TPS3 Facilities Guide for Users

Peter Andrews
Sunil Issar
Dan Nesmith
Frank Pfenning
Hongwei Xi
Matthew Bishop
Chad E. Brown
Rémy Chrétien

August 9, 2011

This material is based upon work supported by NSF grants MCS81-02870, DCR-8402532, CCR-8702699, CCR-9002546, CCR-9201893, CCR-9502878, CCR-9624683, CCR-9732312, CCR-0097179, and a grant from the Center for Design of Educational Computing, Carnegie Mellon University. Any opinions, findings, and conclusions or recommendations are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Contents

Chapter 1

Introduction

This document is a short version of the TPS3 Facilities Guide, and lists TPScommands, flags, etc., excluding those categories which are not useful to the general user. Each chapter lists the members of a TPS category; each chapter is further divided into sections, which group those members of the category by the context in which they were defined.

After each command is listed the arguments it takes (if any). Note that these arguments are not argument types, but rather descriptive identifiers intended to convey the role each argument is playing in the command invocation. The argument types for each command may be obtained by consulting the on-line documentation.

This document is generated automatically by TPS using the `LATEX-DOC` command and, when produced, accurately reflects the current state of the system. All documentation listed in this guide is also available on-line. To produce this document, load the file `latex-facilities-short.lisp`[1] into TPS. The system will produce the LaTeXfile `latex-facilities-short.tex`. You should now run the file `latex-manual-short.tex` through LaTeX, and print the resulting `latex-manual-short.dvi,PS` file.

[1] All files referred to in this chapter are located on the directory doc/facilities.

Chapter 2

Top-Level Commands

The internal name of this category is MEXPR.
A top-level command can be defined using DEFMEXPR. Allowable properties are: `ARGTYPES`, `WFFARGTYPES`, `WFFOP-TYPELIST`, `ARGNAMES`, `ARGHELP`, `DEFAULTFNS`, `MAINFNS`, `ENTERFNS`, `CLOSEFNS`, `PRINT-COMMAND`, `DONT-RESTORE`, `MHELP`.

2.1 Top Levels

BEGIN-PRFW

Begin proofwindow top level. Open Current Subproof, Current Subproof & Line Numbers, and Complete Proof windows with text size determined by the value of the flag CHARSIZE. Printing in various windows can be modified by changing the flags PROOFW-ACTIVE, PROOFW-ALL, PROOFW-ACTIVE+NOS, BLANK-LINES-INSERTED and PRINTLINE-FLAG. The initial size of the windows can be modified with the flags PROOFW-ALL-HEIGHT, PROOFW-ALL-WIDTH, PROOFW-ACTIVE-HEIGHT, PROOFW-ACTIVE-WIDTH, PROOFW-ACTIVE+NOS-HEIGHT, and PROOFW-ACTIVE+NOS-WIDTH; after the windows are open, they can simply be resized as normal. PSTATUS will update the proofwindows manually if necessary. Close the proofwindows with END-PRFW.

DO-GRADES

Invoke the grading package.

ED ***edwff***

Enter the editor on a given wff. Editor windows may be initialized, depending the values of the flags EDWIN-TOP, EDWIN-CURRENT, EDWIN-VPFORM. The flags BLANK-LINES-INSERTED and CHARSIZE determine the layout of these windows. The flags EDWIN-CURRENT,TOP,VPFORM-WIDTH and EDWIN-CURRENT,TOP,VPFORM-HEIGHT determine the intial size of these windows; they may be resized after they are opened in the usual way. WARNING: Since editing is non-destructive, nothing is done with the result of the editing process!

END-PRFW

End proofwindow top level; close all open proofwindows.

EXT-MATE
Enter the EXT-MATE top level for building and manipulating extensional expansion dags (see Chad E. Brown's thesis).

EXT-SEQ
Enter the EXT-SEQ top level for building and manipulating extensional sequent derivations (see Chad E. Brown's thesis).

HISTORY *n reverse*
Show history list. Shows the N most recent events; N defaults to the value of HISTORY-SIZE, showing entire history list. Values of N that are greater than HISTORY-SIZE have the same effect as the default value. REVERSE defaults to NO; if YES, most recent commands will be shown first.

LIB
Enter the library top-level.

See Also: UNIXLIB (an alternative library top level)

MATE *gwff deepen reinit window*
Begin an expansion proof for a gwff.

MODELS
Enter the MODELS top level for working with standard models in which the base types (hence all types) are a power of 2.

MTREE *gwff deepen reset window*
Begin to enter the mating tree top level.

POP
Return from a top level started with PUSH.

PUSH
Start a new top level. This command is almost useless, except from within a prompt (e.g. one can type PUSH in the middle of converting an etree to a ND proof interactively, call SCRIBEPROOF, and then type POP to return to the conversion).

REVIEW
Enter REVIEW to examine and change flags or parameters.

REWRITE *p2 p1 a b p2-hyps p1-hyps*
Rewrite a line of the current natural deduction proof in the REWRITING top level. When finished rewriting, use OK to leave the REWRITING top level, modifying the main proof accordingly.

REWRITE-IN *theory p2 p1 a b p2-hyps p1-hyps*
Rewrite a line in the REWRITING top level using a particular theory.

REWRITING
Enter the REWRITING top level.

TEST ***gwff deepen reinit window***

Enter the test top level. In this top level, the user can search for an optimal mode in which to prove a particular theorem, by defining a list of flags to be varied and then running matingsearch repeatedly with different flag settings. It only works if the value of the flag DEFAULT-MS is one of thse: MS88, MS89, MS90-3, MS90-9, MS91-6, MS91-7, MS92-9, MS93-1, MS98-1, MS03-7 and MS04-2.

UNIFORM-SEARCH ***gwff window mode slist modify***

Enter the test top level to search for any mode that will prove a given theorem. The mode provided by the user should list flag settings that are not to be varied, and the searchlist provided by the user should list all of the flags to be varied. The default settings for the mode and searchlist are UNIFORM-SEARCH-MODE and UNIFORM-SEARCH-2. If you opt for the searchlist to be automatically modified, TPS will inspect the given wff to check whether it is first order, whether it contains any definitions, whether it contains any equalities (and if so whether the LEIBNIZ and ALL instantiations are different), and whether it has any possible primitive substitutions, and will then remove or modify any unnecessary flags from the searchlist (respectively, unification bounds will be deleted, REWRITE-DEFNS will be deleted, REWRITE-EQUALITIES will be deleted or modified, and DEFAULT-MS will be changed to a search without option sets). Also, if you opt for the searchlist to be modified and there is a proof of this theorem in memory, AUTO-SUGGEST will be run and you will be asked whether to modify the searchlist using the results it provides.

After entering the test top level with this command, type GO ! to start searching for a successful mode.

UNIFORM-SEARCH-L ***goal support line-range window mode slist modify***

Enter the test top level to search for any mode that will prove a given lemma. (Compare DIY-L) The mode provided by the user should list flag settings that are not to be varied, and the searchlist provided by the user should list all of the flags to be varied. The default settings for the mode and searchlist are UNIFORM-SEARCH-MODE and UNIFORM-SEARCH-2. If you opt for the searchlist to be automatically modified, TPS will inspect the given wff to check whether it is first order, whether it contains any definitions, whether it contains any equalities (and if so whether the LEIBNIZ and ALL instantiations are different), and whether it has any possible primitive substitutions, and will then remove or modify any unnecessary flags from the searchlist (respectively, unification bounds will be deleted, REWRITE-DEFNS will be deleted, REWRITE-EQUALITIES will be deleted or modified, and DEFAULT-MS will be changed to a search without option sets). After entering the test top level with this command, type GO ! to start searching for a successful mode.

UNIFY

Enter the unification top-level. The user can define disagreement sets using the command ADD-DPAIR available in the unification top-level. If you are entering from the MATE top level, the unification tree associated with the active-mating is passed on to the unification top-level. Any changes made to this tree are destructive. Applicable only for a higher-order unification problem. Uses MS88-style unification.

UNIXLIB

Enter the library top-level with a unix style interface.

The value of the flag CLASS-SCHEME determines what classification scheme is used to determine the virtual directory structure.

If the flag UNIXLIB-SHOWPATH is T, the prompt will be «CLASSS-CHEME>:<PATH TO CLASS><num»

If the flag UNIXLIB-SHOWPATH is NIL, the prompt will be <LIB:<CLASS><num»

See Also: LIB, PSCHEMES, CLASS-SCHEME, UNIXLIB-SHOWPATH, CD, LS, PWD, LN, RM, MKDIR, FETCH, SHOW

2.2 Help

?

Type ? to obtain a list of possible options.

??

Type ?? to get general help on TPS, command completion and history substitution.

ABBREVIATIONS ***show-defns***

This command will list the names of all abbreviations available in TPS.

ENVIRONMENT

Helps to find out about TPS' current environment, i.e. categories of TPS objects, commands, argument types, logical constants, etc.

HELP ***keyword***

Give information about a TPS object like a command or argument type. The amount of help given for inference rules may be changed by setting the flag SHORT-HELP.

Online help can be found at the web site:

http://gtps.math.cmu.edu/tps.html

Typing "?" will show you all available commands at this level.

The web site includes online documentation as well as postscript manuals.

HELP* ***keywords***

Give information about each of a list of TPS objects. This is equivalent to doing HELP on each of them. The amount of help given for inference rules may be changed by setting the flag SHORT-HELP.

HELP-GROUP ***keywords***

Give information about a group of TPS objects; specifically, given the name of a category, a context, or a top level, list the help messages for every object in that class. If given a list of names, it will list the help messages for all the objects that fall into the intersection of these classes (e.g.

HELP-GROUP (MEXPR REWRITING) will show all the top-level commands in the context REWRITING). NOTE: Remember that the name of a context is not necessarily the name that prints on the screen; do HELP CONTEXT to show their real names.

LIST-RULES
List all rules with their suggestion priority.

LIST-RULES*
List all rules with their intermediate rule definition help

OOPS ***position replacement***
Replace the word at a given position in the previous line with another word. Positions start from 0, and the substituted-for command will be entered into the command history list, so for example: <9>HELP GR-FILENAMES <10>OOPS 0 LIST (calls LIST GR-FILENAMES instead) <11>OOPS 1 GR-MISC (calls LIST GR-MISC)

PROBLEMS ***show-defns***
This command will list the names of all exercises available in ETPS.

SEARCH ***phrase search-names***
Look for a key phrase in all help strings (or just all names) of TPS objects. See also KEY, in the review top level (where it searches through the flags) and the library top level (where it searches through the library objects).

2.3 Collecting Help

CHARDOC ***output-style styles filename***
List the special characters of certain output styles in a TeX or Scribe file. The output file can be processed by TeX or Scribe and will have multicolumn format.

COLLECT-HELP ***modules categories filename***
Collect help for the specified modules into a file. Prints out a # every time it finds a help message, and a * every time it finds a TPS object with no help message.

HELP-LIST ***category filename***
List all help available for objects of the given category into a file.

HTML-DOC ***directory***
Produce HTML documentation in the specified directory. This requires an empty directory and a lot of disk space, and will take quite some time to produce.

LATEX-DOC ***category-list context-list filename***
Produce Latex documentation about the specified categories.

LATEX-QUICK-REF ***filename***
Produce a quick Latex reference to the rules available in TPS.

OMDOC-ASSERTION ***wff wff-name filename***
Print a wff in OMDoc notation.

OMDOC-CLASS-SCHEME ***name***
Print the library into OMDoc files using the given Classification Scheme to collect library items into theories.

OMDOC-LIB
Print the library into OMDoc files in OMDoc notation.

OMDOC-PROOF ***filename***
Print the current proof into an OMDoc file in OMDoc notation.

QUICK-REF ***filename***
Produce a quick reference to the rules available in TPS.

SCRIBE-DOC ***category-list context-list filename***
Produce Scribe documentation about the specified categories.

2.4 Concept

LOADKEY ***key mssg***
Load one of the function keys f1-f10 on a concept terminal with a string.

RESET
Put a Concept terminal into correct mode and load the function keys.

2.5 Starting and Finishing

ALIAS ***name def***
Define an alias DEF for the symbol NAME. Works just like the alias command in the Unix csh. If the value of NAME is *ALL*, all aliases will be printed; if the value of DEF is the empty string, then the current alias definition of NAME will be printed. See UNALIAS.

CLEANUP
If the proof is complete, will delete unnecessary lines from a proof. It may also eliminate or suggest eliminating unnecessary hypotheses. If the proof is incomplete, will do a partial cleanup in which only unnecessary lines justified by SAME will be removed.

DONE
Signal that the current proof is complete.

EXERCISE ***excno***
Start the proof of a new exercise.

EXIT
Exit from TPS.

NEWS

Type TPS news on the terminal.

PROVE *wff prefix num*

Start a new proof of a given wff.

RECONSIDER *prefix*

Reconsider a proof. The following proofs are in memory:

For more details, use the PROOFLIST command.

REMARK *remark*

Send a message to the teacher or maintainer.

SUMMARY

Tells the user what exercises have been completed.

UNALIAS *name*

Remove an alias for the symbol NAME. Like the Unix csh unalias, except that NAME must exactly match the existing alias; no filename completion is done.

2.6 Printing

BUILD-PROOF-HIERARCHY

This command builds hierarchical information into the proof outline. The information includes associations between lines and linear chains of inferences which trace the consequences of the most recent hypothesis of a line. That is, a line

ln) Hn,m |- an

would be associated with a linear chain of lines l1,...,ln where m is the line corresponding to the most recent hypothesis and the proof would justify the modified lines

l1) H1,m |- l1 l2) H2,l1 |- l2 l3) H3,l2 |- l3 . . . ln) Hn,ln-1 |- ln

where H1 < H2 < . . . < Hn (subset relation).

That is, we trace the consequences of the hypothesis m to the consequence ln. Such a linear chain is on one level of the hierarchy. One level down on the hierarchy would be the linear chains associated with each of the lines used to justify l1,...,ln (except those which appear in the chain l1,...,ln). If the proof is complete, then lines l1 and m will be the same.

Lines without hypotheses are also associated with such "linear chains", following the rule that l1 < l2 if the proof justifies the inference l1 |- l2.

The resulting hierarchy information is used by PBRIEF, EXPLAIN, and PRINT-PROOF-STRUCTURE to help users focus on the logical structure of a proof.

DEPTH *num*

Causes all subformulas at depth greater than n to be printed as & .

EXPLAIN *line depth*

This command explains a line of a proof outline. In particular, the command BUILD-PROOF-HIERARCHY builds dependency information into a proof outline which allows the proof outline to be viewed as a hierarchy of subproofs (see help for BUILD-PROOF-HIERARCHY). The command EXPLAIN shows the lines included in the levels of this hierarchy (to the specified depth) starting at the level associated with the specified line. Some flags which affect the printing include: PRINT-COMBINED-UIS, PRINT-COMBINED-UGENS, PRINT-COMBINED-EGENS, and PRINT-UNTIL-UI-OR-EGEN.

FIND-LINE *wff vars meta*

Find all lines matching a certain wff, up to alphabetic change of bound variables and (possibly) alphabetic change of a given list of free variables. Optionally, you can treat the remaining free variables as matching any given term (as you might do if you were asserting an axiom). e.g. (suppose P is an abbreviation or constant): FIND-LINE "P a" () NO finds all lines that say "P a" FIND-LINE "P a" ("a") NO also finds "P x" and "P y" FIND-LINE "P a" () YES finds all the above, plus "P [COMPOSE f g]" FIND-LINE "a x" ("x") YES finds all lines of the form "SOME-TERM some-var"

PALL

Print all the lines in the current proof outline.

PBRIEF *depth*

This command prints a proof outline, hiding some lines. In particular, the command BUILD-PROOF-HIERARCHY builds dependency information into a proof outline which allows the proof outline to be viewed as a hierarchy of subproofs (see help for BUILD-PROOF-HIERARCHY). The command PBRIEF shows the lines included in the top levels of this hierarchy (to the specified depth). PBRIEF is essentially a call to the command EXPLAIN with the last line of the proof outline as the LINE argument (see help for EXPLAIN). Some flags which affect the printing include: PRINT-COMBINED-UIS, PRINT-COMBINED-UGENS, PRINT-COMBINED-EGENS, and PRINT-UNTIL-UI-OR-EGEN.

PL *num1 num2*

Print all proof lines in a given range.

PL* *print-ranges*

Print all proof lines in given ranges.

PLINE *line*

Print a specified line.

PPLAN *pline*

Print a planned line and all its supports.

PRINT-PROOF-STRUCTURE

This prints the structure of the proof outline. The structure is generated by BUILD-PROOF-HIERARCHY. Linear chains of line numbers are printed which indicate the logical chains of inferences. Each link in a linear chain is indicated by an arrow (l1)->(l2) where l1 and l2 are line numbers. If line l2 does not follow in a single step from l1 (i.e., by a single application of an inference rules), then PRINT-PROOF-STRUCTURE will also show the linear chains of inference used to justify (l1)->(l2). Some lines (such as those without hypotheses and planned lines) are exceptions. These top level lines are sometimes printed alone (instead of in arrow notation). This could be read TRUE->(l) to maintain consistent notation, but the notation (l) appears more readable in practice.

PRW *gwff*
Print real wff. Turns off special characters (including FACE definitions), infix notation, and dot notation, and then prints the wff.

PW *gwff*
Print gwff.

PWSCOPE *gwff*
print gwff with all brackets restored.

PWTYPES *gwff*
Prints a wff showing types.

SHOWNOTYPES
Suppress the printing of types on all wffs.

SHOWTYPES
From now on show the types on all wffs.

TABLEAU *line*
Print the part of the proof which justifies the given line, in a natural deduction tableau format.

^P
Print current plan-support pair in the proof.

^PN
Print current plan-support pair in the proof, as in ^P, but also print just the line numbers of the other lines in the proof.

2.7 Saving Work

EXECUTE-FILE *comfil execprint outfil stepping*
Execute commands from a SAVE-WORK file. Call this from the main top level or the proofwindows top level of TPS. Note that this will not save subsequent commands in the same file, which distinguishes it from RESTORE-WORK. In the cases where EXECUTE-FILE doesn't work, one can usually just load the .work file into an editor and then cut and paste it, whole, into the TPS window. Single-stepping only works between commands on the main top level; it will not stop at prompts which are internal to a command, nor between commands on a different top level. To force a work-file to stop in such a place, use the PAUSE command

when creating the work file. If you are single-stepping through a file, you can abort at any time by typing ^G<RETURN>.

FINDPROOF ***name***
Searches your home directory and the directories listed in SOURCE-PATH, looking for a proof whose name contains the given string.

FINISH-SAVE
Finishing saving work in a file. The difference between STOP-SAVE and FINISH-SAVE is: the former is temporary because you can use RESUME-SAVE to resume saving work into the same file; the latter closes the output stream, so you can not save work into the same file after executing it.

PAUSE
Force a work file to stop and query the user. PAUSE, like ABORT, is valid both as a top-level command and as a response to a prompt; it prints the message "Press RETURN, or Ctrl-G RETURN to abort.", waits for such a response from the user, and then repeats the original prompt. This command is of no use unless a work file is being created; see EXECUTE-FILE for more details.

RESTORE-WORK ***comfil execprint outfil***
Execute commands from a SAVE-WORK file and continue to save in that file. See EXECUTE-FILE for more information.

RESTOREPROOF ***savefile***
Reads a natural deduction proof from a file created by SAVEPROOF and makes it the current proof. A security feature prevents the restoration of saved proofs which have been altered in any way. Retrieve any definitions which are used in the proof and stored in the library before restoring the proof. If you don't specify a directory, it will first try your home directory and then all the directories listed in SOURCE-PATH.

RESUME-SAVE
Use this command to resume saving commands into the most recent save-work file. Unlike RESTORE-WORK, this command doesn't execute commands from the file, but simply appends subsequent commands to the file. You can not use this command if you are already saving work. Also, you may run into trouble if you forgot to save some commands.

SAVE-FLAGS-AND-WORK ***savefile***
Start saving commands in the specified file, first storing all flag settings.

SAVE-SUBPROOF ***savefile lines subname***
Saves part of the current natural deduction proof to the specified file in a form in which it can be restored. The line ranges specified will be increased to include all the other lines on which the given lines depend. See the help message for LINE-RANGE to find out what a line-range should look like. An example list is: 1–10 15–23 28 34–35 Also creates a new proof in memory with the given name, and makes that the current proof. Use RESTOREPROOF to restore the proof. Overwrites the file if it already exists.

SAVE-WORK ***savefile***

Start saving commands in the specified file. These commands can be executed subsequently by using EXECUTE-FILE or RESTORE-WORK. If you are creating a work file for a demonstration, and need it to pause at certain points as it is reloaded by TPS, then see the help message for EXECUTE-FILE for more information on how to do this.

SAVEPROOF ***savefile***

Saves the current natural deduction proof to the specified file in a form in which it can be restored. Use RESTOREPROOF to restore the proof. Overwrites the file if it already exists.

SCRIPT ***scriptfile if-exists-append***

Saves a transcript of session to a file. If the current setting of STYLE is SCRIBE or TEX, an appropriate header will be output to the script file (unless the file already exists). **NOTE** If you start SCRIPT from a PUSHed top level, be sure to do UNSCRIPT before you POP that top level, or your transcript may be lost. The same also applies to starting SCRIPT from subtoplevels such as MATE; you can enter further subtoplevels like LIB and ED from the MATE top level, and SCRIPT will carry on recording, but before leaving the MATE top level you should type UNSCRIPT or your work will be lost.

STOP-SAVE

Stop saving commands in a SAVE-WORK file.

UNSCRIPT

Closes the most recent file opened with the SCRIPT command.

2.8 Saving Wffs

APPEND-WFF ***weak-label help-string filename***

Append a definition of a weak label to a file. If the file does not yet exist, it will be created. You may wish to use LIB instead.

APPEND-WFFS ***weak-labels filename***

Append the definitions of a list of weak labels to a file. If the file does not yet exist, it will be created. You may wish to use LIB instead.

2.9 Printing Proofs into Files

PRINTPROOF ***filename***

Print the current proof into a file.

SCRIBEPROOF ***filename timing***

Print the current proof into a MSS file. After leaving TPS, run this .MSS file through Scribe and print the resulting file.

SETUP-SLIDE-STYLE

Sets flags to produce slides in scribe style.

SLIDEPROOF ***filename***

Print the current proof into a MSS file. Use this command to make slides. After leaving TPS, run this .MSS file through Scribe and print the resulting file.

TEXPROOF ***filename timing***

Print the current proof into a tex file. After leaving tps, run this .tex file through tex and print the resulting file.

Many flags affect the output of texproof. See: USE-INTERNAL-PRINT-MODE, TURNSTILE-INDENT-AUTO, TURNSTILE-INDENT, LATEX-EMULATION, TEX-MIMIC-SCRIBE, PPWFFLAG, DISPLAYWFF, INFIX-NOTATION, PAGELENGTH, PAGEWIDTH, TEX-BREAK-BEFORE-SYMBOLS, LOCALLEFTFLAG, SCOPE, ALLSCOPEFLAG, USE-DOT, FIRST-ORDER-PRINT-MODE, FILLINEFLAG, ATOMVALFLAG.

2.10 Proof Outline

CREATE-SUBPROOF ***lines subname***

Creates a new proof in memory from the given lines, plus all the lines on which they depend, and makes that the current proof.

LINE-COMMENT ***line comment***

Attach a comment to a given existing line. The comment will be parsed for gwffs and line numbers as follows: anything enclosed in # symbols is assumed to be a gwff, and anything enclosed in $ symbols is assumed to be the number of an existing line. Line numbers in comments will be updated as lines are moved around; gwffs will be printed in the current STYLE. Examples: "1st copy of line $ 5$, instantiated with # COMPOSE# " "2nd copy of line $ 5$, instantiated with ITERATE" "3rd copy of line $ 5$, instantiated with # a OR b# " (The first prints the definition of COMPOSE; the second prints the word "ITERATE", and the third prints the given gwff. If line 5 is subsequently renumbered, the line number will change in all these comments.)

MERGE-PROOFS ***proof subproof***

Merges all of the lines of a subproof into the current proof. If EXPERTFLAG is NIL, no line number may occur in both proofs. If EXPERTFLAG is T, then if a line number occurs in both proofs, the lines to which they refer must be the same (with one exception: if one is a planned line and the other is the same line with a justification, then the justified line will overwrite the planned one). Compare TRANSFER-LINES.

The following proofs are in memory:

For more details, use the PROOFLIST command.

PROOF-COMMENT ***comment***

Attaches a comment to the current proof. The default value is the current comment. Uses the same comment syntax as LINE-COMMENT; see the help message of that command for more information. You can see the comments on all the current proofs by using PROOFLIST.

PROOFLIST

Print a list of all proofs or partial proofs currently in memory. Also prints the final line of each proof and the comment, if any, attached to it.

TRANSFER-LINES ***proof subproof lines***

Copies all of the given lines of a subproof, and all lines on which they depend, into the current proof. If EXPERTFLAG is NIL, no line number may occur in both proofs. If EXPERTFLAG is T, then if a line number occurs in both proofs, the lines to which they refer must be the same (with one exception: if one is a planned line and the other is the same line with a justification, then the justified line will overwrite the planned one). Different comments from two otherwise identical lines will be concatenated to form the comment in the resulting proof.

This is equivalent to CREATE-SUBPROOF followed by MERGE-PROOFS.

The following proofs are in memory:

For more details, use the PROOFLIST command.

2.11 Expansion Trees

PSEQ ***prefix***

Print a Sequent Calculus Derivation

SEE ALSO: pseq-use-labels, pseql

PSEQL ***prefix lbd ubd***

Print a Sequent Calculus Derivation

SEE ALSO: pseq-use-labels, pseq

SEQ-TO-NAT ***sname prefix***

Translates a Sequent Calculus Derivation (possibly with Cuts) to a Natural Deduction Proof

SEQLIST

Print a list of all sequent calculus derivations currently in memory.

2.12 Search Suggestions

AUTO-SUGGEST

Given a completed natural deduction proof (which must be the current dproof; use RECONSIDER to return to an old proof), suggest flag settings for an automatic proof of the same theorem.

This will also automatically remove all uses of SUBST= and SYM= from the proof (you will be prompted before this happens, as it permanently modifies the proof).

This will show all of the instantiations (and primitive substitutions) that are necessary for the proof, and suggest settings for NUM-OF-DUPS, MAX-MATES, DEFAULT-MS, MAX-PRIM-DEPTH, MAX-PRIM-LITS and REWRITE-DEFNS

ETR-AUTO-SUGGEST

Given an eproof, suggest flag settings for an automatic proof of the same theorem. Such an eproof may be the result of translating a natural deduction proof using nat-etree.

This will show all of the instantiations (and primitive substitutions) that are necessary for the proof, and suggest settings for NUM-OF-DUPS, MS98-NUM-OF-DUPS, and MAX-MATES.

2.13 Mating search

CLOSE-TESTWIN

Closes the window that displays the test-top and TPS-TEST summary. Use ..../tps/utilities/vpshow (from a shell, not from TPS) to view the output file again.

DEASSERT-LEMMAS ***prefix***

Combine a collection of natural deduction proofs where some lines contain ASSERT justifications where the asserted line has a natural deduction proof into a single natural deduction proof.

DIY ***goal support window***

DO IT YOURSELF. Calls matingsearch procedure specified by the flag DEFAULT-MS with specified planned line and supports, then translates the resulting proof to natural deduction. Allows some of the output to be sent to a separate vpform window (equivalent to issuing the OPEN-MATEVPW command before typing DIY).

DIY-L ***goal support window range***

DIY for lemmas. Behaves as for DIY, but puts all new lines into a specified range rather than scattering them throughout the proof.

DIY-L-WITH-TIMEOUT ***goal support timeout window***

DIY for lemmas (with timeout). Calls diy-l with a timeout value in seconds. The timeout value applies only to mating search. That is, as long as mating search succeeds within the allotted time, merging and translation to natural deduction can take as long as necessary.

This is only available for TPS running under Lisps with multiprocessing (e.g., Allegro >= 5.0).

See Also: DIY-L, DIY-WITH-TIMEOUT

DIY-WITH-TIMEOUT ***goal support timeout window***

DO IT YOURSELF (with timeout). Calls diy with a timeout value in seconds. The timeout value applies only to mating search. That is, as long as mating search succeeds within the allotted time, merging and translation to natural deduction can take as long as necessary.

This is only available for TPS running under Lisps with multiprocessing (e.g., Allegro >= 5.0).

See Also: DIY, DIY-L-WITH-TIMEOUT

DIY2 ***goal support quiet-run expu newcore output timing testwin***

DO IT YOURSELF 2. Tries to prove an existing line using a variety of given modes. This essentially combines the commands TEST-INIT and TPS-TEST. See the help message for TPS-TEST for more information about options.

See Also: DIY, DIY-L, DIY2-L, PIY, PIY2, TEST-INIT, TPS-TEST

DIY2-L ***goal support line-range quiet-run expu newcore output timing testwin***

DO IT YOURSELF 2 with line range for new lines. Tries to prove an existing line using a variety of given modes. If successful, the new lines are put into the gap specified. This essentially combines the commands TEST-INIT and TPS-TEST. See the help message for TPS-TEST for more information about options.

See Also: DIY, DIY-L, DIY2, PIY, PIY2, TEST-INIT, TPS-TEST

EPROOFLIST ***complete***

Print a list of all expansion proofs currently in memory.

MONITOR

Turns the monitor on, and prints out the current monitor function and parameters. See NOMONITOR. See also QUERY-USER for an alternative way to monitor the progress of the matingsearch. For a list of monitor functions, type MONITORLIST. To change the current monitor function, enter the name of the desired new monitor function from the main top level or the mate top level.

MONITORLIST

List all monitor functions.

NOMONITOR

Turns the monitor off, and prints out the current monitor function and parameters. See MONITOR. For a list of monitor functions, type MONITORLIST. To change the current monitor function, enter the name of the desired new monitor function from the main top level or the mate top level.

PIY ***wff prefix num window***

PROVE IT YOURSELF. Combines the prove command with diy - allowing a choice of a mode for trying to prove a theorem automatically.

PIY2 ***wff prefix num quiet-run expu newcore output timing testwin***

PROVE IT YOURSELF 2. Tries to prove a theorem using a variety of given modes. This essentially combines the commands PROVE, TEST-INIT and TPS-TEST. See the help message for TPS-TEST for more information about options.

See Also: PIY, DIY, DIY-L, DIY2, DIY2-L, TEST-INIT, TPS-TEST

SET-EPROOF ***epf***

Set the current expansion proof.

To see a list of expansion proofs in memory, use EPROOFLIST

2.14 MS91-6 and MS91-7 search procedures

SEARCH-ORDER ***num vpf verb***

Generates the first n option sets that will be searched under the current flag settings (assuming that the first (n-1) searches fail because they run out of time rather than for any other reason). This will show the names and weights of the option sets, the primitive substitutions and duplications. Note : "Ordinary" duplications are duplications that have not had a primsub applied to them. So, for example, "X has 2 primsubs plus 3 ordinary duplications" means that the vpform now contains five copies of the relevant quantifier, two of which have had primsubs applied to them.

2.15 Proof Translation

ETREE-NAT ***prefix num tac mode***

Translates the current expansion proof, which is value of internal variable current-eproof, into a natural deduction style proof. The default value of the tactic is given by the flag DEFAULT-TACTIC.

NAT-ETREE ***prefix***

Translates a natural deduction proof, (which must be the current dproof – use RECONSIDER to return to an old proof in memory), into an expansion proof. This will not work on all proofs: in particular, proofs containing ASSERT of anything but REFL= and SYM=, proofs using rewrite rules and proofs containing SUBST= or SUB= cannot be translated at present.

There are several versions of nat-etree. Set the flag NAT-ETREE-VERSION to determine which version to use.

In all but the OLD version, the user is given the option of removing lines justified by SUBST=, SUB=, or SYM= and replacing the justification with a subproof. This permanently modifies the proof. (AUTO-SUGGEST also gives such an option.)

NORMALIZE-PROOF ***prefix***

Normalize a natural deduction proof. The actual procedure uses DEASSERT-LEMMAS to combine asserted lemmas into one big natural deduction proof. This is the converted into a sequent calculus derivations with cuts. A cut elimination (which may not terminate in principle) creates a cut-free proof which is translated back to a normal natural deduction proof.

To suppress excessive output, try setting the following flags NATREE-DEBUG, ETREE-NAT-VERBOSE and PRINTLINEFLAG to NIL and TACTIC-VERBOSE to MIN.

PFNAT ***proof***

To generate a NATREE from given proof and store it in CURRENT-NATREE. This may evolve into a command for rearranging natural deduction style proofs.

PNTR

Print out the current natree stored in CURRENT-NATREE. Mainly for the purpose of debugging.

TIDY-PROOF ***old-prfname new-prfname***

Translate a ND proof to an eproof and back again (into a proof with a new name) in the hope of tidying it up a bit. Equivalent to NAT-ETREE; MATE ! ; PROP-MSEARCH ; MERGE-TREE ; LEAVE ; ETREE-NAT ; CLEANUP ; SQUEEZE

2.16 Unification

LEAST-SEARCH-DEPTH

Print the least needed unification tree depth for the last proven higher-order theorem. Also suggest to lower flags MAX-SEARCH-DEPTH to the least needed value if they are greater than it.

2.17 Search Analysis

ELIMINATE-ALL-RULEP-APPS ***pfname***

Expands applications of RuleP in the current natural deduction proof into more primitive rules. This works by calling fast propositional search with the current flag settings except USE-RULEP is set to NIL. BASIC-PROP-TAC is used to translate to natural deduction.

This command also eliminates other 'fancy' propositional justifications: Assoc (Assoc-Left), EquivConj (in favor of EquivImplics), Imp-Disj-L, Imp-Disj-R, Imp-Disj, Disj-Imp-L, Disj-Imp-R, and Disj-Imp.

See Also: ELIMINATE-RULEP-LINE - which eliminates a particular application of RuleP. ELIMINATE-CONJ*-RULEP-APPS - which does not depend on automatic search.

ELIMINATE-CONJ*-RULEP-APPS ***pfname***

Expands applications of RuleP in the current natural deduction proof when they can be replaced by a sequence of IConj or EConj applications.

This reverses the effect of the ICONJ* and ECONJ* tactics which are often used when translating from an expansion proof to a natural deduction proof.

SEE ALSO: ELIMINATE-ALL-RULEP-APPS, ELIMINATE-RULEP-LINE

ELIMINATE-RULEP-LINE ***line***

Expands an application of RuleP in the current natural deduction proof into more primitive rules. This works by calling fast propositional search with the current flag settings except USE-RULEP is set to NIL. BASIC-PROP-TAC is used to translate to natural deduction.

This command can also eliminate other 'fancy' propositional justifications: Assoc (Assoc-Left), EquivConj (in favor of EquivImplics), Imp-Disj-L, Imp-Disj-R, Imp-Disj, Disj-Imp-L, Disj-Imp-R, and Disj-Imp.

SEE ALSO: ELIMINATE-ALL-RULEP-APPS, ELIMINATE-CONJ*-RULEP-APPS

SET-BACKGROUND-EPROOF ***epr***

Sets the background eproof to be used by MS98-TRACE. These are automatically set when nat-etree is run.

2.18 Tactics

ECHO ***echothing***

Echo a string.

USE-TACTIC ***tac tac-use tac-mode***

Use a tactic on the current goal. The default tactic is given by the flag DEFAULT-TACTIC.

2.19 suggestions

ADVICE

Give some advice on how to proceed with the current proof.

CHECK-STRUCTURE

Check various structural properties of the current proof. You will be informed about suspect constellations in the incomplete proof which may make it difficult for ETPS to provide advice or for you to finish the proof.

GO

Start producing and applying suggestions until no more are found. Suggestions are treated according to their priority and the state of the global parameter GO-INSTRUCTIONS.

GO2 ***tacmode***
Apply all possible invertible tactics, until no more are possible. This is equivalent to typing USE-TACTIC GO2-TAC NAT-DED. The amount of output to the main window and the proofwindows is determined by the flag ETREE-NAT-VERBOSE.

MONSTRO ***tacmode***
This is equivalent to typing USE-TACTIC MONSTRO-TAC NAT-DED. It applies all the same tactics as GO2, and also ui-herbrand-tac. The amount of output to the main window and the proofwindows is determined by the flag ETREE-NAT-VERBOSE.

SUGGEST ***pline***
Suggest some applicable inference rule for proving a planned line.

2.20 Vpforms

CLOSE-MATEVPW
Closes the window that displays the current vpform and substitution stack. Use ..../tps/utilities/vpshow (from a shell, not from TPS) to view the output file again.

OPEN-MATEVPW ***filename***
Open a window which will display the current vpform and substitution stack, if any. The window can be closed with the command CLOSE-MATEVPW. The size of the text is determined by the flag CHARSIZE, and the current width of the window by the flag VPW-WIDTH. The initial height of the window is determined by VPW-HEIGHT Use ..../tps/utilities/vpshow to view the file from the monitor level.

2.21 Rearranging the Proof

ADD-HYPS ***hyps line***
Weaken a line to include extra hypotheses. Adding the hypotheses to the line may cause some lines to become planned lines. If possible, the user is given the option of adding hypotheses to lines after the given line so that no lines will become planned.

DELETE ***del-lines***
Delete lines from the proof outline.

DELETE* ***ranges***
Delete ranges of lines from the proof outline.

DELETE-HYPS ***hyps line***
Delete some hypotheses from the given line. This may leave the given line as a planned line. The user is given the option of also deleting some hypotheses from lines after the given line. If possible, the user is given the

option of deleting some hypotheses from lines before the given line so that the given line does not become a planned line.

INTRODUCE-GAP ***line num***
Introduce a gap in an existing proof.

LOCK-LINE ***line***
Prevent a line from being deleted.

MAKE-ASSERT-A-HYP ***l***
Take a line justified by Assert, change its justification to Hyp, make lines after it include this as a hypothesis, and perform a Deduct at the end so that the new proof does not depend on the Assert.

We may want to use this before calling nat-etree, since this does not handle most Asserts.

MODIFY-GAPS ***num1 num2***
Remove unnecessary gaps from the proof structure, and modify line numbers so that the length of each gap is neither less than the first argument, nor greater than the second.

MOVE ***old-line new-line***
Renumber one particular line.

MOVE* ***range-to-move new-start***
Move all proof lines in given range to begin at new start number, but preserving the relative distances between the lines.

PLAN ***line***
Change a justified line to a planned line.

RENUMBERALL ***num***
Renumber all the lines in the current proof.

SQUEEZE
Removes unnecessary gaps from the proof structure.

UNLOCK-LINE ***line***
The opposite of LOCK-LINE.

2.22 Status

ARE-WE-USING ***linelist***
Determines if given lines are being used to justify any other lines. Notice that the argument is a list of lines, not a range (i.e. 1 2 3 4 rather than 1–4).

COUNT-LINES
Show the number of lines in the current proof.

PSTATUS

Give the current status information, i.e. planned lines and their supports. If work is being saved, issues an appropriate message.

SPONSOR ***pline linelist***

Add new sponsoring lines to the sponsors of a planned line.

SUBPROOF ***pline***

Concentrate on proving a particular planned line.

UNSPONSOR ***pline linelist***

Remove a list of unwanted sponsoring lines from among the sponsors of a planned line.

2.23 Miscellaneous Rules

ASSERT ***theorem line***

Use a theorem as a lemma in the current proof. If the line already exists, ETPS will check whether it is a legal instance of the theorem schema, otherwise it will prompt for the metavariables in the theorem schema (usually x or P, Q, ...).

ASSERT2 ***theorem line***

Use a theorem as a lemma in the current proof. If the line already exists, ETPS will check whether it is a legal instance of the theorem schema, otherwise it will prompt for the metavariables in the theorem schema (usually x or P, Q, ...). This version of ASSERT ensures correct behaviour for theorems containing bound variables.

HYP ***p2 h1 a b p2-hyps h1-hyps***

Introduce a new hypothesis line into the proof outline.

LEMMA ***p2 p1 a b p2-hyps p1-hyps***

Introduce a Lemma.

SAME ***p2 d1 a p2-hyps d1-hyps***

Use the fact that two lines are identical to justify a planned line.

2.24 Propositional Rules

ASSOC-LEFT ***d1 d2 p assoc-l d1-hyps d2-hyps***

Rule to associate a support line leftwards. Use before calling CASES3 or CASES4.

CASES ***p6 d1 p5 h4 p3 h2 b a c p6-hyps d1-hyps p5-hyps h4-hyps p3-hyps h2-hyps***

Rule of Cases.

CASES3 ***p8 d1 p7 h6 p5 h4 p3 h2 c b a d p8-hyps d1-hyps p7-hyps h6-hyps p5-hyps h4-hyps p3-hyps h2-hyps***
Rule of Cases.

CASES4 ***p10 d1 p9 h8 p7 h6 p5 h4 p3 h2 d c b a e p10-hyps d1-hyps p9-hyps h8-hyps p7-hyps h6-hyps p5-hyps h4-hyps p3-hyps h2-hyps***
Rule of Cases.

DEDUCT ***p3 d2 h1 b a p3-hyps d2-hyps h1-hyps***
The deduction rule.

DISJ-IMP ***d1 d2 b a d1-hyps d2-hyps***
Rule to replace a disjunction by an implication.

DISJ-IMP-L ***d1 d2 b a d1-hyps d2-hyps***
Rule to replace a disjunction by an implication.

DISJ-IMP-R ***d1 d2 b a d1-hyps d2-hyps***
Rule to replace a disjunction by an implication.

ECONJ ***d1 d3 d2 b a d1-hyps d3-hyps d2-hyps***
Rule to infer two conjuncts from a conjunction.

EQUIV-IMPLICS ***d1 d2 r p d1-hyps d2-hyps***
Rule to convert an equivalence into twin implications.

ICONJ ***p3 p2 p1 b a p3-hyps p2-hyps p1-hyps***
Rule to infer a conjunction from two conjuncts.

IDISJ-LEFT ***p2 p1 b a p2-hyps p1-hyps***
Introduce a disjunction (left version).

IDISJ-RIGHT ***p2 p1 a b p2-hyps p1-hyps***
Introduce a disjunction (right version).

IMP-DISJ ***d1 d2 b a d1-hyps d2-hyps***
Rule to replace an implication by a disjunction.

IMP-DISJ-L ***d1 d2 b a d1-hyps d2-hyps***
Rule to replace an implication by a disjunction.

IMP-DISJ-R ***d1 d2 a b d1-hyps d2-hyps***
Rule to replace an implication by a disjunction.

IMPLICS-EQUIV ***p2 p1 r p p2-hyps p1-hyps***
Rule to convert twin implications into an equivalence.

INDIRECT ***p3 p2 h1 a p3-hyps p2-hyps h1-hyps***
Rule of Indirect Proof.

INDIRECT1 ***p3 p2 h1 b a p3-hyps p2-hyps h1-hyps***
Rule of Indirect Proof Using One Contradictory Line.

INDIRECT2 ***p4 p3 p2 h1 b a p4-hyps p3-hyps p2-hyps h1-hyps***
Rule of Indirect Proof Using Two Contradictory Lines.

ITRUTH ***p1 p1-hyps***
Rule to infer TRUTH

MP ***d2 d3 p1 b a d2-hyps d3-hyps p1-hyps***
Modus Ponens.

RULEP ***conclusion antecedents***
Justify the CONSEQUENT line by RULEP using the lines in the list ANTECEDENTS.

SUBST-EQUIV ***d2 d3 p1 p r t s d2-hyps d3-hyps p1-hyps***
Substitution of Equivalence. Usable when R and P are the same modulo the equivalence s EQUIV t.

2.25 Negation Rules

ABSURD ***p2 p1 a p2-hyps p1-hyps***
Rule of Intuitionistic Absurdity.

ENEG ***p3 d1 p2 a p3-hyps d1-hyps p2-hyps***
Rule of Negation Elimination.

INEG ***p3 p2 h1 a p3-hyps p2-hyps h1-hyps***
Rule of Negation Introduction

NNF ***d1 d2 a neg-norm d1-hyps d2-hyps***
Put Wff in Negation Normal Form.

NNF-EXPAND ***p2 p1 a neg-norm p2-hyps p1-hyps***
Expand Wff from Negation Normal Form.

PULLNEG ***p2 p1 a push-negation p2-hyps p1-hyps***
Pull out negation.

PUSHNEG ***d1 d2 a push-negation d1-hyps d2-hyps***
Push in negation.

2.26 Quantifier Rules

AB* ***d1 d2 b a d1-hyps d2-hyps***
Rule to alphabetically change embedded quantified variables.

ABE ***d1 d2 y a x s d1-hyps d2-hyps***
Rule to change a top level occurrence of an existentially quantified variable.

ABU ***p2 p1 y a x s p2-hyps p1-hyps***
Rule to change a top level occurrence of a universally quantified variable.

EGEN ***p2 p1 t a x lcontr p2-hyps p1-hyps***
Rule of Existential Generalization.

RULEC ***p4 d1 d3 h2 y b x a lcontr p4-hyps d1-hyps d3-hyps h2-hyps***
RuleC

RULEC1 ***p4 d1 d3 h2 b x a p4-hyps d1-hyps d3-hyps h2-hyps***
RuleC1 – the special case of RULEC where the chosen variable has the same name as the bound variable.

UGEN ***p2 p1 a x p2-hyps p1-hyps***
Rule of Universal Generalization.

UI ***d1 d2 t a x lcontr d1-hyps d2-hyps***
Rule of Universal Instantiation.

2.27 Substitution Rules

SUBSTITUTE ***d1 d2 x t a s d1-hyps d2-hyps***
Rule to substitute a term for a variable.

TYPESUBST ***d p a b***
Substitute for a type variable in one line to infer another line. The type variable must not appear in any hypothesis.

2.28 Equality Rules

EQUIV-EQ ***d1 d2 b a d1-hyps d2-hyps***
Rule to infer a line from one which is equal up to definitions, lambda conversion, alphabetic change of bound variables and the Leibniz definition of the symbol = . You may use the editor command EXPAND= to create the desired line from the existing one.

EQUIV-EQ-CONTR ***p2 p1 a instantiate-top-equality p2-hyps p1-hyps***
Rule to contract the outermost instance of the Leibniz definition of equality into instances of the symbol = .

EQUIV-EQ-CONTR* ***p2 p1 a instantiate-equalities p2-hyps p1-hyps***
Rule to contract all instances of the Leibniz definition of equality into instances of the symbol = .

EQUIV-EQ-EXPD ***d1 d2 a instantiate-top-equality d1-hyps d2-hyps***
Rule to expand the outermost equality using the Leibniz definition.

EQUIV-EQ-EXPD* ***d1 d2 a instantiate-equalities d1-hyps d2-hyps***
Rule to expand all equalities using the Leibniz definition.

EXT= ***p2 p1 x g f p2-hyps p1-hyps***
Rule of Extensionality.

EXT=0 ***p2 p1 r p p2-hyps p1-hyps***
Rule to convert equality at type o into an equivalence.

LET ***p5 p4 h3 d2 d1 a x c p5-hyps p4-hyps h3-hyps d2-hyps d1-hyps***
Bind a variable to a term.

SUBST= ***d2 d3 p1 p r t s d2-hyps d3-hyps p1-hyps***
Substitution of Equality. Usable when R and P are the same modulo the equality s=t.

SUBST=L ***d2 d3 p1 p r t s d2-hyps d3-hyps p1-hyps***
Substitution of Equality. Replaces some occurrences of the left hand side by the right hand side.

SUBST=R ***d2 d3 p1 p r s t d2-hyps d3-hyps p1-hyps***
Substitution of Equality. Replaces some occurrences of the right hand side by the left hand side.

SYM= ***p2 p1 a b p2-hyps p1-hyps***
Rule of Symmetry of Equality.

2.29 Definition Rules

EDEF ***d1 d2 a inst-def d1-hyps d2-hyps***
Rule to eliminate first definition, left to right.

EQUIV-WFFS ***d1 d2 r p d1-hyps d2-hyps***
Rule to assert equivalence of lines up to definition.

IDEF ***p2 p1 a inst-def p2-hyps p1-hyps***
Rule to introduce a definition.

2.30 Lambda Conversion Rules

BETA* ***d1 d2 b a d1-hyps d2-hyps***
Rule to infer a line from one which is equal up to lambda conversion using beta rule (but NOT eta rule) and alphabetic change of bound variables.

ETA* ***d1 d2 b a d1-hyps d2-hyps***
Rule to infer a line from one which is equal up to lambda conversion using eta rule (but NOT beta rule) and alphabetic change of bound variables.

LAMBDA* ***d1 d2 b a d1-hyps d2-hyps***
Rule to infer a line from one which is equal up to lambda conversion using both beta and eta rules and alphabetic change of bound variables.

LCONTR* ***d1 d2 a lnorm d1-hyps d2-hyps***
Rule to put an inferred line into Lambda-normal form using both beta and eta conversion.

LCONTR*-BETA ***d1 d2 a lnorm-beta d1-hyps d2-hyps***
Rule to put an inferred line into beta-normal form.

LCONTR*-ETA ***d1 d2 a lnorm-eta d1-hyps d2-hyps***
Rule to put an inferred line into eta-normal form.

