathsf{y} \rceil \textbf{ then } \text{conclude}_2(\mathsf{a}^1, \mathsf{a}\text{-color}(\mathsf{t} \mathbin{\triangleright\!\!\triangleright} \mathsf{a}^2),\mathsf{c}) \textbf{ else}$$
$$\textbf{if } \mathsf{a} \overset{\mathrm{r}}{=} \lceil \mathsf{x} \mathbin{@} \mathsf{y} \rceil \textbf{ then } \text{conclude}_2(\mathsf{a}^1, \mathsf{a}\text{-color}(\mathsf{t} \mathbin{@} \mathsf{a}^2),\mathsf{c}) \textbf{ else}$$
$$\textbf{if aspect}(\text{<proof>},\mathsf{a},\mathsf{c}) \textbf{ then } \text{error}(\text{“Lemma expected”},\mathsf{a}) \textbf{ else}$$
$$\textbf{let } \mathsf{d} = \textbf{aspect}(\text{<stmt>},\mathsf{a},\mathsf{c}) \textbf{ in}$$
$$\text{conclude}_3(\mathsf{a}^{\mathrm{I}\triangleright*\triangleright},\mathsf{t},\text{parm}(\mathsf{d}^{32},\mathsf{T},1),\mathsf{T})]^{23}$$

---

[20] $[\mathsf{p}_0 \overset{\text{pyk}}{=} \text{“proof state”}]$

[21] $[\mathsf{x} \gg \mathsf{y} \overset{\text{pyk}}{=} \text{“}*\text{ conclude }*\text{”}]$

[22] $[\text{conclude}_1(\mathsf{t},\mathsf{c}) \overset{\text{pyk}}{=} \text{“conclude one }*\text{ cache }*\text{ end conclude”}]$

[23] $[\text{conclude}_2(\mathsf{a},\mathsf{t},\mathsf{c}) \overset{\text{pyk}}{=} \text{“conclude two }*\text{ proves }*\text{ cache }*\text{ end conclude”}]$

$$
\begin{array}{l}
[\mathrm{conclude}_3(\mathsf{a},\mathsf{t},\mathsf{l},\mathsf{s}) \doteq \mathsf{a!t!l!s!} \\
\textbf{if } \mathsf{l} \stackrel{\mathrm{r}}{=} \lceil \mathsf{x} \vdash \mathsf{y} \rceil \textbf{ then} \\
\mathsf{t} \stackrel{\mathrm{r}}{=} \lceil \mathsf{x} \rhd \mathsf{y} \rceil \left\{ \begin{array}{l} \mathrm{conclude}_3(\mathsf{a}^{\rhd}, \mathsf{t}^1, \mathsf{l}^2, \mathrm{unify}(\mathsf{l}^1 = \mathsf{t}^2, \mathsf{s})) \\ \mathrm{conclude}_3(\mathsf{a}^{\rhd}, \mathsf{t}, \mathsf{l}^2, \mathsf{s}) \end{array} \right. \textbf{else} \\
\textbf{if } \mathsf{l} \stackrel{\mathrm{r}}{=} \lceil \mathsf{x} \Vdash \mathsf{y} \rceil \textbf{ then} \\
\mathsf{t} \stackrel{\mathrm{r}}{=} \lceil \mathsf{x} \Rrightarrow \mathsf{y} \rceil \left\{ \begin{array}{l} \mathrm{conclude}_3(\mathsf{a}^{\rhd}, \mathsf{t}^1, \mathsf{l}^2, \mathrm{unify}(\mathsf{l}^1 = \mathsf{t}^2, \mathsf{s})) \\ \mathrm{conclude}_3(\mathsf{a}^{\rhd}, \mathsf{t}, \mathsf{l}^2, \mathsf{s}) \end{array} \right. \textbf{else} \\
\textbf{if } \mathsf{l} \stackrel{\mathrm{r}}{=} \lceil \forall \mathsf{x}\colon \mathsf{y} \rceil \textbf{ then} \\
\mathsf{t} \stackrel{\mathrm{r}}{=} \lceil \mathsf{x} @ \mathsf{y} \rceil \left\{ \begin{array}{l} \mathrm{conclude}_3(\mathsf{a} @ \mathsf{t}^2, \mathsf{t}^1, \mathsf{l}^2, \mathrm{unify}(\mathsf{l}^1 = \mathsf{t}^2, \mathsf{s})) \\ \mathrm{conclude}_3(\mathsf{a} @ \mathsf{l}^1, \mathsf{t}, \mathsf{l}^2, \mathsf{s}) \end{array} \right. \textbf{else} \\
\textbf{let } \mathsf{s} = \mathrm{unify}(\mathsf{l} = \mathsf{t}, \mathsf{s}) \textbf{ in} \\
\textbf{if } \mathsf{s}^{\mathrm{c}} \textbf{ then } \mathsf{s} \textbf{ else} \\
\mathrm{inst}(\mathsf{a},\mathsf{s})]^{24}
\end{array}
$$

Modus ponens:

$$[\mathsf{x} \rhd \mathsf{y} \doteq \langle \lceil \mathsf{x} \rhd \mathsf{y} \rceil^{\mathrm{R}}, \mathsf{x}, \mathsf{y} \rangle]^{25}$$

Modus probans:

$$[\mathsf{x} \Rrightarrow \mathsf{y} \doteq \langle \lceil \mathsf{x} \Rrightarrow \mathsf{y} \rceil^{\mathrm{R}}, \mathsf{x}, \mathsf{y} \rangle]^{26}$$

2.2.7 Proof constructors

Medium level proofs will be constructed from the constructs listed in the following. In the following the constructs are listed using their texname aspects. This is convenient when talking about the constructs. In the medium level proof above, the constructs are typeset using the tex aspect; the tex aspects include newline and tabulation commands which make proofs pleasing to read.

$$[\mathsf{t}\ \textbf{proof of}\ \mathsf{s}\colon \mathsf{p} \doteq [\textbf{Proof of}\ \mathsf{s}\colon \lambda \mathsf{c}.\lambda \mathsf{x}.\mathcal{P}(\lceil \mathsf{t} \vdash \mathsf{p} \rceil, \mathrm{p}_0, \mathsf{c})]]^{27}$$

$$[\mathrm{Line}\,\mathsf{l} : \mathsf{a} \gg \mathsf{i}; \mathsf{p} \doteq (\mathsf{a} \gg \mathsf{i}; \textbf{let } \mathsf{l} \doteq \mathsf{i} \ \text{ in } \ \mathsf{p})]^{28}$$

$$[\mathrm{Last\ line}\,\mathsf{a} \gg \mathsf{i} \doteq (\mathsf{a} \gg \mathsf{i})]^{29}$$

$$[\mathrm{Line}\,\mathsf{l} : \mathrm{Premise} \gg \mathsf{i}; \mathsf{p} \doteq (\mathsf{i} \vdash \textbf{let } \mathsf{l} \doteq \mathsf{i} \ \text{ in } \ \mathsf{p})]^{30}$$

$$[\mathrm{Line}\,\mathsf{l} : \mathrm{Side\text{-}condition} \gg \mathsf{i}; \mathsf{p} \doteq (\mathsf{i} \Vdash \textbf{let } \mathsf{l} \doteq \mathsf{i} \ \text{ in } \ \mathsf{p})]^{31}$$

---

[24] $[\mathrm{conclude}_3(\mathsf{a},\mathsf{t},\mathsf{l},\mathsf{s}) \stackrel{\mathrm{pyk}}{=}$ "conclude three ∗ proves ∗ lemma ∗ substitution ∗ end conclude"]

[25] $[\mathsf{x} \rhd \mathsf{y} \stackrel{\mathrm{pyk}}{=}$ "∗ modus ponens ∗"]

[26] $[\mathsf{x} \Rrightarrow \mathsf{y} \stackrel{\mathrm{pyk}}{=}$ "∗ modus probans ∗"]

[27] $[\mathsf{t}\ \textbf{proof of}\ \mathsf{l} : \mathsf{p} \stackrel{\mathrm{pyk}}{=}$ "∗ proof of ∗ reads ∗"]