LEXPD* ***p2 p1 a lnorm p2-hyps p1-hyps***
Rule to put a planned line into Lambda-normal form using both beta and eta conversion.

LEXPD*-BETA ***p2 p1 a lnorm-beta p2-hyps p1-hyps***
Rule to put a planned line into beta-normal form.

LEXPD*-ETA ***p2 p1 a lnorm-eta p2-hyps p1-hyps***
Rule to put a planned line into eta-normal form.

2.31 Rewriting commands

ACTIVATE-RULES ***rlist***
Activate a list of rewrite rules. Activating a rule which is already active has no effect.

ACTIVE-THEORY
Show which theory is currently active. Any new derivation in the REWRITING top level will use this theory.

DEACTIVATE-RULES ***rlist***
Deactivate a list of rewrite rules. Deactivating a rule which is already inactive has no effect.

DEACTIVATE-THEORY
Deactivate all the rewrite rules in the active theory.

DELETE-RRULE ***rule***
Delete a rewrite rule from TPS.

LIST-RRULES
Show all the current rewrite rules.

MAKE-ABBREV-RRULE ***name bidir***
Make a rewrite rule corresponding to a known abbreviation.

MAKE-INVERSE-RRULE ***rule newname***
Make the inverse rewrite rule of an existing rule.

MAKE-THEORY ***name extends axioms rrules other sign reflexive congruence mhelp***
Create a new theory. A theory is defined by (optionally) starting from an old theory, and adding rewrite rules and axioms. You can also attach other library objects to the theory, which will then be loaded with it. This will also make an abbreviation of the same name. All of the objects in the theory should be defined in the library.

PERMUTE-RRULES
Permute the list of rewrite rules.

REWRITE-SUPP* ***d1 d2 a apply-rrule-any* d1-hyps d2-hyps***
Rewrite a supporting line using all rewrite rules possible.

REWRITE-SUPP1 ***d1 d2 a apply-rrule-any d1-hyps d2-hyps***
Rewrite a supporting line using the first rewrite rule that applies.

SIMPLIFY-PLAN ***p2 p1 a simplify-up p2-hyps p1-hyps***
Justify a planned line using the first rewrite rule that applies.

SIMPLIFY-PLAN* ***p2 p1 a simplify-up* p2-hyps p1-hyps***
Justify a planned line using the first rewrite rule that applies.

SIMPLIFY-SUPP ***d1 d2 a simplify-down d1-hyps d2-hyps***
Rewrite a supporting line using the first rewrite rule that applies.

SIMPLIFY-SUPP* ***d1 d2 a simplify-down* d1-hyps d2-hyps***
Rewrite a supporting line using the first rewrite rule that applies.

UNREWRITE-PLAN* ***p2 p1 a unapply-rrule-any* p2-hyps p1-hyps***
Justify a planned line using all rewrite rules possible.

UNREWRITE-PLAN1 ***p2 p1 a unapply-rrule-any p2-hyps p1-hyps***
Justify a planned line using the first rewrite rule that applies.

USE-RRULES ***p2 p1 a b p2-hyps p1-hyps***
Rewrite a line. The line may be rewritten several steps, but rewrites may not be nested.

USE-THEORY ***theory***
Activate all the rewrite rules in a theory, and deactivate all other rewrite rules.

2.32 Events

DISABLE-EVENTS
Disable recording of TPS events. You will need to start a new session of TPS to enable recording of events after they have been disabled.

2.33 Statistics

DATEREC ***name type comment***
Records times used in the following processes: DIY, Mating Search, Merging Expansion Tree, Proof Transformation. All times recorded are in seconds. Internal-runtime includes GC-time. GC-time is garbage-collecting-time. I-GC-time is Internal-runtime minus GC-time. DATEREC also records the values of the flags listed in RECORDFLAGS, and will offer the user the chance to reset the provability status of a gwff in the library.

DISPLAY-TIME ***name***

Show time used in several processes: display-time diy: show the time used in DIY process display-time mating: show the time used in mating-search process display-time merge: show the time used in merging-expansion-tree process display-time eproof: show the time used in proof-transformation process display-time all: show all the times above All times are in seconds. Internal-runtime includes GC-time. GC-time is garbage-collecting-time. I-GC-time is Internal-runtime minus GC-time.

2.34 Maintenance

CLOAD ***file***

Compile and load a file.

CLOAD-MODULES ***modules***

Compile and Load a list of modules.

COMPILE-LIST ***directory-list source-only***

Returns a list of files that need to be compiled.

COMPL ***filespeclist***

Compile 1 or more files.

EXTRACT-TEST-INFO ***file***

Extract and report information from a file generated by a run of tps-test. The user has several options for what information to extract.

See Also: TPS-TEST

The options include:

1 - All Theorems Proven 2 - Theorems Proven With Times 3 - Theorems Proven With Successful Modes 4 - Theorems Proven With Times and Successful Modes 5 - Theorems and Modes That Timed Out 6 - Theorems and Modes That Failed

FILETYPE ***filename***

Type a file on the screen. TPS will look for the file in a list of directories.

GENERATE-JAVA-MENUS ***filename***

Generate Java code for menus. This command should only be used by programmers. See the TPS3 Programmer's Guide. This should be run and the resulting code appropriately inserted into TpsWin.java whenever the menu structure has been changed.

LEDIT

Call the resident Lisp editor (if there is one) inside TPS. It takes a filename as an optional argument. In most lisps, this will probably start up Emacs. In CMU lisp, this will start up Hemlock; use ^X^Z to leave Hemlock again. In some lisps, this command may not work at all.

LOAD-SLOW ***filename***

Step through loading a file.

ORGANIZE

Organizes the ENVIRONMENT help tree (e.g. after loading modules).

QLOAD ***filespec***

Load the most recent compiled or uncompiled file from your default directory, home directory, or source path. In general, the following rules are used to determine whether compiled or uncompiled file should be load in: (1) If the file name with extension '.lisp', always load the uncompiled source code. (2) If the file name without extension, then (2.1) if both compiled and uncompiled file exist, and (2.1.1) the compiled one is newer, it is loaded in. (2.1.2) the uncompiled one is newer, (2.1.2.1) if the flag 'expertflag' is NIL, always load the uncompiled source code. (2.1.2.2) if the flag 'expertflag' is T, ask user whether load the uncompiled one, or compile it and load the compiled one then. (2.2) if only the compiled one exists, load it in. (2.3) if only the uncompiled one exists, do the same as case (2.1.2)

SETUP-ONLINE-ACCESS

SETUP-ONLINE-ACCESS allows a user to set up a file of userids and passwords for remote access to a TPS server over the web. For example, this can be used by a teacher to set up a file of userids and passwords for a class to use ETPS online.

See Also: USER-PASSWD-FILE

SYS-LOAD ***modulelist***

Load all the modules in the given list, whether they are loaded already or not.

TEST-INIT

Initialize the flag TEST-THEOREMS to test a collection of theorems on a collection of modes. This command should be followed by TPS-TEST which actually tries to prove the theorems with the modes.

There are currently several possibilities:

1. Set TEST-THEOREMS to test a given set of theorems on a given set of modes. The default set of modes is determined by the value of the flag GOODMODES.

2. Set TEST-THEOREMS to test the set of modes given by the flag GOODMODES on theorems that have a bestmode in the library (determined by DEFAULT-LIB-DIR and BACKUP-LIB-DIR) but are not known to be provable by some mode in the GOODMODES list.

3. Set TEST-THEOREMS to test a set of modes given by the flag GOODMODES on all the theorems the modes are supposed to prove. (This tests whether a list of GOODMODES is still complete with respect to the corresponding list of theorems.)

4. Set TEST-THEOREMS to test all of the best modes known to the library on all the theorems listed with the best modes. By default, this will choose the first mode listed for each theorem in the bestmodes.rec file; if you choose to use multiple modes then it will test each theorem with all of the modes listed for it in that file. The examples are listed in order

from quickest to longest. (This checks that all the theorems associated with bestmodes can still be proven by these bestmodes.)

TLIST ***symbol***

Use a help function to display all of the property list of a symbol.

TLOAD ***filespec***

Load the most recent compiled or uncompiled file from your default directory, home directory, or source-path. In general, the following rules are used to determine whether compiled or uncompiled file should be load in: (1) If both compiled and uncompiled file exist, and (1.1) the compiled one is newer, it is loaded in. (1.2) the uncompiled one is newer, then (1.2.1) if the global variable core::*allow-compile-source* is T, the name of the file contains extension

TPS-TEST ***stop-on-success mate-only record moderec quiet-run expu newcore modify output timing testwin***

Attempt to prove a list of theorems.

The list of theorems, with the modes to be used, is stored as (theorem . mode) pairs in the flag TEST-THEOREMS. These theorems and modes will be fetched from the library, if they cannot be found in TPS and if you have a library. You should set DEFAULT-LIB-DIR and BACKUP-LIB-DIR appropriately. You can only do DATEREC after each theorem if you have a library you can write to.

The first argument STOP-ON-SUCCESS decides whether TPS-TEST should stop trying to prove a particular theorem with different modes after one mode has succeeded. If this is T, then after TPS-TEST proves THM with MODE1, where (THM . MODE1) is on TEST-INIT, TPS-TEST will not try to prove (THM . MODE2) for any (THM . MODE2) on TEST-INIT. It will however, continue to try to prove other theorems on TEST-INIT with different modes (if there are any).

Quiet running uses the mode QUIET to switch off as much screen output as possible.

You can EXPUNGE between proofs (this will reduce the amount of memory required, but will mean that other expansion proofs in the memory may be lost; it will also re-assert your default flag values between each proof). Expunging does not really recover all the space used by TPS, so many repeated proof attempts will result in running out of memory. To remedy this situation, TPS-TEST can start a new core image for each proof attempt. In this case, each core image will start with a fresh memory. (When this option is chosen, expunging is irrelevant.) Certain operating systems and versions of Lisp may not support this option.

If TPS-TEST is running a new core image for each proof attempt, the user can interrupt the slave core image using Control-C. This should throw one to the debugger level of the slave image. In Allegro Lisp, :res will cause the slave to die and throw the user to the debugger level of the master core image. Another :res will return the user to the TPS top level of the master core image.

If the argument MODIFY is T, then the flag TEST-MODIFY can be used to change flag settings after loading each mode but before searching. See the help message for TEST-MODIFY for more information.

In versions of Common Lisp with multiprocessing (e.g., Allegro 5.0 or later), the user can specify a time limit for each proof attempt. The user can also ask TPS-TEST to iterate trying every (THM . MODE) on TEST-THEOREMS, increasing the time limit by a factor on each iteration. A (THM . MODE) is only tried again with a longer time if it timed out on the previous attempt. When multiprocessing is not available (or if the user specifies an INFINITE time limit), TPS will search for a proof using a given mode as long as permitted by that mode.

If TPS-TEST encounters a bug, it will go on to the next (THM . MODE) pair.

The output file is kept independently of DATEREC records, and consists of a record for each (THM . MODE) pair stating that the theorem was proved at a certain time using a certain mode, or that the proof terminated with proof lines still remaining or that tps encountered an error. Timing information can also be sent to the short file if necessary.

If the short file already exists, the old copy will be renamed by adding .bak to its name.

See the help messages for TEST-THEOREMS, TEST-INIT and TEST-MODIFY for more information.

TPS-TEST2 ***searchlist quiet-run expu output testwin***

Like TPS-TEST (see the help message for that command), but calls the TEST top level and attempts to prove one theorem repeatedly with several different values of some crucial flags, to see how the time taken will vary.

TEST-THEOREMS should contain a list of dotted pairs of theorems and modes in which they can be proven; the searchlist which is used should have at least one setting in which the theorem can be proven (otherwise tps-test2 will never finish that theorem).

The output file (by default, tps-test2-output.doc) will contain a summary of the results. If this file already exists, it will be renamed by adding .bak to its name.

TPS3-SAVE

Save the current TPS3 as the new TPS3 core image.

2.35 Modules

LOADED-MODS

Returns list of loaded modules.

MODULES ***modulelist***

Load the specified modules.

UNLOADED-MODS
Returns list of unloaded modules.

2.36 Rules Module

ASSEMBLE-FILE *rule-file part-of*
Parse, build and write every rule in a given rule file. Be sure to set the correct mode (MODE RULES) before using this command.

ASSEMBLE-MOD *module*
Produce a file with rule commands for every rule file in a module.

BUILD *rule*
Process a rule without writing the resulting code to a file.

WRITE-RULE *rule filename*
Write the various functions and definitions for a rule into a file.

2.37 Lisp packages

PACK-STAT
Give information about the current status of the Lisp package structure.

UNUSE *lisp-package*
Make a Lisp package inaccessible.

USE *lisp-package*
Make a Lisp package accessible in the current Lisp package. An error will be issued by Lisp if this leads to name conflicts.

2.38 Display

DISPLAYFILE *filename bigwin*
Open a (big) window in which the contents of the given file will be displayed. Once the end of the file is reached, a message will be printed and some additional blank lines will be added. Once the end of the blank lines is reached, the window will vanish.

LS
List the files in the current directory.

2.39 Best modes

MODEREC
Attempts to create an entry in bestmodes.rec, in a similar way to the way that DATEREC works.

2.40 Library Classification

PSCHEMES
Prints a list of Library Classification Schemes in memory.

See Also: CLASS-SCHEME, CREATE-CLASS-SCHEME, PCLASS, GOTO-CLASS, CLASSIFY-CLASS, UNCLASSIFY-CLASS, CLASSIFY-ITEM, UNCLASSIFY-ITEM, FETCH-LIBCLASS, FETCH-LIBCLASS*

2.41 Bugs

BUG-DELETE *name*
Delete a bug record. Exactly the same as the library DELETE command, but will use the DEFAULT-BUG-DIR if USE-DEFAULT-BUG-DIR is T.

BUG-HELP *name*
Show the help message of a bug record.

BUG-LIST
Show all the saved bugs in the appropriate directory. See USE-DEFAULT-BUG-DIR.

BUG-RESTORE *name*
Restore a bug from the library (see USE-DEFAULT-BUG-DIR). This must have been a bug which was saved with BUG-SAVE; this command will reload all the necessary library objects, reset all the flags and reload the proof. This does NOT create a new mode; it just resets the flags.

BUG-SAVE *name comment*
Records details of a bug. Saves the current flag settings, the output of the HISTORY command, all currently loaded library objects, the current proof, the date and time and any comments (the best idea is to copy any error messages in to the "comments" prompt). This setup can then be retrieved with BUG-RESTORE. The details are saved as a MODE1, under the name that the user provides (in a file of the same name) with the assertion and library objects in other-attributes and other-remarks respectively, and the context set to BUG. The file will be saved in an appropriate directory (see USE-DEFAULT-BUG-DIR).

2.42 Interface

JAVAWIN ***fontsize popups***

Begin a Java Interface window to be used for the remainder of this TPS session.

Chapter 3

Inference Rules

The internal name of this category is SRULE.
An inference rule can be defined using DEFSRULE. Allowable properties are: `MATCHFN`, `MATCH1FN`, `SHORTFN`, `PRIORITY`.

3.1 Miscellaneous Rules

HYP

Introduce a new hypothesis line into the proof outline.

(H1) H1 $\vdash A_o$ Hyp

* (P2) H $\vdash B_o$

Transformation: (P2 ss) ==> (P2 H1 ss)

LEMMA

Introduce a Lemma.

(P1) H1 $\vdash A_o$

* (P2) H2 $\vdash B_o$

Transformation: (P2 ss) ==> (P2 P1 ss) (P1 ss)

SAME

Use the fact that two lines are identical to justify a planned line.

* (D1) H $\vdash A_o$

* (P2) H $\vdash A_o$ Same as: D1

Transformation: (P2 D1 ss) ==>

3.2 Propositional Rules

ASSOC-LEFT

Rule to associate a support line leftwards. Use before calling CASES3 or CASES4.

* (D1) H $\vdash P_o$

(D2) H $\vdash \text{`}(ASSOC - L\, P_o)$ Assoc: D1

Transformation: (pp D1 ss) ==> (pp D2 ss)

CASES

Rule of Cases.

* (D1) H $\vdash A_o \vee B_o$

(H2) H,H2 $\vdash A_o$ Case 1: D1

(P3) H,H2 $\vdash C_o$

(H4) H,H4 $\vdash B_o$ Case 2: D1

(P5) H,H4 $\vdash C_o$

* (P6) H $\vdash C_o$ Cases: D1 P3 P5

Transformation: (P6 D1 ss) ==> (P3 H2 ss) (P5 H4 ss)

CASES3

Rule of Cases.

* (D1) H $\vdash A_o \vee B_o \vee C_o$

(H2) H,H2 $\vdash A_o$ Case 1: D1

(P3) H,H2 $\vdash D_o$

(H4) H,H4 $\vdash B_o$ Case 2: D1

(P5) H,H4 $\vdash D_o$

(H6) H,H6 $\vdash C_o$ Case 3: D1

(P7) H,H6 $\vdash D_o$

* (P8) H $\vdash D_o$ Cases: D1 P3 P5 P7

Transformation: (P8 D1 ss) ==> (P3 H2 ss) (P5 H4 ss) (P7 H6 ss)

CASES4

Rule of Cases.

* (D1) H $\vdash A_o \vee B_o \vee C_o \vee D_o$

(H2) H,H2 $\vdash A_o$ Case 1: D1

(P3) H,H2 $\vdash E_o$

(H4) H,H4 $\vdash B_o$ Case 2: D1

(P5) H,H4 $\vdash E_o$

(H6) H,H6 $\vdash C_o$ Case 3: D1

(P7) H,H6 $\vdash E_o$

(H8) H,H8 $\vdash D_o$ Case 4: D1

(P9) H,H8 $\vdash E_o$

* (P10) H $\vdash E_o$ Cases: D1 P3 P5 P7 P9

Transformation: (P10 D1 ss) ==> (P3 H2 ss) (P5 H4 ss) (P7 H6 ss) (P9 H8 ss)

DEDUCT

The deduction rule.

(H1) H,H1 $\vdash A_o$ Hyp

(D2) H,H1 $\vdash B_o$

* (P3) H $\vdash A_o \supset B_o$ Deduct: D2

Transformation: (P3 ss) ==> (D2 H1 ss)

DISJ-IMP

Rule to replace a disjunction by an implication.

* (D1) H $\vdash \sim A_o \lor B_o$

(D2) H $\vdash A_o \supset B_o$ Disj-Imp: D1

Transformation: (pp D1 ss) ==> (pp D2 ss)

DISJ-IMP-L

Rule to replace a disjunction by an implication.

* (D1) H $\vdash A_o \lor B_o$

(D2) H $\vdash \sim A_o \supset B_o$ Disj-Imp-L: D1

Transformation: (pp D1 ss) ==> (pp D2 ss)

DISJ-IMP-R

Rule to replace a disjunction by an implication.

* (D1) H $\vdash A_o \lor B_o$

(D2) H $\vdash \sim B_o \supset A_o$ Disj-Imp-R: D1

Transformation: (pp D1 ss) ==> (pp D2 ss)

ECONJ

Rule to infer two conjuncts from a conjunction.

* (D1) H $\vdash A_o \land B_o$

(D2) H $\vdash A_o$ Conj: D1

(D3) H $\vdash B_o$ Conj: D1

Transformation: (pp D1 ss) ==> (pp D2 D3 ss)

EQUIV-IMPLICS

Rule to convert an equivalence into twin implications.

* (D1) H $\vdash P_o \equiv R_o$

(D2) H $\vdash [P_o \supset R_o] \land . R \supset P$ EquivImp: D1

Transformation: (pp D1 ss) ==> (pp D2 ss)

ICONJ

Rule to infer a conjunction from two conjuncts.

(P1) H $\vdash A_o$

(P2) H $\vdash B_o$

* (P3) H $\vdash A_o \land B_o$ Conj: P1 P2

Transformation: (P3 ss) ==> (P1 ss) (P2 ss)

IDISJ-LEFT

Introduce a disjunction (left version).

(P1) H $\vdash A_o$

* (P2) H $\vdash A_o \lor B_o$ Idisj-L: P1

Transformation: (P2 ss) ==> (P1 ss)

IDISJ-RIGHT

Introduce a disjunction (right version).

(P1) H $\vdash A_o$

* (P2) H $\vdash B_o \vee A_o$ Idisj-R: P1

Transformation: (P2 ss) ==> (P1 ss)

IMP-DISJ

Rule to replace an implication by a disjunction.

* (D1) H $\vdash A_o \supset B_o$

(D2) H $\vdash \sim A_o \vee B_o$ Imp-Disj: D1

Transformation: (pp D1 ss) ==> (pp D2 ss)

IMP-DISJ-L

Rule to replace an implication by a disjunction.

* (D1) H $\vdash \sim A_o \supset B_o$

(D2) H $\vdash A_o \vee B_o$ Imp-Disj-L: D1

Transformation: (pp D1 ss) ==> (pp D2 ss)

IMP-DISJ-R

Rule to replace an implication by a disjunction.

* (D1) H $\vdash \sim B_o \supset A_o$

(D2) H $\vdash A_o \vee B_o$ Imp-Disj-R: D1

Transformation: (pp D1 ss) ==> (pp D2 ss)

IMPLICS-EQUIV

Rule to convert twin implications into an equivalence.

(P1) H $\vdash [P_o \supset R_o] \wedge . R \supset P$

* (P2) H $\vdash P_o \equiv R_o$ ImpEquiv: P1

Transformation: (P2 ss) ==> (P1 ss)

INDIRECT

Rule of Indirect Proof.

(H1) H,H1 $\vdash \sim A_o$ Assume negation

(P2) H,H1 $\vdash \bot$

* (P3) H $\vdash A_o$ Indirect: P2

Transformation: (P3 ss) ==> (P2 H1 ss)

INDIRECT1

Rule of Indirect Proof Using One Contradictory Line.

(H1) H,H1 $\vdash \sim A_o$ Assume negation

(P2) H,H1 $\vdash B_o \wedge \sim B$

* (P3) H $\vdash A_o$ Indirect: P2

Transformation: (P3 ss) ==> (P2 H1 ss)

INDIRECT2

Rule of Indirect Proof Using Two Contradictory Lines.

(H1) H,H1 $\vdash \sim A_o$ Assume negation

(P2) H,H1 $\vdash B_o$

(P3) H,H1 $\vdash \sim B_o$

* (P4) H $\vdash A_o$ Indirect: P2 P3

Transformation: (P4 ss) ==> (P2 H1 ss) (P3 H1 ss)

ITRUTH

Rule to infer TRUTH

* (P1) H $\vdash \top$ Truth

Transformation: (P1 ss) ==>

MP

Modus Ponens.

(P1) H $\vdash A_o$

* (D2) H $\vdash A_o \supset B_o$

(D3) H $\vdash B_o$ MP: P1 D2

Transformation: (pp D2 ss) ==> (P1 ss) (pp D3 ss P1)

SUBST-EQUIV

Substitution of Equivalence. Usable when R and P are the same modulo the equivalence s EQUIV t.

(P1) H $\vdash P_o$

* (D2) H $\vdash s_o \equiv t_o$

(D3) H $\vdash R_o$ Sub-equiv: P1 D2

Restrictions: (SAME-MODULO-EQUALITY P_o R_o s_o t_o) Transformation: (pp D2 ss) ==>

3.3 Negation Rules

ABSURD

Rule of Intuitionistic Absurdity.

(P1) H $\vdash \bot$

* (P2) H $\vdash A_o$ Absurd: P1

Transformation: (P2 ss) ==> (P1 ss)

ENEG

Rule of Negation Elimination.

* (D1) H $\vdash \sim A_o$

(P2) H $\vdash A_o$

* (P3) H $\vdash \bot$ NegElim: D1 P2

Transformation: (P3 D1 ss) ==> (P2 ss)

INEG

Rule of Negation Introduction

(H1) H,H1 $\vdash A_o$ Hyp

(P2) H,H1 $\vdash \bot$

* (P3) H $\vdash \sim A_o$ NegIntro: P2

Transformation: (P3 ss) ==> (P2 H1 ss)

NNF

Put Wff in Negation Normal Form.

* (D1) H $\vdash A_o$

(D2) H $\vdash$ ‘$(NEG-NORM\, A_o)$ NNF: D1

Restrictions: (NON-ATOMIC-OR-TRUTHVALUE A_o) Transformation: (pp D1 ss) ==> (pp D2 s

NNF-EXPAND

Expand Wff from Negation Normal Form.

(P1) H $\vdash$ ‘$(NEG-NORM\, A_o)$

* (P2) H $\vdash A_o$ NNF-Expand: P1

Restrictions: (NON-ATOMIC A_o) Transformation: (P2 ss) ==> (P1 ss)

PULLNEG

Pull out negation.

(P1) H $\vdash$ ‘$(PUSH-NEGATION[\sim A_o])$

* (P2) H $\vdash \sim A_o$ Neg: P1

Restrictions: (NON-ATOMIC A_o) Transformation: (P2 ss) ==> (P1 ss)

PUSHNEG

Push in negation.

* (D1) H $\vdash \sim A_o$

(D2) H $\vdash$ ‘$(PUSH-NEGATION[\sim A_o])$ Neg: D1

Restrictions: (NON-ATOMIC-OR-TRUTHVALUE A_o) Transformation: (pp D1 ss) ==> (pp D2 s

3.4 Quantifier Rules

AB*

Rule to alphabetically change embedded quantified variables.

* (D1) H $\vdash A_o$

(D2) H $\vdash B_o$ AB: D1

Restrictions: (WFFEQ-AB A_o B_o) Transformation: (pp D1 ss) ==> (pp D2 ss)

ABE

Rule to change a top level occurrence of an existentially quantified variable.

* (D1) H $\vdash \exists x_\alpha A_o$

(D2) H $\vdash \exists y_\alpha$ ‘$(S\, y\, x_\alpha\, A_o)$ AB: y_α D1

Restrictions: (FREE-FOR y_α x_α A_o) (IS-VARIABLE y_α) (IS-VARIABLE x_α) (NOT-FREE

ABU

Rule to change a top level occurrence of a universally quantified variable.

(P1) H ⊢ $\forall y_\alpha$ ‘$(S\, y\, x_\alpha\, A_o)$

* (P2) H ⊢ $\forall x_\alpha\, A_o$ AB: x_α P1

Restrictions: (FREE-FOR y_α x_α A_o) (IS-VARIABLE y_α) (IS-VARIABLE x_α) (NOT-FREE

EGEN

Rule of Existential Generalization.

(P1) H ⊢ ‘$(LCONTR[[\lambda\, x_\alpha\, A_o]\, t_\alpha])$

* (P2) H ⊢ $\exists x_\alpha\, A_o$ EGen: t_α P1

Restrictions: (IS-VARIABLE x_α) Transformation: (P2 ss) ==> (P1 ss)

RULEC

RuleC

* (D1) H ⊢ $\exists x_\alpha\, B_o$

(H2) H,H2 ⊢ ‘$(LCONTR[[\lambda\, x_\alpha\, B_o]\, y_\alpha])$ Choose: y_α D1

(D3) H,H2 ⊢ A_o

* (P4) H ⊢ A_o RuleC: D1 D3

Restrictions: (IS-VARIABLE y_α) (NOT-FREE-IN-HYPS y_α) (NOT-FREE-IN y_α [$\exists x_\alpha$ B_o

RULEC1

RuleC1 – the special case of RULEC where the chosen variable has the same name as the bound variable.

* (D1) H ⊢ $\exists x_\alpha\, B_o$

(H2) H,H2 ⊢ B_o Choose: x_α D1

(D3) H,H2 ⊢ A_o

* (P4) H ⊢ A_o RuleC: D1 D3

Restrictions: (NOT-FREE-IN-HYPS x_α) (IS-VARIABLE x_α) (NOT-FREE-IN x_α A_o) Trans

UGEN

Rule of Universal Generalization.

(P1) H ⊢ A_o

* (P2) H ⊢ $\forall x_\alpha\, A_o$ UGen: x_α P1

Restrictions: (IS-VARIABLE x_α) (NOT-FREE-IN-HYPS x_α) Transformation: (P2 ss) ==>

UI

Rule of Universal Instantiation.

* (D1) H ⊢ $\forall x_\alpha\, A_o$

(D2) H ⊢ ‘$(LCONTR[[\lambda\, x_\alpha\, A_o]\, t_\alpha])$ UI: t_α D1

Restrictions: (IS-VARIABLE x_α) Transformation: (pp D1 ss) ==> (pp D2 D1 ss)

3.5 Substitution Rules

SUBSTITUTE

Rule to substitute a term for a variable.

* (D1) H $\vdash A_o$

(D2) H $\vdash$ ‘$(S\, t_\alpha\, x_\alpha\, A_o)$ Subst: $t_\alpha\, x_\alpha$ D1

Restrictions: (NOT-FREE-IN-HYPS x_α) (IS-VARIABLE x_α) (FREE-FOR t_α x_α A_o) Tran

3.6 Equality Rules

EQUIV-EQ

Rule to infer a line from one which is equal up to definitions, lambda conversion, alphabetic change of bound variables and the Leibniz definition of the symbol $=$. You may use the editor command EXPAND= to create the desired line from the existing one.

* (D1) H $\vdash A_o$

(D2) H $\vdash B_o$ Equiv-eq: D1

Restrictions: (WFFEQ-DEFEQ A_o B_o) Transformation: (pp D1 ss) ==> (pp D2 ss)

EQUIV-EQ-CONTR

Rule to contract the outermost instance of the Leibniz definition of equality into instances of the symbol $=$.

(P1) H $\vdash$ ‘$(INSTANTIATE-TOP-EQUALITY\, A_o)$

* (P2) H $\vdash A_o$ Equiv-eq: P1

Transformation: (P2 ss) ==> (P1 ss)

EQUIV-EQ-CONTR*

Rule to contract all instances of the Leibniz definition of equality into instances of the symbol $=$.

(P1) H $\vdash$ ‘$(INSTANTIATE-EQUALITIES\, A_o)$

* (P2) H $\vdash A_o$ Equiv-eq: P1

Transformation: (P2 ss) ==> (P1 ss)

EQUIV-EQ-EXPD

Rule to expand the outermost equality using the Leibniz definition.

* (D1) H $\vdash A_o$

(D2) H $\vdash$ ‘$(INSTANTIATE-TOP-EQUALITY\, A_o)$

Equiv-eq: D1

Transformation: (pp D1 ss) ==> (pp D2 ss)

EQUIV-EQ-EXPD*

Rule to expand all equalities using the Leibniz definition.

* (D1) H $\vdash A_o$

(D2) H $\vdash$ ‘$(INSTANTIATE-EQUALITIES\, A_o)$

Equiv-eq: D1

Transformation: (pp D1 ss) ==> (pp D2 ss)

EXT=

Rule of Extensionality.

(P1) H ⊢ $\forall x_\beta . f_{\alpha\beta}\, x = g_{\alpha\beta}\, x$

* (P2) H ⊢ $f_{\alpha\beta} = g_{\alpha\beta}$ Ext=: P1

Restrictions: (IS-VARIABLE x_β) (NOT-FREE-IN x_β $f_{\alpha\beta}$) (NOT-FREE-IN x_β $g_{\alpha\beta}$) Trans

EXT=0

Rule to convert equality at type o into an equivalence.

(P1) H ⊢ $P_o \equiv R_o$

* (P2) H ⊢ $P_o = R_o$ Ext=: P1

Transformation: (P2 ss) ==> (P1 ss)

LET

Bind a variable to a term.

(D1) H ⊢ $A_\alpha = A$ Refl=

(D2) H ⊢ $\exists x_\alpha . x = A_\alpha$ EGen: x_α D1

(H3) H,H3 ⊢ $x_\alpha = A_\alpha$ Choose: x_α

(P4) H,H3 ⊢ C_o

* (P5) H ⊢ C_o RuleC: D2 P4

Restrictions: (NOT-FREE-IN-HYPS x_α) (IS-VARIABLE x_α) (NOT-FREE-IN x_α C_o) Transi

SUBST=

Substitution of Equality. Usable when R and P are the same modulo the equality s=t.

(P1) H ⊢ P_o

* (D2) H ⊢ $s_\alpha = t_\alpha$

(D3) H ⊢ R_o Sub=: P1 D2

Restrictions: (SAME-MODULO-EQUALITY P_o R_o s_α t_α) Transformation: (pp D2 ss) ==>

SUBST=L

Substitution of Equality. Replaces some occurrences of the left hand side by the right hand side.

(P1) H ⊢ P_o

* (D2) H ⊢ $s_\alpha = t_\alpha$

(D3) H ⊢ R_o Subst=: P1 D2

Restrictions: (R-PRIME-RESTR s_α P_o t_α R_o) Transformation: (pp D2 ss) ==> (P1 ss

SUBST=R

Substitution of Equality. Replaces some occurrences of the right hand side by the left hand side.

(P1) H ⊢ P_o

* (D2) H ⊢ $t_\alpha = s_\alpha$

(D3) H ⊢ R_o Subst=: P1 D2

Restrictions: (R-PRIME-RESTR s_α P_o t_α R_o) Transformation: (pp D2 ss) ==> (P1 ss

SYM=

Rule of Symmetry of Equality.

(P1) H $\vdash A_\alpha = B_\alpha$

* (P2) H $\vdash B_\alpha = A_\alpha$ Sym=: P1

Transformation: (P2 ss) ==> (P1 ss)

3.7 Definition Rules

EDEF

Rule to eliminate first definition, left to right.

* (D1) H $\vdash A_o$

(D2) H $\vdash$ ‘$(INST - DEF\, A_o)$ Defn: D1

Restrictions: (CONTAINS-DEFN A_o) Transformation: (pp D1 ss) ==> (pp D2 ss)

EQUIV-WFFS

Rule to assert equivalence of lines up to definition.

* (D1) H $\vdash P_o$

(D2) H $\vdash R_o$ EquivWffs: D1

Restrictions: (WFFEQ-DEF P_o R_o) Transformation: (pp D1 ss) ==> (pp D2 ss)

IDEF

Rule to introduce a definition.

(P1) H $\vdash$ ‘$(INST - DEF\, A_o)$

* (P2) H $\vdash A_o$ Defn: P1

Restrictions: (CONTAINS-DEFN A_o) Transformation: (P2 ss) ==> (P1 ss)

3.8 Lambda Conversion Rules

BETA*

Rule to infer a line from one which is equal up to lambda conversion using beta rule (but NOT eta rule) and alphabetic change of bound variables.

* (D1) H $\vdash A_o$

(D2) H $\vdash B_o$ Beta Rule: D1

Restrictions: (WFFEQ-AB-BETA A_o B_o) Transformation: (pp D1 ss) ==> (pp D2 ss)

ETA*

Rule to infer a line from one which is equal up to lambda conversion using eta rule (but NOT beta rule) and alphabetic change of bound variables.

* (D1) H $\vdash A_o$

(D2) H $\vdash B_o$ Eta Rule: D1

Restrictions: (WFFEQ-AB-ETA A_o B_o) Transformation: (pp D1 ss) ==> (pp D2 ss)

LAMBDA*

Rule to infer a line from one which is equal up to lambda conversion using both beta and eta rules and alphabetic change of bound variables.

* (D1) H $\vdash A_o$

(D2) H $\vdash B_o$ Lambda=: D1

Restrictions: (WFFEQ-AB-LAMBDA A_o B_o) Transformation: (pp D1 ss) ==> (pp D2 ss)

LCONTR*

Rule to put an inferred line into Lambda-normal form using both beta and eta conversion.

* (D1) H $\vdash A_o$

(D2) H $\vdash `(LNORM\, A_o)$ Lambda: D1

Transformation: (pp D1 ss) ==> (pp D2 ss)

LCONTR*-BETA

Rule to put an inferred line into beta-normal form.

* (D1) H $\vdash A_o$

(D2) H $\vdash `(LNORM - BETA\, A_o)$ Beta rule: D1

Transformation: (pp D1 ss) ==> (pp D2 ss)

LCONTR*-ETA

Rule to put an inferred line into eta-normal form.

* (D1) H $\vdash A_o$

(D2) H $\vdash `(LNORM - ETA\, A_o)$ Eta rule: D1

Transformation: (pp D1 ss) ==> (pp D2 ss)

LEXPD*

Rule to put a planned line into Lambda-normal form using both beta and eta conversion.

(P1) H $\vdash `(LNORM\, A_o)$

* (P2) H $\vdash A_o$ Lambda: P1

Transformation: (P2 ss) ==> (P1 ss)

LEXPD*-BETA

Rule to put a planned line into beta-normal form.

(P1) H $\vdash `(LNORM - BETA\, A_o)$

* (P2) H $\vdash A_o$ Beta rule: P1

Transformation: (P2 ss) ==> (P1 ss)

LEXPD*-ETA

Rule to put a planned line into eta-normal form.

(P1) H $\vdash `(LNORM - ETA\, A_o)$

* (P2) H $\vdash A_o$ Eta rule: P1

Transformation: (P2 ss) ==> (P1 ss)

3.9 Rewriting commands

REWRITE-SUPP*

Rewrite a supporting line using all rewrite rules possible.

```
*    (D1)   H    ⊢ A_o
     (D2)   H    ⊢ `(APPLY-RRULE-ANY* A_o)          Rewrites:  D1
 Transformation: (pp D1 ss) ==> (pp D2 ss)
```

REWRITE-SUPP1

Rewrite a supporting line using the first rewrite rule that applies.

```
*    (D1)   H    ⊢ A_o
     (D2)   H    ⊢ `(APPLY-RRULE-ANY A_o)            Rewrite:  D1
 Transformation: (pp D1 ss) ==> (pp D2 ss)
```

SIMPLIFY-PLAN

Justify a planned line using the first rewrite rule that applies.

```
(P1)   H    ⊢ `(SIMPLIFY-UP A_o)
 *    (P2)   H    ⊢ A_o                               Rewrite:  P1
 Transformation: (P2 ss) ==> (P1 ss)
```

SIMPLIFY-PLAN*

Justify a planned line using the first rewrite rule that applies.

```
(P1)   H    ⊢ `(SIMPLIFY-UP* A_o)
 *    (P2)   H    ⊢ A_o                               Rewrite:  P1
 Transformation: (P2 ss) ==> (P1 ss)
```

SIMPLIFY-SUPP

Rewrite a supporting line using the first rewrite rule that applies.

```
*    (D1)   H    ⊢ A_o
     (D2)   H    ⊢ `(SIMPLIFY-DOWN A_o)              Rewrite:  D1
 Transformation: (pp D1 ss) ==> (pp D2 ss)
```

SIMPLIFY-SUPP*

Rewrite a supporting line using the first rewrite rule that applies.

```
*    (D1)   H    ⊢ A_o
     (D2)   H    ⊢ `(SIMPLIFY-DOWN* A_o)             Rewrite:  D1
 Transformation: (pp D1 ss) ==> (pp D2 ss)
```

UNREWRITE-PLAN*

Justify a planned line using all rewrite rules possible.

```
(P1)   H    ⊢ `(UNAPPLY-RRULE-ANY* A_o)
 *    (P2)   H    ⊢ A_o                              Rewrites:  P1
 Transformation: (P2 ss) ==> (P1 ss)
```

UNREWRITE-PLAN1

Justify a planned line using the first rewrite rule that applies.

```
(P1)   H    ⊢ `(UNAPPLY-RRULE-ANY A_o)
 *    (P2)   H    ⊢ A_o                               Rewrite:  P1
 Transformation: (P2 ss) ==> (P1 ss)
```

USE-RRULES

Rewrite a line. The line may be rewritten several steps, but rewrites may not be nested.

```
(P1)   H     ⊢ $A_o$
 *    (P2)   H     ⊢ $B_o$                                         Rewrite:  P1
 Restrictions: (INSTANCE-OF-REWRITING  $A_o$  $B_o$) Transformation: (P2 ss) ==> (P1 ss)
```

Chapter 4

Extensional Sequent Commands

The internal name of this category is EXTSEQCMD.
An extensional sequent command can be defined using DEFEXTSEQ. Allowable properties are: `EXTSEQ-ARGTYPES`, `EXTSEQ-ARGNAMES`, `EXTSEQ-ARGHELP`, `EXTSEQ-DEFAULTFNS`, `EXTSEQ-MAINFNS`, `MHELP`.

4.1 Top Levels

LEAVE Leave EXT-SEQ to the next enclosing top level.

4.2 Proof Translation

CUTFREE-TO-EDAG Translate a complete, cut-free extensional sequent derivation to an extensional expansion dag proof. The conclusion of the derivation must be a sequent with a single formula.

4.3 Extensional Sequent Entering

DELETE Delete Lines in an existing derivation

EXPAND-ALL-DERIVED-RULES Remove all applications of derived rules in terms of basic rules. The derived rules include: false-, and-, and+, implies-, implies+, equiv-, equiv+, exists-, exists+

EXPAND-ALL-INITS-AND-REFLS Remove all applications of Inits and Refls in terms of basic rules.

INTRODUCE-GAP Introduce a gap in an existing derivation.

PROOFLIST Print a list of all extensional sequent derivations or partial derivations currently in memory. Also prints the final sequent of each proof.

PROVE Start a sequent calculus derivation for a sequent with one wff. Use WEAKEN to add more wffs to the main sequent.

RECONSIDER Reconsider an extensional sequent derivation. The following proofs are in memory:

For more details, use the PROOFLIST command.

SQUEEZE Removes unnecessary gaps from the sequent derivation.

WEAKEN Weaken the sequent calculus derivation by adding a wff.

4.4 Extensional Sequent Printing

PALL Print all the lines in the current extensional sequent derivation.

PPLAN Print a planned line

PSTATUS Give the current status of the extensional sequent derivation.

4.5 Extensional Sequent Rules

ALL+ Infer (p2) Gamma,[FORALL x M] from (p1) Gamma,[y/x]M.

ALL- Infer (p2) Gamma, [FORALL x M] from (p1) Gamma, [trm/x]M.

CONTR Infer (p2) Gamma,A from (p1) Gamma,A,A.

CUT From (p1) Gamma, C and (p2) Gamma, C infer (p3) Gamma

DEC From (p1) Gamma, [A1 = B1] . . . (pn) Gamma, [An = Bn] infer (p) Gamma, [[H A1 . . . An] = [H B1 . . . Bn]]

DNEG Infer (p2) Gamma, A from (p1) Gamma,A

EQFUNC Infer (p2) Gamma, forall x M from (p1) Gamma, [trm/x]M.

EQO From (p1) Gamma, A, B and (p2) Gamma, A, B infer (p3) Gamma, [A = B]

EQUIVWFFS+ Infer (p2) Gamma,A from (p1) Gamma,B where B is obtained from A by expanding an abbreviation at the head of A if A is not an equation. If A is an equation of base type other than O, the abbreviation must be at the head of the left or right side.

EQUIVWFFS- Infer (p2) Gamma, A from (p1) Gamma, B where B is obtained from A by expanding an abbreviation at the head of A if A is not an equation. If A is an equation of base type other than O, the abbreviation must be at the head of the left or right side.

EUNIF1 From (p1) Gamma, [a = b], [a = c] and (p2) Gamma, [a = b], [b = d] infer (p3) Gamma, [a = b], [c = d]

EUNIF2 From (p1) Gamma, [a = b], [a = d] and (p2) Gamma, [a = b], [b = c] infer (p3) Gamma, [a = b], [c = d]

EXTFUNC Infer (p2) Gamma,forall x M from (p1) Gamma,[a/x]M.

EXTO From (p1) Gamma, A, B and (p2) Gamma, A, B infer (p3) Gamma, [A = B]

INIT Infer (p) Gamma, A, A

INITEQ From (p1) Gamma, [A1 = B1] . . . (pn) Gamma, [An = Bn] infer (p) Gamma, [P A1 . . . An], [P B1 . . . Bn]

INTERNALIZE+ Infer (p2) Gamma,A from (p1) Gamma,# (A) where # (A) is the 'externalized' version of A. This corresponds to the # rule in Chad E. Brown's thesis.

INTERNALIZE- Infer (p2) Gamma, A from (p1) Gamma, # (A) where # (A) is the 'externalized' version of A. This corresponds to the # rule in Chad E. Brown's thesis.

LAM Infer (p2) Gamma,A from (p1) Gamma,N where N is the lambda normal form of A

OR+ From (p1) Gamma, A, B infer (p3) Gamma, [A OR B]

OR- From (p1) Gamma, A and (p2) Gamma, B infer (p3) Gamma, [A OR B]

REFL Infer (p) Gamma, t = t

TRUE+ Infer (p) Gamma, TRUTH

4.6 Extensional Sequent Derived Rules

AND+ From (p1) Gamma, A and (p2) Gamma, B infer (p3) Gamma, [A AND B]

AND- From (p1) Gamma, A, B infer (p3) Gamma, [A AND B]

EQUIV+ From (p1) Gamma, A, B and (p2) Gamma, A, B infer (p3) Gamma, [A EQUIV B]

EQUIV- From (p1) Gamma, A, B and (p2) Gamma, A, B infer (p3) Gamma, [A EQUIV B]

EXISTS+ Infer (p2) Gamma,[EXISTS x M] from (p1) Gamma,[trm/x]M.

EXISTS- Infer (p2) Gamma, [EXISTS x M] from (p1) Gamma, [y/x]M.

FALSE- Infer (p) Gamma, FALSEHOOD

IMPLIES+ From (p1) Gamma, A, B infer (p3) Gamma, [A IMPLIES B]

IMPLIES- From (p1) Gamma, A and (p2) Gamma, B infer (p3) Gamma, [A IMPLIES B]

4.7 Extensional Sequent Files

RESTOREPROOF Reads an extensional sequent derivation from a file created by SAVEPROOF in the EXT-SEQ top level and makes it the current derivation. A security feature prevents the restoration of saved proofs which have been altered in any way. Retrieve any definitions which are used in the proof and stored in the library before restoring the proof. If you don't specify a directory, it will first try your home directory and then all the directories listed in SOURCE-PATH.

SAVEPROOF Saves the current natural deduction proof to the specified file in a form in which it can be restored. Use RESTOREPROOF to restore the proof. Overwrites the file if it already exists.

SCRIBEPROOF Print the current proof into a MSS file. After leaving TPS, run this .MSS file through Scribe and print the resulting file.

TEXPROOF Print the current proof into a tex file. After leaving tps, run this .tex file through tex and print the resulting file.

Many flags affect the output of texproof. See: USE-INTERNAL-PRINT-MODE, TURNSTILE-INDENT-AUTO, TURNSTILE-INDENT, LATEX-EMULATION, TEX-MIMIC-SCRIBE, PPWFFLAG, DISPLAYWFF, INFIX-NOTATION, PAGELENGTH, PAGEWIDTH, TEX-BREAK-BEFORE-SYMBOLS, LOCALLEFTFLAG, SCOPE, ALLSCOPEFLAG, USE-DOT, FIRST-ORDER-PRINT-MODE, FILLINEFLAG, ATOMVALFLAG.

4.8 Compound

GO2 Apply all possible extensional sequent tactics.

Chapter 5

Tactics

The internal name of this category is TACTIC.
A tactic can be defined using DEFTACTIC. Allowable properties are: `NAT-DED`, `ETREE-NAT`, `MATE-SRCH`, `EXT-SEQ`.

5.1 Compound

ALL+TAC Defined for the following uses:

> **EXT-SEQ:** is a primitive tactic.

ALL-TAC Defined for the following uses:

> **EXT-SEQ:** is a primitive tactic.

AND+TAC Defined for the following uses:

> **EXT-SEQ:** is a primitive tactic.

AND-TAC Defined for the following uses:

> **EXT-SEQ:** is a primitive tactic.

AUTO-TAC Defined for the following uses:

> **NAT-DED:** (REPEAT (ORELSE MIN-PROP DIY-TAC)) Does minimal propositional actions then calls mating search if necessary, and translates the resulting proof.

BOOK-TAC Defined for the following uses:

> **ETREE-NAT:** (ORELSE SAME-TAC UNSPONSOR-TAC UNNEC-EXP-TAC)

COMPLETE-TRANSFORM*-TAC Defined for the following uses:

ETREE-NAT: (ORELSE (CALL PRINT-ROUTINES) SAME-TAC NNF-TAC ABSURD-TAC TRUTH-TAC REFL=-TAC (IFTHEN USE-RULEP-TAC RULEP-TAC) (MAKE-ROOM :USE NAT-DED) DUPLICATE-SUPPORT-TAC (THEN** (IFTHEN USE-RULEP-TAC ECONJ*-TAC ECONJ-TAC) UNSPONSOR-TAC) DEDUCT-TAC REWRITE-SLINE-TAC REWRITE-PLINE-TAC RULEC-TAC UGEN-TAC EGEN-TAC (THEN** UI-TAC UNSPONSOR-TAC) (THEN** UNNEC-EXP-TAC UNSPONSOR-TAC) (THEN** IDISJ-TAC UNSPONSOR-TAC) (THEN** (IFTHEN USE-RULEP-TAC ICONJ*-TAC ICONJ-TAC) UNSPONSOR-TAC) (THEN** CASES-TAC UNSPONSOR-TAC) PUSHNEG-TAC ML::NEG-EQUIV-SLINE-TAC NEG-NEG-PLAN-TAC INEG-TAC SUBST=-TAC MP-TAC CLASS-DISJ-TAC INDIRECT2-TAC INESS-PLINE-TAC NEG-AND-SLINE-TAC NEG-AND-PLAN-TAC NEG-EQUAL-SLINE-TAC INDIRECT-TAC)

COMPLETE-TRANSFORM-TAC Defined for the following uses:

ETREE-NAT: (REPEAT COMPLETE-TRANSFORM*-TAC)

CONTRACT-TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

DEC+TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

DIY-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Calls matingsearch procedure specified by the flag DEFAULT-MS on current planned line and its supports, then translates the expansion proof to natural deduction. The actual supports used will be the universal closure of the supports over any free variables which are not free in their hypotheses.

ELIM-DEFNS-TAC Defined for the following uses:

NAT-DED: (ORELSE EDEF-TAC IDEF-TAC)

EQFUNC-TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

EQO-TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

EQUIV+TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

EQUIV-TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

EQUIVWFFS+TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

EQUIVWFFS-TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

EUNIF1-TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

EUNIF2-TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

EXISTS+TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

EXISTS-TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

EXTFUNC+TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

EXTO+TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

FALSE-TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

GO2-TAC Defined for the following uses:

NAT-DED: (REPEAT (ORELSE (CALL PRINT-ROUTINES) SAME-TAC REFL=-TAC SYM=-TAC RULEP-TAC PROP-INTRO-RULES-TAC PROP-ELIM-RULES-TAC PUSHNEG-TAC UGEN-TAC RULEC-TAC SUB=-TAC PULLNEG-TAC INDIRECT-EXISTS-PLINE-TAC INDIRECT-DISJ-PLINE-TAC EQUIV-EQ-CONTR-TAC EQUIV-EQ-EXPD-TAC EXT=-TAC EXT=0-TAC ELIM-DEFNS-TAC LEXPD*-VARY-TAC LCONTR*-VARY-TAC))

EXT-SEQ: (REPEAT (ORELSE (CALL PRINT-ROUTINES) TRUE+TAC FALSE-TAC INIT-TAC REFL+TAC LAMBDA-TAC NOT-TAC OR+TAC AND-TAC IMPLIES+TAC ALL+TAC EXISTS-TAC EXTFUNC+TAC OR-TAC AND+TAC IMPLIES-TAC EQUIV+TAC EQUIV-TAC EXTO+TAC EQO-TAC ALL-TAC EXISTS+TAC EQFUNC-TAC EQUIVWFFS+TAC EQUIVWFFS-TAC INITEQ-TAC DEC+TAC EUNIF1-TAC EUNIF2-TAC CONTRACT-TAC INTERNALIZE+TAC INTERNALIZE-TAC))

IMPLIES+TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

IMPLIES-TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

INIT-TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

INITEQ-TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

INTERNALIZE+TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

INTERNALIZE-TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

LAMBDA-TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

MIN-PROP Defined for the following uses:

NAT-DED: (ORELSE SAME-TAC (IFTHEN USE-RULEP-TAC RULEP-TAC) TRUTH-TAC ABSURD-TAC INDIRECT2-TAC MAKE-ROOM DEDUCT-TAC (IFTHEN USE-RULEP-TAC ECONJ*-TAC ECONJ-TAC) (IFTHEN USE-RULEP-TAC ICONJ*-TAC ICONJ-TAC))

MONSTRO-TAC Defined for the following uses:

NAT-DED: (REPEAT (ORELSE (CALL PRINT-ROUTINES) SAME-TAC REFL=-TAC SYM=-TAC RULEP-TAC PROP-INTRO-RULES-TAC PROP-ELIM-RULES-TAC PUSHNEG-TAC UGEN-TAC RULEC-TAC SUB=-TAC PULLNEG-TAC UI-HERBRAND-TAC INDIRECT-EXISTS-PLINE-TAC INDIRECT-DISJ-PLINE-TAC EQUIV-EQ-CONTR-TAC EQUIV-EQ-EXPD-TAC EXT=-TAC EXT=0-TAC ELIM-DEFNS-TAC LEXPD*-VARY-TAC LCONTR*-VARY-TAC))

NOT-TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

OR+TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

OR-TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

PFENNING*-TAC Defined for the following uses:

ETREE-NAT: (ORELSE (CALL PRINT-ROUTINES) SAME-TAC NNF-TAC ABSURD-TAC TRUTH-TAC REFL=-TAC (MAKE-ROOM :USE NAT-DED) DUPLICATE-SUPPORT-TAC (THEN** ECONJ-TAC UNSPONSOR-TAC) DEDUCT-TAC REWRITE-SLINE-TAC REWRITE-PLINE-TAC RULEC-TAC UGEN-TAC EGEN-TAC (THEN** UI-TAC UNSPONSOR-TAC) (THEN** UNNEC-EXP-TAC UNSPONSOR-TAC) (THEN** IDISJ-TAC UNSPONSOR-TAC) (THEN** ICONJ-TAC UNSPONSOR-TAC) (THEN** CASES-TAC UNSPONSOR-TAC) INEG-TAC ENEG-TAC SUBST=L-TAC SUBST=R-TAC MP-TAC (IFTHEN USE-SYMSIMP-TAC SYMSIMP-TAC CLASS-DISJ-TAC) INESS-PLINE-TAC INDIRECT-TAC NEG-REW-SLINE-TAC ML::NEG-EQUIV-SLINE-TAC) Intended to be the same as the tactics advocated in Pfenning's thesis.

PFENNING-TAC Defined for the following uses:

ETREE-NAT: (REPEAT PFENNING*-TAC) Intended to be the same as the tactics advocated in Pfenning's thesis.

PLINE-TAC Defined for the following uses:

ETREE-NAT: (ORELSE DEDUCT-TAC ICONJ-TAC IDISJ-RIGHT-TAC IDISJ-LEFT-TAC IMPLICS-EQUIV-TAC LEXPD*-VARY-TAC AB-PLAN-TAC EQUIV-WFFS-PLAN-TAC EQUALITY-PLAN-TAC RULEQ-PLAN-TAC UGEN-TAC EGEN-TAC TRUTH-TAC)

PROP-ELIM-RULES-TAC Defined for the following uses:

NAT-DED: (ORELSE INDIRECT2-TAC MAKE-ROOM ECONJ*-TAC CASES-TAC EQUIV-IMPLICS-TAC)

PROP-INTRO-RULES-TAC Defined for the following uses:

NAT-DED: (ORELSE TRUTH-TAC ABSURD-TAC MAKE-ROOM ICONJ*-TAC DEDUCT-TAC INEG-TAC IMPLICS-EQUIV-TAC)

REFL+TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

REWRITE-PLINE-P-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.Returns success if planned line represents a rewrite node.

REWRITE-PLINE-TAC Defined for the following uses:

ETREE-NAT: (IFTHEN (REWRITE-PLINE-P-TAC) (ORELSE AB-PLAN-TAC EQUALITY-PLAN-TAC EQUIV-WFFS-PLAN-TAC RULEQ-PLAN-TAC LEXPD*-VARY-TAC IMPLICS-EQUIV-TAC ML::TRUTHP-REWRITE-PLAN-TAC DISJ-EQUIV-TAC))

REWRITE-SLINE-TAC Defined for the following uses:

ETREE-NAT: (IFTHEN (REWRITE-SLINE-P-TAC) (ORELSE AB-SLINE-TAC EQUALITY-SLINE-TAC EQUIV-WFFS-SLINE-TAC RULEQ-SLINE-TAC LCONTR*-VARY-TAC EQUIV-DISJ-TAC EQUIV-IMPLICS-TAC))

SLINE-TAC Defined for the following uses:

ETREE-NAT: (ORELSE ECONJ-TAC CASES-TAC MP-TAC UI-TAC RULEC-TAC ML::NEG-NEG-TAC NEG-AND-SLINE-TAC NEG-OR-SLINE-TAC NEG-IMP-SLINE-TAC NEG-SEL-SLINE-TAC NEG-EXP-SLINE-TAC EQUIV-DISJ-TAC EQUIV-IMPLICS-TAC LCONTR*-VARY-TAC EQUIV-WFFS-SLINE-TAC AB-SLINE-TAC RULEQ-SLINE-TAC EQUALITY-SLINE-TAC)

SUB=-TAC Defined for the following uses:

NAT-DED: (ORELSE SUBST=L-TAC SUBST=R-TAC)

TRUE+TAC Defined for the following uses:

EXT-SEQ: is a primitive tactic.

5.2 Propositional

ABSURD-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.If a support line is FALSEHOOD applies absurdity rule.

ETREE-NAT: is a primitive tactic.If a support line is FALSEHOOD applies absurdity rule.

BACKCHAIN-LEMMA-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If a support line is an implication, sets up a symmetric simplification problem using the antecedent of the implication in the lemma. Then symmetric simplification is performed.

BASIC-PROP*-TAC Defined for the following uses:

ETREE-NAT: (ORELSE SAME-TAC ABSURD-TAC TRUTH-TAC NEG-ATOM-ELIM-TAC (MAKE-ROOM :USE NAT-DED) DUPLICATE-SUPPORT-TAC (THEN** ECONJ-TAC UNSPONSOR-TAC) DEDUCT-TAC (THEN** IDISJ-TAC UNSPONSOR-TAC) (THEN** ICONJ-TAC UNSPONSOR-TAC) (THEN** CASES-TAC UNSPONSOR-TAC) INEG-TAC MP-TAC IMPLICS-EQUIV-TAC EQUIV-IMPLICS-TAC ML::NEG-EQUIV-SLINE-TAC CLASS-DISJ-TAC NEG-NEG-ELIM-TAC NEG-AND-ELIM-TAC NEG-IMP-ELIM-TAC NEG-OR-ELIM-SIMPLE-TAC NEG-OR-ELIM-DUP-TAC INESS-PLINE-TAC INDIRECT-TAC) Similar to a subset of Pfenning*-tac using only basic propositional rules, avoiding rules such as RuleP

BASIC-PROP-TAC Defined for the following uses:

ETREE-NAT: (REPEAT ML::BASIC-PROP*-TAC) Similar to a subset of Pfenning*-tac using only basic propositional rules, avoiding rules such as RuleP

CASES-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies CASES if a support line is a disjunction.

ETREE-NAT: is a primitive tactic.If a support line is a disjunction, applies rule of cases. Pfenning's tactic 202.

CLASS-DISJ-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned line corresponds to a disjunction, and both of the disjuncts are essential, applies indirect proof. Same as Pfenning's tactic 229.

DEDUCT-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies DEDUCT if planned line is an implication.

ETREE-NAT: is a primitive tactic.Applies deduction rule if planned line corresponds to an implication node. Same as Pfenning's tactic 191.

DISJ-EQUIV-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.

DISJ-IMP-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies DISJ-IMP if a support line is of the form " A or B".

ECONJ*-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies conjunction elimination to a support line if applicable. If support line is a multiple conjunction, completely breaks it up.

ETREE-NAT: is a primitive tactic.Applies conjunction elimination to a support line if applicable. If support line is a multiple conjunction, completely breaks it up.

ECONJ-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies ECONJ if a support line is a conjunction.

ETREE-NAT: is a primitive tactic.Applies conjunction elimination to a support line if applicable. Pfenning's tactics 199-200, but regardless of whether the conjuncts are both essential to proving the planned line.

ENEG-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies ENEG if a support line is a negation and planned line is FALSEHOOD.

ETREE-NAT: (ORELSE NEG-ATOM-ELIM-TAC NEG-NEG-ELIM-TAC NEG-AND-ELIM-TAC NEG-IMP-ELIM-TAC NEG-UNIV-ELIM-TAC NEG-EXISTS-ELIM-SIMPLE-TAC NEG-EXISTS-ELIM-DUP-TAC NEG-OR-ELIM-SIMPLE-TAC NEG-OR-ELIM-DUP-TAC NEG-EQUAL-ELIM-TAC)

EQUIV-DISJ-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If a support line is a rewrite node from an equivalence to a disjunction, carries out the rewrite.

EQUIV-IMPLICS-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies EQUIV-IMPLICS if a support line is an equivalence.

ETREE-NAT: is a primitive tactic.If a support line is a rewrite node for an equivalence to a conjunction, applies the equiv-implics rule.

ICONJ*-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.If planned line corresponds to a conjunction node, splits into subgoals. Will break up a multiple conjunction into separate conjuncts.

ETREE-NAT: is a primitive tactic.If planned line corresponds to a conjunction node, splits into subgoals. Will break up a multiple conjunction into separate conjuncts.

ICONJ-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies ICONJ if the planned line is a conjunction.

ETREE-NAT: is a primitive tactic.Applies ICONJ if planned line corresponds to a conjunction node. Same as Pfenning's tactic 186.

IDISJ-LEFT-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned line corresponds to a disjunction, and the right disjunct is inessential, infers the planned line from the left disjunct by RuleP. Same as Pfenning's tactic 188.

IDISJ-RIGHT-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If the planned line corresponds to a disjunction and the left disjunct is inessential, infers the planned line from the right disjunct by RuleP. Same as Pfenning's tactic 189.

IDISJ-TAC Defined for the following uses:

ETREE-NAT: (ORELSE IDISJ-RIGHT-TAC IDISJ-LEFT-TAC)

IMP-DISJ-TAC Defined for the following uses:

- **NAT-DED:** is a primitive tactic.Applies IMP-DISJ if a support line is an implication.

IMPLICS-EQUIV-TAC Defined for the following uses:

- **NAT-DED:** is a primitive tactic.Applies IMPLICS-EQUIV if planned line is an equivalence.
- **ETREE-NAT:** is a primitive tactic.If the planned line corresponds to a rewrite node with justification equiv-implics, applies implics-equiv rule.

INDIRECT-DISJ-PLINE-TAC Defined for the following uses:

- **NAT-DED:** is a primitive tactic.Applies INDIRECT rule, then pushes negation through quantifier, if planned line is a disjunction.

INDIRECT-EXISTS-PLINE-TAC Defined for the following uses:

- **NAT-DED:** is a primitive tactic.Applies INDIRECT rule, then pushes negation through quantifier, if planned line is an existentially quantified line.

INDIRECT-TAC Defined for the following uses:

- **NAT-DED:** is a primitive tactic.Applies INDIRECT as long as planned line is not FALSEHOOD.
- **ETREE-NAT:** is a primitive tactic.Applies indirect proof. This can almost always be applied when the planned line is not FALSEHOOD. It does not apply if the planned line corresponds to a mated node and one of the support line corresponds to the negation of that node.

INDIRECT2-TAC Defined for the following uses:

- **NAT-DED:** is a primitive tactic.Applies INDIRECT2 if two support lines are contradictory.
- **ETREE-NAT:** is a primitive tactic.If planned line is FALSEHOOD, two support lines are contradictory, and are mated, applies indirect2 rule. Same as Pfenning's tactic 212.

INEG-TAC Defined for the following uses:

- **NAT-DED:** is a primitive tactic.Applies INEG if the planned line is a negated formula.
- **ETREE-NAT:** is a primitive tactic.Applies INEG if planned line is a negation.

MP-TAC Defined for the following uses:

- **NAT-DED:** is a primitive tactic.Applies MP if a support line is an implication.

ETREE-NAT: is a primitive tactic.If a support line is an implication, planned line follows from the succedent and the antecedent is provable, applies Modus Ponens. Same as Pfenning's tactic 209.

NEG-AND-ELIM-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned line is FALSEHOOD, and a support line is a negated conjunction, applies eneg rule. Same as Pfenning's tactic 215.

NEG-AND-PLAN-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned line is a negated conjunction, applies indirect proof, assuming negated planned line with new goal of falsehood.

NEG-AND-SLINE-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If a support line is a negated conjunction, applies indirect proof. Similar to Pfenning's tactic 215.

NEG-ATOM-ELIM-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned line is FALSEHOOD and it has two complementary support lines which are mated, applies eneg rule. Same as Pfenning's tactic 212.

NEG-EQUAL-ELIM-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If a support line is a negated equality and planned line is falsehood, applies eneg. Similar to Pfenning's tactic 217.

NEG-EXISTS-ELIM-DUP-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned line is FALSEHOOD, and a support line is a negated existentially quantified formula with more than one expansion, one of which is admissible, applies eneg rule, adding the line with its other expansions as a support. Same as Pfenning's tactic 221.

NEG-EXISTS-ELIM-SIMPLE-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned line is FALSEHOOD, and a support line is a negated existentially quantified formula with exactly one admissible expansion, applies eneg rule. Same as Pfenning's tactic 220.

NEG-IMP-ELIM-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned line is FALSEHOOD, and a support line is a negated implication, applies eneg rule. Same as Pfenning's tactic 216.

NEG-IMP-PLAN-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned is a negated implication, applies pullneg rule.

NEG-IMP-SLINE-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If a support line is a negated implication, pushes the negation through creating a conjunction.

NEG-NEG-ELIM-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned line is FALSEHOOD and it has doubly-negated support line, applies eneg rule. Same as Pfenning's tactic 214.

NEG-NEG-PLAN-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned line is a double negation, applies the pullneg rule.

NEG-NEG-SLINE-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If a support line is a double negation, removes the negations.

NEG-OR-ELIM-DUP-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned line is FALSEHOOD, and a support line is a negated disjunction both of whose disjuncts is essential, applies eneg rule, adding the line with its other expansions as a support. Same as Pfenning's tactic 219.

NEG-OR-ELIM-SIMPLE-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned line is FALSEHOOD, and a support line is a negated disjunction, one of whose disjuncts is inessential (but not both), applies eneg rule. Same as Pfenning's tactic 218.

NEG-OR-PLAN-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned line is a negated disjunction, applies pullneg rule.

NEG-OR-SLINE-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If a support line is a negated disjunction, pushes the negation through, creating a conjunction.

NEG-UNIV-ELIM-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned line is FALSEHOOD, and a support line is a negated universally quantified formula, applies eneg rule. Same as Pfenning's tactic 217.

OR-LEMMA-LEFT-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.Pfenning's tactic 265.

OR-LEMMA-RIGHT-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.Pfenning's tactic 265.

OR-LEMMA-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.Applies either or-lemma-right-tac or or-lemma-left-tac if applicable.

PROP-PRIM Defined for the following uses:

NAT-DED: (ORELSE SAME-TAC TRUTH-TAC ABSURD-TAC INDIRECT2-TAC MAKE-ROOM ECONJ-TAC ICONJ-TAC EQUIV-IMPLICS-TAC IMPLICS-EQUIV-TAC PUSHNEG-TAC PULLNEG-TAC DEDUCT-TAC MP-TAC CASES-TAC INDIRECT-TAC) Much like tactic defined in Felty's master's thesis, p. 64.

PROPOSITIONAL Defined for the following uses:

NAT-DED: (REPEAT (ORELSE MAKE-ROOM (TRY (REPEAT PROP-PRIM)) (THEN INDIRECT-TAC PROPOSITIONAL))) First tries PROP-PRIM repeatedly. If any goals remain, what work was done is thrown away, indirect proof is applied, and PROPOSITIONAL is called recursively on the new goal.

PULLNEG-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies PULLNEG if the planned line is a negated non-literal formula.

ETREE-NAT: (IFTHEN (NEG-PLINE-P-TAC) (ORELSE NEG-NEG-PLAN-TAC NEG-OR-PLAN-TAC NEG-IMP-PLAN-TAC NEG-SEL-PLAN-TAC NEG-EXP-PLAN-TAC NEG-REW-PLAN-TAC))

PUSHNEG-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies PUSHNEG if a support line is a negated non-literal formula.

ETREE-NAT: (IFTHEN (NEG-SLINE-P-TAC) (ORELSE NEG-NEG-SLINE-TAC NEG-OR-SLINE-TAC NEG-IMP-SLINE-TAC NEG-SEL-SLINE-TAC NEG-EXP-SLINE-TAC NEG-REW-SLINE-TAC))

RULEP-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Attempts to apply RULEP; fails if planned line doesn't follow from supports by RuleP.

ETREE-NAT: is a primitive tactic.Applies RuleP if possible.

SAME-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies SAME if planned line is the same as a support line.

ETREE-NAT: is a primitive tactic.If planned line is the same as a support line, and they are mated, applies SAME. Pfenning's tactic 173.

SUBST=-BACKCHAIN-LEMMA-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If substitution of equality can be applied to a support line, creates a new disjunctive lemma based on the formula to which the equality can be applied. Then symmetric simplification is used to simplify the lemma.

TRUTH-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies RuleP if the planned line is TRUTH.

ETREE-NAT: is a primitive tactic.Applies ITruth if the planned line is TRUTH.

TRUTHP-REWRITE-PLAN-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If the planned line corresponds to a rewrite node with justification truthp, justifies the line by ad hoc Truthp, and makes a new planned line with the rewritten wff.

5.3 Quantifiers

AB-PLAN-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If the planned line corresponds to a rewrite node with justification ab, applies the ab* rule.

AB-SLINE-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If a support line is a rewrite node justified by ab, applies the ab* rule.

ABU-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.If planned line is universally quantified, will apply ABU, prompting for a variable if in interactive mode.

EDEF-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies EDEF if a support line contains a definition.

EGEN-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.If the planned line is existentially quantified, will apply EGEN, prompting for the term if in interactive mode.

ETREE-NAT: is a primitive tactic.If the planned line corresponds to a expansion node with a single admissible expansion term, applies EGEN using that term. Same as Pfenning's tactic 195.

EXISTS-LEMMA-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.Pfenning's tactic 264.

IDEF-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies IDEF if planned line contains a definition.

NEG-EXP-PLAN-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned line is a negated expansion node with only one expansion term, applies pullneg rule.

NEG-EXP-SLINE-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If a support line is a negated expansion node, pushes negation through the quantifier.

NEG-SEL-PLAN-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned is a negated selection node, applies pullneg.

NEG-SEL-SLINE-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If a support line is a negated selection node, pushes the negation through the quantifier.

QUANTIFICATIONAL Defined for the following uses:

NAT-DED: (ORELSE UGEN-TAC (THEN ABU-TAC UGEN-TAC) EGEN-TAC UI-TAC)

RULEC-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.If a support line is existentially quantified, will apply RULEC with a brand new variable.

ETREE-NAT: is a primitive tactic.If a support line corresponds to a selection node, applies RuleC. Same as Pfenning's tactic 207.

RULEQ-PLAN-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If the planned line corresponds to a rewrite node with justification ruleq (minimized quantifier scopes), justifies the line by ad hoc RuleQ, and makes a new planned line with the rewritten wff.

RULEQ-SLINE-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If a support line is a rewrite node justified by ruleq, applies the rewrite.

SYMSIMP-TAC Defined for the following uses:

ETREE-NAT: (ORELSE EXISTS-LEMMA-TAC OR-LEMMA-TAC) Pfenning's symmetric simplification tactics.

UGEN-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies UGEN if planned line is universally quantified.

ETREE-NAT: is a primitive tactic.If the planned line is a skolem or selection node, applies UGEN. Same as Pfenning's tactic 194.

UI-HERBRAND-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.UI-HERBRAND-TAC is a tactic for automatically applying universal instantiation. The terms that are used are generated by finding all subterms of the appropriate type (except quantified variables) and applying to them all functions of the appropriate type to get all possible new terms. I.e., you can think of it as constructing the Herbrand universe one level at a time. The number of times that this can be done for any individual quantified formula is controlled by the flag UI-HERBRAND-LIMIT.

UI-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.If a support line is universally quantified, will instantiate it. In interactive mode will ask for a term, otherwise will use the bound variable itself.

ETREE-NAT: is a primitive tactic.If a support node is an expansion node with an admissible expansion, applies universal instantiation. Pfenning's tactics 204/205. If a support line has multiple expansions, it will be duplicated, with the duplication receiving just the excess expansion terms. The instantiated line will not become a support of any other goal than the current one, since it is not known if it is yet admissible for others. The original support line will be dropped from the supports of the current goal, but remain as a support for any other goals. The new support lines will be supports only for the current goal.

UNNEC-EXP-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If the planned line is an expansion node, deletes any unnecessary expansion terms.

5.4 Equality

EQUALITY-PLAN-TAC Defined for the following uses:

ETREE-NAT: (ORELSE EXT=-PLAN-TAC LEIBNIZ=-PLAN-TAC) If the planned line corresponds to rewrite node with justification for a rewritten equality, justifies the line appropriately, and makes a new planned line with the rewritten wff.

EQUALITY-SLINE-TAC Defined for the following uses:

ETREE-NAT: (ORELSE EXT=-SLINE-TAC LEIBNIZ=-SLINE-TAC) If a support line is a rewrite node rewritten because of an equality, carries out the rewrite.

EXT=-PLAN-TAC Defined for the following uses:

- **ETREE-NAT:** is a primitive tactic.If the planned line corresponds to rewrite node with justification for a rewritten equality using extensionality, justifies the line appropriately, and makes a new planned line with the rewritten wff.

EXT=-SLINE-TAC Defined for the following uses:

- **ETREE-NAT:** is a primitive tactic.If a support line corresponds to rewrite node with justification for a rewritten equality using extensionality, justifies the line appropriately, and makes a new support line with the rewritten wff.

LEIBNIZ=-PLAN-TAC Defined for the following uses:

- **ETREE-NAT:** is a primitive tactic.If the planned line corresponds to rewrite node with justification for a rewritten equality using the Leibniz definition, justifies the line appropriately, and makes a new planned line with the rewritten wff.

LEIBNIZ=-SLINE-TAC Defined for the following uses:

- **ETREE-NAT:** is a primitive tactic.If a support line corresponds to rewrite node with justification for a rewritten equality using the Leibniz definition, justifies the line appropriately, and makes a new support line with the rewritten wff.

NEG-EQUAL-SLINE-TAC Defined for the following uses:

- **ETREE-NAT:** is a primitive tactic.If a support line is a negated equality and planned line is falsehood, applies indirect proof. Similar to Pfenning's tactic 217.

REFL=-TAC Defined for the following uses:

- **NAT-DED:** is a primitive tactic.Applies rule for reflexivity of equality if planned line is of form a=a.
- **ETREE-NAT:** is a primitive tactic.If the planned line is a rewrite node with justification REFL=, applies the ASSERT rule for reflexivity of equality. See Pfenning's theorem 141.1.

SUBST=-TAC Defined for the following uses:

- **ETREE-NAT:** is a primitive tactic.Applies either SUBST=L-TAC or SUBST=R-TAC as appropriate.

SUBST=L-TAC Defined for the following uses:

- **NAT-DED:** is a primitive tactic.Applies SUBST=L if planned line follows by this rule from a support line.
- **ETREE-NAT:** is a primitive tactic.If a support line is an equality, and the planned line follows from the substituting the right-hand-side for the left-hand-side in some wff provable from the other supports, applies Subst=L. See Pfenning's theorem 141.

SUBST=R-TAC Defined for the following uses:

> **NAT-DED:** is a primitive tactic.Applies SUBST=R if planned line follows by this rule from a support line.
>
> **ETREE-NAT:** is a primitive tactic.If a support line is an equality, and the planned line follows from the substituting the left-hand-side for the right-hand-side in some wff provable from the other supports, applies Subst=R. See Pfenning's theorem 141.

SYM=-TAC Defined for the following uses:

> **NAT-DED:** is a primitive tactic.Applies symmetry of equality if planned line follows by that rule from some support line.

5.5 Definitions

EQUIV-WFFS-PLAN-TAC Defined for the following uses:

> **ETREE-NAT:** is a primitive tactic.If the planned line corresponds to a rewrite node with justification equivwffs (instantiated definitions), applies equiv-wffs rule.

EQUIV-WFFS-SLINE-TAC Defined for the following uses:

> **ETREE-NAT:** is a primitive tactic.If a support line is a rewrite node justified by equiv-wffs (instantiating definitions), applies the appropriate rule.

NEG-EQUIV-SLINE-TAC Defined for the following uses:

> **ETREE-NAT:** is a primitive tactic.If a support line is a negated equiv-implics rewrite node, and the planned line is FALSEHOOD, do an eneg to make the support line the planned line without the negation, then do the rewrite.

NEG-REW-PLAN-TAC Defined for the following uses:

> **ETREE-NAT:** is a primitive tactic.If planned line is a negated rewrite node, carry out the rewrite, leaving the negation.

NEG-REW-SLINE-TAC Defined for the following uses:

> **ETREE-NAT:** is a primitive tactic.If a support line is a negated rewrite node, carry out the rewrite, leaving the negation above.

5.6 Lambda

BETA-ETA-SEPARATE-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Returns success if LAMBDA-CONV is BETA-ETA-SEPARATE.

ETREE-NAT: is a primitive tactic.Returns success if LAMBDA-CONV is BETA-ETA-SEPARATE.

BETA-ETA-TOGETHER-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Returns success if LAMBDA-CONV is BETA-ETA-TOGETHER.

ETREE-NAT: is a primitive tactic.Returns success if LAMBDA-CONV is BETA-ETA-TOGETHER.

BETA-ONLY-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Returns success if LAMBDA-CONV is BETA-ONLY.

ETREE-NAT: is a primitive tactic.Returns success if LAMBDA-CONV is BETA-ONLY.

EQUIV-EQ-CONTR-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies EQUIV-EQ-CONTR if planned line is appropriate.

EQUIV-EQ-EXPD-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies EQUIV-EQ-EXPD, if that will change the support line.

EXT=-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies EXT= if planned line is appropriate.

EXT=0-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies EXT=0 if planned line is appropriate.

LCONTR*-BETA-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies LCONTR*-BETA, if that will change the support line.

ETREE-NAT: is a primitive tactic.If a support line is a rewrite node justified by beta, applies lcontr*-beta rule.

LCONTR*-ETA-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies LCONTR*-ETA, if that will change the support line.

ETREE-NAT: is a primitive tactic.If a support line is a rewrite node justified by eta, applies lcontr*-eta rule.

LCONTR*-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies LCONTR*, if that will change the support line.

ETREE-NAT: is a primitive tactic.If a support line is a rewrite node justified by lambda, applies lcontr* rule.

LCONTR*-VARY-TAC Defined for the following uses:

NAT-DED: (ORELSE (IFTHEN BETA-ETA-TOGETHER-TAC LCONTR*-TAC) (IFTHEN BETA-ONLY-TAC LCONTR*-BETA-TAC) (IFTHEN BETA-ETA-SEPARATE-TAC (ORELSE LCONTR*-BETA-TAC LCONTR*-ETA-TAC))) Decides which sort of lambda contraction to do, based on the setting of LAMBDA-CONV.

ETREE-NAT: (ORELSE (IFTHEN BETA-ETA-TOGETHER-TAC LCONTR*-TAC) (IFTHEN BETA-ONLY-TAC LCONTR*-BETA-TAC) (IFTHEN BETA-ETA-SEPARATE-TAC (ORELSE LCONTR*-BETA-TAC LCONTR*-ETA-TAC))) Decides which sort of lambda contraction to do, based on the setting of LAMBDA-CONV.

LEXPD*-BETA-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies LEXPD*-BETA, if that will change the planned line.

ETREE-NAT: is a primitive tactic.If the planned line corresponds to a rewrite node with justification beta, applies lexpd*-beta rule.

LEXPD*-ETA-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies LEXPD*-ETA, if that will change the planned line.

ETREE-NAT: is a primitive tactic.If the planned line corresponds to a rewrite node with justification eta, applies lexpd*-eta rule.

LEXPD*-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Applies LEXPD*, if that will change the planned line.

ETREE-NAT: is a primitive tactic.If the planned line corresponds to a rewrite node with justification lambda, applies lexpd* rule.

LEXPD*-VARY-TAC Defined for the following uses:

NAT-DED: (ORELSE (IFTHEN BETA-ETA-TOGETHER-TAC LEXPD*-TAC) (IFTHEN BETA-ONLY-TAC LEXPD*-BETA-TAC) (IFTHEN BETA-ETA-SEPARATE-TAC (ORELSE LEXPD*-BETA-TAC LEXPD*-ETA-TAC))) Decides which sort of lambda expansion to do, based on the setting of LAMBDA-CONV.

ETREE-NAT: (ORELSE (IFTHEN BETA-ETA-TOGETHER-TAC LEXPD*-TAC) (IFTHEN BETA-ONLY-TAC LEXPD*-BETA-TAC) (IFTHEN BETA-ETA-SEPARATE-TAC (ORELSE LEXPD*-BETA-TAC LEXPD*-ETA-TAC))) Decides which sort of lambda expansion to do, based on the setting of LAMBDA-CONV.

5.7 Auxiliary

DUPLICATE-SUPPORT-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If a support line is part of the mating, the duplicate the line, where the original line will remain a support line and where support line tactics can be applied to the copy. This is needed to make proofs with non-leaf matings translate properly. See Pfenning's Tactic 183.

FINISHED-P Defined for the following uses:

NAT-DED: is a primitive tactic.Returns success if current proof has no remaining planned lines.

INESS-PLINE-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.If planned line is not FALSEHOOD and it is inessential, applies absurdity rule. Same as Pfenning's tactic 224.

MAKE-NICE Defined for the following uses:

NAT-DED: (SEQUENCE (CALL CLEANUP) (CALL SQUEEZE) (CALL PALL)) Cleans up a completed proof.

MAKE-ROOM Defined for the following uses:

NAT-DED: is a primitive tactic.Ensures that there is room for at least four new lines before the planned line.

NEG-PLINE-P-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.Returns success if planned line represents a negation node.

NEG-SLINE-P-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.Returns success if some support line represents a negation node.

NNF-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.Closes a gap when a support line is the same as the planned line up to NNF, and the nodes are mated.

RESTRICT-MATING-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.Restricts the mating of the planned line to only those connections involving the line and its supports. Always succeeds.

REWRITE-SLINE-P-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.Returns success if some support line represents a rewrite node.

SHOW-CURRENT-PLAN Defined for the following uses:

NAT-DED: is a primitive tactic.Shows the current planned line.

ETREE-NAT: is a primitive tactic.Shows the current planned line.

SHOW-PLANS Defined for the following uses:

NAT-DED: is a primitive tactic.Shows current plan support structure for all planned lines.

ETREE-NAT: is a primitive tactic.Shows current plan support structure for all planned lines.

UNIVERSAL-GOAL-P Defined for the following uses:

NAT-DED: is a primitive tactic.Returns success if planned line is universally quantified.

UNSPONSOR-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.Removes any support lines which are not required for the planned line.

USE-RULEP-TAC Defined for the following uses:

NAT-DED: is a primitive tactic.Returns success if value of the flag USE-RULEP is T.

ETREE-NAT: is a primitive tactic.Returns success if value of the flag USE-RULEP is T.

USE-SYMSIMP-TAC Defined for the following uses:

ETREE-NAT: is a primitive tactic.Returns success if value of the flag USE-SYMSIMP is T.

Chapter 6

Tacticals

The internal name of this category is TACTICAL.
A tactical can be defined using DEFTACTICAL. Allowable properties are: `DEFN`, `MHELP`.

6.1 Tactics

CALL (CALL command) will execute command as if it were entered at the top level by the user. CALL is used only for side effects, the goal is always returned.

COMPOSE (COMPOSE tac1 tac2 ... tacn) will apply its argument tactics in order, composing their results until one of them fails.

FAILTAC Tactical which always fails, returns its goal unchanged.

IDTAC Tactical which always succeeds, returns its goal unchanged.

IFTHEN (IFTHEN test tactic1 [tactic2]) will first evaluate test, which may be either a tactical or (if user is an expert) an arbitrary LISP expression. If test is a tactical and does not fail, or is a LISP expression which does not evaluate to nil, then tactic1 will be executed and IFTHEN will return its results. If test fails or is nil, then tactic2 (if present) will be executed and its results returned by IFTHEN. Tactic2 is optional; if not specified, and test fails, IFTHEN will return failure.

NO-GOAL (NO-GOAL) succeeds iff the goal with which it is invoked is nil.

ORELSE Given a list of tactics, ORELSE will apply the first one which succeeds.

REPEAT (REPEAT tactic) will apply tactic repeatedly until it fails on every subgoal which has been created.

SEQUENCE (SEQUENCE TAC1 ... TACn) applies tactics TAC1, ..., TACn in order, regardless of their success or failure.

THEN (THEN tactic1 tactic2) will first apply tactic1; if it fails then failure is returned, otherwise tactic2 is applied to each resulting goal. If tactic2 fails on any of these goals, then failure is returned, otherwise the new goals obtained from the calls to tactic2 are returned.

THEN* (THEN* tactic1 tactic2) will first apply tactic1; if it fails then failure is returned, otherwise tactic2 is applied to each resulting goal. If tactic2 fails on any of these goals, then the new goals obtained as a result of applying tactic1 are returned, otherwise the new goals obtained as the result of applying both tactic1 and tactic2 are returned.

THEN** (THEN** tactic1 tactic2) will first apply tactic1 to the current goal. If it does not fail, tactic2 will be applied to the goals which are produced by tactic1, and success will be returned along with any new goals produced. If tactic1 fails, failure will be returned. Differs from THEN and THEN* in that the current goal will never be copied.

TRY (TRY tactic) will use tactic on the current object. If any goals remain after tactic finishes, then the original object will be restored, otherwise the work done by tactic will be kept.

Chapter 7

Mating-Search Commands

The internal name of this category is MATEOP.
A mating-search command can be defined using DEFMATEOP. Allowable properties are: `MATE-ALIAS`, `MATE-RESULT->`, `MATEWFF-ARGNAME`, `MATE-DEFAULTFNS`, `MATE-APPLICABLE-P`, `MATE-MOVE-FN`, `MHELP`.

7.1 Top Levels

LEAVE Exit mating-search. If the current expansion proof is complete, the user will be prompted as to whether to apply MERGE-TREE before exiting.

7.2 Printing

ETD No more help available. Sorry.

ETP No more help available. Sorry.

P No more help available. Sorry.

PDEEP No more help available. Sorry.

PP No more help available. Sorry.

PPDEEP No more help available. Sorry.

PPF No more help available. Sorry.

PPNODE No more help available. Sorry.

PSH No more help available. Sorry.

PTREE No more help available. Sorry.

PTREE* No more help available. Sorry.

PTREE-FILE No more help available. Sorry.

SHOW-OPTION-TREE Show the current option-tree.

7.3 Recording

O Invert PRINTMATEFLAG, that is switch automatic recording of mating-search into a file either on or off. This has not actually been implemented!

REM Write a remark into the PRINTMATEFILE.

7.4 Expansion Trees

DP No more help available. Sorry.

DP* No more help available. Sorry.

DP= No more help available. Sorry.

DPTREE No more help available. Sorry.

DUP-ALL No more help available. Sorry.

DUP-OUTER No more help available. Sorry.

DUP-VAR No more help available. Sorry.

EXP No more help available. Sorry.

MOD-STATUS No more help available. Sorry.

NAME-PRIM If PRIMSUB-METHOD is something other than PR00, NAME-PRIM lists all possible primitive substitutions for the current shallow formula. See the flags PRIM-BDTYPES, MIN-PRIM-DEPTH, MAX-PRIM-DEPTH and PRIM-QUANTIFIER for information on how to change which substitutions are generated. One can use PRIM-SINGLE to instantiate a set variable with one of the generated primsubs.

If PRIMSUB-METHOD is PR00, this creates a list of instantiated etrees. One can choose to do a mating search on one of these using the mate operation SET-SEARCH-TREE.

PRIM-ALL No more help available. Sorry.

PRIM-OUTER No more help available. Sorry.

PRIM-SINGLE No more help available. Sorry.

PRIM-SUB No more help available. Sorry.

RESTORE-ETREE No more help available. Sorry.

SAVE-ETREE No more help available. Sorry.

SEL No more help available. Sorry.

SET-SEARCH-TREE Set the current etree to be a tree generated and named by NAME-PRIM when PRIMSUB-METHOD is PR00.

SUB No more help available. Sorry.

SUB-ETREE No more help available. Sorry.

TERMS No more help available. Sorry.

7.5 Search Suggestions

ETR-INFO Print information about the expansion tree

7.6 Mating search

ADD-EXT-LEMMAS Automatically add extensionality lemmas to the expansion tree.

See Also: USE-EXT-LEMMAS

GO Start mating search using default mating search (controlled by flag DEFAULT-MS).

NOOP Do nothing. (TPS uses this internally.)

UNIFY No more help available. Sorry.

7.7 MS88 search procedure

ADD-CONN No more help available. Sorry.

ADD-CONN* No more help available. Sorry.

APPLY-SUBSTS No more help available. Sorry.

COMPLETE-P No more help available. Sorry.

INIT-MATING No more help available. Sorry.

MINIMAL-P No more help available. Sorry.

MS88 Call mating search procedure on the current eproof. This procedure uses a naive level-saturation method, exhaustively searching a single jform before applying any duplications. Quantifier duplications are applied uniformly to outermost quantifiers. Will try primitive substitution for outermost variable only. Works on only a single jform at a time.

MS88-SUB No more help available. Sorry.

REM-CONN No more help available. Sorry.

REM-CONN* No more help available. Sorry.

REM-LAST-CONN No more help available. Sorry.

SHOW-MATING No more help available. Sorry.

SHOW-SUBSTS No more help available. Sorry.

7.8 MS89 search procedure

MS89 Begin mating search MS89 on the current expansion proof. Primitive substitutions and duplications are performed systematically, with multiple jforms being worked on simultaneously. On each particular jform, the search procedure MS88 is used. The flags MAX-SEARCH-LIMIT, SEARCH-TIME-LIMIT, and RANK-EPROOF-FN are used to control the search. See also the command SHOW-OPTION-TREE.

7.9 MS90-3 search procedure

EXPAND-ETREE Convert the jform proof found by path-focused duplication procedures MS90-3 and MS90-9 into an expansion proof.

MS90-3 Start mating search procedure MS90-3 on current eproof. This search procedure incorporates Issar's path-focused duplication, but works on just one jform at a time. Only duplications are done, not primitive substitutions. This is not an interactive procedure.

PROP-MSEARCH Start Sunil's propositional mating search procedure. This search procedure only works on propositional jforms.

7.10 MS90-9 search procedure

MS90-9 Begin mating search MS90-9 on the current expansion proof. Primitive substitutions and duplications are performed systematically, with multiple jforms being worked on simultaneously. On each particular jform, the search procedure MS90-3 is used. The flags MAX-SEARCH-LIMIT, SEARCH-TIME-LIMIT, and RANK-EPROOF-FN are used to control the search. See also the command SHOW-OPTION-TREE.

7.11 MS91-6 and MS91-7 search procedures

MS91-6 Begin mating search MS91-6 on the current expansion proof. Primitive substitutions and duplications are performed systematically, with multiple jforms being worked on simultaneously. On each particular jform, the search procedure MS88 is used. The flags MAX-SEARCH-LIMIT and SEARCH-TIME-LIMIT are used to control the amount of time spent on each jform.

The order in which the possible jforms are considered depends on a number of flags. Firstly, the primitive substitutions which are generated are determined by the values of MAX-PRIM-DEPTH, MIN-PRIM-DEPTH, PRIM-QUANTIFIER and NEG-PRIM-SUB. If DUP-ALLOWED is T, then additional options are generated corresponding to duplicated quantifiers. These options are then combined into sets; because there can be many such sets, the flag NEW-OPTION-SET-LIMIT controls how many are generated at once. Each set is given a weighting (see flags WEIGHT-x-COEFFICIENT and WEIGHT-x-FN, for x = A,B,C), and the lowest-weighted set is chosen for searching. If the weight of the lowest-weighted set is too large, TPS may generate more sets; the interpretation of "too large" is given by MS91-WEIGHT-LIMIT-RANGE. If the search fails, it will be discarded; if it runs out of time then it will be re-weighted to be continued later (see RECONSIDER-FN).

MS91-7 Begin mating search MS91-7 on the current expansion proof. Primitive substitutions and duplications are performed systematically, with multiple jforms being worked on simultaneously. On each particular jform, the search procedure MS90-3 is used. The flags MAX-SEARCH-LIMIT and SEARCH-TIME-LIMIT are used to control the amount of time spent on each jform.

The order in which the possible jforms are considered depends on a number of flags. Firstly, the primitive substitutions which are generated are determined by the values of MAX-PRIM-DEPTH, MIN-PRIM-DEPTH, PRIM-QUANTIFIER and NEG-PRIM-SUB. If DUP-ALLOWED is T, then additional options are generated corresponding to duplicated quantifiers. These options are then combined into sets; because there can be many such sets, the flag NEW-OPTION-SET-LIMIT controls how many are generated at once. Each set is given a weighting (see flags WEIGHT-x-COEFFICIENT and WEIGHT-x-FN, for x = A,B,C), and the lowest-weighted set is chosen for searching. If the weight of the lowest-weighted set is too large, TPS may generate more sets; the interpretation of "too large" is given by MS91-WEIGHT-LIMIT-RANGE. If the search fails, it will be discarded; if it runs out of time then it will be re-weighted to be continued later (see RECONSIDER-FN).

7.12 MS92-9 search procedure

MS92-9 Call mating search procedure MS92-9 on the current eproof. This procedure uses a naive level-saturation method, exhaustively searching a single jform before applying any duplications. Quantifier duplications are applied uniformly to outermost quantifiers. Will try primitive substitution for outermost variable only. Works on only a single jform at a time. The procedure is almost identical to MS88, except that the flag NUM-OF-DUPS is used to govern how many times the outermost quantifier may be duplicated. The internal representation of variables is as in MS90-3.

7.13 MS93-1 search procedure

MS93-1 Begin mating search MS93-1 on the current expansion proof. The search is basically identical to MS89, but is performed using the internal variable representations of MS90-9. Primitive substitutions and duplications are performed systematically, with multiple jforms being worked on simultaneously. On each particular jform, the search procedure MS92-9 is used. The flags MAX-SEARCH-LIMIT, SEARCH-TIME-LIMIT, and RANK-EPROOF-FN are used to control the search. See also the command SHOW-OPTION-TREE.

7.14 MS98-1 search procedure

MS98-1 Begin the MS98-1 mating search. See Matt Bishop's thesis for details.

MS98-DUP Make NUM-OF-DUPS duplications in the current etree.

MS98-PRIM Make all possible primitive substitutions and then NUM-OF-DUPS duplications in the current etree.

7.15 Proof Translation

MERGE-TREE If the mating is complete, applies substitutions to the expansion tree, then applies Pfenning's MERGE algorithm, eliminating redundant expansion terms.

7.16 Vpforms

CJFORM No more help available. Sorry.

CW

CWD

CWS

NUM-HPATHS No more help available. Sorry.

NUM-VPATHS No more help available. Sorry.

VP No more help available. Sorry.

VPD No more help available. Sorry.

VPETREE No more help available. Sorry.

VPT No more help available. Sorry.

7.17 Moving Commands

0 Move back to previous node, e.g., undo the last L or R command. Note that 0 stands for the numeral zero.

D Move down one node in etree (to leftmost node if more than one successor).

FB Find the topmost binder.

FI Find an infix node.

GOTO Move to a specified node.

L For an infix etree node, move to the left argument.

R For an infix etree node, move to the right argument.

UP Move up one node in etree.

^ Move upwards to root of expansion tree.

7.18 Statistics

DEL-DUP-CONNS No more help available. Sorry.

STATS No more help available. Sorry.

7.19 Miscellaneous

EXPUNGE Frees up space by getting rid of all expansion proofs and option trees. If you only want to get rid of old expansion proofs and option trees, you can use EXPUNGE-OLD to do you job. Warning : After using EXPUNGE, many commands such as ETD, VP, ..., don't work until you re-initialize the current expansion proof by using commands such as SUB, MATE, ...

EXPUNGE-OLD Frees up space by getting rid of all old expansion proofs and option trees. If you'd like to get rid of all(not only old) expansion proofs and option trees, you must use EXPUNGE to do your job. Warning : Never use EXPUNGE-OLD if you are going to use EXPUNGE, or you cannot get the expected result!

Chapter 8

Extensional Expansion Dag Commands

The internal name of this category is EXTMATECMD.
An extensional expansion dag command can be defined using DEFEXTMATE.
Allowable properties are: `EXTMATE-ARGTYPES`, `EXTMATE-ARGNAMES`, `EXTMATE-ARGHELP`, `EXTMATE-DEFAULTFNS`, `EXTMATE-MAINFNS`, `MHELP`.

8.1 Top Levels

LEAVE Leave EXT-MATE to the next enclosing top level.

8.2 Printing

ETD Show the current the extensional expansion dag, only printing some shallow formulas

ETP Show the current the extensional expansion dag, printing all shallow formulas

P Print the current extensional expansion dag node.

PDEEP Print the deep formula of an extensional expansion dag node.

PP Print an extensional expansion dag with node-names.

PPDEEP Pretty-print the deep formula of an extensional expansion dag node.

PPF Prints information about the current extensional expansion dag.

PSH Print the shallow formula of an extensional expansion dag.

SHOW-EXP-TERMS Show expansion terms in expansion dag.

SHOW-EXP-VARS Show expansion vars in expansion dag.

SHOW-MATING Show the current mating in the extensional expansion dag

SHOW-SEL-VARS Show selection vars in expansion dag.

8.3 Extensional Search

COMPLETE-P Indicate if the current extensional expansion dag is complete, and print an open path if it is not complete.

MS03-LIFT Use lifting to guide the search for a proof using diy with default-ms MS03-7. If successful, values are suggested for many relevant flags in the subject MS03-7.

Setting QUERY-USER to T allows the user more control over lifting.

See Also: LIST MS03-7

MS04-LIFT Use lifting to guide the search for a proof using diy with default-ms MS04-2. If successful, values are suggested for many relevant flags in the subject MS04-2.