[28] $[\mathrm{Line}\,\mathsf{l} : \mathsf{a} \gg \mathsf{i}; \mathsf{p} \stackrel{\mathrm{pyk}}{=}$ "line ∗ because ∗ indeed ∗ cut ∗"]

[29] $[\mathrm{Last\ line}\,\mathsf{a} \gg \mathsf{i} \stackrel{\mathrm{pyk}}{=}$ "because ∗ indeed ∗ qed"]

[30] $[\mathrm{Line}\,\mathsf{l} : \mathrm{Premise} \gg \mathsf{i}; \mathsf{p} \stackrel{\mathrm{pyk}}{=}$ "line ∗ premise ∗ cut ∗"]

[31] $[\mathrm{Line}\,\mathsf{l} : \mathrm{Side\text{-}condition} \gg \mathsf{i}; \mathsf{p} \stackrel{\mathrm{pyk}}{=}$ "line ∗ side condition ∗ cut ∗"]

[Arbitrary $\gg$ $\mathsf{i}; \mathsf{p} \doteq (\forall \mathsf{i}\colon \mathsf{p})$][32]

[Local $\gg$ $\mathsf{a} = \mathsf{i}; \mathsf{p} \doteq (\mathbf{let}\ \mathsf{a} \doteq \mathsf{i}\ \ \mathrm{in}\ \ \mathsf{p})$][33]

A Chores

The name of the page:

[check $\overset{\text{pyk}}{=}$ “check”]

A.1 Line numbers

The following definitions are experimental definitions of line numbers named “ell a”, “ell b”, etc. which expand into [L_{01}], [L_{02}], etc. in such a way that proof lines are numbered in succession. The “ell dummy” operator expands into different line numbers each time it is used and can be used for lines that are never referenced.

[$\mathrm{L}_{\mathrm{a}} \overset{\text{pyk}}{=}$ “ell a”] [$\mathrm{L}_{\mathrm{b}} \overset{\text{pyk}}{=}$ “ell b”] [$\mathrm{L}_{\mathrm{c}} \overset{\text{pyk}}{=}$ “ell c”] [$\mathrm{L}_{\mathrm{d}} \overset{\text{pyk}}{=}$ “ell d”] [$\mathrm{L}_{\mathrm{e}} \overset{\text{pyk}}{=}$ “ell e”]
[$\mathrm{L}_{\mathrm{f}} \overset{\text{pyk}}{=}$ “ell f”] [$\mathrm{L}_{\mathrm{g}} \overset{\text{pyk}}{=}$ “ell g”] [$\mathrm{L}_{\mathrm{h}} \overset{\text{pyk}}{=}$ “ell h”] [$\mathrm{L}_{\mathrm{i}} \overset{\text{pyk}}{=}$ “ell i”] [$\mathrm{L}_{\mathrm{j}} \overset{\text{pyk}}{=}$ “ell j”]
[$\mathrm{L}_{\mathrm{k}} \overset{\text{pyk}}{=}$ “ell k”] [$\mathrm{L}_{\mathrm{l}} \overset{\text{pyk}}{=}$ “ell l”] [$\mathrm{L}_{\mathrm{m}} \overset{\text{pyk}}{=}$ “ell m”] [$\mathrm{L}_{\mathrm{n}} \overset{\text{pyk}}{=}$ “ell n”] [$\mathrm{L}_{\mathrm{o}} \overset{\text{pyk}}{=}$ “ell o”]
[$\mathrm{L}_{\mathrm{p}} \overset{\text{pyk}}{=}$ “ell p”] [$\mathrm{L}_{\mathrm{q}} \overset{\text{pyk}}{=}$ “ell q”] [$\mathrm{L}_{\mathrm{r}} \overset{\text{pyk}}{=}$ “ell r”] [$\mathrm{L}_{\mathrm{s}} \overset{\text{pyk}}{=}$ “ell s”] [$\mathrm{L}_{\mathrm{t}} \overset{\text{pyk}}{=}$ “ell t”]
[$\mathrm{L}_{\mathrm{u}} \overset{\text{pyk}}{=}$ “ell u”] [$\mathrm{L}_{\mathrm{v}} \overset{\text{pyk}}{=}$ “ell v”] [$\mathrm{L}_{\mathrm{w}} \overset{\text{pyk}}{=}$ “ell w”] [$\mathrm{L}_{\mathrm{x}} \overset{\text{pyk}}{=}$ “ell x”] [$\mathrm{L}_{\mathrm{y}} \overset{\text{pyk}}{=}$ “ell y”]
[$\mathrm{L}_{\mathrm{z}} \overset{\text{pyk}}{=}$ “ell z”] [$\mathrm{L}_{\mathrm{A}} \overset{\text{pyk}}{=}$ “ell big a”] [$\mathrm{L}_{\mathrm{B}} \overset{\text{pyk}}{=}$ “ell big b”] [$\mathrm{L}_{\mathrm{C}} \overset{\text{pyk}}{=}$ “ell big c”]
[$\mathrm{L}_{\mathrm{D}} \overset{\text{pyk}}{=}$ “ell big d”] [$\mathrm{L}_{\mathrm{E}} \overset{\text{pyk}}{=}$ “ell big e”] [$\mathrm{L}_{\mathrm{F}} \overset{\text{pyk}}{=}$ “ell big f”] [$\mathrm{L}_{\mathrm{G}} \overset{\text{pyk}}{=}$ “ell big g”]
[$\mathrm{L}_{\mathrm{H}} \overset{\text{pyk}}{=}$ “ell big h”] [$\mathrm{L}_{\mathrm{I}} \overset{\text{pyk}}{=}$ “ell big i”] [$\mathrm{L}_{\mathrm{J}} \overset{\text{pyk}}{=}$ “ell big j”] [$\mathrm{L}_{\mathrm{K}} \overset{\text{pyk}}{=}$ “ell big k”]
[$\mathrm{L}_{\mathrm{L}} \overset{\text{pyk}}{=}$ “ell big l”] [$\mathrm{L}_{\mathrm{M}} \overset{\text{pyk}}{=}$ “ell big m”] [$\mathrm{L}_{\mathrm{N}} \overset{\text{pyk}}{=}$ “ell big n”] [$\mathrm{L}_{\mathrm{O}} \overset{\text{pyk}}{=}$ “ell big o”]
[$\mathrm{L}_{\mathrm{P}} \overset{\text{pyk}}{=}$ “ell big p”] [$\mathrm{L}_{\mathrm{Q}} \overset{\text{pyk}}{=}$ “ell big q”] [$\mathrm{L}_{\mathrm{R}} \overset{\text{pyk}}{=}$ “ell big r”] [$\mathrm{L}_{\mathrm{S}} \overset{\text{pyk}}{=}$ “ell big s”]
[$\mathrm{L}_{\mathrm{T}} \overset{\text{pyk}}{=}$ “ell big t”] [$\mathrm{L}_{\mathrm{U}} \overset{\text{pyk}}{=}$ “ell big u”] [$\mathrm{L}_{\mathrm{V}} \overset{\text{pyk}}{=}$ “ell big v”] [$\mathrm{L}_{\mathrm{W}} \overset{\text{pyk}}{=}$ “ell big w”]
[$\mathrm{L}_{\mathrm{X}} \overset{\text{pyk}}{=}$ “ell big x”] [$\mathrm{L}_{\mathrm{Y}} \overset{\text{pyk}}{=}$ “ell big y”] [$\mathrm{L}_{\mathrm{Z}} \overset{\text{pyk}}{=}$ “ell big z”] [$\mathrm{L}_{?} \overset{\text{pyk}}{=}$ “ell dummy”]
[$\mathrm{L}_{\mathrm{a}} \overset{\text{name}}{=}$ “L_a”] [$\mathrm{L}_{\mathrm{b}} \overset{\text{name}}{=}$ “L_b”] [$\mathrm{L}_{\mathrm{c}} \overset{\text{name}}{=}$ “L_c”] [$\mathrm{L}_{\mathrm{d}} \overset{\text{name}}{=}$ “L_d”] [$\mathrm{L}_{\mathrm{e}} \overset{\text{name}}{=}$ “L_e”]
[$\mathrm{L}_{\mathrm{f}} \overset{\text{name}}{=}$ “L_f”] [$\mathrm{L}_{\mathrm{g}} \overset{\text{name}}{=}$ “L_g”] [$\mathrm{L}_{\mathrm{h}} \overset{\text{name}}{=}$ “L_h”] [$\mathrm{L}_{\mathrm{i}} \overset{\text{name}}{=}$ “L_i”] [$\mathrm{L}_{\mathrm{j}} \overset{\text{name}}{=}$ “L_j”]
[$\mathrm{L}_{\mathrm{k}} \overset{\text{name}}{=}$ “L_k”] [$\mathrm{L}_{\mathrm{l}} \overset{\text{name}}{=}$ “L_l”] [$\mathrm{L}_{\mathrm{m}} \overset{\text{name}}{=}$ “L_m”] [$\mathrm{L}_{\mathrm{n}} \overset{\text{name}}{=}$ “L_n”] [$\mathrm{L}_{\mathrm{o}} \overset{\text{name}}{=}$ “L_o”]
[$\mathrm{L}_{\mathrm{p}} \overset{\text{name}}{=}$ “L_p”] [$\mathrm{L}_{\mathrm{q}} \overset{\text{name}}{=}$ “L_q”] [$\mathrm{L}_{\mathrm{r}} \overset{\text{name}}{=}$ “L_r”] [$\mathrm{L}_{\mathrm{s}} \overset{\text{name}}{=}$ “L_s”] [$\mathrm{L}_{\mathrm{t}} \overset{\text{name}}{=}$ “L_t”]
[$\mathrm{L}_{\mathrm{u}} \overset{\text{name}}{=}$ “L_u”] [$\mathrm{L}_{\mathrm{v}} \overset{\text{name}}{=}$ “L_v”] [$\mathrm{L}_{\mathrm{w}} \overset{\text{name}}{=}$ “L_w”] [$\mathrm{L}_{\mathrm{x}} \overset{\text{name}}{=}$ “L_x”]
[$\mathrm{L}_{\mathrm{y}} \overset{\text{name}}{=}$ “L_y”] [$\mathrm{L}_{\mathrm{z}} \overset{\text{name}}{=}$ “L_z”] [$\mathrm{L}_{\mathrm{A}} \overset{\text{name}}{=}$ “L_A”] [$\mathrm{L}_{\mathrm{B}} \overset{\text{name}}{=}$ “L_B”]
[$\mathrm{L}_{\mathrm{C}} \overset{\text{name}}{=}$ “L_C”] [$\mathrm{L}_{\mathrm{D}} \overset{\text{name}}{=}$ “L_D”] [$\mathrm{L}_{\mathrm{E}} \overset{\text{name}}{=}$ “L_E”] [$\mathrm{L}_{\mathrm{F}} \overset{\text{name}}{=}$ “L_F”]
[$\mathrm{L}_{\mathrm{G}} \overset{\text{name}}{=}$ “L_G”] [$\mathrm{L}_{\mathrm{H}} \overset{\text{name}}{=}$ “L_H”] [$\mathrm{L}_{\mathrm{I}} \overset{\text{name}}{=}$ “L_I”] [$\mathrm{L}_{\mathrm{J}} \overset{\text{name}}{=}$ “L_J”]
[$\mathrm{L}_{\mathrm{K}} \overset{\text{name}}{=}$ “L_K”] [$\mathrm{L}_{\mathrm{L}} \overset{\text{name}}{=}$ “L_L”] [$\mathrm{L}_{\mathrm{M}} \overset{\text{name}}{=}$ “L_M”] [$\mathrm{L}_{\mathrm{N}} \overset{\text{name}}{=}$ “L_N”]