Setting QUERY-USER to T allows the user more control over lifting.

See Also: LIST MS04-2

8.4 Proof Translation

ETREE-NAT Translate a complete edag proof into natural deduction.

MERGE-TREE Merge a complete edag.

8.5 Vpforms

CJFORM Create (or update) for the edag. You can choose to leave out positive and/or negative flexible literals. You can also choose to leave out flex/flex equation goals.

NUM-HPATHS Print the number of horizontal paths in the jform for the edag.

NUM-VPATHS Print the number of vertical paths in the jform for the edag.

VP Print the jform for the edag as a VP diagram.

VPD Save the jform for the edag as a VP diagram in a file The variables VPD-FILENAME, VPD-STYLE, VPD-PTYPES, VPD-BRIEF and VPD-VPFPAGE control this.

8.6 Extensional Expansion Dags

ADD-CONN Add a connection between two atoms or equations in the edag.

ADD-CONN* Repeatedly call add-conn

DUP-AND-IMITATE Duplicate an expansion var and substitute a general imitation term for the original var.

DUP-AND-PROJECT Duplicate an expansion var and substitute a general projection term for the original var.

DUP-AND-SUBST-EXISTS Duplicate an expansion var and substitute a primsub with an existential quantifier for the original var.

DUP-AND-SUBST-FORALL Duplicate an expansion var and substitute a primsub with a universal quantifier for the original var.

DUP-NODE Create a new expansion arc from an expansion node in an expansion dag.

DUP-VAR Duplicate an expansion var in an expansion dag.

EXPAND-EXISTS Given an expansion variable x(A) and a variable y of type B, let p be the primsub using forall of type B. For every expansion arc with expansion term t containing the given expansion variable x, add a new expansion arc using expansion term [p/x]t.

EXPAND-FORALL Given an expansion variable x(A) and a variable y of type B, let p be the primsub using forall of type B. For every expansion arc with expansion term t containing the given expansion variable x, add a new expansion arc using expansion term [p/x]t.

EXPAND-IMITATE Given an expansion variable x(A) and a head H (appropriate for an imitation term of type A), let H' be the general imitation term for H of type x. For every expansion arc with expansion term t containing the given expansion variable x, add a new expansion arc using expansion term [H'/x]t.

EXPAND-PROJECT Given an expansion variable x(A) and integer i (appropriate for a projection term of type A), let p be the i'th projection term for type A. For every expansion arc with expansion term t containing the given expansion variable x, add a new expansion arc using expansion term [p/x]t.

EXPAND-SUBST Given an expansion variable x(A) and a wff W(A), for every expansion arc with expansion term t containing the given expansion variable x, add a new expansion arc using expansion term [W/x]t. (The free variables of W are not considered new expansion variables.)

IMITATE Substitute a general imitation term for the original var.

PROJECT Substitute a general projection term for the original var.

REM-CONN Remove a connection between two atoms or equations in the edag.

REM-CONN* Repeatedly call rem-conn

SUBST Substitute a term for an expansion var in an expansion dag.

SUBST-EXISTS Substitute a primsub with an existential quantifier for the original var.

SUBST-FORALL Substitute a primsub with a universal quantifier for the original var.

8.7 Moving Commands

0 Move back to previous node, e.g., undo the last L or R command. Note that 0 stands for the numeral zero.

D Move down one node in extensional expansion dag (to leftmost node if more than one successor).

FB Move down to the first expansion or selection node (those whose shallow formulas start with a binder).

FI Move down to the first infix node.

GOTO Go to a node in the extensional expansion dag.

L For an infix edag node, move to the left argument.

R For an infix edag node, move to the right argument.

UP Move up one node in the edag.

^ Move up to the root of the edag.

Chapter 9

Matingstree Commands

The internal name of this category is MTREEOP.
A matingstree command can be defined using DEFMTREEOP. Allowable properties are: `MTREE-ALIAS`, `MTREE-MOVE`, `MTREE-PRINT`, `MTREE-DEFAULT`, `MTREE-ARGS`, `MHELP`.

9.1 Top Levels

LEAVE leaving the mtree top level.

9.2 Mtree Operations

ADD-CONN Add a connection. The subsumption is considered. The usage of the command is exactly as the usage of ADD-CONN in MATE.

CHOOSE-BRANCH Remove all matingstree branches except the one leading to the current matingstree node (which must be a leaf of the matingstree, and must be complete). This will also do some preliminary merging, by deleting all of the jforms which are associated with the deleted nodes.

COMPLETE-P Check the completeness of the current mating. The usage of the command is exactly the same as the usage of the mate command COMPLETE-P.

D Go down one level. D <nth> means go down along the nth subnode. Counting begins from 0. Without argument, D means go down along the leftmost subnode.

GOTO GOTO <node> means to go to the given node. If <node> is not given, it means to go to the root of the matingstree

INIT Initialize the matingstree. This is done automatically when you enter the matingstree top level, but can be used subsequently to return everything to the state it was in when you first entered the mtree top level.

KILL KILL <node> means to mark the given node and all nodes below it as dead.

PICK Pick a leaf which you may try to mate with another later. (MB: I think that PICK N behaves as though you had just added a connection to N, and generates the appropriate obligations, without actually demanding another leaf to connect with. I think.)

PRUNE Remove all dead leaves below (but not including) the current matingstree.

REM-NODE Remove the last connection. The subsumption is considered. If the node is the root, the whole matingstree is removed. The usage of the command is exactly the as the usage of REM-LAST-CONN. Please check the help message for REM-LAST-CONN if necessary.

RESURRECT RESURRECT <node> means to mark the given node and all nodes below it as alive.

SHOW-MATING Show the connections in the mating associated with the current node.

SHOW-SUBSTS Show the substitution stack associated with a matingstree node.

SIB Go to the next sibling of this node.

UNIFY Go into UNIFY toplevel and check the UTREE structure associated with the current node in the matingstree. The unification tree associated with the mating is passed on to the unification top-level. Any changes made to this tree are destructive. Applicable only for a higher-order unification problem. Mainly use to check the UTREE structure.

UP Go up one level.

9.3 Mtree Printing

CONNS-ADDED Print out all of the connections which have already been added to the given matingstree node. If no node is given, the current node is used.

LIVE-LEAVES Print out all of the live leaves in the tree below the given matingstree node. If no node is given, the root node is used.

PM-NODE Print out the given matingstree node in detail. If no node is given, the current matingstree is used.

PMTR Print out the given matingstree as a tree, showing the obligations at each node. If no matingstree is given, the current-matingstree is printed out.

Matingstrees enclosed in curly brackets are marked as dead. Branches with *'s denote nodes that are being omitted for lack of space. The cure for this is to either start printing from a node lower in the tree, or make the screen wider. See also PMTR*.

PMTR* Print out the given matingstree as a tree, showing the obligations at each node. If no matingstree is given, the current-matingstree is printed out.

Numbers in round brackets are open obligations. If the brackets end in "..", there are too many open obligations to fit under the mstree label.

Leaves underlined with ^'s are closed matingstrees. Matingstrees enclosed in curly brackets are marked as dead. Branches with *'s denote nodes that are being omitted for lack of space. The cure for this is to either start printing from a node lower in the tree, or make the screen wider. See also PMTR.

PMTR-FLAT Print out the given matingstree in a flat format. If no matingstree is given, the current matingstree is printed out.

POB Print out the vpform associated with the given obligation node. If no obligation is specified, then the first open obligation in the current obligation tree is used. See the flag DEFAULT-OB-DEEP.

POB-LITS Print out the unblocked literals in a given obligation tree. If no argument is given, the current-obligation tree is the default.

POB-NODE Print out the given obligation in detail. If no obligation is given, then the first open obligation in the current obligation tree is used. See the flag DEFAULT-OB-DEEP.

POTR Print out the given obligation tree as a tree. If no obligation is given, the tree below the current obligation is printed out.

Numbers in round brackets are open obligations; those in square brackets are closed. Branches with *'s denote nodes that are being omitted for lack of space. The cure for this is to either start printing from a node lower in the tree, or make the screen wider.

POTR*-FLAT Print out the given obligation tree in flat form, with the jforms attached to all nodes. If no argument is given, the whole obligation tree is printed out.

POTR-FLAT Print out the given obligation tree in flat form, with the jforms attached to the leaves. If no argument is given, the current-obligation tree is printed out.

PPATH Print out the path containing the given obligation. If no obligation is specified, then the first open obligation in the current obligation tree is used. See the flag DEFAULT-OB-DEEP.

PPATH* Print out the path containing the given obligation, and show all of the obligations on this path. If no obligation is specified, then the first open obligation in the current obligation tree is used. See the flag DEFAULT-OB-DEEP.

9.4 Mtree Auto

ADD-ALL-LIT Attempt to mate a literal with all potential mates on the current path.

ADD-ALL-OB Attempt to mate all literals in an obligation with all potential mates on the current path.

EXPAND-LEAVES Apply ADD-ALL-OB to all live leaves of the current matingstree that lie below the given node (or the current node, if no node is given). WARNING: Potential combinatorial explosion!

GO Call the matingstree procedure given in DEFAULT-MS.

MT94-11 Apply EXPAND-LEAVES repeatedly to all live leaves of the current matingstree that lie below the given node (or the current node, if no node is given), until a closed leaf is generated. WARNING: Potential combinatorial explosion!

MT94-12 Least Branching Search: In each leaf node, take the current obligation and find a literal that can be mated, but with as few mates as possible. Add all of these mates as sons to this node. Repeat until a closed leaf is generated. This search is probably not complete.

MT95-1 Fewest Obligations Search: Choose the matingstree node (from the entire tree, not just the tree below the current node) with the fewest open obligations. Go to that node and do one step of MT94-12 (i.e. choose the literal with the fewest number of mates, and generate all of the associated branches of the mtree). Repeat until a closed leaf is generated. This search is probably not complete.

QRY Output a list of literals which can be mated with a given literal.

Chapter 10

Unification Commands

The internal name of this category is UNIFOP.
An unification command can be defined using DEFUNIFOP. Allowable properties are: `UNIF-ARGTYPES`, `UNIF-ARGNAMES`, `UNIF-ARGHELP`, `UNIF-DEFAULTFNS`, `UNIF-APPLICABLEP`, `UNIF-MAINFNS`, `PRINT-COMMAND`, `MOVE-COMMAND`, `MHELP`.

10.1 Top Levels

LEAVE Exit unification.

10.2 Unification

0 Replace the current topnode with the node on top of the nodestack. Generally, typing an integer n will go to the nth son of the current node. Compare the command NTH-SON.

APPLY-SUBST Apply a substitution, suggested by the user, to the current topnode. Modifies the unification tree.

EPROOF-UTREE Create a new utree whose root has all the dpairs associated with the current mating. (The existing utree may have some of the dpairs added lower down the tree; this will bring them all to the top). See also NAME-DPAIR.

GO Call unification in automatic mode. Will search for unifiers only below the current-topnode.

GOTO Go to the specified node in the unification tree.

MATCH This command is applicable only if current-topnode is a non-terminal leaf node. Calls TPS's version of Huet's MATCH algorithm to find substitutions at the current topnode. The pair selected by MATCH is determined by the value of the flag APPLY-MATCH.

MATCH-PAIR This command is applicable only if current-topnode is a non-terminal leaf node. Calls TPS's version of Huet's MATCH algorithm to find substitutions at the current topnode. n refers to the nth dpair, and this must be a flexible-rigid dpair.

NAME-DPAIR Give a name to the dpairset associated with the current topnode. This is most useful when UNIFY has been issued from the MATE top level, and you want to name the current dpair so that you can save it in the library. See also EPROOF-UTREE.

NTH-SON Go to the nth descendant of the current-topnode. Instead of using this command, you can simply type n on the unification top level to go to the nth descendant. It has no effect if the current-topnode has no descendents.

P Displays the current unification node; show its name, measure, number of descendants, substitutions added and free variables. Does not display the disagreement pairs (use PP or PP* for that), or the cumulative substitutions from this node to the root (use SUBST-STACK for that).

PALL Displays all the disagreement pairs at every node below the given node. (Similar to PP, but for the entire tree below the current node.)

PP Displays the disagreement pairs at the current node. See also PP*. More information about the current node is given by the command P.

PP* Displays the disagreement pairs at the current-topnode, including the order of each pair and other information. See also PP. The other information displayed includes (for each wff, each disagreement pair and the whole set of disagreement pairs): 1) the order (e.g. "x(i)" is first order, and so on). 2) whether it is monadic (all function constants are unary). 3) whether it is linear (all free vars occur once only). 4) whether it is a matching problem (one side of a dpair has no free vars). 5) whether it is a relaxed pattern (all free vars have only bound vars as arguments). 6) whether it is a pattern (all free vars have distinct bound vars as arguments). 7) whether a disagreement pair is variable-disjoint (the free vars on the left are disjoint from those on the right). 8) whether the set of disagreement pairs can be partitioned into sets in which each free var in the whole problem occurs in at most one set. 9) whether there are any free vars that occur only once, or not at all, in the whole problem. These conditions all appear in the literature on higher-order unification; see, for example, Prehofer's paper in CADE '94.

More information about the current node is given by the command P.

SIMPLIFY A call to TPS's version of Huet's SIMPL algorithm. Dpairs in the current topnode are replaced by the dpairs returned by the call. It will also find substitutions of the form (var . term) provided 'var' does not occur in 'term'. This command will alter the unification tree.

STATS Statistics about the current unification tree.

SUBST-STACK Displays the substitution stack for the current topnode. See also P, PP, PP* for other information about the current node.

UTREE Displays the unification tree and the associated substitutions at each node which is below the specified node. Display is in a flat format; UTREE* prints the same information in a tree format.

UTREE* Displays the unification tree and the associated substitutions at each node which is below the specified node. Display is in a tree format; UTREE prints the same information in a flat format. Display shows nodes as numbers, followed by I for imitation, P for projection, for negation, A for administrative (e.g. anything generated by SIMPL). Optionally shows the most recent substitution on the subst-stack at each node.

^ Go to the parent node of the current-topnode. (i.e. move up one level in the tree).

^^ Go to the root node in the unification tree (the node with name "0").

10.3 Dpairs

ADD-DPAIR If the disagreement set already exists, insert a disagreement pair at the front. Else create a new disagreement set consisting of this dpair only.

ADD-DPAIRS-TO-NODE Add new dpairs to the disagreement set at the CURRENT-TOPNODE. Applicable only if CURRENT-TOPNODE is a non failure leaf node. ‘Name’, the first argument to this command must already represent a disagreement set. Use the command ADD-DPAIR,etc., to create this set.

ADD-DPAIRS-TO-UTREE Add new dpairs at all non failure leaf nodes.

FIND-NESTING Find the values for MAX-SUBSTS-* implied by the current node.

PRUNE Prune all the branches which have either reached the maximum allowed depth, or which end only in failure nodes.

RM-DPAIR Remove a disagreement pair from a disagreement set.

SHOW-DPAIRSET Show a disagreement set.

UNIF-PROBLEM Set up a new unification problem. ‘Name’, the first argument to this command must already represent a disagreement set. Use the command ADD-DPAIR to create this set. This is in some ways the inverse of the NAME-DPAIR command.

Chapter 11

Test-Top Commands

The internal name of this category is TESTCMD.
A test-top command can be defined using DEFTEST. Allowable properties are: `TEST-ARGTYPES`, `TEST-ARGNAMES`, `TEST-ARGHELP`, `TEST-DEFAULTFNS`, `TEST-MAINFNS`, `MHELP`.

11.1 Top Levels

LEAVE Leave TEST-TOP to the next enclosing top level.

11.2 Mating search

BREADTH-FIRST-SEARCH Equivalent to setting TEST-NEXT-SEARCH-FN to BREADTH-FIRST-SEARCH and then typing GO. Permanently changes TEST-NEXT-SEARCH-FN.

CLOSE-TESTWIN Closes the window that displays the test-top summary. Use ..../tps/utilities/vpshow (from a shell, not from TPS) to view the output file again.

CONTINUE Continue searching with current searchlist & current problem (similar to GO, but will continue from the last point reached rather than restarting at the beginning again).

EXHAUSTIVE-SEARCH Equivalent to setting TEST-NEXT-SEARCH-FN to EXHAUSTIVE-SEARCH and then typing GO. Permanently changes TEST-NEXT-SEARCH-FN.

FIND-BEST-MODE This command effectively runs PUSH-UP until it finds a mode that works, and then runs PRESS-DOWN until it finds the best mode it can. Before using this command, use the MODE command to set up a mode in which the current theorem can not be proven. Also check the value of the TEST-INCREASE-TIME flag (it should probably not be zero). Then PUSH-UP will systematically vary the values of the flags listed in the TEST-EASIER-IF-* flags, using the PUSH-UP search function (see the help message for TEST-NEXT-SEARCH-FN). Once a correct mode is discovered, it will systematically vary the values of the flags listed in the TEST-FASTER-IF-* flags, using the PRESS-DOWN search function, until it finds as good a mode as it can. The values of TEST-REDUCE-TIME, TEST-NEXT-SEARCH-FN, TEST-INCREASE-TIME and TEST-FIX-UNIF-DEPTHS will be permanently changed.

GO Start searching with current searchlist & current problem.

OPEN-TESTWIN Open a window which will display a summary of the test-top output. The window can be closed with the command CLOSE-TESTWIN. The size of the text is determined by the flag CHARSIZE, and the current width of the window by the flag TESTWIN-WIDTH. The initial height of the window is determined by TESTWIN-HEIGHT.

PRESS-DOWN Before using this command, use the MODE command to set up a mode in which the current theorem can be proven. Also check the value of the TEST-INITIAL-TIME-LIMIT flag (it should be high enough that the first attempt at proof will succeed). Then PRESS-DOWN will systematically vary the values of the flags listed in the TEST-FASTER-IF-* flags, using the PRESS-DOWN search function (see the help message for TEST-NEXT-SEARCH-FN). The values of TEST-REDUCE-TIME, TEST-NEXT-SEARCH-FN and TEST-FIX-UNIF-DEPTHS will be permanently changed (to T, PRESS-DOWN and T respectively).

Note that this is NOT the same as PRESS-DOWN-2, since it automatically generates a searchlist rather than relying on the user to provide one.

PRESS-DOWN-2 Equivalent to setting TEST-NEXT-SEARCH-FN to PRESS-DOWN-2 and then typing GO. Permanently changes TEST-NEXT-SEARCH-FN. Note that this is NOT the same as typing PRESS-DOWN; this will use the user-defined searchlist rather than an automatically generated one.

PUSH-UP This command effectively runs PUSH-UP until it finds a mode that works, and then stops. Before using this command, use the MODE command to set up a mode in which the current theorem can not be proven. Also check the value of the TEST-INCREASE-TIME flag (it should probably not be zero). Then PUSH-UP will systematically vary the values of the flags listed in the TEST-EASIER-IF-* flags, using the PUSH-UP search function (see the help message for TEST-NEXT-SEARCH-FN). The value of TEST-NEXT-SEARCH-FN will be changed to PUSH-UP.

Note that this is NOT the same as PUSH-UP-2, since it automatically generates a searchlist rather than relying on the user to provide one

PUSH-UP-2 Equivalent to setting TEST-NEXT-SEARCH-FN to PUSH-UP-2 and then typing GO. Permanently changes TEST-NEXT-SEARCH-FN. Note that this is NOT the same as typing PUSH-UP; this will use the user-defined searchlist rather than an automatically generated one.

SEARCH-ORDER Show the order in which things will be changed if the search is started now using the given searchlist.

11.3 Searchlists

ADD-FLAG Add a single flag to the current searchlist. To change the current searchlist, use NEW-SEARCHLIST.

ADD-FLAG* Repeatedly add new flags to the current searchlist.

ADD-FUNCTION Add a function to a searchlist. This function will be evaluated on every iteration of the search, and will generally reset certain flags. The special functions defined so far are: UNIFORM-SEARCH-FUNCTION sets max-utree-depth, max-search-limit and max-substs-quick using the values of max-search-depth, search-time-limit and max-substs-var respectively, and then sets TEST-INITIAL-TIME-LIMIT to allow 5 option sets on the first try, then 10, then 15, and so on. BASIC-SEARCH-THEN-UNIFORM-SEARCH runs the current searchlist once over, allowing 1 hour for each setting of the flags. Then it switches the searchlist to UNIFORM-SEARCH-2 and continues with that.

ADD-SUBJECTS Add all the flags concerning the given subjects to the current searchlist.

NEW-SEARCHLIST Make a new searchlist; i.e. begin a new list of flags to be varied. This command also changes the current searchlist.

QUICK-DEFINE Define a searchlist the quick and dirty way! If the current flag settings are OK (i.e. are a successful mode), will create a searchlist in which the flags given in the values of the TEST-FASTER-* flags (do LIST TEST-TOP for a listing) vary over values which ought to give a faster search than the current values. If the current flag settings are not OK, will create a searchlist in which the flags given in the values of the TEST-EASIER-* flags vary over values which ought to make the search easier than the current values. The maximum number of values for any flag to take is governed by TEST-MAX-SEARCH-VALUES.

REM-FLAG Remove a single flag from the current searchlist. To change the current searchlist, use NEW-SEARCHLIST.

REM-FLAG* Repeatedly remove flags from the current searchlist.

REVISE-DEFAULTS For each flag in the given searchlist, change the default setting to the current value of the flag, and put the default setting into the range (unless it's already there). This is useful in conjunction with SCALE-UP and SCALE-DOWN; you can keep one searchlist (let's call it MASTER-SLIST) containing all of the flags you're likely to want to vary. Then if the current flag settings are a good mode and you want to try and find a better one, do REVISE-DEFAULTS followed by SCALE-DOWN MASTER-SLIST; if the current settings are a bad mode and you want to try to find one that works, do REVISE-DEFAULTS followed by SCALE-UP MASTER-SLIST.

SCALE-DOWN Rewrites a searchlist under the assumption that the initial values in the searchlist (together with appropriate settings of the other flags) constitute a successful mode, and that TEST is being run in order to find a faster mode. This will discard all settings that would make the search slower, and will arrange the range of values in such a way that the bounds of the search will gradually decrease until the proof cannot be completed. If this makes the range empty or a singleton, the flag is removed from the searchlist. See the TEST-FASTER-* flags

SCALE-UP Rewrites a searchlist under the assumption that the initial values in the searchlist (together with appropriate settings of the other flags) do not constitute a successful mode, and that TEST is being run in order to find a mode that works. This will discard all settings that would make the search harder, and will arrange the range of values in such a way that the bounds of the search will gradually increase until the proof (with a bit of luck) can be completed. If this makes the range empty or a singleton, the flag is removed from the searchlist. See the TEST-EASIER-* flags.

SEARCHLISTS Print a list of all searchlists currently in memory.

SHOW-SEARCHLIST Show contents of a searchlist.

VARY-MODE Go through an existing mode, flag by flag, creating a searchlist by picking out relevant flags from it. All useless flags (i.e. ones that cannot affect the search time) will be automatically stripped out. The default flag value in the searchlist will be its value in the mode. You can also optionally set the current flag values to the values in the mode (equivalent to the MODE command).

11.4 Library

DELETE Delete a saved searchlist or mode (equivalent to the library command DELETE.

FETCH Retrieve a searchlist or mode from the library. Exactly like the library function FETCH, except that when a searchlist is retrieved, it will become the current searchlist.

INSERT Like the library command INSERT; will save a searchlist in the library. Will also save a mode that has been found by using GO.

Chapter 12

Models Commands

The internal name of this category is MODELSCMD.
A models command can be defined using DEFMODELS. Allowable properties are: `MODELS-ARGTYPES`, `MODELS-ARGNAMES`, `MODELS-ARGHELP`, `MODELS-DEFAULTFNS`, `MODELS-MAINFNS`, `MHELP`.

12.1 Top Levels

LEAVE Leave MODELS to the next enclosing top level.

12.2 Printing

PELT Print the integer in notation appropriate to the given type. For example, elements of type (OA) are printed in set notation. The empty set is called EMPTY and the universal set is called FULL.

Constant functions are denoted by Kc.

A few special cases are T and F at type O, NOT at type (OO), the binary connectives AND, OR, IMPLIES, EQUIV and XOR at type (OOO), PI and SIGMA at types of the form (O(OA)), = at types of the form (OAA) and ID at types of the form (AA).

EMPTY at a type (OA) corresponds to the empty set.

FULL at a type (OA) corresponds to the set of all elements of type A.

PI at a type (O(OA)) corresponds to the singleton FULL where FULL corresponds to the set of all elements of type A.

SIGMA at a type (O(OA)) corresponds to the set containing all sets of type A except EMPTY.

For elements of low types the command PELT-REC may also be helpful.

SEE ALSO: PELT-REC

PELT-REC Print the integer in notation appropriate to the given type. For example, elements of type (OA) are printed in set notation. The empty set is called EMPTY and the universal set is called FULL.

Constant functions are denoted by K(c).

A few special cases are T and F at type O, NOT at type (OO), the binary connectives AND, OR, IMPLIES, EQUIV and XOR at type (OOO), PI and SIGMA at types of the form (O(OA)), = at types of the form (OAA) and ID at types of the form (AA).

EMPTY at a type (OA) corresponds to the empty set.

FULL at a type (OA) corresponds to the set of all elements of type A.

PI at a type (O(OA)) corresponds to the singleton FULL where FULL corresponds to the set of all elements of type A.

SIGMA at a type (O(OA)) corresponds to the set containing all sets of type A except EMPTY.

This command is recursive. For low types this is helpful, but the notation becomes unwieldy for higher types. For higher types the command PELT is more appropriate.

SEE ALSO: PELT

PELTS Print all the elements of the given type as both integers and the notation of PELT.

SEE ALSO: PELT

PELTS-REC Print all the elements of the given type as both integers and the notation of PELT-REC.

SEE ALSO: PELT-REC

PSIZE Print the size of the domain of the given type. The elements of the type are 0, . . ., n-1 where n is the size.

SHOW-ASSIGNMENTS Show all currently assigned values. To see the value of any particular variable, use INTERPRET. To assign a value or remove an assignment, use ASSIGN-VAR or UNASSIGN-VAR.

SEE ALSO: ASSIGN-VAR, UNASSIGN-VAR, REMOVE-ALL-ASSIGNMENTS, INTERPRET

12.3 Models

ASSIGN-VAR Assign a value to a variable in the current model.

SEE ALSO: REMOVE-ALL-ASSIGNMENTS, UNASSIGN-VAR, INTERPRET, SHOW-ASSIGNMENTS

CHANGE-BASE-TYPE Change the number of elements in a base type. This must be a power of 2.

COND-PROBABILITY Computes the conditional probability that a wff2 is true if a wff1 is true in the model. Assigned variables are considered fixed. All unassigned variables are allowed to vary over the appropriate domains. The probability is the number of values for these unassigned variables for which wff1 and wff2 are true over the number of values for which wff1 is true.

SEE ALSO: PROBABILITY, INTERPRET, MAX-BINDER-COMPUTATION, MAX-DOMAIN-SIZE

INTERPRET Interpret a formula in the current model. The evaluation is lazy so if a function is constant, the argument is not evaluated. The flags MAX-BINDER-COMPUTATION and MAX-DOMAIN-SIZE bound how complicated the wff can be before interpret will fail.

SEE ALSO: ASSIGN-VAR, SHOW-ASSIGNMENTS, REMOVE-ALL-ASSIGNMENTS, UNASSIGN-VAR, MAX-BINDER-COMPUTATION, MAX-DOMAIN-SIZE

PROBABILITY Computes the probability that a formula is true in the model. Assigned variables are considered fixed. All unassigned variables are allowed to vary over the appropriate domains. The probability is the number of values for these unassigned variables for which the wff is true over the total number of values for the unassigned variables.

SEE ALSO: COND-PROBABILITY, INTERPRET, MAX-BINDER-COMPUTATION, MAX-DOMAIN-SIZE

REMOVE-ALL-ASSIGNMENTS Remove all assignments for variables in the current model.

SEE ALSO: UNASSIGN-VAR, ASSIGN-VAR, INTERPRET, SHOW-ASSIGNMENTS

SOLVE Solve for values for the output variables for any values of the input variables so that the given proposition is true.

If the domains involved are large, TPS will ask the user whether to print the values to the screen or save them to a file.

TPS will always tell the user whether there are no solutions for any inputs, solutions for some but not all inputs, solutions for all inputs and whether there are unique solutions for some inputs.

UNASSIGN-VAR Remove an assignment for a variable in the current model.

SEE ALSO: REMOVE-ALL-ASSIGNMENTS, ASSIGN-VAR, INTERPRET, SHOW-ASSIGNMENTS

Chapter 13

Editor Commands

The internal name of this category is EDOP.
An editor command can be defined using DEFEDOP. Allowable properties are: `ALIAS`, `RESULT->`, `EDWFF-ARGNAME`, `DEFAULTFNS`, `MOVE-FN`, `MHELP`.

13.1 Top Levels

LEAVE Exit the editor with all the changes in place.

NOOP Do nothing.

OK Exit the editor with all the changes in place.

13.2 Printing

P Print a wff using the global settings of all flags.

PP Pretty-print a wff.

PS Print a wff showing all brackets and dots.

PT Print a wff showing types.

13.3 Weak Labels

CW Assigns a label to the edwff, but does not change the edwff. You can use the label to refer to this wff later.

DELWEAK Replaces all occurrences of the label with the wff it represents in the current wff.

DW Replace a top level occurrence of the label by the wff it represents.

DW* Replace all labels in a wff by the wffs represented by them.

NAME Assign a label to the edwff, and replace the edwff with this label.

RW Makes current edwff the new value of label (which must already exist).

13.4 Saving Wffs

SAVE Save a wff by appending it to the file SAVEDWFFS. The weak label name should not already exist (if it does, remove it using RW). The wffs that are saved to this file can be reloaded using the command QLOAD "savedwffs.lisp". This command dates from before the LIBRARY top level was introduced; you should probably avoid it. If you want to save a gwff, use CW to create a weak label, then go into the library with LIB and use INSERT to save the wff.

13.5 Recording

O Invert PRINTEDTFLAG, that is switch automatic recording of wffs in a file either on or off. When switching on, the current wff will be written to the PRINTEDTFILE. Notice that the resulting file will be in Scribe format; if you want something you can reload into TPS, then use the SAVE command.

REM Write a remark into the PRINTEDTFILE.

13.6 Vpforms

CJFORM Converts the given GWFF to JFORM.

DJFORM Converts the given JFORM to GWFF. May not work with skolemized jforms.

NUM-HPATHS Counts the number of horizontal paths through the given jform.

NUM-VPATHS Counts the number of vertical paths through the given jform.

PJ Prints the given gwff, using lists for jforms.

PROP-CJFORM Converts the given GWFF to JFORM.

VP Prints a vertical path diagram. This is like VP in the MATE top level, but will use the current edwff to create a jform if none is currently available.

VPD Use this operation for saving VP diagrams in a file. You may want to change the values of the variables VPD-FILENAME, VPD-STYLE, VPD-PTYPES, VPD-BRIEF, VPD-VPFPAGE.

VPF Prints the vertical path diagram for a JForm or a GWFF.

VPT Prints the path diagram, in a format understood by TeX, for a JForm or a GWFF. At present, it chops off whatever will not fit on one page. The following flags affect the output: 1. VPD-BRIEF controls whether labels or wffs are printed. 2. VPD-PTYPES controls whether types are printed. 3. TEXFORMAT controls whether the vertical or horizontal path diagram is printed. 4. ALLSCOPEFLAG controls where square brackets are printed.

13.7 Moving Commands

0 Move up one-level, i.e., undo the last L, R, D, or A command. Note that 0 stands for the numeral zero.

A for an expression like @wtP x y, delete the rightmost element; in this example the result will be to make @wtPx the current expression. For a quantified expression, it will move to the quantified variable.

D for an expression like @wtP x y, move to the rightmost element; in this example @wty. For a quantified expression it will move to the scope of the quantifier.

FB Find the first binder (left to right)

FI Find an infix operator.

L for an infix-operator, move to the left argument.

R for an infix-operator, move to the right argument.

UNDO Moves up (like 0), but throws away any editing since your last downward moving command (typically A,D,L,or R.

XTR Makes the current edwff the top wff.

^ Move upwards through enclosing wffs all the way to the top.

13.8 Changing Commands

ASRB Apply the following laws to a wff: A and (A or B), (A or B) and A –> A or B A and (B or A), (B or A) and A –> B or A A or (A and B), (A and B) or A –> A (B and A) or A, (B and A) or A –> A.

ASSL Apply the left associative law: A op (B op C) –> (A op B) op C.

ASSR Apply the right associative law: (A op B) op C –> A op (B op C).

CMRG Delete the truth constants from a wff: A and TRUTH, TRUTH and A –> A A and FALSEHOOD, FALSEHOOD and A –> FALSEHOOD A or TRUTH, TRUTH or A –> TRUTH A or FALSEHOOD, FALSEHOOD or A –> A A implies TRUTH –> TRUTH TRUTH implies A –> A A implies FALSEHOOD –> not A FALSEHOOD implies A –> TRUTH A equiv TRUTH, TRUTH equiv A –> A A equiv FALSEHOOD, FALSEHOOD equiv A –> not A not TRUTH –> FALSEHOOD not FALSEHOOD –> TRUTH.

CMUT Apply the commutative laws to a formula: A and B –> B and A A or B –> B or A A implies B –> not B implies not A A equiv B –> B equiv A.

CNTOP Change the top connective of a formula. For example, "cntop or" will change "A and B" into "A or B"; "cntop exists" will change "forall x P x" into "exists x P x".

DIST-CTR Apply the distributivity laws to a wff in the contracting direction: (A and B) or (A and C) –> A and (B or C) (A or B) and (A or C) –> A or (B and C) (B and A) or (C and A) –> (B or C) and A (B or A) and (C or A) –> (B and C) or A.

DIST-EXP Apply the distributivity laws to a wff in the expanding direction: A and (B or C) –> (A and B) or (A and C) A or (B and C) –> (A or B) and (A or C) (B or C) and A –> (B and A) or (C and A) (B and C) or A –> (B or A) and (C or A).

DL Delete the topmost binary connective and its left scope

DNEG Remove a double negation: not not A –> A.

DR Delete the topmost binary connective and its right scope

MRG Apply the following laws to a wff: A and A –> A A or A –> A A implies A –> TRUTH A and not A, not A and A –> FALSEHOOD A or not A, not A or A –> TRUTH A implies not A –> not A not A implies A –> A A equiv not A, not A equiv A –> FALSEHOOD.

PMUT Permute the two components of an infix operator: A op B –> B op A

SUBEQ Apply the following law to a formula: A equiv B –> (A implies B) and (B implies A).

SUBIM Apply the following law to a formula: A implies B –> not A or B.

13.9 Recursively Changing Commands

ASRB* Recursively apply the following laws to a wff: A and (A or B), (A or B) and A –> A or B A and (B or A), (B or A) and A –> B or A A or (A and B), (A and B) or A –> A (B and A) or A, (B and A) or A –> A.

ASSL* Recursively apply the left associative law: A op (B op C) –> (A op B) op C.

ASSR* Recursively apply the right associative law: (A op B) op C –> A op (B op C).

CMRG* Recursively delete the truth constants in a wff: A and TRUTH, TRUTH and A –> A A and FALSEHOOD, FALSEHOOD and A –> FALSEHOOD A or TRUTH, TRUTH or A –> TRUTH A or FALSEHOOD, FALSEHOOD or A –> A A implies TRUTH –> TRUTH TRUTH implies A –> A A implies FALSEHOOD –> not A FALSEHOOD implies A –> TRUTH A equiv TRUTH, TRUTH equiv A –> A A equiv FALSEHOOD, FALSEHOOD equiv A –> not A not TRUTH –> FALSEHOOD not FALSEHOOD –> TRUTH.

CMUT* Recursively apply the commutative laws to a formula: A and B –> B and A A or B –> B or A A implies B –> not B implies not A A equiv B –> B equiv A.

DIST-CTR* Recursively apply the distributive laws to a wff in the contracting direction: (A and B) or (A and C) –> A and (B or C) (A or B) and (A or C) –> A or (B and C) (B and A) or (C and A) –> (B or C) and A (B or A) and (C or A) –> (B and C) or A.

DIST-EXP* Recursively apply the distributive laws to a wff in the expanding direction: A and (B or C) –> (A and B) or (A and C) A or (B and C) –> (A or B) and (A or C) (B or C) and A –> (B and A) or (C and A) (B and C) or A –> (B or A) and (C or A).

DNEG* Recursively remove double negations: not not A –> A.

MRG* Recursively apply the following laws to a wff: A and A –> A A or A –> A A implies A –> TRUTH A equiv A –> TRUTH A and not A, not A and A –> FALSEHOOD A or not A, not A or A –> TRUTH A implies not A –> not A not A implies A –> A A equiv not A, not A equiv A –> FALSEHOOD.

PMUT* Recursively permute the two components of an infix operator: A op B –> B op A

SUBEQ* Recursively apply the following law to a formula: A equiv B –> (A implies B) and (B implies A).

SUBIM* Recursively apply the following law to a formula: A implies B –> not A or B.

13.10 Embedding Commands

MBED-AL Embed the current edwff in the left scope of AND. The right scope is provided by the user.

MBED-AR Embed the current edwff in the right scope of AND. The left scope is provided by the user.

MBED-E Embed the current edwff in the scope of an existential quantifier. The variable of quantification is provided by the user.

MBED-E1 Embed the current edwff in the scope of an exists1 quantifier. The variable of quantification is provided by the user.

MBED-F Embed the current edwff in the scope of a universal quantifier. The variable of quantification is provided by the user.

MBED-IL Embed the current edwff as the antecedent of a conditional. The consequent is provided by the user.

MBED-IR Embed the current edwff as the consequent of a conditional. The antecedent is provided by the user.

MBED-L Embed the current edwff in the scope of lambda. The variable of quantification is provided by the user.

MBED-OL Embed the current edwff in the left scope of OR. The right scope is provided by the user.

MBED-OR Embed the current edwff in the right scope of OR. The left scope is provided by the user.

MBED-QL Embed the current edwff on the left side of equivalence. The right side is provided by the user.

MBED-QR Embed the current edwff on the right side of equivalence. The left side is provided by the user.

MBED=L Embed the current edwff on the left side of equality. The right side is provided by the user.

MBED=R Embed the current edwff on the right side of equality. The left side is provided by the user.

13.11 Rewriting commands

ARR Apply one active rewrite rule to the current edwff; attempt different active rules in the order in which they are listed by LIST-RRULES until one works. If any current rules are bidirectional, you will be prompted about which direction to apply them in.

ARR* Apply one active rewrite rule to the current edwff; attempt different active rules in the order in which they are listed by LIST-RRULES until one works. If any current rules are bidirectional, you will be prompted about which direction to apply them in. Repeat this until no more rules are applicable. CAUTION: may not terminate.

ARR1 Apply a rewrite rule (active or inactive) to the current edwff. If the rule is bidirectional, you will be prompted about which direction to apply it in.

ARR1* Apply a rewrite rule (active or inactive) repeatedly to the current edwff. If the rule is bidirectional, you will be prompted about which direction to apply it in. CAUTION: may not terminate.

MAKE-RRULE Create a rewrite rule whose left-hand side is the current edwff.

UNARR Unapply one active rewrite rule to the current edwff (i.e. apply it in the reverse direction); attempt different active rules in the order in which they are listed by LIST-RRULES until one works. If any current rules are bidirectional, you will be prompted about which direction to apply them in.

UNARR* Unapply one active rewrite rule to the current edwff (i.e. apply it in the reverse direction); attempt different active rules in the order in which they are listed by LIST-RRULES until one works. Repeat this until no more rules are applicable. If any current rules are bidirectional, you will be prompted about which direction to apply them in. CAUTION: may not terminate.

UNARR1 Unapply a rewrite rule (active or inactive) to the current edwff. (i.e. apply it in the reverse direction). If the rule is bidirectional, you will be prompted about which direction to apply it in.

UNARR1* Unapply a rewrite rule (active or inactive) repeatedly to the current edwff. (i.e. apply it in the reverse direction). If the rule is bidirectional, you will be prompted about which direction to apply it in. CAUTION: may not terminate.

13.12 Substitution

AB Alphabetic change of variable at top-level.

IB Instantiate a top-level universal or existential binder with a term.

PRIM-SUBST Replaces a variable with a primitive substitution. Differs from SUBST in that it will also replace quantified variables, and their quantifiers, as necessary.

REW-EQUIV Replaces all occurrences of the form ‘A EQUIV B’ according to the setting of the flag REWRITE-EQUIVS.

RP Replace one occurrence of a symbol (such as AND) by a predefined equivalent wff (such as [lambda p lambda q. .p IMPLIES q]). In this example repsym is AND and rep-by is IMPLIES. To see if a symbol can be replaced by this command, enter HELP symbol; any such replacements will be listed under the heading 'Replaceable Symbols’.

RPALL Replace a all occurrences of a symbol by a predefined equivalent wff.

SUB Replaces the current wff by the wff supplied.

SUBST Substitute a term for the free occurrences of variable in a gwff. Bound variables may be renamed, using the function in the global variable RENVAR-FN.

SUBSTYP Substitutes a type for a type variable in edwff.

13.13 Basic Abbreviations

ABBR Lists all the abbreviations used in a gwff.

CONSTANTS Lists all the logical constants used in a gwff, apart from the primitive constants AND FALSEHOOD IMPLIES NOT OR TRUTH.

EXPAND= Instantiate outermost equality in gwff. Consults the flag REWRITE-EQUALITIES (but ignores it if it's set to NONE).

EXPAND=* Instantiate all equalities in gwff. Consults the flag REWRITE-EQUALITIES (but ignores it if it's set to NONE).

INST Instantiate all occurrences of an abbreviation. The occurrences will be lambda-contracted, but not lambda-normalized.

INST1 Instantiate the first abbreviation, left-to-right.

INSTALL Instantiate all definitions, except the ones specified in the second argument.

INSTALL-REC Recursively instantiate all definitions, except the ones specified in the second argument.

LIB-ABBR Lists all the library abbreviations used in a gwff.

NEW-DEFS Lists all the definitions used in a gwff that are either library abbreviations or weak labels.

13.14 Lambda-Calculus

ABNORM Convert the gwff to alphabetic normal form.

ETAB Eta-expands until original wff is part of a wff of base type.

ETAC Reduces [lambda x.fx] to f at top.

ETAN Reduces [lambda x.fx] to f from inside out.

ETAX Performs a one-step eta expansion.

LETA Returns the long-eta normal form of wff.

LEXP Converts the wff into the application of a function to the term. The function is formed by replacing given valid occurrences of a term with the variable and binding the result.

LNORM Put a wff into lambda-normal form, using beta or beta-eta conversion according to the value of flag LAMBDA-CONV. Compare LNORM-BETA and LNORM-ETA.

LNORM-BETA Put a wff into beta-normal form, not using eta conversion. Compare LNORM and LNORM-ETA.

LNORM-ETA Put a wff into eta-normal form, not using beta conversion. Compare LNORM-BETA and LNORM.

RED Lambda-contract a top-level reduct. Bound variables may be renamed using REN-VAR-FN

ULNORM Convert a untyped wff into lambda-normal form. Be aware of unterminated reduction in untyped lambda calculus.

13.15 Negation movers

NEG Negates current wff, erasing double negations.

NNF Return the negation normal form of the given wff.

PULL-NEG Pulls negations out one level.

PUSH-NEG Pushes negation through the outermost operator or quantifier.

13.16 Primitive Substitutions

NAME-PRIM Creates weak labels for primitive substitutions for the head variables of a wff.

PRT-PRIM Prints primitive substitutions for the head variables of a wff.

13.17 Miscellaneous

CLAUSE-FORM Converts the given wff to clause form, as if the resulting wff is to be given to a resolution theorem prover. The gwff is skolemized, rectified, etc.

CNF Find the conjunctive normal form of a wff.

HEAD Find the head of a gwff.

HVARS Find all head variables of a wff.

MIN-SCOPE Minimize the scope of quantifiers in a gwff. Deletes vacuous quantifiers. During proof transformation, the gap between a formula and its min-quant-scope version is filled by RULEQ.

SUBFORMULAS Find all subformulas of a given type in a wff.

13.18 RuleP

SAT Check whether a propositional wff is satisfiable.

VALID Check whether a propositional wff is valid.

13.19 Skolemizing

SK1 Skolemize a wff using method S1. See page 127 of Andrews' book. If equivalences are present, you must eliminate them first by REW-EQUIV.

SK3 Skolemize a wff using method S3. At the moment it takes only those free variables which are universally quantified somewhere before, all other variables are considered to be constants. See page 127 of Andrews' book. If equivalences are present, you must eliminate them first by REW-EQUIV.

13.20 Quantifier Commands

DB Delete the leftmost binder in a wff.

EP Delete all accessible essentially existential quantifiers.

OP Delete all accessible essentially universal quantifiers.

13.21 Wellformedness

DUPW duplicates wff across connective.

EDILL Find a minimal ill-formed subformula.

ILL Return a list of messages, each the describing the error in a minimal ill-formed subparts of the argument.

TP Return the type of a gwff.

WFFP Test for a gwff (general well-formed formula).

Chapter 14

Replaceable Symbols

The internal name of this category is REPSYMBOL.
A replaceable symbol can be defined using DEFREPSYMBOL. Allowable properties are: `EQUIV-TO`, `MHELP`.

14.1 Basic Abbreviations

AND `AND may be replaced by any of: INVERSE` $\lambda p_o \lambda q_o . q \wedge p$ `IMPLIES` $\lambda p_o \lambda q_o . \sim . p \supset \sim q$ `OR` $\lambda p_o \lambda q_o . \sim . \sim p \vee \sim q$

IMPLIES `IMPLIES may be replaced by any of: INVERSE` $\lambda p_o \lambda q_o . \sim q \supset \sim p$ `AND` $\lambda p_o \lambda q_o . \sim . p \wedge \sim q$ `OR` $\lambda p_o \lambda q_o . \sim p \vee q$

OR `OR may be replaced by any of: INVERSE` $\lambda p_o \lambda q_o . q \vee p$ `IMPLIES` $\lambda p_o \lambda q_o . \sim p \supset q$ `AND` $\lambda p_o \lambda q_o . \sim . \sim p \wedge \sim q$

SUBSET `SUBSET may be replaced by any of: INTERSECT` $\lambda p_{o\alpha} \lambda q_{o\alpha} . p \cap q = p$ `IMPLIES` $\lambda p_{o\alpha} \lambda q_{o\alpha} \; \forall x_\alpha . p x \supset q x$ `INVERSE` $\lambda p_{o\alpha} \lambda q_{o\alpha} . \; q \subseteq p$

Chapter 15

Theorems

15.1 Book Theorems

DESCR Axiom of description at all types.
$\iota[= Y_\alpha] = Y$

EXT Axiom of extensionality at all types.
$\forall x_\beta [f_{\alpha\beta} x = g_{\alpha\beta} x] \supset f = g$

EXT-LEIB Extensional equality of f and g implies Leibniz equality of f and g.
$\forall f_{\alpha\beta} \; \forall g_{\alpha\beta} . \; \forall x_\beta [f x = g x] \supset \forall q_{o(\alpha\beta)} . q f \supset q g$

REFL= Reflexivity of Equality.
$A_\alpha = A$

SYM= Symmetry of Equality.
$A_\alpha = B_\alpha \supset B = A$

T5302 Symmetry of Equality.
$x_\alpha = y_\alpha = . y = x$

T5310 Theorem about descriptions.
$\forall z_\alpha [p_{o\alpha} z \equiv y_\alpha = z] \supset \iota p = y$

T5310A Theorem about descriptions.
$\forall z_\alpha [p_{o\alpha} z \equiv z = y_\alpha] \supset \iota p = y$

15.2 First-Order Logic

X2106
$\forall x [R x \supset P x] \wedge \forall x [\sim Q x \supset R x] \supset \forall x . P x \vee Q x$

X2107
$R a b \wedge \forall x \; \forall y [R x y \supset R y x \wedge Q x y] \wedge \forall u \; \forall v [Q u v \supset Q u u] \supset Q a a$
$\wedge Q b b$

X2108
$\forall x\ \exists y. Px \supset Py$

X2109
$\exists x[p \wedge Qx] \equiv p \wedge \exists x Qx$

X2110
$\exists x Rx \wedge \forall y[Ry \supset \exists z Qyz] \wedge \forall x\ \forall y[Qxy \supset Qxx] \supset \exists x\ \exists y. Qxy \wedge Ry$

X2111
$\forall x[\ \exists y Pxy \supset \forall y Qxy] \wedge \forall z\ \exists y Pzy \supset \forall y\ \forall x Qxy$

X2112
$\exists v\ \forall x Pxv \wedge \forall x[Sx \supset \exists y Qyx] \wedge \forall x\ \forall y[Pxy \supset \sim Qxy] \supset \exists u. \sim Su$

X2113
$\forall y\ \exists w Ryw \wedge \exists z\ \forall x[Px \supset \sim Rzx] \supset \exists x. \sim Px$

X2114
$\forall x Rxb \wedge \forall y[\ \exists z Ryz \supset Ray] \supset \exists u\ \forall v Ruv$

X2115
$\forall x[\ \exists y Pxy \supset \forall z Pzz] \wedge \forall u\ \exists v[Puv \vee Mu \wedge Q. fuv] \wedge \forall w[Qw \supset \sim M. gw] \supset \forall u\ \exists v. P[gu]v \wedge Puu$

X2116
$\forall x\ \exists y[Px \supset Rx[g. hy] \wedge Py] \wedge \forall w[Pw \supset P[gw] \wedge P. hw] \supset \forall x. Px \supset \exists y. Rxy \wedge Py$

X2117
$\forall u\ \forall v[Ruu \equiv Ruv] \wedge \forall w\ \forall z[Rww \equiv Rzw] \supset .\ \exists x Rxx \supset \forall y Ryy$

X2118
$\forall x[p \wedge Qx \vee \sim p \wedge Rx] \supset \forall x Qx \vee \forall x Rx$

X2119
$\exists y\ \forall x. Py \supset Px$

X2120
$\forall u\ \forall v\ \forall w[Puv \vee Pvw] \supset \exists x\ \forall y Pxy$

X2121
$\exists v\ \forall y\ \exists z. Pay[hy] \vee Pvy[fy] \supset Pvyz$

X2122
$\exists x Rxx \supset \forall y Ryy \supset \exists u\ \forall v. Ruu \supset Rvv$

X2123
$\exists y[Py \supset Qx] \supset \exists y. Py \supset Qy$

X2124
$\exists x[Px \supset Qx] \equiv \forall x Px \supset \exists x Qx$

X2125
$\exists x\ \forall y[Px \equiv Py] \equiv .\ \exists x Px \equiv \forall y Py$

X2126
$\forall x[Px \equiv \exists y Py] \equiv .\ \forall x Px \equiv \exists y Py$

X2127
$\exists x\ \forall y[Py \equiv Px] \supset \forall x Px \vee \forall x.\ \sim Px$

X2128
$\forall x[Px \equiv \forall y Py] \equiv .\ \exists x Px \equiv \forall y Py$

X2129
$\exists x\ \forall y[Px \equiv Py] \equiv [\ \exists x Qx \equiv \forall y Py] \equiv .\ \exists x\ \forall y[Qx \equiv Qy] \equiv .$
$\exists x Px \equiv \forall y Qy$

X2130
$\forall x Px \supset \sim \exists y Qy \vee \exists z. Pz \supset Qz$

X2131
$\forall x Px \supset \exists y.\ \forall x\ \forall z Qxyz \supset \sim \forall z. Pz \wedge \sim Qyyz$

X2132
$\forall w[\ \sim Rww] \supset \exists x\ \exists y.\ \sim Rxy \wedge . Qyx \supset \forall z Qzz$

X2133
$\forall x[\ \exists y Qxy \supset Px] \wedge \forall v\ \exists u Quv \wedge \forall w\ \forall z[Qwz \supset Qzw \vee Qzz]$
$\supset \forall z Pz$

X2134
$\forall z\ \exists x[\ \forall y Pxy \vee Qxz] \supset \forall y\ \exists x. Pxy \vee Qxy$

X2135
$\exists x\ \forall y. Px \wedge Qy \supset Qx \vee Py$

X2136
$\exists x\ \exists y\ \forall u. Pxyz \supset Puxx$

X2137
$\exists x\ \forall y. Px \supset Qx \vee Py$

X2138
$\forall x\ \exists y Fxy \wedge \exists x\ \forall e\ \exists n\ \forall w[Snw \supset Dwxe] \wedge \forall e\ \exists d\ \forall a\ \forall b[Dabd$
$\supset \forall y\ \forall z. Fay \wedge Fbz \supset Dyze] \supset \exists y\ \forall e\ \exists m\ \forall w. Smw \supset \forall z. Fwz$
$\supset Dzye$

15.3 Higher-Order Logic

X5200
$x_{o\alpha} \cup y_{o\alpha} = \bigcup .\lambda v_{o\alpha}. v = x \vee v = y$

X5201
$x_{o\alpha} \cap y_{o\alpha} = \bigcap .\lambda v_{o\alpha}. v = x \vee v = y$

X5202
$\# f_{\alpha\beta}[x_{o\beta} \cup y_{o\beta}] = \# f\, x \cup \# f\, y$

X5203
$\# f_{\alpha\beta}[x_{o\beta} \cap y_{o\beta}] \subseteq \# f\, x \cap \# f\, y$

X5204
$\# f_{\alpha\beta}[\bigcup w_{o(o\beta)}] = \bigcup .\#[\# f]\, w$

X5205
$\# f_{\alpha\beta}[\bigcap w_{o(o\beta)}] \subseteq \bigcap .\#[\# f]\, w$

X5206
$\# f_{\alpha\beta}[x_{o\beta} \cup y_{o\beta}] = \# f\, x \cup \# f\, y$

X5207
$\# f_{\alpha\beta}[x_{o\beta} \cap y_{o\beta}] \subseteq \# f\, x \cap \# f\, y$

X5208
$\exists S_{o\iota}\ \forall x_\iota[[S\,x \lor P_{o\iota}\,x] \land . \sim S\,x \lor Q_{o\iota}\,x] \equiv \forall y_\iota . P\,y \lor Q\,y$

X5209
$\mathcal{P}_{o(o\alpha)(o\alpha)}[D_{o\alpha} \cap E_{o\alpha}] = \mathcal{P}\,D \cap \mathcal{P}\,E$

X5210
$[= x_\alpha] = \lambda z_\alpha\ \exists y_\alpha . y = x \land z = y$

X5211
$y_{o\alpha} = \bigcup .\lambda z_{o\alpha}\ \exists x_\alpha . y\,x \land z = [= x]$

X5212
$\lambda z_\alpha\ \exists x_\beta[g_{o\beta}\,x \land z = f_{\alpha\beta}\,x] = \# f\, g$

X5303
$= = \lambda x_\alpha \lambda y_\alpha\ \forall p_{o\alpha\alpha} .\ \forall z_\alpha\, p\,z\,z \supset p\,x\,y$

X5304
$\sim \exists g_{o\alpha\alpha}\ \forall f_{o\alpha}\ \exists j_\alpha . g\,j = f$

X5305
$\forall s_{o\alpha} . \sim \exists g_{o\alpha\alpha}\ \forall f_{o\alpha} . f \subseteq s \supset \exists j_\alpha . s\,j \land g\,j = f$

X5308
$\exists j_{\beta(o\beta)}\ \forall p_{o\beta}[\ \exists x_\beta\, p\,x \supset p . j\,p] \supset .\ \forall x_\alpha\ \exists y_\beta\, r_{o\beta\alpha}\,x\,y \equiv \exists f_{\beta\alpha}\ \forall x\, r\,x . f\,x$

X5309
$\sim \exists h_{\iota(o\iota)}\ \forall p_{o\iota}\ \forall q_{o\iota} . h\,p = h\,q \supset p = q$

X5310
$\forall r_{o\beta(o\beta)}[\ \forall x_{o\beta}\ \exists y_\beta\, r\,x\,y \supset \exists f_{\beta(o\beta)}\ \forall x\, r\,x . f\,x] \supset \exists j_{\beta(o\beta)}\ \forall p_{o\beta} .\ \exists z_\beta\, p\,z$
$\supset p . j\,p$

X5500
$\forall P_{o\beta}[\ \exists x_\beta\, P\,x \supset P . J_{\beta(o\beta)}\,P] \supset \forall f_{\alpha\beta}\ \forall g_{\alpha\beta} . f[J.\lambda x. \sim . f\,x = g\,x] =$
$g[J.\lambda x. \sim . f\,x = g\,x] \supset f = g$

X6004

$E_{o(o\alpha)(o\beta)}[= x_{\beta}]. = y_{\alpha}$

X6101

$\overline{\mathbf{1}} = \Sigma^{1}_{o(o\iota)}$

X6104

$\exists i_{o(\alpha\alpha)(\alpha\alpha)}. \forall g_{\alpha\alpha}[i\, g[\lambda x_{\alpha}\, x] \wedge i\, g.\lambda x\, g.\, g\, x] \wedge \forall f_{\alpha\alpha}\ \forall y_{\alpha}. i[\lambda x\, y]\, f \supset f\, y = y$

X6105 This is a lemma for X6106. You may need to ASSERT DESCR or T5310 or T5310A

$\forall n_{o(o\iota)}. \mathbf{NAT}\, n \supset \forall q_{o\iota}. n\, q \supset \exists j_{\iota(o\iota)}\ \forall r_{o\iota}. r \subseteq q \wedge \exists x_{\iota}\, r\, x \supset r. j\, r$

X6106

$\mathbf{FINITE}[\lambda x_{\iota}\ \top] \supset \exists j_{\iota(o\iota)}\ \forall r_{o\iota}. \exists x\, r\, x \supset r. j\, r$

X6201

$\exists r_{o\alpha\alpha}\ \forall x_{\alpha}\ \forall y_{\alpha}\ \forall z_{\alpha}[\ \exists w_{\alpha}\, r\, x\, w \wedge \sim r\, x\, x \wedge. r\, x\, y \supset . r\, y\, z \supset r\, x\, z]$
$\supset \exists R_{o(o\alpha)(o\alpha)}\ \forall X_{o\alpha}\ \forall Y_{o\alpha}\ \forall Z_{o\alpha}. \exists W_{o\alpha}\, R\, X\, W \wedge \sim R\, X\, X \wedge. R\, X\, Y$
$\supset . R\, Y\, Z \supset R\, X\, Z$

X8030A

$[g_{oo}\ \top \wedge g\ \bot] = \forall x_{o}\, g\, x$

Chapter 16

Logical Abbreviations

The internal name of this category is ABBREV.
A logical abbreviation can be defined using DEF-ABBREV. Allowable properties are: `TYPE`, `TYPELIST`, `DEFN`, `DEFN-FUN`, `MHELP`, and more.

16.1 Basic Abbreviations

<= 7 (Infix) $\lambda x_\sigma \lambda y_\sigma\ \forall p_{o\sigma}.\, p\, x \wedge \forall z_\sigma [p\, z \supset p.\ \mathbf{SUCC}_{\sigma\sigma}\, z] \supset p\, y.$

COND $\lambda x_\xi \lambda y_\xi \lambda p_o \texttt{THAT}\ q_\xi.\, p \wedge x = q \vee \sim p \wedge y = q.$

EQP E $\lambda p_{o\beta} \lambda q_{o\alpha}\ \exists s_{\alpha\beta}.\ \forall x_\beta [p\, x \supset q.\, s\, x] \wedge \forall y_\alpha.\, q\, y \supset \exists_1 x.\, p\, x \wedge y = s\, x.$

EQUIV $\equiv$ 2 (Infix) $=.$

FINITE $\lambda p_{o\iota}\ \exists n_{o(o\iota)}.\ \mathbf{NAT}\, n \wedge n\, p.$

MU μ $\lambda p_{o\sigma} \texttt{THAT}\ x_\sigma.\ \mathbf{NAT}\, x \wedge p\, x \wedge \texttt{FORALLN}\ y_\sigma.\, p\, y \supset x \leq y.$

NAT $\lambda n_{o(o\iota)}\ \forall p_{o\sigma}.\, p\ \overline{\mathbf{0}}_\sigma \wedge \forall x_\sigma [p\, x \supset p.\ \mathbf{SUCC}_{\sigma\sigma}\, x] \supset p\, n.$

NC $\lambda u_{o(o\beta)}\ \exists p_{o\beta}.\, u = E_{o(o\beta)(o\beta)}\, p.$

ONE $\overline{\mathbf{1}}$ $\mathbf{SUCC}_{\sigma\sigma}\ \overline{\mathbf{0}}_\sigma.$

RECURSION $\lambda h_{\sigma\sigma\sigma} \lambda g_\sigma \lambda n_{o(o\iota)} \texttt{THAT}\ m_\sigma\ \forall w_{o\sigma\sigma}.\, w\ \overline{\mathbf{0}}_\sigma\, g \wedge \forall x_\sigma\ \forall y_\sigma [w\, x\, y \supset w[\ \mathbf{SUCC}_{\sigma\sigma}\, x].\, h$

SIGMA1 Σ^1 $\lambda P_{o\alpha}\ \exists y_\alpha.\, P = [= y].$

SUBSET $\subseteq$ 8 (Infix) $\lambda P_{o\alpha} \lambda R_{o\alpha}\ \forall x_\alpha.\, P\, x \supset R\, x.$

SUCC $\lambda n_{o(o\iota)} \lambda p_{o\iota}\ \exists x_\iota.\, p\, x \wedge n.\lambda t_\iota.\ \sim [t = x] \wedge p\, t.$

UNITSET $\mathcal{U}$ $\lambda x_\alpha \lambda y_\alpha.\, x = y.$

ZERO $\lambda p_{o\iota}.\ \sim \exists x_\iota\, p\, x.$

16.2 Set Abbreviations

% $\lambda f_{\alpha\beta}\lambda x_{o\beta}\lambda z_{\alpha}\ \exists t_{\beta}.\, x\, t\ \wedge z\, =\, f\, t.$

COMPLEMENT 11 (Prefix) $\lambda S_{o\alpha}\lambda x_{\alpha}.\ \sim\, S\, x.$

EQUIVS $\equiv^{s}$ 7 (Infix) $\lambda P_{o\alpha}\lambda R_{o\alpha}\ \forall x_{\alpha}.\, P\, x\ \equiv\ R\, x.$

INTERSECT $\cap$ 10 (Infix) $\lambda P_{o\alpha}\lambda R_{o\alpha}\lambda x_{\alpha}.\, P\, x\ \wedge R\, x.$

POWERSET $\mathcal{P}$ $\lambda P_{o\alpha}\lambda R_{o\alpha}.\, R\ \subseteq\ P.$

SETEQUIV $\equiv^{s}$ 7 (Infix) $\lambda P_{o\alpha}\lambda R_{o\alpha}.\, P\ \subseteq\ R\ \wedge R\ \subseteq\ P.$

SETINTERSECT $\bigcap$ $\lambda D_{o(o\alpha)}\lambda x_{\alpha}\ \forall S_{o\alpha}.\, D\, S\ \supset\ S\, x.$

SETUNION $\bigcup$ $\lambda D_{o(o\alpha)}\lambda x_{\alpha}\ \exists S_{o\alpha}.\, D\, S\ \wedge S\, x.$

UNION $\cup$ 9 (Infix) $\lambda P_{o\alpha}\lambda R_{o\alpha}\lambda z_{\alpha}.\, P\, z\ \vee R\, z.$

Chapter 17

Binders

The internal name of this category is BINDER.
A binder can be defined using DEF-BINDER. Allowable properties are: `TYPELIST`, `VAR-TYPE`, `SCOPE-TYPE`, `WFF-TYPE`, `DEF-VAR`, `DEF-SCOPE`, `DEFN`, `MHELP`, and more.

17.1 wff Primitives

LAMBDA λ 100 (Prefix) Church's lambda binder.

17.2 Basic Abbreviations

EXISTS $\exists$ 100 (Prefix) Existential quantifier.

EXISTS1 $\exists_1$ 100 (Prefix) $\Sigma^1_{o(o\alpha)}.\lambda\, x_\alpha\, A_o$.

EXISTSN 100 (Prefix) $\exists\, z_\sigma.\ \mathbf{NAT}\, z\ \wedge A_o$.

FORALL $\forall$ 100 (Prefix) Universal quantifier.

FORALLN 100 (Prefix) $\forall\, z_\sigma.\ \mathbf{NAT}\, z\ \supset\ A_o$.

MU-BIND μ 100 (Prefix) $\mu.\lambda\, z_\sigma\, A_o$.

THAT 100 (Prefix) $\iota.\lambda\, z_\xi\, A_o$. Description binder: Selects the unique term such that.

Chapter 18

Logical Constants

The internal name of this category is LOGCONST.
A logical constant can be defined using DEF-LOGCONST. Allowable properties are: `TYPE`, `MHELP`, and more.

18.1 wff Primitives

AND $\land$ 5 (Infix) Denotes conjunction.

FALSEHOOD $\bot$ Denotes falsehood.

IMPLIES $\supset$ 3 (Infix) Denotes implication.

NOT $\sim$ 8 (Prefix) Denotes negation.

OR $\lor$ 4 (Infix) Denotes (inclusive) disjunction.

TRUTH $\top$ Denotes truth.

Chapter 19

Polymorphic Proper Symbols

The internal name of this category is PMPROPSYM.
A polymorphic proper symbol can be defined using DEF-PMPROPSYM. Allowable properties are: `TYPE`, `TYPELIST`, `MHELP`, and more.

19.1 wff Primitives

= Equality

IOTA Description operator

Chapter 20

Typeconstants

The internal name of this category is TYPECONST.
A typeconstant can be defined using DEF-TYPECONST. Allowable properties are: `DEFN`, `MHELP`.

20.1 wff Primitives

I The type of individuals.

O The type of truth values.

Chapter 21

Type Abbreviations

The internal name of this category is TYPEABBREV.
A type abbreviation can be defined using DEF-TYPEABBREV. Allowable properties are: `TYPE-DEFN`, `MHELP`.

21.1 wff Primitives

S The type of natural numbers.

Chapter 22

Library Commands

The internal name of this category is LIBRARYCMD.
A library command can be defined using DEFLIBRARY. Allowable properties are: `LIB-ARGTYPES`, `LIB-ARGNAMES`, `LIB-ARGHELP`, `LIB-DEFAULTFNS`, `LIB-MAINFNS`, `MHELP`.

22.1 Top Levels

LEAVE Leave LIBRARY to the next enclosing top level.

22.2 Display

KEY Search for a string in the names of all library objects. If the given string is also a keyword (see SHOW-KEYWORDS), then the keywords for each library object will also be searched. This command does not search the help messages of library objects.

LIBFILES Lists all library files in the current default directories, or in a single chosen directory.

LIBOBJECTS-IN-FILE Lists the contents of a file.

If more than one file of the given name is found in the library directories in DEFAULT-LIB-DIR and BACKUP-LIB-DIR, the user is prompted to choose one.

LIST-OF-LIBOBJECTS List all objects or all objects of specified TYPE.

SCRIBE-ALL-WFFS Write all wffs in all files in DEFAULT-LIB-DIR (and optionally BACKUP-LIB-DIR) to an mss file. The three verbosity settings are: MIN, which just shows the names of the objects, MED, which shows the help messages, keywords, provability and wffs as well, and MAX, which shows everything. As a filter, you can select any known keywords; only the wffs which satisfy all of the given keywords will be shown. See SHOW-KEYWORDS for a list of keywords.

SCRIBELIBDIR Print all the library files in a given directory into MSS files. See SCRIBELIBFILE for details.

SCRIBELIBFILE Print the specified library files into MSS files. The three verbosity settings are: MIN, which just shows the names of the objects, MED, which shows the help messages, keywords, provability and wffs as well, and MAX, which shows everything. It can take a list of filenames and a corresponding list of output files; if the latter list is too long it will be truncated, and if it is too short then the last filename given will be used for all the remaining output (so you can write a group of library files to a single output file by only supplying one output filename). After leaving TPS, run the .mss files through Scribe and print the resulting files.

Some files in the list of library files may be ambiguous, in the sense that more than one file of the given name exists in the library directories in DEFAULT-LIB-DIR and BACKUP-LIB-DIR. In this case, the user is prompted to disambiguate each ambiguous filename from first to last.

SEARCH Search the entire library, including all comments, for any one of a given list of strings, and return the names of all objects which contain such a string. This is useful for finding out, for example, which gwffs can be proven using either MS88 or MS89. WARNING: THIS COMMAND IS SLOW, AND CAN USE A LOT OF MEMORY. You might want to think about using the Unix "grep" command instead.

SEARCH2 Search the entire library, including all comments, for a given combination of strings. See also SEARCH. The syntax for the given list is essentially conjunctive normal form – it should be a list of conjuncts, each of which is a list of disjuncts. For example: ((MS88) (THM)) finds everything containing THM and MS88 ((MS88 THM)) finds everything containing THM or MS88 ((MS88 MS89) (THM EXERCISE)) finds everything containing (MS88 or MS89) and (THM or EXERCISE). WARNING: THIS COMMAND IS SLOW, AND CAN USE A LOT OF MEMORY. You might want to think about using the Unix "grep" command instead.

SHOW Display a library object.

If more than one library object of this name is stored in the library and SHOW-ALL-LIBOBJECTS is set to T, the user is prompted to disambiguate.

SHOW*-WFF Display the wff of a gwff in the library, with the associated help message, keywords and provability status. Also shows any needed objects, such as the definition and help for abbrevations used in the gwff.

SHOW-ALL-WFFS Show all wffs in all files in DEFAULT-LIB-DIR (and optionally BACKUP-LIB-DIR). As a filter, you can select any known keywords; only the wffs which satisfy all of the given keywords will be shown. See SHOW-KEYWORDS for a list of keywords.

SHOW-HELP Display the help message associated with a library object.

If more than one library object of this name is stored in the library and SHOW-ALL-LIBOBJECTS is set to T, the user is prompted to disambiguate.

SHOW-OBJECTS-IN-FILE Lists all the objects of the given type (or types) in a file.

If more than one file of the given name is found in the library directories in DEFAULT-LIB-DIR and BACKUP-LIB-DIR, the user is prompted to choose one.

SHOW-TIMING Display the timing information of a gwff in the library. NOTE: Will only display timing information that has been recorded in standard DATEREC format. If you opt for output to go to a file as well as to the screen, the format of the file will be SCRIBE or TEX if this is the current value of the STYLE flag, and GENERIC otherwise.

SHOW-WFF Display the wff of a gwff in the library.

If more than one library object of this name is stored in the library and SHOW-ALL-LIBOBJECTS is set to T, the user is prompted to disambiguate.

SHOW-WFF& HELP Display the wff of a gwff in the library, with the associated help message, keywords and provability status.

SHOW-WFFS-IN-FILE Lists the wffs in a file.

TEX-ALL-WFFS Write all wffs in all files in DEFAULT-LIB-DIR (and optionally BACKUP-LIB-DIR) to a TeX file. The three verbosity settings are: MIN, which just shows the names of the objects, MED, which shows the help messages, provability and wffs as well, and MAX, which shows everything. As a filter, you can select any known keywords; only the wffs which satisfy all of the given keywords will be shown. See SHOW-KEYWORDS for a list of keywords.

TEXLIBDIR Print all the library files in a given directory into TEX files. See TEXLIBFILE for details.

TEXLIBFILE Print the specified library files into TeX files. The three verbosity settings are: MIN, which just shows the names of the objects, MED, which shows the help messages, keywords, provability and wffs as well, and MAX, which shows everything. It can take a list of filenames and a corresponding list of output files; if the latter list is too long it will be truncated, and if it is too short then the last filename given will be used for all the remaining output (so you can write a group of library files to a single output file by only supplying one output filename). After leaving TPS, run the .tex files through TeX and print the resulting files.

22.3 Reading

DESTROY Remove a library object from TPS (the object will remain stored in the library).

FETCH Make a library object available in TPS. Will create a new TPS object if EXPERTFLAG is set to T, otherwise will create a weak label for the new library object.

If more than one library object of this name is stored in the library and SHOW-ALL-LIBOBJECTS is set to T, the user is prompted to disambiguate.

FIND-PROVABLE Look for gwffs with a certain provability status.

RESTORE-MASTERINDEX Restore library master index. Normally this need not be done by the user as it is done automatically when TPS is first entered. However, if the contents of the library may have been changed from outside of TPS (e.g. by a text editor) since TPS was started, then this command will re-initialize the library index.

RETRIEVE-FILE Make all objects in a library file available in TPS. Objects in a file are retrieved in the same order as they are stored in the file.

If more than one file of the given name is found in the library directories in DEFAULT-LIB-DIR and BACKUP-LIB-DIR, the user is prompted to choose one.

22.4 Library Structure

COPY-LIBDIR COPY-LIBDIR can be used to copy a library directory into a new library directory which TPS will automatically create, or it can be used to copy the contents of a library directory into an existing library directory. If COPY-LIBDIR is copying into an existing directory, and an object of the same name and type exists in both the source and destination directory, the original object remains in the destination directory instead of being overwritten. The user has the option of omitting the other-remarks property of the library objects. If any needed-objects are left over, the user is given the option of copying these extra needed-objects into a new library file in the destination library directory.

COPY-LIBDIR will also copy the bestmodes and keywords files, if they exist. If the target directory already has a bestmodes or keywords file, then the corresponding files will be merged.

COPY-LIBFILE Copy a file of library objects. The source file will be found among the directories in DEFAULT-LIB-DIR and BACKUP-LIB-DIR (the user will be prompted if more than one such file exists, and also if there is a choice of directories for the new file). Needed objects are not copied.

CREATE-LIB-DIR Create a directory to store files containing library items. This will not only create the directory, but create a file libindex.rec so that TPS will recognize the directory as a library directory. This command can be executed for the latter purpose even if the directory already exists. This command will automatically add the directory to DEFAULT-LIB-DIR in the current session of TPS.

CREATE-LIB-SUBDIR Creates a subdirectory of a current library directory in DEFAULT-LIB-DIR to store files containing library items. This will not only create the directory, but also creates a LIB-MASTERINDEX-FILE so that TPS will recognize the directory as a library directory. This command will also add the subdirectory to DEFAULT-LIB-DIR. TPS automatically looks for subdirectories when setting DEFAULT-LIB-DIR, so there is no need to add the subdirectory to the DEFAULT-LIB-DIR setting in the tps3.ini file.

DELETE-LIB-DIR Deletes a library directory and removes it from DEFAULT-LIB-DIR. The command will fail if the directory contains any library objects (i.e., if the index file is not empty).

DELETE-LIBFILE Delete a Library File

MOVE-LIBFILE Move a file of library objects. The source file will be found among the directories in DEFAULT-LIB-DIR and BACKUP-LIB-DIR (the user will be prompted if more than one such file exists, and also if there is a choice of directories for the new file). Needed objects are not moved.

RENAME-LIBDIR Rename a Library Directory

RENAME-LIBFILE Rename a Library File (within the same library directory)

UPDATE-LIBDIR UPDATE-LIBDIR can be used to update a (common) library directory by copying any object from a directory DEFAULT-LIB-DIR or BACKUP-LIB-DIR into the (common) library directory, if it is not already there. Before updating from a library directory, the user is asked whether to update from this directory. This is so one can choose a collection of library directories to combine into the common destination directory.

This has the same effect of

1. calling COPY-LIBDIR with copying from each (chosen) directory in DEFAULT-LIB-DIR and BACKUP-LIB-DIR into the (common) destination library directory.

2. Calling IMPORT-NEEDED-OBJECTS to ensure all needed-objects are also put into the destination directory.

If one wants to get the latest version of all library items, specify the complete pathname of a nonexistent directory when TPS prompts for a destination directory.

22.5 Editing

ADD-GOODMODES Add modes to a list of goodmodes. Also, add theorems that these goodmodes can prove.

CHANGE-PROVABILITY Change the PROVABILITY attribute of a stored gwff.

If more than one library object of this name is stored in the library and SHOW-ALL-LIBOBJECTS is set to T, the user is prompted to disambiguate.

CHECK-NEEDED-OBJECTS Checks for library objects which are not stored in the chosen directory, but are needed by some object in that directory.

COPY-LIBOBJECT Copy an object from some specified directory to the default directory. Does not copy the library entries of needed objects.

If more than one library object of this name is stored in the library and SHOW-ALL-LIBOBJECTS is set to T, the user is prompted to disambiguate.

DELETE Delete an object from the library.

If more than one library object of this name is stored in the library, the user is prompted to disambiguate.

FIX-MODES Change all references to obsolete flags into the appropriate new flag setting, for every mode in your library directory. You only need to do this once. You will be prompted before anything is changed, and you should probably keep a backup copy of your old library in case disaster strikes! THE CODE FOR THIS COMMAND SHOULD BE REWRITTEN FOR EACH RELEVANT CHANGE TO THE TPS FLAGS. At the minute, it's set up to remove references to REWRITE-DEFNS-EAGER, REWRITE-EQUAL-EXT and REWRITE-ONLY-EXT, which have been removed, and to reset REWRITE-DEFNS and REWRITE-EQUALITIES to appropriate values. It also puts LAST-MODE-NAME at the head of all settings for RECORDFLAGS.

IMPORT-NEEDED-OBJECTS Copies library objects which are not stored in the chosen directory, but are needed by some object in that directory, into the directory. If there is a choice of objects to import, and SHOW-ALL-LIBOBJECTS is set to T, then the user is prompted to choose one.

INSERT Insert an item in the library. The INSERT command can be used to create a new library object or to modify existing entries in the library. If SHOW-ALL-LIBOBJECTS is set to T, the user is prompted to indicate which existing library object to modify or which library directory into which the new object should be inserted. If AUTO-KEYWORDS is set to T, executing INSERT-LIBOBJECT requires expanding all definitions, which can take an enormous amount of time when definitions are deeply nested.

All the items will be replaced by whatever you write (or kept the same if you use the default) except for "additional remarks"; what you specify here will be added to whatever is already there. If you don't want to add additional remarks, respond with <space><return>. Use your favorite editor to make any changes within the existing comment.

INSERT-TPTP Insert a TPTP Problem into the library. The INSERT-TPTP command can be used to create a new library file containing abbreviations and theorems from a TPTP formatted file (.tps file). The destination directory is set according to the flag AUTO-LIB-DIR. It will not overwrite or delete any existing items: new items need to have different names, hence the suffix. If AUTO-KEYWORDS is set to T, executing INSERT-TPTP requires expanding all definitions, which can take an enormous amount of time when definitions are deeply nested.

INSERT-TPTP* For each TPTP Problem in the source directory, insert a new file into the library. The INSERT-TPTP* command can be used to create new library files containing abbreviations and theorems from a directory of TPTP formatted files (.tps file). The destination directory is set accordig to the flag AUTO-LIB-DIR. It will not overwrite or delete any existing items: new items need to have different names. If AUTO-KEYWORDS is set to T, executing INSERT-TPTP* requires expanding all definitions, which can take an enormous amount of time when definitions are deeply nested.

MOVE-LIBOBJECT Move an object from one library file to another. This command will also move a list of objects (either all of the same type, or all of type NIL), into a single named file.

REFORMAT Reformat the specified file. Will attempt to load all the objects in a given file and then to rewrite that file in the standard library format. This can be useful if you manually edit your library files a lot and they've started to look a little disorganized. To reformat all files in your directories, use SPRING-CLEAN.

REINDEX Reindex and reformat the specified file — i.e. reconstruct the entries in the library master index relating to the objects in a particular file (you should only need this if you've been manually editing the libindex.rec file and have accidentally lost some entries...), and then attempt to load and rewrite the file. To reindex all files in your directories, use SPRING-CLEAN. If you get an error because of parsing problems, try again but answer no to "Reformat?" (it is not possible to format a file without parsing it).

REMOVE-GOODMODES Remove modes from a list of goodmodes. Also, remove theorems that these goodmodes can prove.

RENAME-OBJECT Change the name of a library object. Does not move the object or alter it in any other way.

SORT Sort the specified file into alphabetical order, except for the given list of objects which are put at the head of the file (if they were originally in the file). This command reads in the entire file and then rewrites it; it will incidentally also catch any parsing errors.

SPRING-CLEAN Will do its best to reindex, reformat and/or sort every file in the default library directory. If your files are a real mess, you might consider using emacs to get rid of the worst of the problems before using SPRING-CLEAN. It will also delete any file in the directory that doesn't belong there Generally this means everything except .lib and libindex.rec files; you will be asked for confirmation before each file is deleted. If you get an error because of parsing problems, try again but answer no to "Reformat?" and "Sort?" (it is not possible to reformat or sort a file that cannot be parsed). Better yet, delete the unparseable entry and try again.

22.6 Keywords

ADD-KEYWORD Add a keyword to the keywords.rec file in your default directory. This must be done before the keyword can be used anywhere else in the library.

CHANGE-KEYWORDS Change the keywords attribute of a stored library object. NOTE: not all keywords can be changed. TPS may modify your list of keywords – for example, if you specify FIRST-ORDER for a problem that is higher-order, TPS will change it.

SHOW-KEYWORDS List all of the current acceptable keywords for the library.

UPDATE-KEYWORDS For each library entry, update the keywords field to include all of those keywords that can be determined automatically. Any other keywords will be left untouched. If you answer NO to the question about checking existing keywords, then this command will just attempt to fill in keywords for those objects which have none. If you answer YES, keywords will be generated for all of the objects (but existing user-defined keywords will not be overwritten).

This command will almost certainly crash if it discovers any untypable definitions, missing needed-objects, circular definitions, misprints, etc... in your library. This probably won't damage your library, but you might want to make a backup of all your files before you call this, just in case...

22.7 Best modes

ADD-BESTMODE Add a mode for the specified theorem to the list in your bestmodes.rec file. If the theorem and mode are already present in the list (either in your directory or in another user's), you will be asked to confirm the creation of a new entry. If they are already present in your own directory, you will be given the option of overwriting them.

The TEST-INIT command sets the flag TEST-THEOREMS to a collection of theorems associated with bestmodes. TPS-TEST uses this list to perform automatic testing. ADD-BESTMODE gives you the option (using the argument AUTO-TEST) of having TEST-INIT include the new theorem/bestmode pair for automatic testing. (The default is to include it.) If the mode is intended to be used interactively (e.g., for a demo), then it should not be included for automatic testing.

See Also: TPS-TEST, TEST-INIT, TEST-THEOREMS

DELETE-BESTMODE Remove an existing entry in your own bestmodes.rec file. Attempting to remove an entry in another user's bestmode.rec file will fail.

FIND-DUP-MODES List all potential duplicates in the bestmodes.rec file.

MODIFY-BESTMODE Edit an existing entry in the bestmodes.rec file. Attempting to modify a read-only mode (i.e. one in another user's directory) will create a modified copy in your own directory.

SHOW-BESTMODE List all of the current best modes for theorems in the library. Shows mode name, date, time for proof, and whether the mode is read/write (in your library) or read-only (in someone else's library).

SHOW-BESTMODE-THMS List all of the theorems that have bestmodes in bestmodes.rec files.

SHOW-NEW-BESTMODES List all of the best modes which have been added since the given date. This will search all available bestmodes.rec files, including those in other people's library directories.

UPDATE-PROVABILITY Update the PROVABILITY attribute of all the gwffs for which a best mode is known.

22.8 Library Classification

CLASSIFY-CLASS Classifies class1 under class2 within the current library classification scheme.

See Also: UNCLASSIFY-CLASS, CLASSIFY-ITEM, UNCLASSIFY-ITEM, GOTO-CLASS, CLASS-SCHEME, CREATE-CLASS-SCHEME, PCLASS, PSCHEMES, PCLASS-SCHEME-TREE, PCLASS-TREE, FETCH-LIBCLASS, FETCH-LIBCLASS*

CLASSIFY-ITEM Puts the library item into the given class within the current library classification scheme. If the item has needed objects, TPS also offers to classify these. If the flag CLASS-DIRECTION is set to UP, the needed objects must be classified in ancestors of the given class. If the flag CLASS-DIRECTION is set to DOWN, the needed objects must be classified in descendants of the given class.

See Also: CLASSIFY-CLASS, UNCLASSIFY-CLASS, UNCLASSIFY-ITEM, GOTO-CLASS, CLASS-SCHEME, CREATE-CLASS-SCHEME, PCLASS, PSCHEMES, PCLASS-SCHEME-TREE, PCLASS-TREE, FETCH-LIBCLASS, FETCH-LIBCLASS*

CREATE-CLASS-SCHEME Create a classification scheme for the library. A classification scheme is a way of organizing library items into a tree (actually a directed acyclic graph) of classes. Each class can have classes as children. Each class has associated libitems.

This classification scheme can itself be saved in the library and retrieved from the library as an object of type LIBCLASS.

A classification scheme can also be used to access the TPS library using a Unix-style interface. Use the command UNIXLIB to enter the Unix-style top level for the library.

See Also: UNIXLIB, PSCHEMES, CLASS-SCHEME, GOTO-CLASS, CREATE-LIBCLASS, CLASSIFY-CLASS, CLASSIFY-ITEM, PCLASS-SCHEME, PCLASS-SCHEME-TREE, PCLASS-TREE, FETCH-LIBCLASS, FETCH-LIBCLASS*

CREATE-LIBCLASS Creates a new class in the current classification scheme.

See Also: CREATE-CLASS-SCHEME, CLASSIFY-CLASS, UNCLASSIFY-CLASS, CLASSIFY-ITEM, UNCLASSIFY-ITEM, GOTO-CLASS, CLASS-SCHEME, PSCHEMES, FETCH-LIBCLASS, FETCH-LIBCLASS*, PCLASS, PCLASS-SCHEME-TREE, PCLASS-TREE

FETCH-DOWN Fetches all the library items classified in the current class and in all the descendents of that class are also fetched.

See Also: CLASS-DIRECTION, FETCH-LIBCLASS*, FETCH-UP, FETCH-LIBCLASS, CLASS-SCHEME, CLASSIFY-CLASS, UNCLASSIFY-CLASS, CLASSIFY-ITEM, UNCLASSIFY-ITEM, GOTO-CLASS, PSCHEMES, PCLASS, PCLASS-SCHEME-TREE, PCLASS-TREE, CREATE-CLASS-SCHEME

FETCH-LIBCLASS Fetches all the library items classified in the current class within the current library classification scheme.

See Also: FETCH-LIBCLASS*, CLASS-SCHEME, CLASSIFY-CLASS, UNCLASSIFY-CLASS, CLASSIFY-ITEM, UNCLASSIFY-ITEM, GOTO-CLASS, PSCHEMES, PCLASS, PCLASS-SCHEME-TREE, PCLASS-TREE, CREATE-CLASS-SCHEME

FETCH-LIBCLASS* Fetches all the library items classified in the current class within the current library classification scheme. If the flag CLASS-DIRECTION is set to Up, then FETCH-LIBCLASS* also fetches all the libitems classified in ancestor classes. If the flag CLASS-DIRECTION is set to Down, then FETCH-LIBCLASS* also fetches all the libitems classified in descendant classes.

See Also: FETCH-UP, FETCH-DOWN, FETCH-LIBCLASS, CLASS-DIRECTION, CLASS-SCHEME, CLASSIFY-CLASS, UNCLASSIFY-CLASS, CLASSIFY-ITEM, UNCLASSIFY-ITEM, GOTO-CLASS, ROOT-CLASS, PSCHEMES, PCLASS, PCLASS-SCHEME-TREE, PCLASS-TREE, CREATE-CLASS-SCHEME

FETCH-UP Fetches all the library items classified in the current class and in all the ancestors of that class are also fetched.

See Also: SUBCLASS-DIRECTION, FETCH-LIBCLASS*, FETCH-DOWN, FETCH-LIBCLASS, CLASS-SCHEME, CLASSIFY-CLASS, UNCLASSIFY-CLASS, CLASSIFY-ITEM, UNCLASSIFY-ITEM, GOTO-CLASS, PSCHEMES, PCLASS, PCLASS-SCHEME-TREE, PCLASS-TREE, CREATE-CLASS-SCHEME

GENERATE-CLASS-SCHEME Generate a classification scheme for all abbreviations, constants, and gwffs. TPS does some of the work, and prompts the user to interactively make other choices.

This command can also be used to update an existing class-scheme by including all library items which are not classified in the existing class-scheme.

NOTE: It is best to run this with a fresh core image. Otherwise, TPS may confuse items previously fetched from the library with objects defined in the core TPS image.

GOTO-CLASS Searches for classes of the given name within the current library classification scheme. If one is found, that class is made the current class. If several are found, the user is asked to choose.

See Also: CLASS-SCHEME, ROOT-CLASS, CREATE-CLASS-SCHEME, PCLASS, PSCHEMES, PCLASS-SCHEME-TREE, PCLASS-TREE, CLASSIFY-CLASS, UNCLASSIFY-CLASS, CLASSIFY-ITEM, UNCLASSIFY-ITEM, FETCH-LIBCLASS, FETCH-LIBCLASS*

PCLASS Prints information about the current library class in the current classification scheme.

See Also: CLASS-SCHEME, CREATE-CLASS-SCHEME, PSCHEMES, PCLASS-SCHEME-TREE, PCLASS-TREE, GOTO-CLASS, CLASSIFY-CLASS, UNCLASSIFY-CLASS, CLASSIFY-ITEM, UNCLASSIFY-ITEM, FETCH-LIBCLASS, FETCH-LIBCLASS*

PCLASS-SCHEME-TREE Prints the classification scheme as a tree starting from the root class. A list of known classification schemes is printed by PSCHEMES.

See Also: PCLASS, PSCHEMES, PCLASS-TREE, CREATE-CLASS-SCHEME, CLASSIFY-CLASS, UNCLASSIFY-CLASS, CLASSIFY-ITEM, UNCLASSIFY-ITEM, GOTO-CLASS, CLASS-SCHEME, FETCH-LIBCLASS, FETCH-LIBCLASS*

PCLASS-TREE Prints the current class and its children as a tree.

See Also: PCLASS, PSCHEMES, PCLASS-SCHEME-TREE, CREATE-CLASS-SCHEME, CLASSIFY-CLASS, UNCLASSIFY-CLASS, CLASSIFY-ITEM, UNCLASSIFY-ITEM, GOTO-CLASS, CLASS-SCHEME, FETCH-LIBCLASS, FETCH-LIBCLASS*

PINTERSECT Print the objects that are classified in all the specified classes.

See Also: pintersect*

PINTERSECT* Finds and prints the name of all the objects which, for each specified class, are classified in the class or a 'subclass'.

If CLASS-DIRECTION is set to DOWN, 'subclass' means a descendant class.

If CLASS-DIRECTION is set to UP, 'subclass' means a ancestor class.

See Also: pintersect

PSCHEMES Prints a list of Library Classification Schemes in memory.

See Also: CLASS-SCHEME, CREATE-CLASS-SCHEME, PCLASS, GOTO-CLASS, CLASSIFY-CLASS, UNCLASSIFY-CLASS, CLASSIFY-ITEM, UNCLASSIFY-ITEM, FETCH-LIBCLASS, FETCH-LIBCLASS*

ROOT-CLASS Makes the root class of the current library classification scheme the current class.

See Also: CLASS-SCHEME, GOTO-CLASS.

UNCLASSIFY-CLASS Removes class1 from class2 within the current library classification scheme.

See Also: CLASSIFY-CLASS, CLASSIFY-ITEM, UNCLASSIFY-ITEM, GOTO-CLASS, CLASS-SCHEME, CREATE-CLASS-SCHEME, PCLASS, PSCHEMES, PCLASS-SCHEME-TREE, PCLASS-TREE, FETCH-LIBCLASS, FETCH-LIBCLASS*

UNCLASSIFY-ITEM Removes the library item from the given class within the current library classification scheme.

See Also: CLASSIFY-CLASS, UNCLASSIFY-CLASS, CLASSIFY-ITEM, GOTO-CLASS, CLASS-SCHEME, CREATE-CLASS-SCHEME, PCLASS, PSCHEMES, PCLASS-SCHEME-TREE, PCLASS-TREE, FETCH-LIBCLASS, FETCH-LIBCLASS*

Chapter 23

Library Objects

The internal name of this category is LIBOBJECT.
A library object can be defined using DEFLIBOBJECT. Allowable properties are: `LIB-PROMPTFN`, `LIB-DESCR-READFN`, `LIB-ATTR-READFN`, `LIB-TPSOBJECT`, `LIB-PRINTFN`, `MHELP`.

23.1 Miscellaneous

ABBR Saving abbreviations. Abbreviations should be closed wffs.

CLASS-SCHEME Classification Scheme for a library. A classification scheme is a way of organizing library items into a tree (actually a directed acyclic graph) of classes. Each class can have classes as children. Each class has associated libitems.

To see what classification schemes are available call: LIST-OF-LIBOBJECTS CLASS-SCHEME from the lib top level.

See Also: CREATE-CLASS-SCHEME, PSCHEMES, PCLASS-SCHEME-TREE, PCLASS-TREE, CREATE-LIBCLASS, CLASSIFY-CLASS, CLASSIFY-ITEM, FETCH-LIBCLASS, FETCH-LIBCLASS*

DPAIRSET Set of disagreement pairs.

GWFF Gwff

LIB-CONST Constants and Polymorphic Proper Symbols. These are like abbreviations, but will never be expanded by TPS and hence have no definition.

MODE Define a new mode, and save it in the library. Note that you will have to explicitly set the all the flag settings that you want to save even if the mode already exists in the library. Also see MODE1.

MODE1 Define a new mode, and save it in the library. All the current flag settings for the subjects that you specify will be saved. Also see MODE.

MODES-GWFFS A list of 'good' modes. Generally, this should be a list of modes which can be used to prove many theorems automatically. We usually want a list of goodmodes to be 'complete' in the following sense: For any theorem that has a bestmode, there is some goodmode that proves the theorem.

SEE ALSO: GOODMODES, TEST-INIT, ADD-GOODMODES, REMOVE-GOODMODES

RRULE Rewrite rule

THEORY A theory (a set of axioms and rewrite rules).

23.2 Library

SLIST The library object corresponding to a searchlist.

Chapter 24

Classification Scheme For The Library.s

The internal name of this category is CLASS-SCHEME.
A Classification Scheme for the library. can be defined using DEF-CLASS-SCHEME. Allowable properties are: `CLASS-DIRECTION`, `LIBCLASS`.

24.1 Modules

LIBDIR LIBDIR is a classification scheme built based purely on the directory structure of the library directories in DEFAULT-LIB-DIR and BACKUP-LIB-DIR. Other classification schemes may be stored in and retrieved from the library.

See Also: UNIXLIB, DEFAULT-LIB-DIR, BACKUP-LIB-DIR

Chapter 25

Library Command Using A Unix Style Interfaces

The internal name of this category is UNIX-LIBRARYCMD.
A library command using a unix style interface can be defined using DEFUNIXLIBRARY. Allowable properties are: `ULIB-ARGTYPES`, `ULIB-ARGNAMES`, `ULIB-ARGHELP`, `ULIB-DEFAULTFNS`, `ULIB-MAINFNS`, `MHELP`.

25.1 Top Levels

LEAVE Leave the Unix-style LIBRARY to the next enclosing top level.

25.2 Display

FIND-GENERATED-CLASS Find a class with an automatically generated name, i.e., one with prefix 'GEN.' and cd to its parent. This command is important when GENERATE-CLASS-SCHEME has been used to automatically generate a class-scheme and one wants to rename and manipulate these classes.

See Also: GENERATE-CLASS-SCHEME, CD, MV, RENAME-CLASS, IMPORT-CLASS

GENERATE-CLASS-SCHEME Generate a classification scheme for all abbreviations, constants, and gwffs. TPS does some of the work, and prompts the user to interactively make other choices.

NOTE: It is best to run this with a fresh core image. Otherwise, TPS may confuse items previously fetched from the library with objects defined in the core TPS image.

See Also: FIND-GENERATED-CLASS, MV, RENAME-CLASS, IMPORT-CLASS

IMPORT-CLASS Move the items and subclasses of a class to its parent and delete the class.

LOCATE Locate a class or classified item in the classification scheme.

LS-ITEMS* List all accessable Library Items accessable from the current class.

See Also: UNIXLIB, CLASS-DIRECTION, LS

PDOWN Print all the subpaths of the current class.

PINTERSECT Print the objects that are classified in all the specified classes.

See Also: pintersect*

PINTERSECT* Finds and prints the name of all the objects which, for each specified class, are classified in the class or a 'subclass'.

If CLASS-DIRECTION is set to DOWN, 'subclass' means a descendant class.

If CLASS-DIRECTION is set to UP, 'subclass' means a ancestor class.

See Also: pintersect

PUP Print all the paths up to the current class.

PWD Print the path to the current class.

See Also: UNIXLIB

SHOW Display a library object.

If more than one library object of this name is stored in the class and SHOW-ALL-LIBOBJECTS is set to T, the user is prompted to disambiguate.

SHOW-ALL-WFFS Show all wffs in all files in the current library class. As a filter, you can select any known keywords; only the wffs which satisfy all of the given keywords will be shown. See SHOW-KEYWORDS for a list of keywords.

SHOW-HELP Display the help message associated with a library object.

If more than one library object of this name is stored in the library and SHOW-ALL-LIBOBJECTS is set to T, the user is prompted to disambiguate.

SHOW-WFF Display the wff of a gwff in the library.

If more than one library object of this name is stored in the class and SHOW-ALL-LIBOBJECTS is set to T, the user is prompted to disambiguate.

SHOW-WFF& HELP Display the wff of a gwff in the library, with the associated help message, keywords and provability status.

25.3 Reading

DESTROY Remove a library object from TPS (the object will remain stored in the library).

FETCH Make a library object from the current class available in TPS. Will create a new TPS object if EXPERTFLAG is set to T, otherwise will create a weak label for the new library object.

If more than one library object of this name is stored in the library and SHOW-ALL-LIBOBJECTS is set to T, the user is prompted to disambiguate.

See Also: CD, LS, PWD, LN, RM, MKDIR, SHOW

25.4 Library Classification

CD Change to a different Library Class.

See Also: UNIXLIB

CLASSIFY-ITEM Classify a Library Item in the Current Library Class.

See Also: UNIXLIB

COPY-CLASS-SCHEME Copy an existing classification scheme to a new name so the new scheme can be modified and saved in the library under this new name without changing the old classification scheme.

CP Copy a Library Item to another Class. If the user specifies a Class to copy from, then all the library items in that class are copied.