[32] [Arbitrary $\gg$ $\mathsf{i}; \mathsf{p} \overset{\text{pyk}}{=}$ “arbitrary $*$ cut $*$”]

[33] [Local $\gg$ $\mathsf{u} = \mathsf{v}; \mathsf{p} \overset{\text{pyk}}{=}$ “locally define $*$ as $*$ cut $*$”]

[L_O $\stackrel{\text{name}}{=}$ “L_O”] [L_P $\stackrel{\text{name}}{=}$ “L_P”] [L_Q $\stackrel{\text{name}}{=}$ “L_Q”] [L_R $\stackrel{\text{name}}{=}$ “L_R”]
[L_S $\stackrel{\text{name}}{=}$ “L_S”] [L_T $\stackrel{\text{name}}{=}$ “L_T”] [L_U $\stackrel{\text{name}}{=}$ “L_U”] [L_V $\stackrel{\text{name}}{=}$ “L_V”]
[L_W $\stackrel{\text{name}}{=}$ “L_W”] [L_X $\stackrel{\text{name}}{=}$ “L_X”] [L_Y $\stackrel{\text{name}}{=}$ “L_Y”] [L_Z $\stackrel{\text{name}}{=}$ “L_Z”]
[$L_?$ $\stackrel{\text{name}}{=}$ “L_?”]

[L_a $\stackrel{\text{tex}}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_a \else
\if \relax \csname lgwella\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwella {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwella \fi ”]
[L_b $\stackrel{\text{tex}}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_b \else
\if \relax \csname lgwellb\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellb {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellb \fi ”]
[L_c $\stackrel{\text{tex}}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_c \else
\if \relax \csname lgwellc\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellc {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellc \fi ”]
[L_d $\stackrel{\text{tex}}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_d \else
\if \relax \csname lgwelld\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwelld {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwelld \fi ”]
[L_e $\stackrel{\text{tex}}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_e \else
\if \relax \csname lgwelle\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwelle {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwelle \fi ”]
[L_f $\stackrel{\text{tex}}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_f \else
\if \relax \csname lgwellf\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellf {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellf \fi ”]
[L_g $\stackrel{\text{tex}}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_g \else
\if \relax \csname lgwellg\endcsname

```
\global \advance \lgwproofline by 1
\xdef \lgwellg {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellg \fi "]
    [L_h ^^ex= "
\if \relax \csname lgwprooflinep\endcsname L_h \else
\if \relax \csname lgwellh\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellh {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellh \fi "]
    [L_i ^^ex= "
\if \relax \csname lgwprooflinep\endcsname L_i \else
\if \relax \csname lgwelli\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwelli {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwelli \fi "]
    [L_j ^^ex= "
\if \relax \csname lgwprooflinep\endcsname L_j \else
\if \relax \csname lgwellj\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellj {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellj \fi "]
    [L_k ^^ex= "
\if \relax \csname lgwprooflinep\endcsname L_k \else
\if \relax \csname lgwellk\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellk {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellk \fi "]
    [L_l ^^ex= "
\if \relax \csname lgwprooflinep\endcsname L_l \else
\if \relax \csname lgwelll\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwelll {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwelll \fi "]
    [L_m ^^ex= "
\if \relax \csname lgwprooflinep\endcsname L_m \else
\if \relax \csname lgwellm\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellm {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellm \fi "]
    [L_n ^^ex= "
\if \relax \csname lgwprooflinep\endcsname L_n \else
\if \relax \csname lgwelln\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwelln {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
```

```
\fi \lgwelln \fi "]
    [L_o ^^= "
\if \relax \csname lgwprooflinep\endcsname L_o \else
\if \relax \csname lgwello\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwello {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwello \fi "]
    [L_p ^^= "
\if \relax \csname lgwprooflinep\endcsname L_p \else
\if \relax \csname lgwellp\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellp {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellp \fi "]
    [L_q ^^= "
\if \relax \csname lgwprooflinep\endcsname L_q \else
\if \relax \csname lgwellq\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellq {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellq \fi "]
    [L_r ^^= "
\if \relax \csname lgwprooflinep\endcsname L_r \else
\if \relax \csname lgwellr\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellr {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellr \fi "]
    [L_s ^^= "
\if \relax \csname lgwprooflinep\endcsname L_s \else
\if \relax \csname lgwells\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwells {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwells \fi "]
    [L_t ^^= "
\if \relax \csname lgwprooflinep\endcsname L_t \else
\if \relax \csname lgwellt\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellt {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellt \fi "]
    [L_u ^^= "
\if \relax \csname lgwprooflinep\endcsname L_u \else
\if \relax \csname lgwellu\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellu {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellu \fi "]
```

[$L_v \overset{tex}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_v \else
\if \relax \csname lgwellv\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellv {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellv \fi ”]
[$L_w \overset{tex}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_w \else
\if \relax \csname lgwellw\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellw {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellw \fi ”]
[$L_x \overset{tex}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_x \else
\if \relax \csname lgwellx\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellx {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellx \fi ”]
[$L_y \overset{tex}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_y \else
\if \relax \csname lgwelly\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwelly {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwelly \fi ”]
[$L_z \overset{tex}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_z \else
\if \relax \csname lgwellz\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellz {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellz \fi ”]
[$L_A \overset{tex}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_A \else
\if \relax \csname lgwellbiga\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbiga {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbiga \fi ”]
[$L_B \overset{tex}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_B \else
\if \relax \csname lgwellbigb\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigb {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigb \fi ”]
[$L_C \overset{tex}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_C \else

```
\if \relax \csname lgwellbigc\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigc {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigc \fi "]
    [L_D ^{tex} =  "
\if \relax \csname lgwprooflinep\endcsname L_D \else
\if \relax \csname lgwellbigd\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigd {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigd \fi "]
    [L_E ^{tex} =  "
\if \relax \csname lgwprooflinep\endcsname L_E \else
\if \relax \csname lgwellbige\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbige {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbige \fi "]
    [L_F ^{tex} =  "
\if \relax \csname lgwprooflinep\endcsname L_F \else
\if \relax \csname lgwellbigf\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigf {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigf \fi "]
    [L_G ^{tex} =  "
\if \relax \csname lgwprooflinep\endcsname L_G \else
\if \relax \csname lgwellbigg\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigg {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigg \fi "]
    [L_H ^{tex} =  "
\if \relax \csname lgwprooflinep\endcsname L_H \else
\if \relax \csname lgwellbigh\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigh {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigh \fi "]
    [L_I ^{tex} =  "
\if \relax \csname lgwprooflinep\endcsname L_I \else
\if \relax \csname lgwellbigi\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigi {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigi \fi "]
    [L_J ^{tex} =  "
\if \relax \csname lgwprooflinep\endcsname L_J \else
\if \relax \csname lgwellbigj\endcsname
\global \advance \lgwproofline by 1
```

```
\xdef \lgwellbigj {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigj \fi "]
    [L_K ^{tex} = "
\if \relax \csname lgwprooflinep\endcsname L_K \else
\if \relax \csname lgwellbigk\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigk {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigk \fi "]
    [L_L ^{tex} = "
\if \relax \csname lgwprooflinep\endcsname L_L \else
\if \relax \csname lgwellbigl\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigl {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigl \fi "]
    [L_M ^{tex} = "
\if \relax \csname lgwprooflinep\endcsname L_M \else
\if \relax \csname lgwellbigm\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigm {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigm \fi "]
    [L_N ^{tex} = "
\if \relax \csname lgwprooflinep\endcsname L_N \else
\if \relax \csname lgwellbign\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbign {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbign \fi "]
    [L_O ^{tex} = "
\if \relax \csname lgwprooflinep\endcsname L_O \else
\if \relax \csname lgwellbigo\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigo {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigo \fi "]
    [L_P ^{tex} = "
\if \relax \csname lgwprooflinep\endcsname L_P \else
\if \relax \csname lgwellbigp\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigp {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigp \fi "]
    [L_Q ^{tex} = "
\if \relax \csname lgwprooflinep\endcsname L_Q \else
\if \relax \csname lgwellbigq\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigq {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigq \fi "]
```

```
    [L_R ^{tex} =“
\if \relax \csname lgwprooflinep\endcsname L_R \else
\if \relax \csname lgwellbigr\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigr {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigr \fi ”]
    [L_S ^{tex} =“
\if \relax \csname lgwprooflinep\endcsname L_S \else
\if \relax \csname lgwellbigs\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigs {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigs \fi ”]
    [L_T ^{tex} =“
\if \relax \csname lgwprooflinep\endcsname L_T \else
\if \relax \csname lgwellbigt\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigt {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigt \fi ”]
    [L_U ^{tex} =“
\if \relax \csname lgwprooflinep\endcsname L_U \else
\if \relax \csname lgwellbigu\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigu {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigu \fi ”]
    [L_V ^{tex} =“
\if \relax \csname lgwprooflinep\endcsname L_V \else
\if \relax \csname lgwellbigv\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigv {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigv \fi ”]
    [L_W ^{tex} =“
\if \relax \csname lgwprooflinep\endcsname L_W \else
\if \relax \csname lgwellbigw\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigw {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigw \fi ”]
    [L_X ^{tex} =“
\if \relax \csname lgwprooflinep\endcsname L_X \else
\if \relax \csname lgwellbigx\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigx {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigx \fi ”]
    [L_Y ^{tex} =“
\if \relax \csname lgwprooflinep\endcsname L_Y \else
```

\if \relax \csname lgwellbigy\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigy {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigy \fi ”]
[$\mathrm{L_Z} \overset{\text{tex}}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_Z \else
\if \relax \csname lgwellbigz\endcsname
\global \advance \lgwproofline by 1
\xdef \lgwellbigz {L\ifnum \lgwproofline <10 0\fi \number \lgwproofline }
\fi \lgwellbigz \fi ”]
[$\mathrm{L_?} \overset{\text{tex}}{=}$ “
\if \relax \csname lgwprooflinep\endcsname L_? \else
\global \advance \lgwproofline by 1
L\ifnum \lgwproofline <10 0\fi \number \lgwproofline
\fi ”]

B TEX definitions

[$\mathrm{parm}(\mathsf{t},\mathsf{s},\mathsf{n}) \overset{\text{tex}}{=}$ “
parm(#1.
,#2.
,#3.
)”]

[$\mathrm{parm}^*(\mathsf{t},\mathsf{s},\mathsf{n}) \overset{\text{tex}}{=}$ “
parm^*(#1.
,#2.
,#3.
)”]

[$\mathrm{inst}(\mathsf{t},\mathsf{s}) \overset{\text{tex}}{=}$ “
inst(#1.
,#2.
)”]

[$\mathrm{inst}^*(\mathsf{t},\mathsf{s}) \overset{\text{tex}}{=}$ “
inst^*(#1.
,#2.
)”]

[$\mathrm{occur}(\mathsf{t},\mathsf{u}) \overset{\text{tex}}{=}$ “
occur(#1.
,#2.
)”]

[$\mathrm{occur}^*(\mathsf{t},\mathsf{u})$ $\stackrel{\text{tex}}{=}$ “
occur^*(#1.
,#2.
)”]

[$\mathrm{circular}(\mathsf{t}=\mathsf{u},\mathsf{s})$ $\stackrel{\text{tex}}{=}$ “
circular(#1.
=#2.
,#3.
)”]

[$\mathrm{circular}^*(\mathsf{t}=\mathsf{u},\mathsf{s})$ $\stackrel{\text{tex}}{=}$ “
circular^*(#1.
=#2.
,#3.
)”]

[$\mathrm{unify}(\mathsf{t}=\mathsf{u},\mathsf{s})$ $\stackrel{\text{tex}}{=}$ “
unify(#1.
=#2.
,#3.
)”]

[$\mathrm{unify}^*(\mathsf{t}=\mathsf{u},\mathsf{s})$ $\stackrel{\text{tex}}{=}$ “
unify^*(#1.
=#2.
,#3.
)”]

[$\mathrm{unify}_2(\mathsf{t}=\mathsf{u},\mathsf{s})$ $\stackrel{\text{tex}}{=}$ “
unify_2(#1.
=#2.
,#3.
)”]

[Reflexivity $\stackrel{\text{tex}}{=}$ “
Reflexivity”]

[Commutativity $\stackrel{\text{tex}}{=}$ “
Commutativity”]

[$\mathrm{Reflexivity}_1$ $\stackrel{\text{tex}}{=}$ “
Reflexivity_1”]

[<tactic> $\stackrel{\text{tex}}{=}$ “
{<}tactic{>}”]

$[\mathsf{tactic} \overset{\text{tex}}{=} \text{“}$
tactic”]

$[[\mathsf{x} \overset{\mathsf{tactic}}{=} \mathsf{y}] \overset{\text{tex}}{=} \text{“}$
[#1/tex name/tex.
\stackrel {tactic}{=}#2.
]”]

$[\mathcal{P}(\mathsf{t}, \mathsf{s}, \mathsf{c}) \overset{\text{tex}}{=} \text{“}$
{\cal P}(#1.
, #2.
, #3.
)”]

$[\mathcal{P}^*(\mathsf{t}, \mathsf{s}, \mathsf{c}) \overset{\text{tex}}{=} \text{“}$
{\cal P}^*(#1.
, #2.
, #3.
)”]

$[\mathsf{p}_0 \overset{\text{tex}}{=} \text{“}$
p_0”]

$[\mathsf{x} \gg \mathsf{y} \overset{\text{tex}}{=} \text{“}$#1.
\gg #2.”]