See Also: UNIXLIB, CLASSIFY-CLASS, LN

LN Classify a Class by essentially creating a Link to another place it is classified.

Example:

ln /B/C /A

makes the class C a child of A.

Unlike the Unix ln command, the link must have the same name as the target class. Also, users are not allowed to create a link that results in a cycle in the heirarchy.

See Also: UNIXLIB, CLASSIFY-CLASS, CP

LS List the Library Items and the Subclasses in the current class.

See Also: UNIXLIB

MKDIR Creates a new Library Class classified under the Current Library Class.

See Also: UNIXLIB, CREATE-LIBCLASS

MV Move an item or class to another class.

See Also: UNIXLIB, CP, RM, CD, MKDIR

RENAME-CLASS Renames a class.

NOTE: This changes the name of the class everywhere in the structure.

See Also: UNIXLIB, CP, RM, CD, MKDIR

RM Remove a Classification of a Class or Item. This DOES NOT remove a library item from the library. It only removes the classification.

If a class has more than one parent and the class is removed from its primary parent, a secondary parent becomes the primary parent. For example, if C has two parents A and B with A as the primary parent. Here, the full path to C is

/A/C

After performing

cd /A rm C

the full path to C will be

/B/C

See Also: UNIXLIB, LN, MKDIR

Chapter 26

Review Commands

The internal name of this category is REVIEWCMD.
A review command can be defined using DEFREVIEW. Allowable properties are: `ARGTYPES`, `ARGNAMES`, `ARGHELP`, `DEFAULTFNS`, `MAINFNS`, `CLOSEFNS`, `MHELP`.

26.1 Top Levels

LEAVE Leave REVIEW to the next enclosing top level.

26.2 Flags

CHANGED-FLAGS List all those flags whose current value is not the default value.

DESCRIBE Describe a flag.

DESCRIBE* List all flags under the subjects requested, along with their descriptions.

KEY Look for a key phrase in the help strings (or just the names) of flags of given subjects. See also SEARCH, at the main top level.

LIST List all flags in the given subjects with their current value.

SAVE-FLAG-RELEVANCY-INFO Save Flag Relevancy Info built from Lisp Source Files

SEE ALSO: UPDATE-RELEVANT, SHOW-RELEVANCE-PATHS

SET Directly set the value of a flag.

SETFLAG Set the value of a flag after examining it.

SETFLAGS1 Simultaneously sets multiple flags of the form ((FLAG1 . VALUE1) (FLAG2 . VALUE2)...) (the dots may be omitted); intended for use when cutting and pasting records from library or bug files. The opening and closing parentheses must be supplied.

SETFLAGS2 Simultaneously sets multiple flags of the form "FLAG1: VALUE1 FLAG2: VALUE2 ...". Intended for use when cutting and pasting records from library or bug files. User must provide double quotes before and after pasting the record, and each flag and value pair should be separated by a newline. Flag-names containing double quotes must be set separately. This command cannot handle such cases.

SHOW-RELEVANCE-PATHS Given a function F or flag A to start from and a flag B to end at, show all paths which explain why the flag B should be relevant when F is called or when the flag A has a certain value.

SUBJECTS Print a list of currently defined subjects for REVIEW.

UPDATE Update all the flags concerning the given subjects. ! will leave the remaining flags unchanged.

UPDATE-RELEVANT Update a flag and flags that are known to be relevant to the value given. For example,

update-relevant DEFAULT-MS

will allow the user to first set DEFAULT-MS. If the user sets DEFAULT-MS to MS98-1, then TPS will ask the user to set flags relevant to MS98-1.

When update-relevant is called, the user is given the option of using the current flag relevancy information in memory, loading flag relevancy information saved to a file using SAVE-FLAG-RELEVANCY, or rebuilding flag relevancy information from the Lisp source files.

26.3 Modes

ADD-FLAG-TO-MODE Add a flag to a mode. The flag will be added with its current setting. If the flag is already present, its value in the mode will be changed to its current setting.

COMPARE-MODES Compare two different modes; print a list of the values on which they differ.

COPY-MODE Make a copy of a mode, with a new name. To delete the old mode from memory, use DESTROY.

MODE Set a group of flags by switching to a mode.

REMOVE-FLAG-FROM-MODE Delete a flag from a mode. If the flag is not present in the mode, this command will do nothing.

26.4 Unification

UNIF-DEPTHS Turn off all the MAX-SUBSTS checking in unification, and use only the flags MAX-SEARCH-DEPTH, MAX-UTREE-DEPTH and MIN-QUICK-DEPTH.

UNIF-NODEPTHS Turn off all the depth checking in unification, and set the MAX-SUBSTS-VAR and MAX-SUBSTS-QUICK flags.

26.5 Best modes

FIND-MODE Find a mode from bestmodes.rec for the given theorem, and (after prompting the user) switch to the selected mode. This will search all of the bestmodes.rec files which occur in any of the directories in DEFAULT-LIB-DIR and BACKUP-LIB-DIR.

Chapter 27

Subjects

The internal name of this category is REVIEW-SUBJECT.
A subject can be defined using DEFSUBJECT. Allowable properties are: `MHELP`.

27.1 Top Levels

EDITOR Flags concerning the operation of the wff editor.

blank-lines-inserted
charsize
edppwfflag
edprintdepth
edwin-current
edwin-current-height
edwin-current-width
edwin-top
edwin-top-height
edwin-top-width
edwin-vpform
edwin-vpform-height
edwin-vpform-width
printedtfile
printedtflag
printedtflag-slides
printedtops
printvpdflag
untyped-lambda-calculus

TEST-TOP About the test-top top level.

test-easier-if-high
test-easier-if-low
test-easier-if-nil
test-easier-if-t
test-faster-if-high
test-faster-if-low
test-faster-if-nil
test-faster-if-t
test-fix-unif-depths
test-increase-time
test-initial-time-limit
test-max-search-values
test-next-search-fn
test-reduce-time
test-verbose
testwin-height
testwin-width

27.2 OTL Object

OTL-VARS Variables needed by the otlnl (outline) package.

cleanup-rulec
cleanup-same
history-size
print-dots
printlineflag
proofw-active
proofw-active+nos
proofw-active+nos-height
proofw-active+nos-width
proofw-active-height
proofw-active-width
proofw-all
proofw-all-height
proofw-all-width
scribe-line-width
short-help
slides-turnstile-indent
slides-turnstyle-indent
support-numbers
tex-line-width
turnstile-indent
turnstile-indent-auto
turnstyle-indent
turnstyle-indent-auto
use-diy

OUTLINE Flags having to do with outline manipulations.

auto-generate-hyps
default-wffeq
print-comments
support-numbers

27.3 Printing

PRINTING About printing wffs.

allscopeflag
alpha-lower-flag
atomvalflag
blank-lines-inserted
charsize
displaywff
edppwfflag
edprintdepth
edwin-current
edwin-top
edwin-vpform
elim-defns
etree-nat-verbose
fillineflag
first-order-print-mode
flushleftflag
infix-notation
leftmargin
localleftflag
pagelength
ppwfflag
print-combined-egens
print-combined-ugens
print-combined-uis
print-comments
print-deep
print-dots
print-meta
print-nodenames
print-until-ui-or-egen
print-weak
printdepth
printedtfile
printedtflag
printedtflag-slides
printedtops
printlineflag
printmatefile
printmateflag
printmateflag-slides
printmateops
printtypes
printtypes-all
proofw-active
proofw-active+nos
proofw-all
retain-initial-type
rightmargin
scope
scribe-postamble
scribe-preamble
slides-preamble
style
suppress-flags
suppress-flags-list
suppress-irrelevance-warnings
turnstile-indent
turnstile-indent-auto
turnstyle-indent
turnstyle-indent-auto
use-dot
use-internal-print-mode

PRINTING-TEX About formatting TeX output.

displaywff
in-tex-math-mode
infix-notation
latex-emulation
latex-postamble
latex-preamble
pagelength
pagewidth
ppwfflag
tex-1-postamble
tex-1-preamble
tex-break-before-symbols
tex-mimic-scribe
tex-postamble
tex-preamble
tpstex
turnstile-indent
turnstile-indent-auto
turnstyle-indent
turnstyle-indent-auto
use-internal-print-mode
vpdtex

WINDOW-PROPS Properties of windows (e.g., editor, proof windows, vpform windows).

blank-lines-inserted
edwin-current-height
edwin-current-width
edwin-top-height
edwin-top-width
edwin-vpform-height
edwin-vpform-width
etree-nat-verbose
proofw-active
proofw-active+nos
proofw-active+nos-height
proofw-active+nos-width
proofw-active-height
proofw-active-width
proofw-all
proofw-all-height
proofw-all-width
testwin-height
testwin-width
use-window-style
vpw-height
vpw-width
window-style

27.4 Flavors of Labels

INTERNAL-NAMES Choice of names for flavors of internal labels.

meta-bdvar-name
meta-label-name
meta-var-name

27.5 Saving Work

SAVING-WORK About saving and restoring work.

save-interval
save-work-on-start-up
save-work-p

27.6 Expansion Trees

ETREES Variables associated with expansion trees.

add-truth
default-ob
econj-name
edisj-name
empty-dup-info-name
eproof-name
expansion-name
false-name
imp-name
lambda-conv
leaf-name
mating-name
matingstree-name
merge-minimize-mating
min-quant-etree
min-quantifier-scope
mt-dups-per-quant
mt94-12-trigger
mtree-filter-dups
mtree-stop-immediately
neg-name
print-deep
print-nodenames
remove-leibniz
rewrite-name
selection-name
skolem-selection-name
true-name
truthvalues-hack

27.7 Mtree Operations

MTREE Flags concerning matingstree.

MTREE-TOP Flags concerning the operation of the matingstree top level.

default-expand
default-mate
default-ms
default-ob
matingstree-name
mt-default-ob-mate
mt-dups-per-quant
mt-subsumption-check
mt94-12-trigger
mtree-filter-dups
mtree-stop-immediately
tag-conn-fn
tag-mating-fn

27.8 Mating search

IMPORTANT The crucial flags that need to be set for automatic proofs.

bad-var-connected-prune
default-ms
include-coinduction-principle
include-induction-principle
max-constraint-size
max-mates
max-num-constraints
max-prim-depth
max-prim-lits
max-search-depth
max-search-limit
max-substs-quick
max-substs-var
max-utree-depth
min-prim-depth
min-prim-lits
num-of-dups
order-components
pr00-num-iterations
pr97c-max-abbrevs
pr97c-prenex
prim-bdtypes
prim-bdtypes-auto
primsub-method
rewrite-defns
rewrite-equalities
rewrite-equivs
search-time-limit
total-num-of-dups
which-constraints

MATING-SEARCH Flags concerning mating search.

add-truth
allow-nonleaf-conns
bad-var-connected-prune
default-expand
default-mate
default-ms
dissolve
dup-allowed
duplication-strategy
duplication-strategy-pfd
excluding-gc-time
first-order-mode-ms
include-coinduction-principle
include-induction-principle
initial-bktrack-limit
interrupt-enable
last-mode-name
mate-ffpair
mate-up-to-nnf
mating-verbose
max-constraint-size
max-dup-paths
max-mates
max-num-constraints
max-search-limit
merge-minimize-mating
min-quant-etree
min-quantifier-scope
monitorflag
ms-dir
ms-init-path
ms-split
ms90-3-dup-strategy
ms98-external-rewrites
ms98-pollute-global-rewrites
natree-debug
new-mating-after-dup
num-of-dups
occurs-check
order-components
prim-quantifier
print-mating-counter
printmatefile
printmateflag
printmateflag-slides
printmateops
prop-strategy
query-user
rank-eproof-fn
recordflags
remove-leibniz
rewrite-defns
rewrite-equalities
rewrite-equivs
rulep-wffeq
search-complete-paths
search-time-limit
show-time
skolem-default
timing-named
total-num-of-dups
truthvalues-hack
unify-verbose
use-diy
use-ext-lemmas
use-fast-prop-search
use-rulep
use-symsimp
which-constraints

TRANSMIT Flags which should be transmitted from a slave tps to a master tps when piy2 or diy2 is used. This is so the appropriate flag values can be recorded by a daterec after such a run.

add-truth
allow-nonleaf-conns
apply-match
assert-lemmas
bad-var-connected-prune
break-at-quantifiers
countsubs-first
default-expand
default-mate
default-ms
default-ob
default-tactic
delay-setvars
dissolve
dneg-imitation
dup-allowed
duplication-strategy
duplication-strategy-pfd
eta-rule
etree-nat-verbose
ext-search-limit
ff-delay
first-order-mode-ms
hpath-threshold
imitation-first
include-coinduction-principle
include-induction-principle
initial-bktrack-limit
last-mode-name
leibniz-sub-check
mate-ffpair
mate-up-to-nnf
mating-verbose
max-constraint-size
max-dup-paths
max-mates
max-num-constraints
max-prim-depth
max-prim-lits
max-search-depth
max-search-limit
max-substs-proj
max-substs-proj-total
max-substs-quick
max-substs-var
max-utree-depth
maximize-first
measurements
merge-minimize-mating
min-prim-depth
min-prim-lits
min-quant-etree
min-quantifier-scope
min-quick-depth
ms-dir
ms-init-path
ms-split
ms03-dup-method
ms03-quick-eunification-limit
ms03-solve-rigid-parts
ms03-solve-rigid-parts-allow-reconnects
ms03-use-jforms
ms03-use-set-constraints
ms03-verbose
ms03-weight-banned-sels
ms03-weight-change-dups
ms03-weight-disj-eunif
ms03-weight-disj-mate
ms03-weight-disj-unif
ms03-weight-dup-var
ms03-weight-eunif1
ms03-weight-eunif2
ms03-weight-flexflexdiff
ms03-weight-flexflexdiff-o
ms03-weight-flexflexsame
ms03-weight-flexflexsame-o
ms03-weight-flexrigid-branch
ms03-weight-flexrigid-eqn
ms03-weight-flexrigid-flexeqn
ms03-weight-flexrigid-mate
ms03-weight-flexrigid-noeqn
ms03-weight-flexrigid-o
ms03-weight-imitate
ms03-weight-occurs-check
ms03-weight-primsub-falsehood
ms03-weight-primsub-first-and
ms03-weight-primsub-first-equals
ms03-weight-primsub-first-exists
ms03-weight-primsub-first-forall
ms03-weight-primsub-first-not-equals
ms03-weight-primsub-first-not-proj
ms03-weight-primsub-first-or
ms03-weight-primsub-first-proj
ms03-weight-primsub-next-and
ms03-weight-primsub-next-equals
ms03-weight-primsub-next-exists
ms03-weight-primsub-next-forall
ms03-weight-primsub-next-not-equals
ms03-weight-primsub-next-not-proj
ms03-weight-primsub-next-or
ms03-weight-primsub-next-proj
ms03-weight-primsub-truth
ms03-weight-project
ms03-weight-rigid-mate
ms03-weight-rigidrigid-eqn
ms03-weight-rigidrigid-flexeqn
ms03-weight-rigidrigid-noeqn
ms03-weight-rigidrigiddiff-o
ms03-weight-rigidrigidsame-o
ms04-allow-flex-eunifs
ms04-allow-flexrigid-proj-mate
ms04-backtrack-method
ms04-check-unif-depth
ms04-delay-flexrigid-mates
ms04-delay-unif-constraints
ms04-dup-early
ms04-dup-weight
ms04-eager-unif-subst
ms04-incr-depth
ms04-initial-depth

27.9 MS88 search procedure

MS88 Flags relevant to the MS88 mating-search procedure.

default-expand
default-mate
default-ms
dup-allowed
duplication-strategy
first-order-mode-ms
initial-bktrack-limit
interrupt-enable
mate-ffpair
max-dup-paths
max-mates
max-prim-depth
max-prim-lits
merge-minimize-mating
min-prim-depth
min-prim-lits
min-quantifier-scope
ms-dir
ms-init-path
ms-split
natree-debug
new-mating-after-dup
occurs-check
order-components
pr97c-max-abbrevs
pr97c-prenex
prim-quantifier
primsub-method
prop-strategy
query-user
remove-leibniz
rewrite-defns
rewrite-equalities
rewrite-equivs
rigid-path-ck
rulep-wffeq
search-complete-paths
skolem-default
unify-verbose
use-rulep
use-symsimp

27.10 MS89 search procedure

MS89 Flags relevant to the MS89 mating-search procedure.

default-expand
default-mate
default-ms
dup-allowed
first-order-mode-ms
initial-bktrack-limit
interrupt-enable
mate-ffpair
max-dup-paths
max-mates
max-prim-depth
max-prim-lits
max-search-limit
merge-minimize-mating
min-prim-depth
min-prim-lits
min-quantifier-scope
ms-dir
ms-init-path
ms-split
ms90-3-dup-strategy
natree-debug
new-mating-after-dup
occurs-check
order-components
pr97c-max-abbrevs
pr97c-prenex
prim-quantifier
primsub-method
prop-strategy
query-user
rank-eproof-fn
remove-leibniz
rewrite-defns
rewrite-equalities
rewrite-equivs
rigid-path-ck
rulep-wffeq
search-complete-paths
search-time-limit
skolem-default
unify-verbose
use-rulep
use-symsimp

27.11 MS90-3 search procedure

MS90-3 Flags relevant to the MS90-3 mating-search procedure.

default-expand
default-mate
default-ms
dup-allowed
duplication-strategy-pfd
first-order-mode-ms
initial-bktrack-limit
interrupt-enable
max-dup-paths
max-mates
max-prim-depth
max-prim-lits
merge-minimize-mating
min-prim-depth
min-prim-lits
min-quant-etree
min-quantifier-scope
ms-init-path
ms90-3-dup-strategy
ms90-3-quick
natree-debug
new-mating-after-dup
num-frpairs
num-of-dups
order-components
pr97c-max-abbrevs
pr97c-prenex
prim-quantifier
primsub-method
print-mating-counter
prop-strategy
query-user
remove-leibniz
rewrite-defns
rewrite-equalities
rewrite-equivs
rigid-path-ck
rulep-wffeq
show-time
skolem-default
total-num-of-dups
unify-verbose
use-rulep
use-symsimp

27.12 MS90-9 search procedure

MS90-9 Flags relevant to the MS90-9 mating-search procedure.

default-expand
default-mate
default-ms
dup-allowed
duplication-strategy-pfd
first-order-mode-ms
initial-bktrack-limit
interrupt-enable
max-dup-paths
max-mates
max-prim-depth
max-prim-lits
max-search-limit
merge-minimize-mating
min-prim-depth
min-prim-lits
min-quant-etree
min-quantifier-scope
ms-init-path
ms90-3-dup-strategy
ms90-3-quick
natree-debug
new-mating-after-dup
num-frpairs
num-of-dups
order-components
pr97c-max-abbrevs
pr97c-prenex
prim-quantifier
primsub-method
print-mating-counter
prop-strategy
query-user
rank-eproof-fn
remove-leibniz
rewrite-defns
rewrite-equalities
rewrite-equivs
rigid-path-ck
rulep-wffeq
search-time-limit
show-time
skolem-default
unify-verbose
use-rulep
use-symsimp

27.13 MS91-6 and MS91-7 search procedures

MS91-6 Flags relevant to the MS91-6 mating-search procedure.

default-expand
default-mate
default-ms
dup-allowed
first-order-mode-ms
initial-bktrack-limit
interrupt-enable
mate-ffpair
max-dup-paths
max-mates
max-prim-depth
max-prim-lits
max-search-limit
merge-minimize-mating
min-prim-depth
min-prim-lits
min-quantifier-scope
ms-dir
ms-init-path
ms-split
ms91-interleave
ms91-prefer-smaller
ms91-time-by-vpaths
ms91-weight-limit-range
natree-debug
new-mating-after-dup
new-option-set-limit
occurs-check
options-generate-arg
options-generate-fn
options-generate-update
options-verbose
order-components
penalty-for-each-primsub
penalty-for-multiple-primsubs
penalty-for-multiple-subs
penalty-for-ordinary-dup
pr97c-max-abbrevs
pr97c-prenex
prim-quantifier
primsub-method
prop-strategy
query-user
reconsider-fn
remove-leibniz
rewrite-defns
rewrite-equalities
rewrite-equivs
rigid-path-ck
rulep-wffeq
search-complete-paths
search-time-limit
skolem-default
unify-verbose
use-rulep
use-symsimp
weight-a-coefficient
weight-a-fn
weight-b-coefficient
weight-b-fn
weight-c-coefficient
weight-c-fn

MS91-7 Flags relevant to the MS91-7 mating-search procedure.

default-expand
default-mate
default-ms
dup-allowed
duplication-strategy-pfd
first-order-mode-ms
initial-bktrack-limit
interrupt-enable
max-dup-paths
max-mates
max-prim-depth
max-prim-lits
max-search-limit
merge-minimize-mating
min-prim-depth
min-prim-lits
min-quant-etree
min-quantifier-scope
ms-init-path
ms90-3-dup-strategy
ms90-3-quick
ms91-interleave
ms91-prefer-smaller
ms91-time-by-vpaths
ms91-weight-limit-range
natree-debug
new-mating-after-dup
new-option-set-limit
num-frpairs
num-of-dups
options-generate-arg
options-generate-fn
options-generate-update
options-verbose
order-components
penalty-for-each-primsub
penalty-for-multiple-primsubs
penalty-for-multiple-subs
penalty-for-ordinary-dup
pr97c-max-abbrevs
pr97c-prenex
prim-quantifier
primsub-method
print-mating-counter
prop-strategy
query-user
reconsider-fn
remove-leibniz
rewrite-defns
rewrite-equalities
rewrite-equivs
rigid-path-ck
rulep-wffeq
search-time-limit
show-time
skolem-default
unify-verbose
use-rulep
use-symsimp
weight-a-coefficient
weight-a-fn
weight-b-coefficient
weight-b-fn
weight-c-coefficient
weight-c-fn

27.14 MS92-9 search procedure

MS92-9 Flags relevant to the MS92-9 mating-search procedure.

default-expand
default-mate
default-ms
dup-allowed
duplication-strategy-pfd
first-order-mode-ms
initial-bktrack-limit
interrupt-enable
max-dup-paths
max-mates
max-prim-depth
max-prim-lits
merge-minimize-mating
min-prim-depth
min-prim-lits
min-quant-etree
min-quantifier-scope
ms-init-path
ms90-3-dup-strategy
ms90-3-quick
natree-debug
new-mating-after-dup
num-frpairs
num-of-dups
order-components
pr97c-max-abbrevs
pr97c-prenex
prim-quantifier
primsub-method
prop-strategy
query-user
remove-leibniz
rewrite-defns
rewrite-equalities
rewrite-equivs
rigid-path-ck
rulep-wffeq
show-time
skolem-default
unify-verbose
use-rulep
use-symsimp

27.15 MS93-1 search procedure

MS93-1 Flags relevant to the MS93-1 mating-search procedure.

default-expand
default-mate
default-ms
dup-allowed
duplication-strategy-pfd
first-order-mode-ms
initial-bktrack-limit
interrupt-enable
max-dup-paths
max-mates
max-prim-depth
max-prim-lits
max-search-limit
merge-minimize-mating
min-prim-depth
min-prim-lits
min-quant-etree
min-quantifier-scope
ms-init-path
ms90-3-dup-strategy
ms90-3-quick
natree-debug
new-mating-after-dup
num-frpairs
num-of-dups
order-components
pr97c-max-abbrevs
pr97c-prenex
prim-quantifier
primsub-method
prop-strategy
query-user
rank-eproof-fn
remove-leibniz
rewrite-defns
rewrite-equalities
rewrite-equivs
rigid-path-ck
rulep-wffeq
search-time-limit
show-time
skolem-default
unify-verbose
use-rulep
use-symsimp

27.16 MS98-1 search procedure

MS98-1 Pertaining to the component search MS98-1.

break-at-quantifiers
default-ms
first-order-mode-ms
max-mates
max-substs-quick
max-substs-var
merge-minimize-mating
min-quantifier-scope
ms98-base-prim
ms98-external-rewrites
ms98-first-fragment
ms98-fragment-order
ms98-init
ms98-max-prims
ms98-measure
ms98-num-of-dups
ms98-pollute-global-rewrites
ms98-primsub-count
ms98-rewrite-depth
ms98-rewrite-size
ms98-rewrite-unif
ms98-rewrites
ms98-use-colors
ms98-verbose
num-of-dups
rewrite-defns
rewrite-equalities
rewrite-equivs
skolem-default

MS98-MINOR Less important flags for MS98-1.

ff-delay
hpath-threshold
maximize-first
ms98-dup-below-primsubs
ms98-dup-primsubs
ms98-force-h-o
ms98-low-memory
ms98-max-components
ms98-merge-dags
ms98-minimality-check
ms98-rew-primsubs
ms98-rewrite-model
ms98-rewrite-prune
ms98-trace
ms98-unif-hack
ms98-unif-hack2
ms98-valid-pair
ms98-variable-order

27.17 Extensional Search

EXT-SEARCH Flags concerning extensional proof search. These include all flags relevant to either of the search procedures MS03-7 or MS04-2.

ext-mate-recompute-jforms
ext-search-limit
ms03-dup-method
ms03-quick-eunification-limit
ms03-solve-rigid-parts
ms03-solve-rigid-parts-allow-reconnects
ms03-use-jforms
ms03-use-set-constraints
ms03-verbose
ms03-weight-banned-sels
ms03-weight-change-dups
ms03-weight-disj-eunif
ms03-weight-disj-mate
ms03-weight-disj-unif
ms03-weight-dup-var
ms03-weight-eunif1
ms03-weight-eunif2
ms03-weight-flexflexdiff
ms03-weight-flexflexdiff-o
ms03-weight-flexflexsame
ms03-weight-flexflexsame-o
ms03-weight-flexrigid-branch
ms03-weight-flexrigid-eqn
ms03-weight-flexrigid-flexeqn
ms03-weight-flexrigid-mate
ms03-weight-flexrigid-noeqn
ms03-weight-flexrigid-o
ms03-weight-imitate
ms03-weight-occurs-check
ms03-weight-primsub-falsehood
ms03-weight-primsub-first-and
ms03-weight-primsub-first-equals
ms03-weight-primsub-first-exists
ms03-weight-primsub-first-forall
ms03-weight-primsub-first-not-equals
ms03-weight-primsub-first-not-proj
ms03-weight-primsub-first-or
ms03-weight-primsub-first-proj
ms03-weight-primsub-next-and
ms03-weight-primsub-next-equals
ms03-weight-primsub-next-exists
ms03-weight-primsub-next-forall
ms03-weight-primsub-next-not-equals
ms03-weight-primsub-next-not-proj
ms03-weight-primsub-next-or
ms03-weight-primsub-next-proj
ms03-weight-primsub-truth
ms03-weight-project
ms03-weight-rigid-mate
ms03-weight-rigidrigid-eqn
ms03-weight-rigidrigid-flexeqn
ms03-weight-rigidrigid-noeqn
ms03-weight-rigidrigiddiff-o
ms03-weight-rigidrigidsame-o
ms04-allow-flex-eunifs
ms04-allow-flexrigid-proj-mate
ms04-backtrack-method
ms04-check-unif-depth
ms04-delay-flexrigid-mates
ms04-delay-unif-constraints
ms04-dup-early
ms04-dup-weight
ms04-eager-unif-subst
ms04-incr-depth
ms04-initial-depth
ms04-max-delayed-conns
ms04-max-depth
ms04-max-dups
ms04-max-eunif1s
ms04-max-eunif2s
ms04-max-flex-eunifs
ms04-max-flexrigid-mates
ms04-max-flexrigid-neg-mates
ms04-max-flexrigid-neg-proj-mates
ms04-max-flexrigid-proj-mates
ms04-max-imits
ms04-max-primsub-and
ms04-max-primsub-equals
ms04-max-primsub-exists
ms04-max-primsub-forall
ms04-max-primsub-not
ms04-max-primsub-not-equals
ms04-max-primsub-not-proj
ms04-max-primsub-or
ms04-max-primsub-proj
ms04-max-projs
ms04-max-rigid-mates
ms04-mp-options
ms04-prenex-primsubs
ms04-semantic-pruning
ms04-solve-unif-depth
ms04-trace
ms04-use-semantics
ms04-use-set-constraints
ms04-verbose
ms04-weight-add-set-constraint
ms04-weight-delay-unif
ms04-weight-eunif-decs
ms04-weight-eunif-diff-heads
ms04-weight-flex-eunif
ms04-weight-flexrigid-proj-mate
ms04-weight-multiple-eunif1s
ms04-weight-multiple-eunif2s
ms04-weight-multiple-mates
ms04-weight-primsub-first-not
ms04-weight-primsub-next-not
ms04-weight-primsub-nexttp
ms04-weight-primsub-occurs-const
ms04-weight-solve-set-constraints

MS03-7 Flags concerning the proof search procedure MS03-7 which incorporates extensional reasoning, equality reasoning, and set constraints. This uses extensional expansion dags instead of expansion trees. See Chad E. Brown's thesis.

default-ms
ext-search-limit
ms03-dup-method
ms03-quick-eunification-limit
ms03-solve-rigid-parts
ms03-solve-rigid-parts-allow-reconnects
ms03-use-jforms
ms03-use-set-constraints
ms03-verbose
ms03-weight-banned-sels
ms03-weight-change-dups
ms03-weight-disj-eunif
ms03-weight-disj-mate
ms03-weight-disj-unif
ms03-weight-dup-var
ms03-weight-eunif1
ms03-weight-eunif2
ms03-weight-flexflexdiff
ms03-weight-flexflexdiff-o
ms03-weight-flexflexsame
ms03-weight-flexflexsame-o
ms03-weight-flexrigid-branch
ms03-weight-flexrigid-eqn
ms03-weight-flexrigid-flexeqn
ms03-weight-flexrigid-mate
ms03-weight-flexrigid-noeqn
ms03-weight-flexrigid-o
ms03-weight-imitate
ms03-weight-occurs-check
ms03-weight-primsub-falsehood
ms03-weight-primsub-first-and
ms03-weight-primsub-first-equals
ms03-weight-primsub-first-exists
ms03-weight-primsub-first-forall
ms03-weight-primsub-first-not-equals
ms03-weight-primsub-first-not-proj
ms03-weight-primsub-first-or
ms03-weight-primsub-first-proj
ms03-weight-primsub-next-and
ms03-weight-primsub-next-equals
ms03-weight-primsub-next-exists
ms03-weight-primsub-next-forall
ms03-weight-primsub-next-not-equals
ms03-weight-primsub-next-not-proj
ms03-weight-primsub-next-or
ms03-weight-primsub-next-proj
ms03-weight-primsub-truth
ms03-weight-project
ms03-weight-rigid-mate
ms03-weight-rigidrigid-eqn
ms03-weight-rigidrigid-flexeqn
ms03-weight-rigidrigid-noeqn
ms03-weight-rigidrigiddiff-o
ms03-weight-rigidrigidsame-o
query-user

MS04-2 Flags concerning the proof search procedure MS04-2 which incorporates extensional reasoning, equality reasoning, and set constraints. This uses extensional expansion dags instead of expansion trees. See Chad E. Brown's thesis.

default-ms
max-domain-size
ms03-weight-banned-sels
ms03-weight-eunif2
ms03-weight-flexflexdiff-o
ms03-weight-flexflexsame-o
ms03-weight-flexrigid-eqn
ms03-weight-flexrigid-mate
ms03-weight-flexrigid-o
ms03-weight-occurs-check
ms03-weight-primsub-first-equals
ms03-weight-primsub-first-forall
ms03-weight-primsub-first-not-proj
ms03-weight-primsub-first-proj
ms03-weight-primsub-next-equals
ms03-weight-primsub-next-forall
ms03-weight-primsub-next-not-proj
ms03-weight-primsub-next-proj
ms03-weight-rigid-mate
ms03-weight-rigidrigid-flexeqn
ms03-weight-rigidrigiddiff-o
ms04-allow-flex-eunifs
ms04-backtrack-method
ms04-delay-flexrigid-mates
ms04-dup-early
ms04-eager-unif-subst
ms04-initial-depth
ms04-max-depth
ms04-max-eunif1s
ms04-max-flex-eunifs
ms04-max-flexrigid-neg-mates
ms04-max-flexrigid-proj-mates
ms04-max-primsub-and
ms04-max-primsub-exists
ms04-max-primsub-not
ms04-max-primsub-not-proj
ms04-max-primsub-proj
ms04-max-rigid-mates
ms04-prenex-primsubs
ms04-solve-unif-depth
ms04-use-semantics
ms04-verbose
ms04-weight-delay-unif
ms04-weight-eunif-diff-heads
ms04-weight-flexrigid-proj-mate
ms04-weight-multiple-eunif2s
ms04-weight-primsub-first-not
ms04-weight-primsub-nexttp
ms04-weight-solve-set-constraints
max-binder-computation
ms03-quick-eunification-limit
ms03-weight-eunif1
ms03-weight-flexflexdiff
ms03-weight-flexflexsame
ms03-weight-flexrigid-branch
ms03-weight-flexrigid-flexeqn
ms03-weight-flexrigid-noeqn
ms03-weight-imitate
ms03-weight-primsub-first-and
ms03-weight-primsub-first-exists
ms03-weight-primsub-first-not-equals
ms03-weight-primsub-first-or
ms03-weight-primsub-next-and
ms03-weight-primsub-next-exists
ms03-weight-primsub-next-not-equals
ms03-weight-primsub-next-or
ms03-weight-project
ms03-weight-rigidrigid-eqn
ms03-weight-rigidrigid-noeqn
ms03-weight-rigidrigidsame-o
ms04-allow-flexrigid-proj-mate
ms04-check-unif-depth
ms04-delay-unif-constraints
ms04-dup-weight
ms04-incr-depth
ms04-max-delayed-conns
ms04-max-dups
ms04-max-eunif2s
ms04-max-flexrigid-mates
ms04-max-flexrigid-neg-proj-mates
ms04-max-imits
ms04-max-primsub-equals
ms04-max-primsub-forall
ms04-max-primsub-not-equals
ms04-max-primsub-or
ms04-max-projs
ms04-mp-options
ms04-semantic-pruning
ms04-trace
ms04-use-set-constraints
ms04-weight-add-set-constraint
ms04-weight-eunif-decs
ms04-weight-flex-eunif
ms04-weight-multiple-eunif1s
ms04-weight-multiple-mates
ms04-weight-primsub-next-not
ms04-weight-primsub-occurs-const

27.18 Proof Translation

ETR-NAT Pertaining to the translation from expansion tree proofs to natural deduction proofs.

assert-lemmas
etree-nat-verbose
lambda-conv
merge-minimize-mating
nat-etree-version
pseq-use-labels
remove-leibniz
use-diy
use-rulep
use-symsimp

27.19 Unification

UNIFICATION Variables associated with Unification

apply-match
countsubs-first
dneg-imitation
eta-rule
imitation-first
leibniz-sub-check
max-search-depth
max-substs-proj
max-substs-proj-total
max-substs-quick
max-substs-var
max-utree-depth
min-quick-depth
ms03-weight-banned-sels
ms03-weight-eunif1
ms03-weight-eunif2
ms03-weight-flexflexdiff
ms03-weight-flexflexdiff-o
ms03-weight-flexflexsame
ms03-weight-flexflexsame-o
ms03-weight-flexrigid-branch
ms03-weight-flexrigid-eqn
ms03-weight-flexrigid-flexeqn
ms03-weight-flexrigid-mate
ms03-weight-flexrigid-noeqn
ms03-weight-flexrigid-o
ms03-weight-imitate
ms03-weight-occurs-check
ms03-weight-project
ms03-weight-rigid-mate
ms03-weight-rigidrigid-eqn
ms03-weight-rigidrigid-flexeqn
ms03-weight-rigidrigid-noeqn
ms03-weight-rigidrigiddiff-o
ms03-weight-rigidrigidsame-o
ms04-weight-flex-eunif
ms04-weight-flexrigid-proj-mate
ms90-3-quick
num-frpairs
pr00-max-substs-var
pruning
reduce-double-neg
rigid-path-ck
stop-at-tsn
subsumption-check
subsumption-depth
subsumption-nodes
uni-search-heuristic
unif-counter
unif-counter-output
unif-trigger
unify-verbose

27.20 Tactics

TACTICS Flags concerning tactics.

default-tactic
lambda-conv
tacmode
tactic-verbose
tacuse
ui-herbrand-limit
use-diy
use-rulep
use-symsimp

27.21 suggestions

SUGGESTS About SUGGESTIONS and GO.

go-instructions
quietly-use-defaults
resolve-conflict

27.22 Vpforms

JFORMS Variables associated with jforms.

lit-name
order-components
print-lit-name
printvpdflag
renumber-leaves
rulep-wffeq
texformat
vpd-brief
vpd-filename
vpd-lit-name
vpd-ptypes
vpd-style
vpd-vpfpage
vpform-labels
vpform-tex-magnification
vpform-tex-nest
vpform-tex-preamble
vpw-height
vpw-width

27.23 Semantics

SEMANTIC-BOUNDS Bounds related to models

max-binder-computation
max-domain-size

27.24 wff Primitives

WFF-PRIMS Flags for wff primitives, not related to parsing or printing.

name-skolem-fn
ren-var-fn
rename-all-bd-vars
rewrite-equalities

27.25 Wff Parsing

PARSING About parsing wffs.

base-type
first-order-mode-parse
lowercaseraise
make-wffops-labels
type-iota-mode

27.26 Primitive Substitutions

PRIMSUBS Variables associated with primitive substitutions.

bad-var-connected-prune
delay-setvars
include-coinduction-principle
include-induction-principle
max-constraint-size
max-num-constraints
max-prim-depth
max-prim-lits
min-prim-depth
min-prim-lits
ms03-use-set-constraints
ms03-weight-primsub-falsehood
ms03-weight-primsub-first-and
ms03-weight-primsub-first-equals
ms03-weight-primsub-first-exists
ms03-weight-primsub-first-forall
ms03-weight-primsub-first-not-equals
ms03-weight-primsub-first-not-proj
ms03-weight-primsub-first-or
ms03-weight-primsub-first-proj
ms03-weight-primsub-next-and
ms03-weight-primsub-next-equals
ms03-weight-primsub-next-exists
ms03-weight-primsub-next-forall
ms03-weight-primsub-next-not-equals
ms03-weight-primsub-next-not-proj
ms03-weight-primsub-next-or
ms03-weight-primsub-next-proj
ms03-weight-primsub-truth
ms04-prenex-primsubs
ms04-weight-primsub-first-not
ms04-weight-primsub-next-not
ms04-weight-primsub-nexttp
ms04-weight-primsub-occurs-const
ms91-interleave
neg-prim-sub
pr00-allow-subnode-conns
pr00-max-substs-var
pr00-num-iterations
pr00-require-arg-deps
pr97c-max-abbrevs
pr97c-prenex
prim-bdtypes
prim-bdtypes-auto
prim-prefix
prim-quantifier
primsub-method
primsub-var-select
which-constraints

27.27 Events

EVENTS Dealing with EVENTS.

added-conn-enabled
advice-asked-enabled
advice-file
command-enabled
command-file
considered-conn-enabled
done-exc-enabled
dupe-enabled
dupe-var-enabled
error-enabled
error-file
event-cycle
events-enabled
incomp-mating-enabled
input-error-enabled
input-error-file
mate-subsumed-test-enabled
mate-subsumed-true-enabled
mating-changed-enabled
primsub-enabled
proof-action-enabled
proof-file
quiet-events
rec-ms-file
rec-ms-filename
removed-conn-enabled
rule-error-enabled
rule-error-file
score-file
start-time-enabled
stop-time-enabled
unif-subsumed-test-enabled
unif-subsumed-true-enabled
user-passwd-file

27.28 Grader

GR-FILENAMES Files used by the grading package.

etps-file
grade-dir
grade-file
letter-grade-file
old-grade-file
old-totals-grade-file
patch-file
totals-grade-file

GR-MISC Miscellaneous variables associated with the grading package.

cal-percentage
course-name
default-penalty-fn
drop-min
due-date-flag
letter-grade-flag
new-item
print-n-digits
statistical-options

27.29 Maintenance

MAINTAIN Flags useful for system maintainers

compiled-extension
completion-options
diy2-init-time-limit
diy2-num-iterations
diy2-time-increase-factor
expertflag
goodmodes
history-size
init-dialogue
init-dialogue-fn
java-comm
load-warn-p
news-dir
omdoc-aut-creator
omdoc-catalogue
omdoc-rights
omdoc-source
omdoc-trc-creator
omdoc-type
read-lload-sources-p
save-file
show-all-packages
source-extension
source-path
test-modify
test-theorems

SYSTEM Flags containing system constants.

excluding-gc-time
lisp-implementation-type
machine-instance
machine-type
short-site-name
timing-named
xterm-ansi-bold

27.30 Rules object

RULES-MOD Flags having to do with the operation of the rules module.

27.31 Library

LIBRARY About the library facility.

add-subdirectories
auto-keywords
auto-lib-dir
backup-lib-dir
class-direction
class-scheme
default-bug-dir
default-lib-dir
default-libfile-type
default-libindex-type
elim-defns
lib-bestmode-file
lib-keyword-file
lib-masterindex-file
measurements
recordflags
remove-trailing-dir
show-all-libobjects
use-default-bug-dir

Chapter 28

Flag Or Parameters

The internal name of this category is FLAG.
A flag or parameter can be defined using DEFFLAGAllowable properties are: `FLAGTYPE`, `DEFAULT`, `PRE-CHANGE-FN`, `CHANGE-FN`, `SUBJECTS`, `RELEVANCY-PRECONDITIONS`, `IRRELEVANCY-PRECONDITIONS`, `RELEVANT-KIDS`, `IRRELEVANT-KIDS`, `MHELP`.

28.1 Top Levels

EXT-MATE-RECOMPUTE-JFORMS If T, JForms are eagerly recomputed after modifications are made to extensional expansion dags in the EXT-MATE top level. Otherwise, the user must use the command CJFORM to update the JForm. Even if the value is T, CJFORM is useful for obtaining special JForms where Flex-Flex or Flexible nodes are left out. It takes values of type BOOLEAN and belongs to subjects `EXT-SEARCH`. The default value is:

T

MT-DUPS-PER-QUANT The maximum number of times that each individual quantifier may be duplicated in the MATINGSTREE search procedures. This flag is overridden by NUM-OF-DUPS, which governs the maximum total number of duplications of all quantifiers in the matingstree search. It takes values of type INTEGER+-OR-INFINITY and belongs to subjects `TRANSMIT`, `ETREES`, `MTREE-TOP`. The default value is:

INFINITY

PROOFW-ACTIVE If T, active lines of the current proof are printed in the Current Subproof window, if this window exists. It takes values of type BOOLEAN and belongs to subjects `WINDOW-PROPS`, `PRINTING`, `OTL-VARS`. The default value is:

T

PROOFW-ACTIVE+NOS If T, active lines of the current proof are printed in the Current Subproof & Line Numbers window, if this window exists.

It takes values of type BOOLEAN and belongs to subjects `WINDOW-PROPS`, `PRINTING`, `OTL-VARS`. The default value is:

T

PROOFW-ACTIVE+NOS-HEIGHT Controls the initial height of the Current Subproof & Line Numbers window. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `OTL-VARS`. The default value is:

24

PROOFW-ACTIVE+NOS-WIDTH Controls the initial width of the Current Subproof & Line Numbers window. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `OTL-VARS`. The default value is:

80

PROOFW-ACTIVE-HEIGHT Controls the initial height of the Current Subproof window. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `OTL-VARS`. The default value is:

24

PROOFW-ACTIVE-WIDTH Controls the initial width of the Current Subproof window. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `OTL-VARS`. The default value is:

80

PROOFW-ALL If T, entire proof so far is printed in the Complete Proof window, if this window exists. It takes values of type BOOLEAN and belongs to subjects `WINDOW-PROPS`, `PRINTING`, `OTL-VARS`. The default value is:

T

PROOFW-ALL-HEIGHT Controls the initial height of the Complete Proof window. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `OTL-VARS`. The default value is:

24

PROOFW-ALL-WIDTH Controls the initial width of the Complete Proof window. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `OTL-VARS`. The default value is:

80

UNIXLIB-SHOWPATH If T, print the current class as a directory in the prompt in the Unix Style Library Top Level.

If the value is T, the prompt will be «CLASSSCHEME>:<PATH TO CLASS><num»

If the value is NIL, the prompt will be <LIB:<CLASS><num»

See Also: UNIXLIB, PSCHEMES, CLASS-SCHEME, CD, LS, PWD, LN, RM, MKDIR, FETCH, SHOW It takes values of type BOOLEAN and belongs to subjects The default value is:

T

28.2 Style

STYLE The style of the terminal output device. It takes values of type DEVSTYLE and belongs to subjects `PRINTING`. The default value is:

GENERIC

28.3 Review

ALPHA-LOWER-FLAG If T, output from ? will be made more readable (alphabetized, smaller left margin, mostly lower case) If NIL, output is in the old style (non-alphabetized, large left margin, mostly block capitals). It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

NIL

LAST-MODE-NAME LAST-MODE-NAME contains the name of the last MODE used. There is no point in the user's altering its value, since TPS only ever writes to it, and never reads from it. It takes values of type STRING and belongs to subjects `TRANSMIT`, `MATING-SEARCH`. The default value is:

""

28.4 Flags

SUPPRESS-IRRELEVANCE-WARNINGS If SUPPRESS-IRRELEVANCE-WARNINGS is T, TPS does not warn when the user sets a flag that has no effect given the current settings of other flags. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

NIL

28.5 Modes

SUPPRESS-FLAGS If T, will suppress the printing of any flags in SUPPRESS-FLAGS-LIST by the HELP MODE, COMPARE-MODES, LIST, DESCRIBE*, UPDATE and CHANGED-FLAGS commands. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

NIL

SUPPRESS-FLAGS-LIST If SUPPRESS-FLAGS is T, these flags will not be printed. SUPPRESS-FLAGS-LIST itself is always suppressed, because it's very large. It takes values of type TPSFLAGLIST and belongs to subjects `PRINTING`. The default value is:

()

28.6 Help

SHOW-ALL-PACKAGES Determines whether ENVIRONMENT will show symbols in all packages or merely accessible symbols. It takes values of type BOOLEAN and belongs to subjects `MAINTAIN`. The default value is:

NIL

28.7 Collecting Help

OMDOC-AUT-CREATOR The aut creator listed in metadata of TPS omdoc files. It takes values of type STRING and belongs to subjects `MAINTAIN`. The default value is:

```
"The␣TPS␣Project"
```

OMDOC-CATALOGUE The omdoc catalogue location. It takes values of type STRING and belongs to subjects `MAINTAIN`. The default value is:

```
"../logics/catalogue.omdoc"
```

OMDOC-RIGHTS The rights listed in metadata of TPS omdoc files. It takes values of type STRING and belongs to subjects `MAINTAIN`. The default value is:

```
"The␣formalization␣can␣be␣freely␣distributed,␣
    maintaining␣reference␣to␣the␣TPS␣source."
```

OMDOC-SOURCE The source listed in metadata of TPS omdoc files. It takes values of type STRING and belongs to subjects `MAINTAIN`. The default value is:

```
"The␣TPS␣library:␣http://gtps.math.cmu.edu/tps.html"
```

OMDOC-TRC-CREATOR The trc creator listed in metadata of TPS omdoc files. If this is the empty string, the userid is used. It takes values of type STRING and belongs to subjects `MAINTAIN`. The default value is:

```
""
```

OMDOC-TYPE The type listed in metadata of TPS omdoc files. It takes values of type STRING and belongs to subjects `MAINTAIN`. The default value is:

```
"Dataset"
```

28.8 Starting and Finishing

COMPLETION-OPTIONS If T, then the user will be offered a choice between multiple completions of a command. Also, the commands offered will come from the current top level, the main top level and the flags. If NIL, command completion will try first the current top level, then the main top level, and then the flags, and will fail if the first of these which contains any completions also contains multiple completions. For example (when T) <1>displ&

3 matching commands or flags have been found. 1) DISPLAYFILE 2) DISPLAY-TIME 3) DISPLAYWFF 4) None of these. Input a number between 1 and 4: [1]>

(when NIL) <2>displ& TPS error while reading. Multiple completions for DISPL: DISPLAYFILE DISPLAY-TIME It takes values of type BOOLEAN and belongs to subjects `MAINTAIN`. The default value is:

T

HISTORY-SIZE Maximum number of commands to save. If NIL, all commands will be saved. It takes values of type NULL-OR-POSINTEGER and belongs to subjects `MAINTAIN`, `OTL-VARS`. The default value is:

25

28.9 OTL Object

ASSERT-RRULES When T, PROVE adds to the asserted line the active rewrite rules as equational premises. It takes values of type BOOLEAN and belongs to subjects `OTL-OBJECT`. The default value is:

NIL

AUTO-GENERATE-HYPS If T, hypotheses for lines computed and filled in automatically, if NIL, the user will be asked for confirmation for each set of hypotheses. It takes values of type BOOLEAN and belongs to subjects `OUTLINE`. The default value is:

T

CLEANUP-RULEC If T, cleanup-same works on lines with multiple-line justifications. It takes values of type BOOLEAN and belongs to subjects `OTL-VARS`. The default value is:

T

CLEANUP-SAME If NIL, identical lines are not replaced when doing CLEANUP. It takes values of type BOOLEAN and belongs to subjects `OTL-VARS`. The default value is:

T

DEFAULT-WFFEQ The name of the functions which checks for equality of wffs. It takes values of type SYMBOL and belongs to subjects `OUTLINE`. The default value is:

WFFEQ–AB

PRINT-DOTS If nil, ... are not printed before a plan line. It takes values of type BOOLEAN and belongs to subjects `PRINTING`, `OTL-VARS`. The default value is:

T

PRINTLINEFLAG If nil, lines in the proof outline are not printed. It takes values of type BOOLEAN and belongs to subjects `PRINTING`, `OTL-VARS`. The default value is:

T

SHORT-HELP If T, only the rule specification will be shown when asking for help on a rule, and the command format of a command will not be shown. It takes values of type BOOLEAN and belongs to subjects `OTL-VARS`. The default value is:

NIL

28.10 Printing

PRINT-COMBINED-EGENS When set to t, the commands PBRIEF and EXPLAIN will combine lines which are a sequence of existential generalizations and print a single line. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

T

PRINT-COMBINED-UGENS When set to t, the commands PBRIEF and EXPLAIN will combine lines which are a sequence of universal generalizations and print a single line. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

T

PRINT-COMBINED-UIS When set to t, the commands PBRIEF and EXPLAIN will combine lines which are a sequence of universal instantiations and print a single line. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

T

PRINT-UNTIL-UI-OR-EGEN When set to t, the commands PBRIEF and EXPLAIN will continue to print beyond the depth specified until a line justified by UI or Egen is encountered. The intuition is that these are the real choice points in the proof. When set to nil, PBRIEF and EXPLAIN print only to the depth specified. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

NIL

28.11 Printing

ALLSCOPEFLAG If T, all brackets will be printed; no implicit scoping is assumed. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

NIL

ATOMVALFLAG If T, the name of every atom will be printed below its value. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

NIL

BLANK-LINES-INSERTED Number of blank lines printed in the proofwindows between different stages of each proof. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `PRINTING`, `EDITOR`. The default value is:

CHARSIZE Should be one of MIN, MED or MAX. Determines the size of characters used by Proofwindows and Editor Windows. Currently, MIN and MED are the same size. It takes values of type SYMBOL and belongs to subjects `PRINTING`, `EDITOR`. The default value is:

MED

DISPLAYWFF If T, formulas are printed on separate lines. It takes values of type BOOLEAN and belongs to subjects `PRINTING-TEX`, `PRINTING`. The default value is:

NIL

ELIM-DEFNS When printing a wff, first instantiate all of the definitions and lambda-normalize. This instantiation will ignore REWRITE-DEFNS, but will use the current setting of REWRITE-EQUALITIES. It's best to leave this at NIL (i.e. off), since output with it set to T can be confusing. It takes values of type BOOLEAN and belongs to subjects `LIBRARY`, `PRINTING`. The default value is:

NIL

FILLINEFLAG If NIL, every argument of an associative infix operator will have a separate line. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

NIL

FIRST-ORDER-PRINT-MODE If T, formulas are printed so they can be parsed when FIRST-ORDER-MODE-PARSE is set to T. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

NIL

FLUSHLEFTFLAG Currently this flag does nothing. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

NIL

LEFTMARGIN The global left margin of the terminal in characters. It takes values of type INTEGER+ and belongs to subjects `PRINTING`. The default value is:

0

LOCALLEFTFLAG If T, arguments of infix operators start in the same column as the operator. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

NIL

PPWFFLAG If T, formulas will generally be pretty-printed (except for the editor). For pretty-printing to work properly, the flag INFIX-NOTATION must be set to T. It takes values of type BOOLEAN and belongs to subjects `PRINTING-TEX`, `PRINTING`. The default value is:

T

PRINTDEPTH If 0, all printing will be done to arbitrary recursive depth, if n > 0 subformulas of depth n will be replaced by '& '. It takes values of type INTEGER+ and belongs to subjects `PRINTING`. The default value is:

0

PRINTTYPES If NIL, type symbols will never be printed. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

T

PRINTTYPES-ALL This flag only applies when the flag PRINTTYPES is T. If PRINTTYPES-ALL is NIL, type symbols will be printed only on the first occurrence of a variable name. If it is T, type symbols will be printed on every occurrence of a variable name. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

NIL

RETAIN-INITIAL-TYPE If T, type property is inherited from the previous occurrence (if any) of the logical symbols. Else, it is modified whenever the parser encounters a fresh occurrence. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

T

RIGHTMARGIN The global right margin of the terminal in characters.