$[\mathrm{conclude}_1(\mathsf{t}, \mathsf{c}) \overset{\text{tex}}{=} \text{“}$
conclude_1 (#1.
, #2.
)”]

$[\mathrm{conclude}_2(\mathsf{a}, \mathsf{t}, \mathsf{c}) \overset{\text{tex}}{=} \text{“}$
conclude_2 (#1.
, #2.
, #3.
)”]

$[\mathrm{conclude}_3(\mathsf{a}, \mathsf{t}, \mathsf{l}, \mathsf{s}) \overset{\text{tex}}{=} \text{“}$
conclude_3 (#1.
, #2.
, #3.
, #4.
)”]

$[\mathsf{x} \rhd \mathsf{y} \overset{\text{tex}}{=} \text{“}$#1.
\rhd #2.”]

```
[x ⧐ y ≝tex "#1.
        \mathrel {\makebox [0mm][l]{$\rhd $}\, {\rhd }} #2."]

[t proof of s : p ≝tex "
        \if\relax\csname lgwprooflinep\endcsname
        \def\lgwprooflinep{x}
        \newcount\lgwproofline
        \fi
        \begingroup
        \def\insideproof{x}
        \lgwproofline=0 #1.
        \mathbf {\ proof\ of\ } #2.
        \colon #3.
        \gdef\lgwella{\relax}
        \gdef\lgwellb{\relax}
        \gdef\lgwellc{\relax}
        \gdef\lgwelld{\relax}
        \gdef\lgwelle{\relax}
        \gdef\lgwellf{\relax}
        \gdef\lgwellg{\relax}
        \gdef\lgwellh{\relax}
        \gdef\lgwelli{\relax}
        \gdef\lgwellj{\relax}
        \gdef\lgwellk{\relax}
        \gdef\lgwelll{\relax}
        \gdef\lgwellm{\relax}
        \gdef\lgwelln{\relax}
        \gdef\lgwello{\relax}
        \gdef\lgwellp{\relax}
        \gdef\lgwellq{\relax}
        \gdef\lgwellr{\relax}
        \gdef\lgwells{\relax}
        \gdef\lgwellt{\relax}
        \gdef\lgwellu{\relax}
        \gdef\lgwellv{\relax}
        \gdef\lgwellw{\relax}
        \gdef\lgwellx{\relax}
        \gdef\lgwelly{\relax}
        \gdef\lgwellz{\relax}
        \gdef\lgwellbiga{\relax}
        \gdef\lgwellbigb{\relax}
        \gdef\lgwellbigc{\relax}
        \gdef\lgwellbigd{\relax}
        \gdef\lgwellbige{\relax}
        \gdef\lgwellbigf{\relax}
        \gdef\lgwellbigg{\relax}
```

```
\gdef\lgwellbigh{\relax}
\gdef\lgwellbigi{\relax}
\gdef\lgwellbigj{\relax}
\gdef\lgwellbigk{\relax}
\gdef\lgwellbigl{\relax}
\gdef\lgwellbigm{\relax}
\gdef\lgwellbign{\relax}
\gdef\lgwellbigo{\relax}
\gdef\lgwellbigp{\relax}
\gdef\lgwellbigq{\relax}
\gdef\lgwellbigr{\relax}
\gdef\lgwellbigs{\relax}
\gdef\lgwellbigt{\relax}
\gdef\lgwellbigu{\relax}
\gdef\lgwellbigv{\relax}
\gdef\lgwellbigw{\relax}
\gdef\lgwellbigx{\relax}
\gdef\lgwellbigy{\relax}
\gdef\lgwellbigz{\relax}
\endgroup "]

[t proof of s : p ^name= "#1.
\mathbf{\ proof\ of\ } #2.
: #3."]

[Line l : a ≫ i; p ^tex= "
\newline \makebox [0.1\textwidth]{}%
\parbox [b]{0.4\textwidth }{\raggedright
\setlength {\parindent }{-0.1\textwidth }%
\makebox [0.1\textwidth ][l]{$#1.
$:}$#2.
{}\gg {}$}\quad
\parbox [t]{0.4\textwidth }{$#3.
$\hfill \makebox [0mm][l]{\quad ;}}#4."]

[Line l : a ≫ i; p ^name= "
Line \, #1.
: #2.
\gg #3.
; #4."]

[Last line a ≫ i ^tex= "
\newline \makebox [0.1\textwidth]{}%
\parbox [b]{0.4\textwidth }{\raggedright
\setlength {\parindent }{-0.1\textwidth }%
\makebox [0.1\textwidth ][l]{$
\if \relax \csname lgwprooflinep\endcsname L_? \else
```

```
\global \advance \lgwproofline by 1
L\ifnum \lgwproofline <10 0\fi \number \lgwproofline
\fi
$:}$#1.
{}\gg {}$}\quad
\parbox [t]{0.4\textwidth }{$#2.
$\hfill \makebox [0mm][l]{\quad \makebox[0mm]{$\Box$}}}"]

[Last line a ≫ i ≝ "
Last\ line \, #1.
\gg #2."]

[Line l : Premise ≫ i; p ≝ "
\newline \makebox [0.1\textwidth ][l]{$#1.
$:}\makebox [0.4\textwidth ][l]{$Premise{}\gg{}$}\quad
\parbox [t]{0.4\textwidth }{$#2.
$\hfill \makebox [0mm][l]{\quad ; }}#3."]

[Line l : Premise ≫ i; p ≛ "
Line \, #1.
: Premise \gg #2.
; #3."]

[Line l : Side-condition ≫ i; p ≝ "
\newline \makebox [0.1\textwidth ][l]{$#1.
$:}\makebox [0.4\textwidth ][l]{
$\mbox{Side-condition}{}\gg{}$}\quad
\parbox [t]{0.4\textwidth }{$#2.
$\hfill \makebox [0mm][l]{\quad ; }}#3."]

[Line l : Side-condition ≫ i; p ≛ "
Line \, #1.
: \mbox{Side-condition} \gg #2.
; #3."]

[Arbitrary ≫ i; p ≝ "
\newline \makebox [0.1\textwidth ][l]{$
\if \relax \csname lgwprooflinep\endcsname L_? \else
\global \advance \lgwproofline by 1
L\ifnum \lgwproofline <10 0\fi \number \lgwproofline
\fi
$:}\makebox [0.4\textwidth ][l]{$Arbitrary{}\gg{}$}\quad
\parbox [t]{0.4\textwidth }{$#1.
$\hfill \makebox [0mm][l]{\quad ; }}#2."]

[Arbitrary ≫ i; p ≛ "
Arbitrary \gg #1.
; #2."]
```

$[\mathrm{Local} \gg \mathsf{a} = \mathsf{i}; \mathsf{p} \overset{\mathrm{tex}}{=} \text{“}$
\newline\makebox[0.1\textwidth]{$
\if \relax \csname lgwprooflinep\endcsname L_? \else
\global \advance \lgwproofline by 1
L\ifnum \lgwproofline <10 0\fi \number \lgwproofline
\fi
$:}%
\makebox[0.4\textwidth]{$Local{}\gg{}$}%
\quad%
\parbox[t]{0.4\textwidth}{$#1.
= #2.
$\hfill\makebox[0mm][l]{\quad ;}}#3.”]

$[\mathrm{Local} \gg \mathsf{a} = \mathsf{i}; \mathsf{p} \overset{\mathrm{name}}{=} \text{“}$
Local \gg #1.
= #2.
; #3.”]

C Test

Test cases are listed here. To avoid TEX errors about missing items, a trivial test has been included.

$[\mathrm{inst}(\mathrm{parm}(\lceil \forall \mathsf{x}\colon \forall \mathsf{y}\colon \mathsf{x} + \mathsf{y} \rceil, \mathsf{T}, 4), \mathsf{T}[4 \to \lceil \mathsf{u} \rceil][8 \to \lceil \mathsf{v} \rceil]) \overset{\mathrm{t}}{=} \lceil \forall \mathsf{u}\colon \forall \mathsf{v}\colon \mathsf{u} + \mathsf{v} \rceil]^{\cdot}$

math test occur eight in parameter term quote not all var x indeed var x end quote end occur end test end math

math false test occur nine in parameter term quote not all var x indeed var x end quote end occur end test end math

$[\mathrm{unify}(\lceil 2 \rceil = \lceil 2 \rceil, \mathsf{T})]^{\cdot}$

$[\mathrm{unify}(\lceil 2 \rceil = \lceil 3 \rceil, \mathsf{T}) \approx 0]^{\cdot}$

$[\mathrm{unify}(\lceil 2 + 3 \rceil = \lceil 2 + 3 \rceil, \mathsf{T})]^{\cdot}$

$[\mathrm{unify}(\lceil 2 + 3 \rceil = \lceil 2 + 4 \rceil, \mathsf{T}) \approx 0]^{\cdot}$

$[\mathrm{unify}(\lceil 2 \rceil = 3, \mathsf{T})[3] \overset{\mathrm{t}}{=} \lceil 2 \rceil]^{\cdot}$

$[\mathrm{unify}(3 = \lceil 2 \rceil, \mathsf{T})[3] \overset{\mathrm{t}}{=} \lceil 2 \rceil]^{\cdot}$

$[\mathrm{unify}(\langle \lceil \mathsf{x} + \mathsf{y} \rceil^{\mathrm{R}}, 1, \lceil 3 \rceil \rangle = \lceil 2 + 3 \rceil, \mathsf{T})[1] \overset{\mathrm{t}}{=} \lceil 2 \rceil]^{\cdot}$

$[\mathrm{unify}(\langle \lceil \mathsf{x} + \mathsf{y} \rceil^{\mathrm{R}}, 1, \lceil 3 \rceil \rangle = \langle \lceil \mathsf{x} + \mathsf{y} \rceil^{\mathrm{R}}, \lceil 2 \rceil, 2 \rangle, \mathsf{T})[1] \overset{\mathrm{t}}{=} \lceil 2 \rceil]^{\cdot}$

$[\mathrm{unify}(\mathrm{parm}(\lceil \mathcal{A} + 3 \rceil, \langle \lceil \mathcal{A} \rceil :: 1 \rangle, 1) = \mathrm{parm}(\lceil 2 + \mathcal{B} \rceil, \langle \lceil \mathcal{B} \rceil :: 2 \rangle, 1), \mathsf{T})[1] \overset{\mathrm{t}}{=} \lceil 2 \rceil]^{\cdot}$

$[\mathrm{unify}(\mathrm{parm}(\lceil \mathcal{A} + 3 \rceil, \langle \lceil \mathcal{A} \rceil :: 1 \rangle, 1) = \mathrm{parm}(\lceil 2 + \mathcal{B} \rceil, \langle \lceil \mathcal{B} \rceil :: 2 \rangle, 1), \mathsf{T})[2] \stackrel{\mathrm{t}}{=} \lceil 3 \rceil]^{\cdot}$

$[\mathrm{inst}(\mathrm{parm}(\lceil \mathcal{A} + \mathcal{B} \rceil, \langle \lceil \mathcal{A} \rceil :: 1, \lceil \mathcal{B} \rceil :: 2 \rangle, 1), \mathrm{unify}(\mathrm{parm}(\lceil \mathcal{A} + 3 \rceil, \langle \lceil \mathcal{A} \rceil :: 1 \rangle, 1) = \mathrm{parm}(\lceil 2 + \mathcal{B} \rceil, \langle \lceil \mathcal{B} \rceil :: 2 \rangle, 1), \mathsf{T})) \stackrel{\mathrm{t}}{=} \lceil 2 + 3 \rceil]^{\cdot}$