See Also: PAGEWIDTH It takes values of type INTEGER+ and belongs to subjects `PRINTING`. The default value is:

79

SCOPE If T, all wffs will be enclosed in square brackets. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

NIL

SLIDES-PREAMBLE The preamble that is printed into the first lines of all the Scribe slides files produced by TPS. See also SCRIBE-PREAMBLE. It takes values of type STRING and belongs to subjects `PRINTING`. The default value is:

`""`

USE-DOT If T, formulas are printed using Church's dot notation. If NIL, only brackets will be used. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

T

USE-INTERNAL-PRINT-MODE If T, the internally-defined modes SCRIBE-OTL, TEX-OTL and TEX-1-OTL will be used for printing Scribe and TeX output. (See the help message for TEX-MIMIC-SCRIBE for help on the difference between the last two.) These are usually good enough, but if you want to use a custom-defined flag setting, then set this flag to NIL to override the internal modes. This may cause problems, in which case set this flag back to T. It takes values of type BOOLEAN and belongs to subjects `PRINTING-TEX`, `PRINTING`. The default value is:

NIL

28.12 Internal for Printing

INFIX-NOTATION If T, infix notation can be used for connectives and abbreviations which have an INFIX property. If NIL, infix notation is disallowed. (Note: If you set this to NIL, library objects saved with infix notation will become unreadable. Also, if you set this to NIL, you should also set PPWFFLAG to NIL since pretty-printing will not work properly without using infix notation.) It takes values of type BOOLEAN and belongs to subjects `PRINTING-TEX`, `PRINTING`. The default value is:

T

28.13 TeX

IN-TEX-MATH-MODE If T, $'s will not be printed around wffs in style TeX. It takes values of type BOOLEAN and belongs to subjects `PRINTING-TEX`. The default value is:

NIL

LATEX-EMULATION If T, all of the printing commands that produce TeX output will produce output suitable for LaTeX instead. See LATEX-PREAMBLE, LATEX-POSTAMBLE. It takes values of type BOOLEAN and belongs to subjects `PRINTING-TEX`. The default value is:

NIL

PAGELENGTH Number of lines on an output page. Used by printing routines to determine where to break output. It takes values of type POSINTEGER and belongs to subjects `PRINTING-TEX`, `PRINTING`. The default value is:

55

PAGEWIDTH Width of a page. When creating a TeX file, RIGHTMARGIN gets temporarily set to this value.

See Also: RIGHTMARGIN It takes values of type POSINTEGER and belongs to subjects `PRINTING-TEX`. The default value is:

85

TEX-BREAK-BEFORE-SYMBOLS A list of symbols that TeX will allow linebreaks before (when the flags PPWFFLAG and DISPLAYWFF are NIL). The command TEXPROOF already allows line breaks before logical constants, quantifiers, abbreviations and infix constants.

Users normally don't need to change this flag. It takes values of type SYMBOLLIST and belongs to subjects `PRINTING-TEX`. The default value is:

()

TEX-MIMIC-SCRIBE If T, TEXPROOF will give a good-looking tex output. If NIL, TEXPROOF cannot break formulas in terms of the connectives in it. So the output is a little bit ugly. Change the flag into NIL only when you cannot get a good-looking output by setting it to T. It takes values of type BOOLEAN and belongs to subjects `PRINTING-TEX`. The default value is:

T

28.14 X Windows

USE-WINDOW-STYLE If T, uses the style given by WINDOW-STYLE for output to windows other than the main one. If NIL, windows will all be in the style given by STYLE. It takes values of type BOOLEAN and belongs to subjects `WINDOW-PROPS`. The default value is:

T

WINDOW-STYLE The style of output that will be used in all the windows besides the main one, if USE-WINDOW-STYLE is T. Ignored if USE-WINDOW-STYLE is NIL. It takes values of type DEV-STYLE and belongs to subjects `WINDOW-PROPS`. The default value is:

XTERM

XTERM-ANSI-BOLD The number corresponding to the ANSI code for switching to bold font. The default is 53 (ASCII for character 5) which corresponds to blink (often displayed as bold). An alternative is 49 (ASCII for character 1) which is the ANSI standard for bold.

Further information is contained in the User's Manual and Programmer's Guide. It takes values of type INTEGER+ and belongs to subjects `SYSTEM`. The default value is:

53

28.15 Weak Labels

PRINT-WEAK If T, weak labels are printed, otherwise they wff the represent will be printed. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

T

28.16 Flavors of Labels

MAKE-WFFOPS-LABELS If T, meta labels are created by the parser, if NIL, wffops are evaluated at parse-time. It takes values of type BOOLEAN and belongs to subjects `PARSING`. The default value is:

NIL

META-LABEL-NAME The prefix for names of meta labels (from wffops). It takes values of type SYMBOL and belongs to subjects `INTERNAL-NAMES`. The default value is:

ML

PRINT-META If T, meta labels are printed, otherwise the wffop they represent will be printed. It takes values of type BOOLEAN and belongs to subjects `PRINTING`. The default value is:

NIL

28.17 Saving Work

SAVE-INTERVAL Interval of file-write of saved commands. It takes values of type INTEGER+ and belongs to subjects `SAVING-WORK`. The default value is:

5

SAVE-WORK-ON-START-UP If T, work is saved automatically whenever TPS3 is started. It takes values of type BOOLEAN and belongs to subjects `SAVING-WORK`. The default value is:

NIL

SAVE-WORK-P If T, work is saved automatically. It takes values of type BOOLEAN and belongs to subjects `SAVING-WORK`. The default value is:

T

28.18 Recording

PRINTEDTFILE The name of the file in which wffs are recorded. It takes values of type FILESPEC and belongs to subjects `PRINTING`, `EDITOR`. The default value is:

"edt.mss"

PRINTEDTFLAG If T, editor operations are recorded into open transcript files. It takes values of type BOOLEAN and belongs to subjects `PRINTING`, `EDITOR`. The default value is:

NIL

PRINTEDTFLAG-SLIDES If T, editor operations are recorded in slides style. This flag has no effect unless PRINTEDTFLAG is T. It takes values of type BOOLEAN and belongs to subjects `PRINTING`, `EDITOR`. The default value is:

NIL

PRINTEDTOPS The function or name of the function which test whether the result of a particular edop should be written to a file. It takes values of type ANYTHING and belongs to subjects `PRINTING`, `EDITOR`. The default value is:

ALWAYS–TRUE

PRINTMATEFILE The name of the file in which mateops are recorded. This has not yet been implemented, although one can record remarks (only) into the file. It takes values of type FILESPEC and belongs to subjects `PRINTING`, `MATING-SEARCH`. The default value is:

"mate.mss"

PRINTMATEFLAG If T, mating-search operations are recorded into open transcript files. Not currently implemented. It takes values of type BOOLEAN and belongs to subjects `PRINTING`, `MATING-SEARCH`. The default value is:

NIL

PRINTMATEFLAG-SLIDES If T, mating-search operations are recorded in slides style. This flag has no effect unless PRINTMATEFLAG is T. (In fact, it has no effect even if PRINTMATEFLAG is T, since it hasn't been implemented.) It takes values of type BOOLEAN and belongs to subjects `PRINTING`, `MATING-SEARCH`. The default value is:

NIL

PRINTMATEOPS The function or name of the function which test whether the result of a particular mateop should be written to a file. This has not been implemented. It takes values of type ANYTHING and belongs to subjects `PRINTING`, `MATING-SEARCH`. The default value is:

ALWAYS–TRUE

28.19 Printing Proofs into Files

LATEX-POSTAMBLE The standard way in which TPS will end a TeX file when LATEX-EMULATION is T. It takes values of type STRING and belongs to subjects `PRINTING-TEX`. The default value is:

```
"\\end{document}"
```

LATEX-PREAMBLE The preamble that is printed into the beginning of all TeX files produced by TPS when LATEX-EMULATION is T. It takes values of type STRING and belongs to subjects `PRINTING-TEX`. The default value is:

```
"\\documentclass{article}
\\setlength{\\parindent}{0pt}
\\topmargin␣0in
\\footskip␣0pt
\\textheight␣8.5in
\\oddsidemargin␣0in
\\evensidemargin␣0pt
\\textwidth␣7in
\\def\\endf{\\end{document}}
\\input␣/afs/andrew/mcs/math/TPS/doc/lib/tps.sty
\\input␣/afs/andrew/mcs/math/TPS/doc/lib/tps.tex
\\input␣/afs/andrew/mcs/math/TPS/doc/lib/vpd.tex
\\newcommand{\\markhack}[1]{\\vspace*{-0.6in}{#1}\\
    vspace*{0.35in}\\markright{{#1}}}
%a␣hack␣to␣get␣us␣a␣fake␣header␣on␣page␣1␣without␣
    having␣to␣do␣begin{titlepage}␣~␣end{titlepage}
\\begin{document}
"
```

SCRIBE-LINE-WIDTH Width of a proofline in characters. It takes values of type INTEGER+ and belongs to subjects `OTL-VARS`. The default value is:

75

SCRIBE-POSTAMBLE The postamble that is printed into all Scribe files immediately before they are closed by TPS. See SCRIBE-PREAMBLE. It takes values of type STRING and belongs to subjects `PRINTING`. The default value is:

```
""
```

SCRIBE-PREAMBLE The preamble that is printed into the first lines of all the Scribe files produced by TPS, except those that are in SLIDES style. See also SLIDES-PREAMBLE, TEX-PREAMBLE. It takes values of type STRING and belongs to subjects `PRINTING`. The default value is:

```
""
```

TEX-1-POSTAMBLE Another TeX postamble, used when TEX-MIMIC-SCRIBE is T. See TEX-POSTAMBLE. It takes values of type STRING and belongs to subjects `PRINTING-TEX`. The default value is:

```
"\\vfill\\eject\\end"
```

TEX-1-PREAMBLE Another TeX preamble, used when TEX-MIMIC-SCRIBE is T. See TEX-PREAMBLE. It takes values of type STRING and belongs to subjects `PRINTING-TEX`. The default value is:

```
"\\parindent=0pt
"
```

TEX-LINE-WIDTH width of a proofline in characters. It takes values of type INTEGER+ and belongs to subjects `OTL-VARS`. The default value is:

75

TEX-POSTAMBLE The standard way in which TPS will end a TeX file. See TEX-PREAMBLE, TEX-1-POSTAMBLE. It takes values of type STRING and belongs to subjects `PRINTING-TEX`. The default value is:

```
"\\eject\\end"
```

TEX-PREAMBLE The preamble that is printed into the beginning of all TeX files produced by TPS. See also VPFORM-TEX-PREAMBLE, TEX-1-PREAMBLE, TEX-POSTAMBLE. It takes values of type STRING and belongs to subjects `PRINTING-TEX`. The default value is:

```
""
```

TPSTEX The pathname of the tps.tex file on your system. Should be initialized by the tps3.ini file. It takes values of type STRING and belongs to subjects `PRINTING-TEX`. The default value is:

```
""
```

VPDTEX The pathname of the vpd.tex file on your system. Should be initialized by the tps3.ini file. It takes values of type STRING and belongs to subjects `PRINTING-TEX`. The default value is:

""

28.20 Proof Outline

PRINT-COMMENTS If T, print the comments attached to lines and proofs. See LINE-COMMENT and PROOF-COMMENT. It takes values of type BOOLEAN and belongs to subjects `OUTLINE`, `PRINTING`. The default value is:

T

SLIDES-TURNSTILE-INDENT Number of columns (from leftmargin) that turnstile should be indented when making slides. Compare TURNSTILE-INDENT. This flag and SLIDES-TURNSTYLE-INDENT are synonymous. It takes values of type INTEGER+ and belongs to subjects `OTL-VARS`. The default value is:

4

SLIDES-TURNSTYLE-INDENT Number of columns (from leftmargin) that turnstile should be indented when making slides. Compare TURNSTYLE-INDENT. This flag and SLIDES-TURNSTILE-INDENT are synonymous. It takes values of type INTEGER+ and belongs to subjects `OTL-VARS`. The default value is:

4

SUPPORT-NUMBERS This has three possible settings: GAP: new support lines will be put in the gap between the current planned line and the previous line, whatever it is. PLAN: new support lines will be put immediately after the previous (lower-numbered) planned line, if there is one (and as for NIL if there isn't). NIL (or anything else): new support lines will be put in whatever seems to be a sensible place.

This flag may well be useless (although non-NIL values will force it to do the right thing, TPS will probably do the right thing anyway). It takes values of type SYMBOL and belongs to subjects `OUTLINE`, `OTL-VARS`. The default value is:

NIL

TURNSTILE-INDENT Number of columns (from leftmargin) that turnstile should be indented when writing proofs in a SCRIBE file. Notice that slides use a different flag, SLIDES-TURNSTILE-INDENT. This flag and TURNSTYLE-INDENT are synonymous. It takes values of type INTEGER+ and belongs to subjects `OTL-VARS`, `PRINTING`, `PRINTING-TEX`. The default value is:

13

TURNSTILE-INDENT-AUTO Decides how turnstiles are printed in proofs. This flag works in all styles other than TEX; in particular, it works in XTERM, GENERIC, SCRIBE and SLIDES styles. There are four possible settings: FIX : put the turnstile in the column indicated by TURNSTYLE-INDENT (or SLIDES-TURNSTYLE-INDENT, in style SLIDES). MIN : print the turnstile as far to the left as possible while still having it in the same column on every line. (If this puts it off the right margin, then this will default to the same behaviour as FIX.) COMPRESS : similar to VARY, but also removes spaces at other points in the proof (e.g. around dots, and between line numbers and hypotheses). VARY : print the turnstile one space after the hypotheses in each line (so it will move from line to line). It takes values of type INDENTATION and belongs to subjects `OTL-VARS`, `PRINTING`, `PRINTING-TEX`. The default value is:

VARY

TURNSTYLE-INDENT Number of columns (from leftmargin) that turnstile should be indented when writing proofs in a SCRIBE file or on the screen. Notice that slides use a different flag, SLIDES-TURNSTYLE-INDENT. This flag and TURNSTILE-INDENT are synonymous. It takes values of type INTEGER+ and belongs to subjects `PRINTING-TEX`, `PRINTING`, `OTL-VARS`. The default value is:

13

TURNSTYLE-INDENT-AUTO Decides how turnstiles are printed in proofs. This flag works in all styles other than TEX; in particular, it works in XTERM, GENERIC, SCRIBE and SLIDES styles. There are four possible settings: FIX : put the turnstile in the column indicated by TURNSTYLE-INDENT (or SLIDES-TURNSTYLE-INDENT, in style SLIDES). MIN : print the turnstile as far to the left as possible while still having it in the same column on every line. (If this puts it off the right margin, then this will default to the same behaviour as FIX.) COMPRESS : similar to VARY, but also removes spaces at other points in the proof (e.g. around dots, and between line numbers and hypotheses). VARY : print the turnstile one space after the hypotheses in each line (so it will move from line to line). It takes values of type INDENTATION and belongs to subjects `PRINTING-TEX`, `PRINTING`, `OTL-VARS`. The default value is:

VARY

28.21 Expansion Trees

ADD-TRUTH When set to IF-NEEDED, tests whether the etree has any path of length 1; if it does, then adds a conjunct TRUTH to the vpform. When set to T, it will always add this conjunct. When set to NIL, it will never add this conjunct. (When TRUTHVALUES-HACK is NIL, it will also add a conjunct NOT FALSEHOOD). It takes values of type SYMBOL and belongs to subjects TRANSMIT, MATING-SEARCH, ETREES. The default value is:

IF−NEEDED

DUPLICATION-STRATEGY The name of a duplication strategy. Currently, either DUP-ALL or DUP-OUTER. Only applies to MS88. It takes values of type SYMBOL and belongs to subjects TRANSMIT, MS88, MATING-SEARCH. The default value is:

DUP−OUTER

DUPLICATION-STRATEGY-PFD The name of a duplication strategy for path-focused procedures. It may have either of two values: DUP-INNER and DUP-OUTER. DUP-INNER means inner quantifiers get duplicated before outer ones, while DUP-OUTER means vice versa. It takes values of type SYMBOL and belongs to subjects TRANSMIT, MS93-1, MS92-9, MS91-7, MS90-9, MS90-3, MATING-SEARCH. The default value is:

DUP−INNER

ECONJ-NAME Prefix for labels associated with conjunction nodes. It takes values of type SYMBOL and belongs to subjects ETREES. The default value is:

CONJ

EDISJ-NAME Prefix for labels associated with disjunction nodes. It takes values of type SYMBOL and belongs to subjects ETREES. The default value is:

DISJ

EMPTY-DUP-INFO-NAME Prefix for labels associated with empty-dup-info nodes. It takes values of type SYMBOL and belongs to subjects ETREES. The default value is:

EMP

EPROOF-NAME Prefix for names of expansion proofs. It takes values of type SYMBOL and belongs to subjects ETREES. The default value is:

EPR

EXPANSION-NAME Prefix for labels associated with expansion nodes. It takes values of type SYMBOL and belongs to subjects ETREES. The default value is:

EXP

FALSE-NAME Prefix for labels associated with FALSEHOOD nodes. It takes values of type SYMBOL and belongs to subjects `ETREES`. The default value is:

FALSE

IMP-NAME Prefix for labels associated with implication nodes. It takes values of type SYMBOL and belongs to subjects `ETREES`. The default value is:

IMP

INITIAL-BKTRACK-LIMIT Initial backtrack limit. If a mating exceeds this limit, a new mating will be started, and the limit incremented. If the value of the flag is set to INFINITY, then this will never happen. It takes values of type INTEGER+-OR-INFINITY and belongs to subjects `TRANSMIT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `MATING-SEARCH`. The default value is:

INFINITY

LEAF-NAME Prefix for labels associated with leaf nodes. It takes values of type SYMBOL and belongs to subjects `ETREES`. The default value is:

LEAF

MATING-NAME Prefix for names of matings. It takes values of type SYMBOL and belongs to subjects `ETREES`. The default value is:

MAT

MIN-QUANTIFIER-SCOPE When this flag is T, the scope of quantifiers is minimized before starting expansion proofs. If an eproof is found with this flag set to T, during the translation of the eproof to an ND proof RULEQ is called to fill the gap between the theorem as originally stated and its min-quantifier-scope version. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-1`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `MATING-SEARCH`, `ETREES`. The default value is:

NIL

NEG-NAME Prefix for labels associated with negation nodes. It takes values of type SYMBOL and belongs to subjects `ETREES`. The default value is:

NEG

PRINT-DEEP T will print the deep formula of an expansion or selection node, NIL will print the shallow formula, both only if PRINT-NODENAMES is NIL. It takes values of type BOOLEAN and belongs to subjects `ETREES`, `PRINTING`. The default value is:

T

PRINT-NODENAMES T will print the names of expansion and selection nodes, NIL will print either the deep or shallow formula of the node. (see the flag PRINT-DEEP). It takes values of type BOOLEAN and belongs to subjects `ETREES`, `PRINTING`. The default value is:

T

PSEQ-USE-LABELS Set to T if pseq should abbreviate formulas and print a legend. It takes values of type BOOLEAN and belongs to subjects `ETR-NAT`. The default value is:

T

REWRITE-DEFNS A list whose first element is one of NONE, EAGER, LAZY1 and DUAL, and whose other (optional) elements are lists whose first element is one of these four options and whose other elements are the names of definitions. The first element is the default behaviour for rewriting definitions, and the other lists are lists of exceptions to this default, with a different behaviour specified. NONE: do not rewrite this definition at all. EAGER: rewrite all of these definitions, in one big step, as soon as possible. LAZY1: rewrite these, one step at a time, when there are no more EAGER rewrites to do. DUAL: as LAZY1, but rewrite these abbreviations A to a conjunction of A and A, and then deepen only one of these conjuncts. (e.g. TRANSITIVE p becomes TRANSITIVE p AND FORALL x y z . [pxy AND pyz] IMPLIES pxz LAZY2: synonym for DUAL.

For example: the value (EAGER) would be interpreted as "Rewrite every definition in one step."

(DUAL (EAGER TRANSITIVE) (NONE INJECTIVE SURJECTIVE)) would be interpreted as "Rewrite TRANSITIVE whenever it appears. Don't ever rewrite INJECTIVE or SURJECTIVE. Rewrite every other definition in the DUAL way." It takes values of type REWRITE-DEFNS-LIST and belongs to subjects `TRANSMIT`, `MS98-1`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `IMPORTANT`, `MATING-SEARCH`. The default value is:

(EAGER)

REWRITE-NAME Prefix for labels associated with rewrite nodes. It takes values of type SYMBOL and belongs to subjects `ETREES`. The default value is:

REW

SELECTION-NAME Prefix for labels associated with selection nodes (in a non-skolem etree). It takes values of type SYMBOL and belongs to subjects `ETREES`. The default value is:

SEL

SHOW-SKOLEM When true, skolem terms are shown when a wff containing them is printed, otherwise a parameter is printed instead. It takes values of type BOOLEAN and belongs to subjects The default value is:

NIL

SKOLEM-DEFAULT Default method for skolemizing, in which wffs of the form EXISTS y . M are replaced by M(g(...)). There are three possible ways to do this: SK1 is the original method due to Skolem, where the Skolem constants g take as arguments all the x such that FORALL x occurs in the wff and EXISTS y . M is in its scope. SK3 is the method in which the arguments of g are the free variables of EXISTS y . M. NIL means don't Skolemize at all; use selection nodes instead. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `MS98-1`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `MATING-SEARCH`. The default value is:

SK1

SKOLEM-SELECTION-NAME Prefix for labels associated with selection nodes (in a skolem etree). It takes values of type SYMBOL and belongs to subjects `ETREES`. The default value is:

SKOL

TRUE-NAME Prefix for labels associated with TRUTH nodes. It takes values of type SYMBOL and belongs to subjects `ETREES`. The default value is:

TRUE

TRUTHVALUES-HACK When this flag is T, leaves of truthvalues will not deepened into an empty disjunction or an empty conjunction. this allows us to deal with truthvalues in formulas, especially, higher-order formulas. In order to deal with truthvalues in definitions, such as NULLSET, the definitions containing falsehood should be rewritten. Please put new definitions containing falsehood into truthvalues-hack-updatelist so that they can be rewritten appropriately. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MATING-SEARCH`, `ETREES`. The default value is:

NIL

28.22 Mtree Operations

DEFAULT-OB If DEEPEST, the default next obligation is found by depth-first search of the obtree, if HIGHEST it is found by breadth-first-search, if D-SMALLEST then the deepest of the set of smallest obligations (i.e.

the set of all obligations with the fewest possible literals) is chosen, if H-SMALLEST then the highest of this set is chosen. It takes values of type OBDEFAULT and belongs to subjects `TRANSMIT`, `ETREES`, `MTREE-TOP`. The default value is:

D–SMALLEST

MT-DEFAULT-OB-MATE Determines how ADD-CONN chooses the default obligation for the second literal of the given pair (it is possible that this literal will occur several times on the path, in several different obligations). Options are: LOWEST : Chooses the obligation which lies lowest (i.e. furthest from the root) HIGHEST : Chooses the obligation nearest to the root (but not the root). HI-LO : Finds the obligation which occurs lowest; this obligation was first added at some point in the matingstree. Then chooses the highest obligation which was added at the same point in the matingstree. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `MTREE-TOP`. The default value is:

LOWEST

28.23 Mtree Auto

MT-SUBSUMPTION-CHECK If SAME-CONNS or T, will check whether the node about to be added is duplicated elsewhere in the tree, and will reject it if it is. (This will use the SAME-TAG function described below, and then do a more thorough check if the tags match.)

If SUBSET-CONNS, will check whether the connections at the node about to be added are a subset of those at some other node. (This is only really useful in MT94-11, where all possible new nodes are added, breadth-first, to the tree. It is probably too restrictive for the other mtree searches.)

If SAME-TAG will check whether the tag (an integer generated from the list of connections) is the same as any other existing tag, and will reject it if it is. See TAG-CONN-FN and TAG-LIST-FN. (Note that most tag functions can produce the same tag for different matings, so this may reject connections unnecessarily.)

If NIL, will turn off subsumption checking altogether. It takes values of type MT-SUBSUMPTION and belongs to subjects `TRANSMIT`, `MTREE-TOP`. The default value is:

SAME–CONNS

MT94-12-TRIGGER If the current obligation contains fewer than MT94-12-TRIGGER literals, MT94-12 will behave in the same way as MT94-11 If it contains MT94-12-TRIGGER or more, MT94-12 will choose a literal with as few mates as possible. There are two extrema: infinity means that the least branch will only be chosen if the obligation is as big as the initial obligation; 0 means that the least branch will always be chosen. It takes values of type INTEGER+-OR-INFINITY and belongs to subjects `TRANSMIT`, `ETREES`, `MTREE-TOP`. The default value is:

INFINITY

MTREE-FILTER-DUPS If T, will not add the same link to a mating twice on the same branch of a matingstree during automatic search. If NIL, will add it as many times as it wants to. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `ETREES`, `MTREE-TOP`. The default value is:

T

MTREE-STOP-IMMEDIATELY If T, will stop an automatic search as soon as a closed node is found. If NIL, will continue to generate whatever level of the tree it was working on, and will check for closed nodes when it finishes. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `ETREES`, `MTREE-TOP`. The default value is:

T

TAG-CONN-FN Determines how the tag (a number attached to each mating) is calculated. Should be the name of a function which, given a connection, will generate an integer from it. See MT-SUBSUMPTION-CHECK and TAG-MATING-FN.

Current settings are TAG-CONN-QUICK, which uses TPS's internal number for the connection. (Actually, it uses (1 + this number), so as to avoid multiplying by one.) TAG-CONN-LEAFNO, which multiplies the integer parts of the two leaf names in the given connection. It takes values of type SYMBOL and belongs to subjects `MTREE-TOP`. The default value is:

TAG–CONN–LEAFNO

TAG-MATING-FN Determines how the tags for each connection are combined to produce a tag for the entire mating. Should be the name of a function which, given two integers, will generate a third integer. See MT-SUBSUMPTION-CHECK and TAG-MATING-FN.

Current settings are MULTIPLY-TAG-LIST, which simply multiplies the numbers together. It takes values of type SYMBOL and belongs to subjects `MTREE-TOP`. The default value is:

MULTIPLY–TAG–LIST

28.24 Mating search

ASSERT-LEMMAS If this is set to T, Lemmas are justified in the natural deduction proofs using an Assert. The Assert gives the name of the proof of the Lemma.

Lemmas may be introduced in the following circumstances:

. when extensionality is used (USE-EXT-LEMMAS must be set to T) . when set variables are solved instantiated using constraints (DELAY-SETVARS must be set to T)

If lemmas L1, . . ., Ln are used to prove A, then the full proof consists of proofs of each of the Li and a proof of A using the lemmas Li. In other words, it is a proof of

[L1 and . . . and Ln] and [[L1 and . . . and Ln] implies A] It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `ETR-NAT`. The default value is:

T

DEFAULT-EXPAND Used with DEFAULT-MATE to determine a setting for DEFAULT-MS. Combinations marked N/A will result in DEFAULT-MS being set to NIL. Notice that for otree and oset searches, the actual primsubs generated will depend on the setting of PRIMSUB-METHOD. Takes values: none, ms98-1, ms03-7, ms04-2, otree and oset. The values MS98-1, MS03-7 and MS04-2 are exceptional settings used for both this flag and DEFAULT-MATE to denote the MS98-1, MS03-7 and MS04-2 procedures. Changes DEFAULT-MS as follows: DEFAULT-EXPAND: | NONE | OTREE | OSET | ========================+=========+========= DEFAULT-MATE: NPFD | MS88 | MS89 | MS91-6 | ————+——— +———+———+ NPFD-1 | MS92-9 | MS93-1 | N/A | ————+—— —+———+———+ PFD | MS90-3 | MS90-9 | MS91-7 | ————+—— —+———+———+ MTREE | MT94-11 | N/A | N/A | ————+—— —+———+———+ MTREE-1 | MT94-12 | N/A | N/A | ————+— ——+———+———+ MTREE-2 | MT95-1 | N/A | N/A | ————+— ——+———+———+ It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `MTREE-TOP`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `MATING-SEARCH`. The default value is:

OTREE

DEFAULT-MATE Used with DEFAULT-EXPAND to determine a setting for DEFAULT-MS. Combinations marked N/A will result in DEFAULT-MS being set to NIL. (Notice that for otree and oset searches, the actual primsubs generated will depend on the setting of PRIMSUB-METHOD.) Takes values: ms98-1, ms03-7, ms04-2, npfd, npfd-1, pfd, mtree, mtree-1 and mtree-2. The values MS98-1, MS03-7 and MS04-2 are exceptional settings used for both this flag and DEFAULT-EXPAND to denote the MS98-1, MS03-7 and MS04-2 procedures. Changes DEFAULT-MS as follows: DEFAULT-EXPAND: | NONE | OTREE | OSET | ========================+===== DEFAULT-MATE: NPFD | MS88 | MS89 | MS91-6 | ————+——— +———+———+ NPFD-1 | MS92-9 | MS93-1 | N/A | ————+—— —+———+———+ PFD | MS90-3 | MS90-9 | MS91-7 | ————+—— —+———+———+ MTREE | MT94-11 | N/A | N/A | ————+—— —+———+———+ MTREE-1 | MT94-12 | N/A | N/A | ————+— ——+———+———+ MTREE-2 | MT95-1 | N/A | N/A | ————+— ——+———+———+ It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `MTREE-TOP`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `MATING-SEARCH`. The default value is:

PFD

DEFAULT-MS The default mating search procedure to be used when either the DIY command or the mate level GO command is invoked. This will be changed if you set the DEFAULT-MATE and DEFAULT-EXPAND flags (they may also change DEFAULT-MS to NIL, if you pick a non-existent combination – see the help messages for those flags). Conversely, setting DEFAULT-MS will set the values of DEFAULT-MATE and DEFAULT-EXPAND, as follows: (Notice that for otree and oset searches, the actual primsubs generated will depend on the setting of PRIMSUB-METHOD.)
DEFAULT-EXPAND: | NONE | OTREE | OSET | ==========================+======
DEFAULT-MATE: NPFD | MS88 | MS89 | MS91-6 | ————+———
+———+———+ NPFD-1 | MS92-9 | MS93-1 | N/A | ————+——
—+———+———+ PFD | MS90-3 | MS90-9 | MS91-7 | ————+——
—+———+———+ MTREE | MT94-11 | N/A | N/A | ————+——
—+———+———+ MTREE-1 | MT94-12 | N/A | N/A | ————+—
——+———+———+ MTREE-2 | MT95-1 | N/A | N/A | ————+—
——+———+———+

(Setting DEFAULT-MS to MS98-1, MS03-7 or MS04-2 will also set both DEFAULT-EXPAND and DEFAULT-MATE to MS98-1, MS03-7 or MS04-2, since those procedures don't really fit into the above table.) Possible values are MS88, MS89, MS90-3, MS90-9, MS91-6, MS91-7, MS92-9, MS93-1, MT94-11, MT94-12, MT95-1, MS98-1, MS03-7 and MS04-2. It takes values of type SEARCHTYPE and belongs to subjects `TRANSMIT`, `IMPORTANT`, `MTREE-TOP`, `MS04-2`, `MS03-7`, `MS98-1`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `MATING-SEARCH`. The default value is:

MS90−3

DIY2-INIT-TIME-LIMIT Initial time limit for running DIY2 and PIY2 iteratively with increasing time limits. It takes values of type INTEGER+-OR-INFINITY and belongs to subjects `MAINTAIN`. The default value is:

2

DIY2-NUM-ITERATIONS Number of iterations for DIY2 and PIY2 to run on the same mode with increasing time limits. It takes values of type INTEGER+-OR-INFINITY and belongs to subjects `MAINTAIN`. The default value is:

1

DIY2-TIME-INCREASE-FACTOR Factor to increase time limit on each iteration when running DIY2 and PIY2. It takes values of type POSNUMBER and belongs to subjects `MAINTAIN`. The default value is:

2

INTERRUPT-ENABLE When true, allows user to interrupt mating search by typing a <RETURN>; otherwise mating search will continue until it succeeds or is aborted by a CTRL-G. You may want to set this flag to nil if you are going to have input commands (examples to run, etc.) read in from a file. It takes values of type BOOLEAN and belongs to subjects `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `MATING-SEARCH`. The default value is:

T

MATING-VERBOSE Should be one of SILENT, MIN, MED, or MAX. Determines the amount of information given about the current mating process. It takes values of type VERBOSE and belongs to subjects `TRANSMIT`, `MATING-SEARCH`. The default value is:

MED

MONITORFLAG The monitor is switched on if this flag is T and off if it is NIL. This flag is set by the command MONITOR, and unset by the command NOMONITOR (and may of course also be set manually). It takes values of type BOOLEAN and belongs to subjects `MATING-SEARCH`. The default value is:

NIL

NEW-MATING-AFTER-DUP This flag affects the way a complete mating is constructed after duplication. If nil, mating search attempts to extend only those matings which were inextensible earlier. Otherwise, it starts constructing new matings. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `MATING-SEARCH`. The default value is:

NIL

QUERY-USER Has the following effects according to its value: T : User will be queried by the mating search process as to whether a duplication of variables should occur, unification depth should be increased, etc. NIL : The mating search process will take some action that makes sense. QUERY-JFORMS : The mating search process will stop after printing each vpform and ask whether to search on this vpform or to generate another. (Note: in MS90-3, this is pointless, since the vpform never changes.) SHOW-JFORMS : Like QUERY-JFORMS, but automatically answers no to each question (and hence never actually proceeds with a search). QUERY-SLISTS : In the TEST top level, stops after each setting of the flags and asks whether to search with those settings. It takes values of type QUERYTYPE and belongs to subjects `TRANSMIT`, `MS03-7`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `MATING-SEARCH`. The default value is:

NIL

REC-MS-FILE If true, mating search events are recorded in file named by flag rec-ms-filename. This only works for npfd procedures MS88, MS89 and MS91-6. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

NIL

REC-MS-FILENAME Name of file in which mating search events are recorded. (See REC-MS-FILE.) It takes values of type FILESPEC and belongs to subjects `EVENTS`. The default value is:

"mating.rec"

USE-DIY When T, proof lines which are proven by DIY, DIY-L or UNIFORM-SEARCH-L will not be translated into natural deduction style, but will instead be justified in a single step, as "Automatic" from the support lines. A comment will be added to the relevant line of the proof showing the time taken and the mode used for the automatic proof.

Obviously, ND proofs containing justifications of this sort cannot be translated by NAT-ETREE. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `OTL-VARS`, `TACTICS`, `MATING-SEARCH`, `ETR-NAT`. The default value is:

NIL

USE-EXT-LEMMAS If this is set to T, then diy finds all positive and negative literals which have a proper subterm of propositional, set, or relation types. For example, the jform may have a positive literal P X(OA) and a negative literal P Y(OA). For each pair of subterms such as X and Y, extensionality lemmas of the form

forall x [X x EQUIV Y x] implies X = Y

are added to the expansion tree before beginning mating search. Note that the type A is determined by the types of the subterms X and Y.

See Also: ADD-EXT-LEMMAS It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MATING-SEARCH`. The default value is:

NIL

USE-FAST-PROP-SEARCH If T, will attempt to use the path-focused fast propositional theorem prover on all problems, before switching to the usual default mating-search if this fails. If NIL, will use the default mating-search only. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MATING-SEARCH`. The default value is:

T

28.25 MS88 search procedure

ADDED-CONN-ENABLED If NIL, recording events of type ADDED-CONN is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

CONSIDERED-CONN-ENABLED If NIL, recording events of type CONSIDERED-CONN is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

DUP-ALLOWED If T mating search duplicates quantifiers whenever necessary. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `MATING-SEARCH`. The default value is:

T

DUPE-ENABLED If NIL, recording events of type DUPE is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

DUPE-VAR-ENABLED If NIL, recording events of type DUPE-VAR is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

EXCLUDING-GC-TIME If T, we can use the function get-net-internal-run-time to exclude the gc time in recordings. Otherwise, get-net-internal-run-time is the same as get-internal-run-time. The value of the flag should not be changed. This is a nominal flag, whose value does not affect the system at all except telling users the message above. Check the flags SEARCH-TIME-LIMIT and MAX-SEARCH-LIMIT to get more information. It takes values of type BOOLEAN and belongs to subjects `MATING-SEARCH`, `SYSTEM`. The default value is:

NIL

FIRST-ORDER-MODE-MS If T first-order unification is called during mating search, else higher-order unification is used. TPS changes the value of this flag to T when it is called by DIY to work on a first-order problem, but not when it is called from MATE. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-1`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `MATING-SEARCH`. The default value is:

NIL

INCOMP-MATING-ENABLED If NIL, recording events of type INCOMP-MATING is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

MATE-FFPAIR Controls whether to consider a pair of literals with flexible heads as a potential connection. The MS controller will locally modify it under certain conditions; in particular, it will always be set locally to T in the following cases, among others: a) for first-order problems (when FIRST-ORDER-MODE-MS is T). b) when a mating is removed because it is incompatible with the etree. c) when using the interactive command ADD-CONN. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS91-6`, `MS89`, `MS88`, `MATING-SEARCH`. The default value is:

NIL

MATE-SUBSUMED-TEST-ENABLED If NIL, recording events of type MATE-SUBSUMED-TEST is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

MATE-SUBSUMED-TRUE-ENABLED If NIL, recording events of type MATE-SUBSUMED-TRUE is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

MATING-CHANGED-ENABLED If NIL, recording events of type MATING-CHANGED is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

MS-INIT-PATH If NIL MS considers the current path when a new mating is started. Otherwise, starts from the beginning in the natural ordering on paths in a jform. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `MATING-SEARCH`. The default value is:

NIL

MS-SPLIT If T mating search attempts to split the proof. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS91-6`, `MS89`, `MS88`, `MATING-SEARCH`. The default value is:

T

OCCURS-CHECK This flag is not effective unless FIRST-ORDER-MODE-MS is T. If its value is T, occurs check in first-order unification is postponed till a mating is complete. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS91-6`, `MS89`, `MS88`, `MATING-SEARCH`. The default value is:

T

PRIM-QUANTIFIER When NIL, primitive substitutions containing new quantifiers will not be applied. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `MATING-SEARCH`, `PRIMSUBS`. The default value is:

T

PRIMSUB-ENABLED If NIL, recording events of type PRIMSUB is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

PROP-STRATEGY This flag is only used in PROPOSITIONAL proof search, which can be one of (1) allow-duplicates (2) hash-table (3) pushnew (1) Adds CONNECTION to the mating even though it might already be in the mating. In case of (2) and (3) adds CONNECTION to the mating only if it is not already in the mating. (2) uses HASH-TABLE to determine this. (3) uses CLISP macro PUSHNEW to determine this. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `MATING-SEARCH`. The default value is:

ALLOW–DUPLICATES

REMOVED-CONN-ENABLED If NIL, recording events of type REMOVED-CONN is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

SEARCH-COMPLETE-PATHS Not yet implemented. If NIL paths are generated only to a length until a connection can be located on it. Otherwise full paths are generated. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS91-6`, `MS89`, `MS88`, `MATING-SEARCH`. The default value is:

NIL

START-TIME-ENABLED If NIL, recording events of type START-TIME is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

STOP-TIME-ENABLED If NIL, recording events of type STOP-TIME is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

TIMING-NAMED If T, the labels printed by display-time will be shortened to allow room for the name of the current dproof, if there is one. If NIL, then they won't. Abbreviations used are: PRE - preprocessing, MS - mating search, U - unification, PPR - postprocessing, MRG - merging, TRA - translation, PRT - printing. It takes values of type BOOLEAN and belongs to subjects `MATING-SEARCH`, `SYSTEM`. The default value is:

NIL

UNIF-SUBSUMED-TEST-ENABLED If NIL, recording events of type UNIF-SUBSUMED-TEST is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

UNIF-SUBSUMED-TRUE-ENABLED If NIL, recording events of type UNIF-SUBSUMED-TRUE is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

28.26 MS89 search procedure

MAX-SEARCH-LIMIT If integer-valued, is an upper limit on the TOTAL amount of time (in seconds) which can be spent on searching for a proof in any particular option. If null, then search time is unbounded. The flag is not affected by Garbage Collecting time whenever the value of the flag excluding-gc-time is T. Please read the help message for EXCLUDING-GC-TIME for more information. It takes values of type NULL-OR-POSINTEGER and belongs to subjects `TRANSMIT`, `MS93-1`, `MS91-7`, `MS91-6`, `MS90-9`, `MS89`, `IMPORTANT`, `MATING-SEARCH`. The default value is:

NIL

RANK-EPROOF-FN The name of a function which should take as its single argument an incomplete expansion proof, and return a nonnegative integer ranking the proof's likelihood of success, with 0 meaning no success (so don't try), and, otherwise, the better the likelihood, the lower the returned value. The only currently defined value for this flag is NUM-VPATHS-RANKING. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `MS93-1`, `MS90-9`, `MS89`, `MATING-SEARCH`. The default value is:

NUM–VPATHS–RANKING

SEARCH-TIME-LIMIT If integer-valued, is an upper limit on the CONTINUAL amount of time (in seconds) which can be spent on searching for a proof in any particular option. If null, then an ad hoc bound is used by the search procedure. The flag is not affected by Garbage Collecting time whenever the value of the flag excluding-gc-time is T. Please read the help message for EXCLUDING-GC-TIME for more information. It takes values of type NULL-OR-POSINTEGER and belongs to subjects `TRANSMIT`, `MS93-1`, `MS91-7`, `MS91-6`, `MS90-9`, `MS89`, `IMPORTANT`, `MATING-SEARCH`. The default value is:

NIL

28.27 MS90-3 search procedure

MAX-MATES Max number of mates for a literal. If the search attempts to add a mate that would exceed this limit, then this connection is not added. Copies of a literal created by path-focused duplication are regarded as the same when computing this number. Set MAX-MATES to INFINITY to allow an unlimited number of mates for any literal. It takes values of type POSINTEGER-OR-INFINITY and belongs to subjects `TRANSMIT`, `IMPORTANT`, `MS98-1`, `MS93-1`, `MS92-9`, `MS88`, `MS89`, `MS91-6`, `MS91-7`, `MS90-9`, `MS90-3`, `MATING-SEARCH`. The default value is:

2

MIN-QUANT-ETREE Only affects path-focused search procedures. When this flag is T, the scope of quantifiers is minimized in primsubs appearing in the expansion proof after searching is done and before the propositional proof checker starts. This allows the corresponding instantiation terms in the ND proof to be in non-prenex form, often giving more readable proofs. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS90-9`, `MS90-3`, `MATING-SEARCH`, `ETREES`. The default value is:

T

MS90-3-DUP-STRATEGY 1 to select any combination of duplications (2 1 3 1 is allowed), any thing else to select duplications in non decreasing order only. (2 1 3 1 is not allowed, but 1 1 2 3 is allowed.) It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS90-9`, `MS90-3`, `MS89`, `MATING-SEARCH`. The default value is:

1

NUM-FRPAIRS The match routine considers at most NUM-FRPAIRS frpairs, before selecting a frpair. However, if it finds a pair that has at most 1 substitution, it will automatically select this pair. Applies to UN90 only. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS90-9`, `MS90-3`, `UNIFICATION`. The default value is:

5

PRINT-MATING-COUNTER Prints the current mating after this many iterations in the top level ms90-3 search. Applicable only for path-focused duplication search procedures It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS91-7`, `MS90-9`, `MS90-3`, `MATING-SEARCH`. The default value is:

300000

SHOW-TIME When true, print the time taken by MS90-3 and MS90-9. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS90-9`, `MS90-3`, `MATING-SEARCH`. The default value is:

T

28.28 MS91-6 and MS91-7 search procedures

MS91-INTERLEAVE In MS91-*, primitive substitutions are generated by NAME-PRIM, and they are applied to the master eproof before the search mechanism chooses particular parts of that eproof (and hence particular substitutions) to try and prove.

If MS91-INTERLEAVE is NIL, all of the substitutions generated by NAME-PRIM are applied at once, and then the search mechanism chooses among them, probably in the order in which they were generated. The process of applying them to the eproof can take a very long time.

If MS91-INTERLEAVE is an integer n, we take n primsubs at a time for each variable which has primsubs, and apply only those to the eproof. Once we have searched through those (to be specific, once we decide to generate new options), we take the next n primsubs for each variable and apply them, and so on. This is much quicker, and has the advantage of not having to work through every primsub for the first variable before starting work on the next variable.