$[\mathsf{T}]^{\cdot}$

D Priority table

Priority table
Preassociative
[check], [base], [bracket $*$ end bracket], [big bracket $*$ end bracket], [math $*$ end math], [**flush left** $[*]$], [$\underline{\mathsf{x}}$], [$\underline{\mathsf{y}}$], [$\underline{\mathsf{z}}$], $[[* \bowtie *]]$, $[[* \stackrel{*}{\rightarrow} *]]$, [pyk], [tex], [name], [prio], $[*]$, [T], [if$(*,*,*)$], $[[* \stackrel{*}{\Rightarrow} *]]$, [val], [claim], [$\underline{\perp}$], [f$(*)$], $[(*)^{\mathrm{I}}]$, [F], [$\underline{0}$], [$\underline{1}$], [$\underline{2}$], [$\underline{3}$], [$\underline{4}$], [$\underline{5}$], [$\underline{6}$], [$\underline{7}$], [$\underline{8}$], [$\underline{9}$], [0], [1], [2], [3], [4], [5], [6], [7], [8], [9], [a], [b], [c], [d], [e], [f], [g], [h], [i], [j], [k], [l], [m], [n], [o], [p], [q], [r], [s], [t], [u], [v], [w], $[(*)^{\mathrm{M}}]$, [If$(*,*,*)$], [array$\{*\}$ $*$ end array], [l], [c], [r], [empty], $[\langle * \mid * := * \rangle]$, $[\mathcal{M}(*)]$, $[\tilde{\mathcal{U}}(*)]$, $[\mathcal{U}(*)]$, $[\mathcal{U}^{\mathrm{M}}(*)]$, [**apply**$(*,*)$], [**apply**$_1(*,*)$], [identifier$(*)$], [identifier$_1(*,*)$], [array-plus$(*,*)$], [array-remove$(*,*,*)$], [array-put$(*,*,*,*)$], [array-add$(*,*,*,*,*)$], [bit$(*,*)$], [bit$_1(*,*)$], [rack], ["vector"], ["bibliography"], ["dictionary"], ["body"], ["codex"], ["expansion"], ["code"], ["cache"], ["diagnose"], ["pyk"], ["tex"], ["texname"], ["value"], ["message"], ["macro"], ["definition"], ["unpack"], ["claim"], ["priority"], ["lambda"], ["apply"], ["true"], ["if"], ["quote"], ["proclaim"], ["define"], ["introduce"], ["hide"], ["pre"], ["post"], $[\mathcal{E}(*,*,*)]$, $[\mathcal{E}_2(*,*,*,*,*)]$, $[\mathcal{E}_3(*,*,*,*)]$, $[\mathcal{E}_4(*,*,*,*,*)]$, [**lookup**$(*,*,*)$], [**abstract**$(*,*,*,*)$], $[[*]]$, $[\mathcal{M}(*,*,*)]$, $[\mathcal{M}_2(*,*,*,*)]$, $[\mathcal{M}^*(*,*,*)]$, [macro], [s$_0$], [**zip**$(*,*)$], [**assoc**$_1(*,*,*)$], $[(*)^{\mathbf{P}}]$, [self], $[[* \doteq\!\doteq *]]$, $[[* \doteq *]]$, $[[* \doteqdot *]]$, $[[* \stackrel{\mathrm{pyk}}{=} *]]$, $[[* \stackrel{\mathrm{tex}}{=} *]]$, $[[* \stackrel{\mathrm{name}}{=} *]]$, [**Priority table**$[*]$], $[\tilde{\mathcal{M}}_1]$, $[\tilde{\mathcal{M}}_2(*)]$, $[\tilde{\mathcal{M}}_3(*)]$, $[\tilde{\mathcal{M}}_4(*,*,*,*)]$, $[\tilde{\mathcal{M}}(*,*,*)]$, $[\tilde{\mathcal{Q}}(*,*,*)]$, $[\tilde{\mathcal{Q}}_2(*,*,*)]$, $[\tilde{\mathcal{Q}}_3(*,*,*,*)]$, $[\tilde{\mathcal{Q}}^*(*,*,*)]$, $[(*)]$, [**aspect**$(*,*)$], [**aspect**$(*,*,*)$], $[\langle * \rangle]$, [**tuple**$_1(*)$], [**tuple**$_2(*)$], [let$_2(*,*)$], [let$_1(*,*)$], $[[* \stackrel{\mathrm{claim}}{=} *]]$, [checker], [**check**$(*,*)$], [**check**$_2(*,*,*)$], [**check**$_3(*,*,*)$], [**check**$^*(*,*)$], [**check**$^*_2(*,*,*)$], $[[*]^{\cdot}]$, $[[*]^{-}]$, $[[*]^{\circ}]$, [msg], $[[* \stackrel{\mathrm{msg}}{=} *]]$, [<stmt>], [stmt], $[[* \stackrel{\mathrm{stmt}}{=} *]]$, [HeadNil$'$], [HeadPair$'$], [Transitivity$'$], [$\underline{\perp\!\!\perp}$], [Contra$'$], [T$'_{\mathrm{E}}$], [L$_1$], [$\underline{*}$], $[\mathcal{A}]$, $[\mathcal{B}]$, $[\mathcal{C}]$, $[\mathcal{D}]$, $[\mathcal{E}]$, $[\mathcal{F}]$, $[\mathcal{G}]$, $[\mathcal{H}]$, $[\mathcal{I}]$, $[\mathcal{J}]$, $[\mathcal{K}]$, $[\mathcal{L}]$, $[\mathcal{M}]$, $[\mathcal{N}]$, $[\mathcal{O}]$, $[\mathcal{P}]$, $[\mathcal{Q}]$, $[\mathcal{R}]$, $[\mathcal{S}]$, $[\mathcal{T}]$, $[\mathcal{U}]$, $[\mathcal{V}]$, $[\mathcal{W}]$, $[\mathcal{X}]$, $[\mathcal{Y}]$, $[\mathcal{Z}]$, $[\langle * \mid * := * \rangle]$, $[\langle {}^* * \mid * := * \rangle]$, $[\emptyset]$, [Remainder], $[(*)^{\mathbf{v}}]$, [error$(*,*)$], [error$_2(*,*)$], [proof$(*,*,*)$], [proof$_2(*,*)$], $[\mathcal{S}(*,*)]$, $[\mathcal{S}^{\mathrm{I}}(*,*)]$, $[\mathcal{S}^{\triangleright}(*,*)]$, $[\mathcal{S}^{\triangleright}_1(*,*,*)]$, $[\mathcal{S}^{\mathrm{E}}(*,*)]$, $[\mathcal{S}^{\mathrm{E}}_1(*,*,*)]$, $[\mathcal{S}^{+}(*,*)]$, $[\mathcal{S}^{+}_1(*,*,*)]$, $[\mathcal{S}^{-}(*,*)]$, $[\mathcal{S}^{-}_1(*,*,*)]$, $[\mathcal{S}^{*}(*,*)]$, $[\mathcal{S}^{*}_1(*,*,*)]$, $[\mathcal{S}^{*}_2(*,*,*,*)]$, $[\mathcal{S}^{@}(*,*)]$, $[\mathcal{S}^{@}_1(*,*,*)]$, $[\mathcal{S}^{\vdash}(*,*)]$, $[\mathcal{S}^{\vdash}_1(*,*,*,*)]$, $[\mathcal{S}^{\Vdash}(*,*)]$, $[\mathcal{S}^{\Vdash}_1(*,*,*,*)]$, $[\mathcal{S}^{\mathrm{i.e.}\cdot}(*,*)]$, $[\mathcal{S}^{\mathrm{i.e.}\cdot}_1(*,*,*,*)]$, $[\mathcal{S}^{\mathrm{i.e.}\cdot}_2(*,*,*,*,*)]$, $[\mathcal{S}^{\forall}(*,*)]$, $[\mathcal{S}^{\forall}_1(*,*,*,*)]$, $[\mathcal{S}^{\cdot}(*,*)]$, $[\mathcal{S}^{\cdot}_1(*,*,*)]$, $[\mathcal{S}^{\cdot}_2(*,*,*,*)]$, $[\mathcal{T}(*)]$, [claims$(*,*,*)$], [claims$_2(*,*,*)$], [<proof>], [proof], $[[$**Lemma** $*\colon *]]$, $[[$**Proof of** $*\colon *]]$, $[[*$ **lemma** $*\colon *]]$, $[[*$ **antilemma** $*\colon *]]$, $[[*$ **rule** $*\colon *]]$, $[[*$ **antirule** $*\colon *]]$, [verifier], $[\mathcal{V}_1(*)]$, $[\mathcal{V}_2(*,*)]$, $[\mathcal{V}_3(*,*,*,*)]$, $[\mathcal{V}_4(*,*)]$, $[\mathcal{V}_5(*,*,*,*)]$, $[\mathcal{V}_6(*,*,*,*)]$, $[\mathcal{V}_7(*,*,*,*)]$,

[Cut(∗, ∗)], [Head$_{\oplus}$(∗)], [Tail$_{\oplus}$(∗)], [rule$_1$(∗, ∗)], [rule(∗, ∗)], [Rule tactic], [Plus(∗, ∗)], [[**Theory** ∗]], [theory$_2$(∗, ∗)], [theory$_3$(∗, ∗)], [theory$_4$(∗, ∗, ∗)], [HeadNil″], [HeadPair″], [Transitivity″], [Contra″], [HeadNil], [HeadPair], [Transitivity], [Contra], [T$_{\mathrm{E}}$], [ragged right], [ragged right expansion], [color(∗ : ∗)], [color*(∗ : ∗)], [vars(∗)], [vars*(∗)], [instantiate(∗ + ∗ : ∗)], [instantiate*(∗ + ∗ : ∗)], [unify(∗ : ∗ = ∗ : ∗ + ∗)], [unify*(∗ : ∗ = ∗ : ∗ + ∗)], [unify$_1$(∗ : ∗ = ∗ : ∗ + ∗)], [unify$_2$(∗ + ∗)], [unify$_3$(∗ = ∗ + ∗)], [parm(∗, ∗, ∗)], [parm*(∗, ∗, ∗)], [inst(∗, ∗)], [inst*(∗, ∗)], [occur(∗, ∗)], [occur*(∗, ∗)], [circular(∗ = ∗, ∗)], [circular*(∗ = ∗, ∗)], [unify(∗ = ∗, ∗)], [unify*(∗ = ∗, ∗)], [unify$_2$(∗ = ∗, ∗)], [L$_{\mathrm{a}}$], [L$_{\mathrm{b}}$], [L$_{\mathrm{c}}$], [L$_{\mathrm{d}}$], [L$_{\mathrm{e}}$], [L$_{\mathrm{f}}$], [L$_{\mathrm{g}}$], [L$_{\mathrm{h}}$], [L$_{\mathrm{i}}$], [L$_{\mathrm{j}}$], [L$_{\mathrm{k}}$], [L$_{\mathrm{l}}$], [L$_{\mathrm{m}}$], [L$_{\mathrm{n}}$], [L$_{\mathrm{o}}$], [L$_{\mathrm{p}}$], [L$_{\mathrm{q}}$], [L$_{\mathrm{r}}$], [L$_{\mathrm{s}}$], [L$_{\mathrm{t}}$], [L$_{\mathrm{u}}$], [L$_{\mathrm{v}}$], [L$_{\mathrm{w}}$], [L$_{\mathrm{x}}$], [L$_{\mathrm{y}}$], [L$_{\mathrm{z}}$], [L$_{\mathrm{A}}$], [L$_{\mathrm{B}}$], [L$_{\mathrm{C}}$], [L$_{\mathrm{D}}$], [L$_{\mathrm{E}}$], [L$_{\mathrm{F}}$], [L$_{\mathrm{G}}$], [L$_{\mathrm{H}}$], [L$_{\mathrm{I}}$], [L$_{\mathrm{J}}$], [L$_{\mathrm{K}}$], [L$_{\mathrm{L}}$], [L$_{\mathrm{M}}$], [L$_{\mathrm{N}}$], [L$_{\mathrm{O}}$], [L$_{\mathrm{P}}$], [L$_{\mathrm{Q}}$], [L$_{\mathrm{R}}$], [L$_{\mathrm{S}}$], [L$_{\mathrm{T}}$], [L$_{\mathrm{U}}$], [L$_{\mathrm{V}}$], [L$_{\mathrm{W}}$], [L$_{\mathrm{X}}$], [L$_{\mathrm{Y}}$], [L$_{\mathrm{Z}}$], [L$_?$], [Reflexivity], [Reflexivity$_1$], [Commutativity], [<tactic>], [tactic], [[∗ $\overset{\text{tactic}}{=}$ ∗]], [$\mathcal{P}$(∗, ∗, ∗)], [$\mathcal{P}^*$(∗, ∗, ∗)], [p$_0$], [conclude$_1$(∗, ∗)], [conclude$_2$(∗, ∗, ∗)], [conclude$_3$(∗, ∗, ∗, ∗)];

Preassociative

[∗-{∗}], [∗′], [∗[∗]], [∗[∗→∗]], [∗[∗⇒∗]];

Preassociative

[“ ∗ ”], [], [(∗)$^{\mathbf{t}}$], [string(∗) + ∗], [string(∗) ++ ∗], [∗], [∗], [!∗], [”∗], [#∗], [$∗], [%∗], [&∗], [’∗], [(∗], [)∗], [∗∗], [+∗], [, ∗], [-∗], [.∗], [/∗], [0∗], [1∗], [2∗], [3∗], [4∗], [5∗], [6∗], [7∗], [8∗], [9∗], [:∗], [; ∗], [<∗], [=∗], [>∗], [?∗], [@∗], [A∗], [B∗], [C∗], [D∗], [E∗], [F∗], [G∗], [H∗], [I∗], [J∗], [K∗], [L∗], [M∗], [N∗], [O∗], [P∗], [Q∗], [R∗], [S∗], [T∗], [U∗], [V∗], [W∗], [X∗], [Y∗], [Z∗], [[∗], [\∗], []∗], [ˆ∗], [_∗], [‘∗], [a∗], [b∗], [c∗], [d∗], [e∗], [f∗], [g∗], [h∗], [i∗], [j∗], [k∗], [l∗], [m∗], [n∗], [o∗], [p∗], [q∗], [r∗], [s∗], [t∗], [u∗], [v∗], [w∗], [x∗], [y∗], [z∗], [{∗], [|∗], [}∗], [˜∗], [**Preassociative** ∗; ∗], [**Postassociative** ∗; ∗], [[∗], ∗], [priority ∗ end], [newline ∗], [macro newline ∗];

Preassociative

[∗0], [∗1], [0b], [∗-color(∗)], [∗-color*(∗)];

Preassociative

[∗ ’ ∗], [∗ ‘ ∗];

Preassociative

[∗$^{\mathrm{H}}$], [∗$^{\mathrm{T}}$], [∗$^{\mathrm{U}}$], [∗$^{\mathrm{h}}$], [∗$^{\mathrm{t}}$], [∗$^{\mathrm{s}}$], [∗$^{\mathrm{c}}$], [∗$^{\mathrm{d}}$], [∗$^{\mathrm{a}}$], [∗$^{\mathrm{C}}$], [∗$^{\mathrm{M}}$], [∗$^{\mathrm{B}}$], [∗$^{\mathrm{r}}$], [∗$^{\mathrm{i}}$], [∗$^{\mathrm{d}}$], [∗$^{\mathrm{R}}$], [∗^{0}], [∗^{1}], [∗^{2}], [∗^{3}], [∗^{4}], [∗^{5}], [∗^{6}], [∗^{7}], [∗^{8}], [∗^{9}], [∗$^{\mathrm{E}}$], [∗$^{\mathcal{V}}$], [∗$^{\mathcal{C}}$], [∗$^{\mathcal{C}^*}$];

Preassociative

[∗ · ∗], [∗ ·$_0$ ∗];

Preassociative

[∗ + ∗], [∗ +$_0$ ∗], [∗ +$_1$ ∗], [∗ − ∗], [∗ −$_0$ ∗], [∗ −$_1$ ∗];

Preassociative

[∗ ∪ {∗}], [∗ ∪ ∗], [∗\{∗}];

Postassociative

[∗ ∴ ∗], [∗ $\underline{\therefore}$ ∗], [∗ $\underline{::}$ ∗], [∗ $\underline{+2*}$ ∗], [∗ : : ∗], [∗ +2∗ ∗];

Postassociative

[∗, ∗];

Preassociative

$[* \overset{\mathrm{B}}{\approx} *], [* \overset{\mathrm{D}}{\approx} *], [* \overset{\mathrm{C}}{\approx} *], [* \overset{\mathrm{P}}{\approx} *], [* \approx *], [* = *], [* \overset{+}{\to} *], [* \overset{\mathrm{t}}{=} *], [* \overset{\mathrm{t}^*}{=} *], [* \overset{\mathrm{r}}{=} *],$
$[* \in_{\mathrm{t}} *], [* \subseteq_{\mathrm{T}} *], [* \overset{\mathrm{T}}{=} *], [* \overset{\mathrm{s}}{=} *],$ [* free in *], [* free in* *], [* free for * in *],
[* free for* * in *], $[* \in_{\mathrm{c}} *], [* < *], [* <' *], [* \leq' *];$
Preassociative
$[\neg *];$
Preassociative
$[* \wedge *], [* \ddot{\wedge} *], [* \tilde{\wedge} *], [* \wedge_{\mathrm{c}} *];$
Preassociative
$[* \vee *], [* \parallel *], [* \ddot{\vee} *];$
Postassociative
$[* \Rrightarrow *];$
Postassociative
$[* : *], [*!*];$
Preassociative
$[* \begin{cases} * \\ * \end{cases}];$
Preassociative
$[\lambda * .*], [\Lambda *],$ [**if** * **then** * **else** *], [**let** * = * **in** *], [**let** * $\ddot{=}$ * in *];
Preassociative
$[*^{\mathrm{I}}], [*^{\triangleright}], [*^{\mathrm{V}}], [*^{+}], [*^{-}], [*^{*}];$
Preassociative
$[* @ *], [* \triangleright *], [* \Rrightarrow *], [* \gg *];$
Postassociative
$[* \vdash *], [* \Vdash *],$ [* i.e. *];
Preassociative
$[\forall *: *];$
Postassociative
$[* \oplus *];$
Postassociative
$[*; *];$
Preassociative
[* proves *];
Preassociative
[* **proof of** * : *], [Line * : * $\gg$ *; *], [Last line * $\gg$ *], [Line * : Premise $\gg$ *; *],
[Line * : Side-condition $\gg$ *; *], [Arbitrary $\gg$ *; *], [Local $\gg$ * = *; *];
Postassociative
[* then *], [*[*]*];
Preassociative
[*&*];
Preassociative
[**]; **End table**

E Index

F Bibliography

[1] K. Grue. Logiweb. In Fairouz Kamareddine, editor, *Mathematical Knowledge Management Symposium 2003*, volume 93 of *Electronic Notes in Theoretical Computer Science*, pages 70–101. Elsevier, 2004.

[2] C. P. Wadsworth M. J. Gordon, A. J. Milner. *Edinburgh LCF, A mechanised logic of computation*, volume 78 of *Lecture Notes in Computer Science*. Springer-Verlag, 1979.