If MS91-INTERLEAVE is non-NIL, and NEW-OPTION-SET-LIMIT is greater than MS91-INTERLEAVE * (# of vars that have primsubs), then TPS will reduce NEW-OPTION-SET-LIMIT. This ensures that single substitutions are generated before multiple substitutions. It takes values of type NULL-OR-POSINTEGER and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS91-7`, `MS91-6`. The default value is:

5

MS91-PREFER-SMALLER When T, smaller option-sets will be preferred to any larger ones. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

T

MS91-TIME-BY-VPATHS When T, the amount of time given by SEARCH-TIME-LIMIT and MAX-SEARCH-LIMIT will be multiplied by the number of vertical paths through the vpform and then divided by the number of paths through the initial vpform (so the first vpform will get SEARCH-TIME-LIMIT seconds, and if the next has twice as many paths it will get twice as many seconds, and so on...). When NIL, every option set will get the same search time. This flag only applies in MS91 procedures. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

NIL

MS91-WEIGHT-LIMIT-RANGE New option-sets, when constructed, will be accepted if their weights lie in the range [current weight limit, current weight limit + MS91-WEIGHT-LIMIT-RANGE]. Hence increasing this value means that more option-sets will be acceptable during the creation stage. If this range is very small, there is a risk that no option sets at all will be accepted and the search will waste time recreating these sets with a higher current weight limit. If it is too large, then there is a risk that high-weighted sets will be considered before lower-weighted ones. Note: option sets of weight INFINITY will never be accepted, no matter what. It takes values of type INTEGER+-OR-INFINITY and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

1

NEW-OPTION-SET-LIMIT The maximum number of new option-sets that can be created at any one time. See MS91-INTERLEAVE. It takes values of type POSINTEGER and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

20

OPTIONS-GENERATE-ARG The argument used by the function given in the flag OPTIONS-GENERATE-FN. If this argument is INFINITY then new options will never be generated. See the help message for OPTIONS-GENERATE-FN. It takes values of type INTEGER+-OR-INFINITY and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

75

OPTIONS-GENERATE-FN This is the function for deciding when to add new options to the list from which option sets are generated. This is only called when new option sets are being generated, so if you are generating large numbers of options sets at a time then you might not see an effect until some time after your given criterion is satisfied. (Check the value of NEW-OPTION-SETS-LIMIT if this seems to be the case.) The argument for this function is in the flag OPTIONS-GENERATE-ARG, and the function to update that argument is in the flag OPTIONS-GENERATE-UPDATE. The options are: * ADD-OPTIONS-ORIGINAL generates new options when over OPTIONS-GENERATE-ARG percent of the possible option sets have been used, and each option appears in at least one option set. * ADD-OPTIONS-COUNT generates new options when more than OPTIONS-GENERATE-ARG different option sets have been tried. * ADD-OPTIONS-WEIGHT generates new options when the lower end of the acceptable weight bracket for a new option set exceeds OPTIONS-GENERATE-ARG. * ADD-OPTIONS-SUBS generates new options when the number of substitutions and duplications in the next option set (i.e. its SIMPLEST-WEIGHT-B) exceeds OPTIONS-GENERATE-ARG. If OPTIONS-GENERATE-ARG is INFINITY, no new options are ever generated. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

ADD−OPTIONS−ORIGINAL

OPTIONS-GENERATE-UPDATE The function used to update the value of the flag OPTIONS-GENERATE-ARG. Current possibilities are: * IDENT-ARG leaves the value unchanged. * DOUBLE-ARG doubles the value. * SQUARE-ARG squares the value. * INF-ARG makes the value INFINITY. Note that a value of INFINITY means that new options will never be generated. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

IDENT−ARG

OPTIONS-VERBOSE If T, will output extra information about the options being considered. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

NIL

PENALTY-FOR-EACH-PRIMSUB Used in computing weight-b in MS91 search procedures. Should be a nonnegative integer or the symbol INFINITY, and will be the amount of penalty given for using each primitive substitution. See WEIGHT-B-FN. It takes values of type INTEGER+-OR-INFINITY and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

3

PENALTY-FOR-MULTIPLE-PRIMSUBS Used in computing weight-b in MS91 search procedures. Should be a nonnegative integer or the symbol INFINITY, and will be the amount of penalty given for using more than one primitive substitution for a single variable. See WEIGHT-B-FN. It takes values of type INTEGER+-OR-INFINITY and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

5

PENALTY-FOR-MULTIPLE-SUBS Used in computing weight-b in MS91 search procedures. Should be a nonnegative integer or the symbol INFINITY, and will be the amount of penalty given for using more than one substitution for a single variable. See WEIGHT-B-FN. It takes values of type INTEGER+-OR-INFINITY and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

5

PENALTY-FOR-ORDINARY-DUP Used in computing weight-b in MS91 search procedures. Should be a nonnegative integer or the symbol INFINITY, and will be the amount of penalty given for each duplicate copy of a quantifier which is not used by a primitive substitution. See WEIGHT-B-FN. It takes values of type INTEGER+-OR-INFINITY and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

INFINITY

RECONSIDER-FN A function that should take a weight as argument and return a value to be used as a new weight after the associated option set runs out of time. Currently, the predefined functions are INF-WEIGHT SQUARE-WEIGHT, DOUBLE-WEIGHT and INCREMENT-WEIGHT (which, respectively, make reconsidering an old option set impossible, very unlikely, quite likely and probable). INCREMENT-WEIGHT actually adds 10 to the weight of an option set, as adding 1 is insignificant under most circumstances. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

DOUBLE–WEIGHT

WEIGHT-A-COEFFICIENT Coefficient to be used in multiplying weight-a of options in the option-set of which we are computing weight-d. See WEIGHT-A-FN. The total weight of a set of options is the weight-a of each option plus the weight-b of the set plus the weight-c of the set. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

0

WEIGHT-A-FN A function that should take an option as argument and return a value to be used as its weight-a. Currently, the only such predefined function is EXPANSION-LEVEL-WEIGHT-A, which returns the expansion level of the option to be used as a weight. The total weight of a set of options is the weight-a of each option plus the weight-b of the set plus the weight-c of the set. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

EXPANSION–LEVEL–WEIGHT–A

WEIGHT-B-COEFFICIENT Coefficient to be used in multiplying weight-b of option/option-subset pairs for the option-set of which we are computing weight-d. See WEIGHT-B-FN. The total weight of a set of options is the weight-a of each option plus the weight-b of the set plus the weight-c of the set. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

1

WEIGHT-B-FN A function that should take an option set and return a value to be used as its weight-b. Currently, the only such predefined functions are: * SIMPLE-WEIGHT-B-FN, which returns the sum of the penalties for the primsubs, multiple subs and duplications used in the option set (see the flags PENALTY-FOR-EACH-PRIMSUB, PENALTY-FOR-MULTIPLE-PRIMSUBS and PENALTY-FOR-MULTIPLE-SUBS for more information), * ALL-PENALTIES-FN which is much the same as SIMPLE-WEIGHT-B-FN but also adds a penalty for extra duplications given by the PENALTY-FOR-ORDINARY-DUP flag, and * SIMPLEST-WEIGHT-B-FN, which returns 1 for the original option set and adds 1 for each primsub or duplication (the idea is to set the coefficients of weight-a and weight-c to zero while using SIMPLEST-WEIGHT-B-FN). The total weight of a set of options is the weight-a of each option plus the weight-b of the set plus the weight-c of the set. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

SIMPLEST−WEIGHT−B−FN

WEIGHT-C-COEFFICIENT Coefficient to be used in multiplying weight-c of options in the option-set of which we are computing weight-d. See WEIGHT-C-FN. The total weight of a set of options is the weight-a of each option plus the weight-b of the set plus the weight-c of the set. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

0

WEIGHT-C-FN A function that should take an list of options as argument and return a value to be used as its weight-c. Currently, the only such predefined functions are OPTION-SET-NUM-VPATHS, which returns the number of vertical paths through the relevant etree, and OPTION-SET-NUM-LEAVES, which returns the number of leaves in the relevant etree. The total weight of a set of options is the weight-a of each option plus the weight-b of the set plus the weight-c of the set. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `MS91-7`, `MS91-6`. The default value is:

OPTION−SET−NUM−LEAVES

28.29 MS98-1 search procedure

BREAK-AT-QUANTIFIERS Applies only to quantifiers which cannot be duplicated later in the search. If T, then fragments will be broken so as not to contain any quantifiers; if NIL, fragments may contain quantifiers of the sort specified. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-1`. The default value is:

NIL

FF-DELAY If T, delay unifying f-f pairs for single connections, and unify them in context when some f-r pairs are added. If NIL, unify them as usual. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

NIL

HPATH-THRESHOLD If NIL, break on major conjunctions. If n, break at conjunctions and also on disjunctions having more than n hpaths. It takes values of type NULL-OR-POSINTEGER and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

1

MAXIMIZE-FIRST For each component which is being extended, do not create any new components which exceed MAX-MATES 1 until there are no other ways to extend the component. This only works for higher-order problems, and will be ignored in the first-order case. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

NIL

MEASUREMENTS A flag set by the system to give information about the complexity of the last problem worked on by TPS. Should be included in the value of RECORDFLAGS so that daterec will record the information.

Currently this records the number of vertical and horizontal paths, number of literals, and number of acceptable connections. It takes values of type SYMBOL-DATA-LIST and belongs to subjects `TRANSMIT`, `LIBRARY`. The default value is:

()

MS98-BASE-PRIM If T, we allow the search to begin with a fragment which is part of a primitive substitution. If NIL, we always choose a fragment which is outside the primitive substitutions (if possible). It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-1`. The default value is:

NIL

MS98-DUP-BELOW-PRIMSUBS When T, duplicate the quantifiers which occur below a primitive substitution NUM-OF-DUPS times. When NIL, don't. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

NIL

MS98-DUP-PRIMSUBS When T, MS98-DUP duplicates variables which have primsubs; when NIL, it doesn't. (Note that duplicating the variable will not duplicate the primsub; it will produce another copy of the unsubstituted-for tree below that expansion node.) It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

NIL

MS98-EXTERNAL-REWRITES When set to T, MS98-1 uses the currently active rewrite rules as global rewrites in addition to those it extracts from the formula. See Matt Bishop's thesis for details on rewriting in MS98-1. If MS98-REWRITES is set to NIL, this flag is irrelevant. It takes values of type BOOLEAN and belongs to subjects `MS98-1`, `MATING-SEARCH`. The default value is:

NIL

MS98-FIRST-FRAGMENT If non-NIL, this will move a single fragment to the beginning of the literal ordering, as follows: T : set of support strategy, more or less. The starting fragment will be the last non-duplicate fragment enumerated. This will be the rightmost part of the wff to be proven. n : (for integer n) the starting fragment will be whichever fragment contains LEAFn. If this leaf is part of a duplicate fragment, or does not exist at all, then this will behave like T.

NB: This flag overrides MS98-BASE-PRIM; the chosen fragment may always be part of a primitve substitution. See also MS98-FRAGMENT-ORDER. It takes values of type SYMBOL-OR-INTEGER and belongs to subjects `TRANSMIT`, `MS98-1`. The default value is:

NIL

MS98-FORCE-H-O If T, use higher-order unification graphs even for first-order searches. If NIL, use the normal first-order unification. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

NIL

MS98-FRAGMENT-ORDER The order in which the fragments are considered. This principally affects which fragment will become the starting point of the search, and which of the touched but not blocked fragments will be blocked next. See also MS98-FIRST-FRAGMENT. 0 : consider the number of ways to block the given fragment. 1 : consider the number of ways that the results for 0 might be extended (i.e. look ahead two steps in the search process) 2 : as for 1, but then weight in favour of ground fragments (i.e. those containing no variables). It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS98-1`. The default value is:

1

MS98-INIT Before doing ms98-1 search: If 0, do nothing at first; after each failure, duplicate one more quantifier. If 1, duplicate all outer quantifiers NUM-OF-DUPS times. If 2, apply primsubs and duplicate all outer quantifiers NUM-OF-DUPS times. If 3, cycle through primsubs one at a time, and duplicate all outer quantifiers NUM-OF-DUPS times. The time spent on each primsub will be at least MAX-SEARCH-LIMIT seconds, unless the search fails before then. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS98-1`. The default value is:

0

MS98-LOW-MEMORY If T, try to keep memory use low. This will probably make the search take longer. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

NIL

MS98-MAX-COMPONENTS If non-NIL, the maximum number of components that can be considered on any iteration of the MS98 search. It takes values of type NULL-OR-POSINTEGER and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

NIL

MS98-MAX-PRIMS The maximum number of primsubs allowed in any component. It takes values of type NULL-OR-POSINTEGER and belongs to subjects `TRANSMIT`, `MS98-1`. The default value is:

1

MS98-MEASURE Determines the measure which is used on components. If 0, count the components blocked and then weight heavily against the situation described by MS98-VALID-PAIR. If 1, the standard measure using the # of components blocked and touched If 2, as for 1 but also take account of the number of dups If 3, just count the number of components blocked If 4, as for 2 but also count the no of matings for the smallest component touched If 5, multiply the no of matings for the smallest touched by the number of subs. If 6, use the ratio of blocked to touched components and the ratio of the number of blocked components to the number of connections. If 7, prefer matings where positive leaves are mated to negative leaves and vice versa. If 8, use the ratio of blocked to touched components. If 9, favour large components satisfying max-mates 1. If 10, do as for 9 and then weight heavily against the situation described by MS98-VALID-PAIR. If 11, do as for 6 and then weight heavily against the situation described by MS98-VALID-PAIR. If 12, do as for 8 and then weight heavily against the situation described by MS98-VALID-PAIR. If 13, weight in favour of components with max-mates 1 and then weight heavily against the situation described by MS98-VALID-PAIR. If 14, do as for 7 and then weight heavily against the situation described by MS98-VALID-PAIR. If 15, take the average of 11 and 14. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS98-1`. The default value is:

0

MS98-MERGE-DAGS For higher-order searches only. Affects the way in which the unification graphs of elementary components are computed. 0 : Check that the graphs of the connections are pairwise compatible. Only compute the full graph of a component when necessary. 1 : Check that the graphs of the connections are compatible taken all together. (This can take a while for large sets of connections.) Only compute the full graph when necessary. 2 : Always compute the full graph. This overrides FF-DELAY. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

0

MS98-MINIMALITY-CHECK If T, check each new component for minimality and reject those which are non-minimal. If NIL, don't bother. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

NIL

MS98-NUM-OF-DUPS If NIL, we can use every duplication that's present. If some positive integer n, we reject any component using more than n of the duplications. It takes values of type NULL-OR-POSINTEGER and belongs to subjects `TRANSMIT`, `MS98-1`. The default value is:

NIL

MS98-POLLUTE-GLOBAL-REWRITES When set to T, rewrites generated by MS98-1 are not removed from the list of global rewrite rules after the search is complete. See Matt Bishop's thesis for details on rewriting in MS98-1. If MS98-REWRITES is set to NIL, this flag is irrelevant. It takes values of type BOOLEAN and belongs to subjects `MS98-1`, `MATING-SEARCH`. The default value is:

NIL

MS98-PRIMSUB-COUNT The maximum number of primsubs to be applied each set variable in the expansion tree. It takes values of type NULL-OR-POSINTEGER and belongs to subjects `TRANSMIT`, `MS98-1`. The default value is:

3

MS98-REW-PRIMSUBS When T, MS98-DUP does primsubs for Leibniz variables which have become rewrites; when NIL, it doesn't. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

NIL

MS98-REWRITE-DEPTH When attempting to rewrite one term into another, the maximum number of steps of rewriting that are allowed. It takes values of type POSINTEGER and belongs to subjects `TRANSMIT`, `MS98-1`. The default value is:

2

MS98-REWRITE-MODEL If T, ask the user for a model of the rewrite rules to help slim down the unification tree. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

NIL

MS98-REWRITE-PRUNE If T, delete any unifiers which are duplicates modulo rewriting (this can be slow). If NIL, don't. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

T

MS98-REWRITE-SIZE The maximum size of a (lambda-normalized) term that can be produced by rewriting, measured as the number of nodes in the parse tree of that term. NIL means that there is no maximum. It takes values of type NULL-OR-POSINTEGER and belongs to subjects `TRANSMIT`, `MS98-1`. The default value is:

NIL

MS98-REWRITE-UNIF When a rewrite rule can introduce a new variable, this flag governs the size of the allowed substitutions for that variable. Essentially, this is a special case of MAX-SUBSTS-VAR. It takes values of type NULL-OR-POSINTEGER and belongs to subjects `TRANSMIT`, `MS98-1`. The default value is:

NIL

MS98-REWRITES When T, make all of the global equalities into rewrites. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-1`. The default value is:

NIL

MS98-TRACE Given a mating in advance, this is used to trace the progress of MS98-1 search for a mating. This is a list of symbols which indicate what to trace. The possible symbols are:

1. MATING - Search as usual, keeping a record of when good connections and components are formed. The value of *ms98-trace-file* is a string giving the name of a file into which this information is stored. 2. MATING-FILTER - The search is filtered to only consider good connections and components. This is useful for a quick check if the search can possibly succeed. Typically, when MATING-FILTER is on the list, then so is MATING.

If the list is nonempty at all, then the trace is considered 'on'. The consequence of this is that duplications and primsubs are skipped at the beginning of search, and that the output of the trace will be sent to the file indicated by the global variable *ms98-trace-file*. It takes values of type SYMBOLLIST and belongs to subjects `MS98-MINOR`. The default value is:

()

MS98-UNIF-HACK If T, do not introduce new constants during unification. (NOTE: This is a hack; we *do* need to introduce new constants, in general, but in most cases we needn't bother.) It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

NIL

MS98-UNIF-HACK2 If T, during the generation of unifiers, prevent the occurrence of subformulas of type o which contain no variables (except for TRUTH and FALSEHOOD, if they are allowed by MS98-UNIF-HACK). If NIL, allow these to be generated. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

NIL

MS98-USE-COLORS It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-1`. The default value is:

T

MS98-VALID-PAIR Given two disjuncts X OR Y and A OR B, this flag determines when we are allowed to make a component containing connections X-A and Y-B (assuming they're unifiable connections). The higher the number, the more stringent (and more time-consuming) the test; any correct mating is guaranteed to pass any of these tests: 1: MAX-MATES is not 1. 2: As for 1, plus we require an extra mate for each of X,Y,A and B. 3: As for 2, plus we require that all of these new mates be pairwise compatible with each other. 4: As for 3, plus we require that all of these new mates be simultaneously compatible with each other.

3 and 4 are only applicable to higher-order searches.

There is an extra value, 0, which rejects any such connections even if max-mates is not 1. This results in an incomplete search, but is often acceptable. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

1

MS98-VARIABLE-ORDER Determines the variable ordering for the unification graph. Only affects higher-order searches. Suppose N is the maximum number of unifiers for a given list of variables, and K is the length of the list. For values 0–3, the variables are first grouped into lists of duplicate copies (so each variable is listed with its duplicates, if any) 0

: Sort by N, largest first. 1 : Sort by N, smallest first. 2 : Sort by K, largest first. 3 : Sort by K, smallest first. 10–13 : Group the variables into lists of length 1, and then proceed as 0–3. 20–23 : Group the variables into lists that occur together (i.e. two variables go into the same list if their expansion nodes are not separated by any junctive node in the etree) and then proceed as for 0–3. 30–33 : Group the variables as for 0–3, and then reduce the lists to length 1 while keeping the variables in the same order. 40–43 : Group the variables as for 20–23, and then reduce the lists to length 1 while keeping the variables in the same order. Other values X will behave like (X div 10) for variable grouping and (X mod 10) for ordering the groups. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS98-MINOR`. The default value is:

1

MS98-VERBOSE If T, print extra information during MS98-1 search. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-1`. The default value is:

NIL

28.30 Extensional Search

EXT-SEARCH-LIMIT If EXT-SEARCH-LIMIT is an integer which will place a limit on the extensional search procedure MS03-7. Given such a limit, search is incomplete and guaranteed to eventually terminate. If EXT-SEARCH-LIMIT is set to infinity, then the search may not terminate. It takes values of type INTEGER+-OR-INFINITY and belongs to subjects `TRANSMIT`, `MS03-7`, `EXT-SEARCH`. The default value is:

INFINITY

MS03-DUP-METHOD The method by which different duplication options are considered by the MS03-7 search procedure.

1. Simply add the oldest expansion arc that has not been considered yet (and any arcs related to it) each time a new option is tried. This will lead to extremely large jforms in most cases.

2. Works like 1 except with respect to expansion arcs that either contain a nontrivial set substitution (ie, one with logical connectives) or are associated with a set existence lemma. With respect to these 'set expansion arcs', we remove whatever such arcs are in the current option and replace them with a new set expansion arc (thus considering a new set expansion option). If every single set expansion option has been considered, we begin considering two at a time, and so on.

3. Works like 2 except we treat every expansion of set type as a set expansion arc instead of just the ones with nontrivial set substitutions.

See Also: MS03-WEIGHT-CHANGE-DUPS, MAX-SEARCH-LIMIT It takes values of type POSNUMBER and belongs to subjects `TRANSMIT`, `MS03-7`, `EXT-SEARCH`. The default value is:

1

MS03-QUICK-EUNIFICATION-LIMIT This provides a bound on how much E-unification MS03-7 and MS04-2 attempt to do before deciding what to mate. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

50

MS03-SOLVE-RIGID-PARTS If T, MS03-7 tries to find quick solutions to the rigid parts of a problem. This only applies when MS03-USE-JFORMS is T. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS03-7`, `EXT-SEARCH`. The default value is:

T

MS03-SOLVE-RIGID-PARTS-ALLOW-RECONNECTS When trying to solve the rigid part of a jform, we might consider connecting two literals that are already connected. Sometimes this speeds up the search, presumably by keeping us from looking at possible connections beneath connections (needed to show equivalences). It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS03-7`, `EXT-SEARCH`. The default value is:

T

MS03-USE-JFORMS If T, MS03-7 uses (dissolved) jforms during search. Constructing and dissolving jforms can be time consuming, but in principle can restrict the branching of search. If NIL, jforms are not used, which may result in the consideration of connections which only span paths already spanned by other connections. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS03-7`, `EXT-SEARCH`. The default value is:

T

MS03-USE-SET-CONSTRAINTS If this flag and MS03-USE-JFORMS are T, MS03-7 uses set constraints in addition to primsubs to determine potential set substitutions. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS03-7`, `EXT-SEARCH`. The default value is:

NIL

MS03-VERBOSE If T, print extra information during MS03-7 search. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS03-7`, `EXT-SEARCH`. The default value is:

NIL

MS03-WEIGHT-BANNED-SELS Controls the penalty for trying to unify two terms that require getting around using a banned selected variable (using duplication or equational reasoning). It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

300

MS03-WEIGHT-CHANGE-DUPS If MAX-SEARCH-LIMIT is NIL, then MS03-WEIGHT-CHANGE-DUPS controls how often MS03-7 changes which expansion terms are considered.

SEE ALSO: MS03-DUP-METHOD It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS03-7`, `EXT-SEARCH`. The default value is:

100

MS03-WEIGHT-DISJ-EUNIF When attempting to E-unify two literals a and b, this weight is multiplied by disjdepth(a) * disjdepth(b) where disjdepth of a literal is the number of disjunctions above the literal on the jform. The effect of this is to prefer mating nodes that are closer to being 'global'.

If MS03-USE-JFORMS is set to NIL, the disjdepth of a node is measured by the number of disjunctive nodes above the node in the edag. This measure is less precise, since dissolution isn't used.

See Also: MS03-WEIGHT-DISJ-MATE, MS03-WEIGHT-DISJ-UNIF It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS03-7`, `EXT-SEARCH`. The default value is:

10

MS03-WEIGHT-DISJ-MATE When attempting to mate two literals a and b, this weight is multiplied by disjdepth(a) * disjdepth(b) where disjdepth of a literal is the number of disjunctions above the literal on the jform. The effect of this is to prefer mating nodes that are closer to being 'global'.

If MS03-USE-JFORMS is set to NIL, the disjdepth of a node is measured by the number of disjunctive nodes above the node in the edag. This measure is less precise, since dissolution isn't used.

See Also: MS03-WEIGHT-DISJ-EUNIF, MS03-WEIGHT-DISJ-UNIF It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS03-7`, `EXT-SEARCH`. The default value is:

10

MS03-WEIGHT-DISJ-UNIF When performing a unification (imitation or projection) step on a negative equation literal, this value is multiplied by the disjdepth of the literal. The disjdepth is the number of disjunctions above the literal in the jform.

If MS03-USE-JFORMS is set to NIL, the disjdepth of the negative equation node is measured by the number of disjunctive nodes above the node in the edag.

See Also: MS03-WEIGHT-DISJ-MATE, MS03-WEIGHT-DISJ-EUNIF It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS03-7`, `EXT-SEARCH`. The default value is:

10

MS03-WEIGHT-DUP-VAR Controls how often MS03-7 tries to duplicate an expansion variable in order to substitute a banned selected variable for the new expansion variable. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS03-7`, `EXT-SEARCH`. The default value is:

300

MS03-WEIGHT-EUNIF1 This value is added to the weight for adding any eunif1 (E-unification without symmetry) between two equation literals. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

1

MS03-WEIGHT-EUNIF2 This value is added to the weight for adding any eunif2 (E-unification with symmetry) between two equation literals. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

1

MS03-WEIGHT-FLEXFLEXDIFF Controls the penalty for trying to unify two terms that require unifying two flexible terms of a base type other than O with different heads. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

3

MS03-WEIGHT-FLEXFLEXDIFF-O Controls the penalty for trying to unify two terms that require unifying two flexible terms of type O with different heads. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

10

MS03-WEIGHT-FLEXFLEXSAME Controls the penalty for trying to unify two terms that require unifying two flexible terms of a base type other than O with the same head. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

5

MS03-WEIGHT-FLEXFLEXSAME-O Controls the penalty for trying to unify two terms that require unifying two flexible terms of type O with the same head. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

20

MS03-WEIGHT-FLEXRIGID-BRANCH Controls the penalty for trying to unify two terms that require solving a branching (higher-order) flex-rigid disagreement pair of a base type other than O. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

6

MS03-WEIGHT-FLEXRIGID-EQN Controls the penalty for trying to unify two terms that require a solving a flex-rigid pair of a base type other than O when no imitation and no projection is appropriate and there is an an equation which is between a pair of rigid terms sharing a head with the disagreement pair. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

100

MS03-WEIGHT-FLEXRIGID-FLEXEQN Controls the penalty for trying to unify two terms that require a solving a flex-rigid pair of a base type other than O when no imitation and no projection is appropriate and there is a flex-rigid equation between terms of the same base type. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

100

MS03-WEIGHT-FLEXRIGID-MATE This value is added to the weight for adding any connection between any rigid literal and flexible literal. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

1

MS03-WEIGHT-FLEXRIGID-NOEQN Controls the penalty for trying to unify two terms that require a solving a flex-rigid pair of a base type other than O when no imitation and no projection is appropriate and there are no flex-rigid equations between terms of the same base type. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

500

MS03-WEIGHT-FLEXRIGID-O Controls the penalty for trying to unify two terms that require solving a branching (higher-order) flex-rigid disagreement pair of type O. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

20

MS03-WEIGHT-IMITATE This value is added to the weight for any imitation unification steps. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

1

MS03-WEIGHT-OCCURS-CHECK Controls the penalty for trying to unify two terms that require getting around an occurs check (using equational reasoning). It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

150

MS03-WEIGHT-PRIMSUB-FALSEHOOD Controls how often MS03-7 tries a primsub using FORALL It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS03-7`, `EXT-SEARCH`. The default value is:

50

MS03-WEIGHT-PRIMSUB-FIRST-AND Controls when MS03-7 or MS04-2 first tries a primsub using AND. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

200

MS03-WEIGHT-PRIMSUB-FIRST-EQUALS Controls when MS03-7 or MS04-2 first tries a primsub using equality at a base type. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

200

MS03-WEIGHT-PRIMSUB-FIRST-EXISTS Controls when MS03-7 or MS04-2 first tries a primsub using EXISTS. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

200

MS03-WEIGHT-PRIMSUB-FIRST-FORALL Controls when MS03-7 or MS04-2 first tries a primsub using FORALL. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

200

MS03-WEIGHT-PRIMSUB-FIRST-NOT-EQUALS Controls when MS03-7 or MS04-2 first tries a primsub using negation and equality at a base type. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

200

MS03-WEIGHT-PRIMSUB-FIRST-NOT-PROJ Controls when MS03-7 or MS04-2 first tries a primsub using negation and a projection. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

500

MS03-WEIGHT-PRIMSUB-FIRST-OR Controls when MS03-7 or MS04-2 first tries a primsub using OR. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

200

MS03-WEIGHT-PRIMSUB-FIRST-PROJ Controls when MS03-7 or MS04-2 first tries a primsub using a projection. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

500

MS03-WEIGHT-PRIMSUB-NEXT-AND Controls how often MS03-7 or MS04-2 tries a primsub using AND after the first time. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

200

MS03-WEIGHT-PRIMSUB-NEXT-EQUALS Controls how often MS03-7 or MS04-2 tries a primsub using equality at a base type after the first time. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

200

MS03-WEIGHT-PRIMSUB-NEXT-EXISTS Controls how often MS03-7 or MS04-2 tries a primsub using EXISTS at various types after the first time. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

200

MS03-WEIGHT-PRIMSUB-NEXT-FORALL Controls how often MS03-7 or MS04-2 tries a primsub using FORALL at various types after the first time. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

200

MS03-WEIGHT-PRIMSUB-NEXT-NOT-EQUALS Controls how often MS03-7 or MS04-2 tries a primsub using negation and equality at a base type after the first time. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

200

MS03-WEIGHT-PRIMSUB-NEXT-NOT-PROJ Controls how often MS03-7 or MS04-2 tries a primsub using negation and a projection after the first time. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

500

MS03-WEIGHT-PRIMSUB-NEXT-OR Controls how often MS03-7 or MS04-2 tries a primsub using OR after the first time. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

200

MS03-WEIGHT-PRIMSUB-NEXT-PROJ Controls how often MS03-7 or MS04-2 tries a primsub using a projection after the first time. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

500

MS03-WEIGHT-PRIMSUB-TRUTH Controls how often MS03-7 tries a primsub using TRUTH It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS03-7`, `EXT-SEARCH`. The default value is:

50

MS03-WEIGHT-PROJECT This value is added to the weight for any projection unification steps. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

1

MS03-WEIGHT-RIGID-MATE This value is added to the weight for adding any connection between two rigid literals. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

1

MS03-WEIGHT-RIGIDRIGID-EQN Controls the penalty for trying to unify two terms that require a solving a rigid-rigid pair of a base type other than O in the presence of an equation which is between a pair of rigid terms sharing a head with the disagreement pair. Some form of equational reasoning is required to solve these cases. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

50

MS03-WEIGHT-RIGIDRIGID-FLEXEQN Controls the penalty for trying to unify two terms that require a solving a rigid-rigid pair of a base type other than O in the presence of an equation which is between a rigid and a flexible term. Some form of equational reasoning is required to solve these cases. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

60

MS03-WEIGHT-RIGIDRIGID-NOEQN Controls the penalty for trying to unify two terms that require a solving a rigid-rigid pair of a base type other than O in the absence of any equations of the same base type. Some form of equational reasoning is required to solve these cases, but we may need to mate two nodes before an appropriate equation has appeared in the search. Such a case is unusual so it makes sense for this flag to be set to a high value. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

500

MS03-WEIGHT-RIGIDRIGIDDIFF-O Controls the penalty for trying to unify two terms that require a solving a rigid-rigid pair of type O with the different heads. Extensionality is required to solve these cases. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

40

MS03-WEIGHT-RIGIDRIGIDSAME-O Controls the penalty for trying to unify two terms that require a solving a rigid-rigid pair of type O with the same head. Extensionality is required to solve these cases. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `MS03-7`, `EXT-SEARCH`. The default value is:

15

MS04-ALLOW-FLEX-EUNIFS If MS04-ALLOW-FLEX-EUNIFS is T, then MS04-2 will try to mate flexible nodes with positive equation nodes and negative equation goal nodes. To do this, MS04-2 will imitate the equality (or negation of equality) first. This is not necessary for completeness (since an equality primsub will eventually be considered), but is sometimes helpful. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

T

MS04-ALLOW-FLEXRIGID-PROJ-MATE If MS04-ALLOW-FLEXRIGID-PROJ-MATE is T, then MS04-2 will try to mate flexible nodes with atoms using a projection. This is not necessary for completeness (since a projection primsub will eventually be considered), but is sometimes helpful. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

T

MS04-BACKTRACK-METHOD Determines which choices are used for backtracking.

1. Backtrack on all choices.

2. Do not backtrack over connections.

3. Do not backtrack over connections or duplications. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

1

MS04-CHECK-UNIF-DEPTH If MS04-DELAY-UNIF-CONSTRAINTS is T, MS04-CHECK-UNIF-DEPTH determines how deeply MS04-2 will try to unify in order to prune out states where the unification problem is unsolvable.

See Also: MS04-DELAY-UNIF-CONSTRAINTS It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

3

MS04-DELAY-FLEXRIGID-MATES If MS04-DELAY-UNIF-CONSTRAINTS is T and MS04-DELAY-FLEXRIGID-MATES is T, then potential connections between flexible nodes and atomic nodes are delayed and the dpair is added to the unification problem. In particular, this may allow projections to be used to unify flexible nodes with atomic nodes. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

T

MS04-DELAY-UNIF-CONSTRAINTS If set to T, the MS04-2 search procedure will delay considering vertical paths that contain certain equation goals which are being used to weight further options. The procedure is complete with this set to T or NIL. Setting it to T creates more nondeterminism, but can lead to faster proofs. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

T

MS04-DUP-EARLY If set to T, MS04-2 will only duplicate expansion nodes before making any substitutions or connections (on paths that share the expansion node). Originally, MS04-2 always did this, but only MS04-2 with duplications allowed anytime (when the value of MS04-DUP-EARLY is NIL) is shown complete in Chad E. Brown's thesis. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

NIL

MS04-DUP-WEIGHT Sets the weight for duplicating an expansion node in MS04-2. This controls how often MS04-2 will duplicate expansion nodes. The higher the weight, the less often duplication occurs. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

300

MS04-EAGER-UNIF-SUBST If set to T (and MS04-DELAY-UNIF-CONSTRAINTS is T), the MS04-2 search procedure will substitute for parts of the pattern part of the current unification problem. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

T

MS04-INCR-DEPTH Every time MS04-2 has completed the search space up to a given bound, the bound is increased by MS04-INCR-DEPTH.

SEE ALSO: MS04-INITIAL-DEPTH, MS04-MAX-DEPTH It takes values of type POSINTEGER and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

100

MS04-INITIAL-DEPTH This sets the initial bound for the depth of the search procedure MS04-2. Once the search to this depth has failed, MS04-INCR-DEPTH is used to increase the bound.

SEE ALSO: MS04-INCR-DEPTH, MS04-MAX-DEPTH It takes values of type POSINTEGER and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

100

MS04-MAX-DELAYED-CONNS The maximum number of delayed connections (waiting to be unified) MS04-2 will consider (on the first iteration of search). It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

10

MS04-MAX-DEPTH This sets an absolute maximum on the depth of the search. For completeness, this should be set to infinity.

SEE ALSO: MS04-INITIAL-DEPTH, MS04-INCR-DEPTH It takes values of type INTEGER+-OR-INFINITY and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

INFINITY

MS04-MAX-DUPS The maximum number of duplications MS04-2 will consider (on the first iteration of search). It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

3

MS04-MAX-EUNIF1S The maximum number of E-unification connections MS04-2 will consider (on the first iteration of search). It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

3

MS04-MAX-EUNIF2S The maximum number of symmetric E-unification connections MS04-2 will consider (on the first iteration of search). It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

3

MS04-MAX-FLEX-EUNIFS The maximum number of times MS04-2 will instantiate the head of a flexible node with an equality of base type (or the negation of an equality) in order to E-unify the instantiated node with a positive equation node or an equation goal node. This flag is only relevant if MS04-ALLOW-FLEX-EUNIFS is set to T. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

2

MS04-MAX-FLEXRIGID-MATES The maximum number of mates between a flexible node and a rigid atom of opposite polarity MS04-2 will consider (by imitating the head of the rigid atom). This value is increased by 1 after each failed iteration of the search. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

5

MS04-MAX-FLEXRIGID-NEG-MATES The maximum number of mates between a flexible node and a rigid atom of the same polarity MS04-2 will consider (by using a negation and imitating the head of the rigid atom). This value is increased by 1 after each failed iteration of the search. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

5

MS04-MAX-FLEXRIGID-NEG-PROJ-MATES The maximum number of mates between a flexible node and a rigid atom of the same polarity MS04-2 will consider using projections with a negation instead of imitations. This flag is only relevant if MS04-ALLOW-FLEXRIGID-PROJ-MATE is T. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

5

MS04-MAX-FLEXRIGID-PROJ-MATES The maximum number of mates between a flexible node and a rigid atom of opposite polarity MS04-2 will consider using projections instead of imitations. This flag is only relevant if MS04-ALLOW-FLEXRIGID-PROJ-MATE is T. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

5

MS04-MAX-IMITS The maximum number of imitations (for unification) MS04-2 will attempt during an iteration of the search. The value is increased by 1 after every failed iteration. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

10

MS04-MAX-PRIMSUB-AND The maximum number of conjunction primsubs MS04-2 will attempt during an iteration of the search. Conjunction primsubs are only tried if MS04-PRENEX-PRIMSUBS is T. The value is increased by 1 after every failed iteration. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

1

MS04-MAX-PRIMSUB-EQUALS The maximum number of primsubs using equality (at base type) MS04-2 will attempt during an iteration of the search. The value is increased by 1 after every failed iteration. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

1

MS04-MAX-PRIMSUB-EXISTS The maximum number of EXISTS primsubs MS04-2 will attempt during an iteration of the search. Conjunction primsubs are only tried if MS04-PRENEX-PRIMSUBS is T. The value is increased by 1 after every failed iteration. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

1

MS04-MAX-PRIMSUB-FORALL The maximum number of FORALL primsubs MS04-2 will attempt during an iteration of the search. The value is increased by 1 after every failed iteration. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

1

MS04-MAX-PRIMSUB-NOT The maximum number of negation primsubs MS04-2 will attempt during an iteration of the search. Negation primsubs are only tried if MS04-PRENEX-PRIMSUBS is NIL. The value is increased by 1 after every failed iteration. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

1

MS04-MAX-PRIMSUB-NOT-EQUALS The maximum number of primsubs using negated equality (at base type) MS04-2 will attempt during an iteration of the search. Negated equality primsubs are only tried if MS04-PRENEX-PRIMSUBS is T. The value is increased by 1 after every failed iteration. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

1

MS04-MAX-PRIMSUB-NOT-PROJ The maximum number of negated projection primsubs MS04-2 will attempt during an iteration of the search. Negated projection primsubs are only tried if MS04-PRENEX-PRIMSUBS is T. The value is increased by 1 after every failed iteration. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

1

MS04-MAX-PRIMSUB-OR The maximum number of disjunction primsubs MS04-2 will attempt during an iteration of the search. The value is increased by 1 after every failed iteration. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

1

MS04-MAX-PRIMSUB-PROJ The maximum number of projection primsubs MS04-2 will attempt during an iteration of the search. The value is increased by 1 after every failed iteration. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

1

MS04-MAX-PROJS The maximum number of projections (for unification) MS04-2 will attempt during an iteration of the search. The value is increased by 1 after every failed iteration. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

10

MS04-MAX-RIGID-MATES The maximum number of mates between nodes which are already rigid MS04-2 will consider (on the first iteration of search). It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

10

MS04-MP-OPTIONS In Allegro, any MS04-2 option listed in the value of this flag will cause TPS to use multiprocessing to consider the option in parallel to consideration of other options.

The main MS04-2 options which may be included on the list are DUP, PRIMSUB and ADD-SET-CONSTRAINT. Other MS04-2 options which may be included are MATE, EUNIF1, EUNIF2, SUBST, MATE-FLEXRIGID, MATE-FLEXRIGID-NEG, MATE-FLEXRIGID-PROJ, MATE-FLEXRIGID-NEG-PROJ, FLEX-EUNIF, PRIMSUB-QUANT-GENTP, DELAY-UNIF, DELAY-CONN and SOLVE-SET-CONSTRAINTS. It takes values of type SYMBOLLIST and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

()

MS04-PRENEX-PRIMSUBS If T, only primsubs in conjunctive-prenex normal forms will be generated. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `EXT-SEARCH`. The default value is:

T

MS04-SEMANTIC-PRUNING If set to T, the MS04-2 search procedure will try to prune search states using semantics.

See Also: MODELS, MAX-DOMAIN-SIZE, MAX-BINDER-COMPUTATION It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

NIL

MS04-SOLVE-UNIF-DEPTH If MS04-DELAY-UNIF-CONSTRAINTS is T, MS04-SOLVE-UNIF-DEPTH determines how deeply MS04-2 will try to solve unification constraints after every vertical path can be solved by the delayed unification constraints.

See Also: MS04-DELAY-UNIF-CONSTRAINTS It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

5

MS04-TRACE If T, MS04-2 will gather information about the search which will be used to suggest values for flag settings (if search is successful). It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

NIL

MS04-USE-SEMANTICS If set to T, the MS04-2 search procedure will use semantics to guide the search.

See Also: MODELS, MAX-DOMAIN-SIZE, MAX-BINDER-COMPUTATION It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

NIL

MS04-USE-SET-CONSTRAINTS If set to T, the MS04-2 search procedure will use set constraints and set existence lemmas to solve for set variables. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

NIL

MS04-VERBOSE Determines level of verbosity of MS04-2 search. Value should be MIN, MED or MAX. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

MED

MS04-WEIGHT-ADD-SET-CONSTRAINT If MS04-USE-SET-CONSTRAINTS is T, this weight is used to determine when to add another constraint for a set variable.

See Also: MS04-USE-SET-CONSTRAINTS, MAX-NUM-CONSTRAINTS, MAX-CONSTRAINT-SIZE It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

1

MS04-WEIGHT-DELAY-UNIF If MS04-DELAY-UNIF-CONSTRAINTS is T, this weight is used to determine when to add an equation goal node to the collection of delayed unification constraints.

See Also: MS04-DELAY-UNIF-CONSTRAINTS It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

0

MS04-WEIGHT-EUNIF-DECS Controls how often EUnification is applied to equation goals that are decomposable, i.e., have shallow formula of the form:

[H . . .] = [H . . .]

There are cases where one needs to do this, but often one wants to avoid it. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

1000

MS04-WEIGHT-EUNIF-DIFF-HEADS An extra weight on EUNIF1 steps of the form [A = B]^+ to [C = D]^- where the heads of A and C are different and the heads of B and D are different. The weight is also added to EUNIF2 steps when the heads A and D are different and the heads of B and C are different. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

2000

MS04-WEIGHT-FLEX-EUNIF This value is added to the weight for adding any connection between any flexible literal and an equation. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `EXT-SEARCH`. The default value is:

2

MS04-WEIGHT-FLEXRIGID-PROJ-MATE This value is added to the weight for adding any connection between a flexible literal and an atom using a projection on the head of the flexible literal. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`, `MS04-2`, `EXT-SEARCH`. The default value is:

2

MS04-WEIGHT-MULTIPLE-EUNIF1S This controls the extra weight every time a node is eunified more than once. This is similar to MAX-MATES. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

10

MS04-WEIGHT-MULTIPLE-EUNIF2S This controls the extra weight every time a node is symmetrically eunified more than once. This is similar to MAX-MATES. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

10

MS04-WEIGHT-MULTIPLE-MATES This controls the extra weight every time a node is mated more than once. This is similar to MAX-MATES. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

10

MS04-WEIGHT-PRIMSUB-FIRST-NOT Controls when MS04-2 first tries a primsub using a negation.

This is only used when MS04-PRENEX-PRIMSUBS is NIL. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `EXT-SEARCH`. The default value is:

1000

MS04-WEIGHT-PRIMSUB-NEXT-NOT Controls when MS04-2 tries a primsub using a negation after the first time.

This is only used when MS04-PRENEX-PRIMSUBS is NIL. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `EXT-SEARCH`. The default value is:

1000

MS04-WEIGHT-PRIMSUB-NEXTTP Sets the weight for each higher type we generate for a primsub using either FORALL or EXISTS. This controls how often MS04-2 will use primsubs with higher types. The higher the weight, the less often higher types are used. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `EXT-SEARCH`. The default value is:

100

MS04-WEIGHT-PRIMSUB-OCCURS-CONST Some logical constants occur embedded in the terms of a theorem. This flag controls when MS04-2 tries a primsub using one of these logical constants if the logical constant will not be tried by other primsubs. This is only used if MS04-PRENEX-PRIMSUBS is NIL.

See Also: MS04-PRENEX-PRIMSUBS It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `MS04-2`, `EXT-SEARCH`. The default value is:

1000

MS04-WEIGHT-SOLVE-SET-CONSTRAINTS If MS04-USE-SET-CONSTRAINTS is T, this weight is used to determine when to stop adding constraints for a set variable.

See Also: MS04-USE-SET-CONSTRAINTS, MAX-NUM-CONSTRAINTS, MAX-CONSTRAINT-SIZE It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS04-2`, `EXT-SEARCH`. The default value is:

1

28.31 Proof Translation

ETREE-NAT-VERBOSE Should be a list of print-functions (see the help message for PRINT-FUNCTION), which will be executed after each tactic during ETREE-NAT. It takes values of type PRINT-FUNCTION-LIST and belongs to subjects `TRANSMIT`, `WINDOW-PROPS`, `PRINTING`, `ETR-NAT`. The default value is:

(PRFW–PALL PRFW–^P PRFW–^PN ^PN)

MATINGSTREE-NAME Prefix for labels associated with nodes in a matingstree. It takes values of type SYMBOL and belongs to subjects `ETREES`, `MTREE-TOP`. The default value is:

MSTREE

MERGE-MINIMIZE-MATING If T, merging will attempt to minimize the mating by removing any unnecessary connections. If NIL, it won't. T will sometimes produce a more readable ND proof, but can also take a very long time. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS98-1`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `ETR-NAT`, `ETREES`, `MATING-SEARCH`. The default value is:

T

NAT-ETREE-VERSION Determines which version of NAT-ETREE to use: OLD – The original version. HX – Hongwei Xi's version which is intended to work on any natural deduction proof, normal or not. This version has problems, but might work. CEB – Which is intended to only work on normal proofs, and should in principle always work on normal proofs. It takes values of type NAT-ETREE-VERSION-TYPE and belongs to subjects `ETR-NAT`. The default value is:

CEB

NATREE-DEBUG To invoke the debugging facilities mentioned in the Programmers Guide associated with NAT-ETREE. If NATREE-VERSION is set to CEB and NATREE-DEBUG is set to T, then the code doublechecks that a mating exists, giving the user lots of information. This should eventually evolve into a flag with more choices. It takes values of type BOOLEAN and belongs to subjects MS93-1, MS92-9, MS88, MS89, MS91-6, MS91-7, MS90-9, MS90-3, MATING-SEARCH. The default value is:

NIL

REMOVE-LEIBNIZ If TRUE, selection parameters corresponding to Leibniz equality definitions will be removed from expansion proofs during merging (cf. Pfenning's thesis, theorem 138). It takes values of type BOOLEAN and belongs to subjects TRANSMIT, MS93-1, MS92-9, MS91-7, MS91-6, MS90-9, MS90-3, MS89, MS88, ETR-NAT, ETREES, MATING-SEARCH. The default value is:

T

RENUMBER-LEAVES If this flag is T, copies of leafN will be numbered leafN.1, leafN.2, etc. If the flag is NIL, they will be given the next available number, as determined by an internal counter. It takes values of type BOOLEAN and belongs to subjects JFORMS. The default value is:

T

28.32 Unification

APPLY-MATCH Heuristic to decide the pair that should be given to match. UN88 procedures: APPLY-MATCH-ALL-FRDPAIRS applies match to all flexible-rigid pairs and chooses whichever will have fewest substitutions. APPLY-MATCH-ALL-FRDPAIRS-MSV does the same, but also checks for MAX-SUBSTS-VAR violations at the same time. APPLY-MATCH-MAX-SUBSTS applies match to whichever flexible-rigid pair is closest to exceeding the bound in MAX-SUBSTS-VAR. If it finds one with a unique substitution, it uses that. APPLY-MATCH-MIN-SUBSTS is like the above, but chooses the pair which is farthest from the MAX-SUBSTS-VAR bound. APPLY-MATCH-MOST-CONSTS applies match to whichever flex-rigid pair contains the most constant symbols. (The last two of these are all but useless; both of the SUBSTS versions will be disastrous if MAX-SUBSTS-VAR is NIL...)

UN90 procedures: This flag is almost always ignored (the default behaviour is much like APPLY-MATCH-ALL-FRDPAIRS, but see NUM-FRPAIRS and COUNTSUBS-FIRST for more details). The exception is if it is APPLY-MATCH-MAX-SUBSTS, in which case it will go for whichever pair is closest to exceeding the MAX-SUBSTS-VAR bound (but will still use NUM-FRPAIRS and COUNTSUBS-FIRST). It takes values of type SYMBOL and belongs to subjects TRANSMIT, UNIFICATION. The default value is:

APPLY–MATCH–ALL–FRDPAIRS

COUNTSUBS-FIRST if NIL, the substitutions which MATCH generates for each dpair in the unification process are generated and counted, and then MATCH is actually applied to the variable for which this number is smallest; if T, the substitutions are counted before they are generated, and only those which will be applied are actually generated. Applies to UN90 only. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

NIL

DNEG-IMITATION Determine when to produce imitation terms that contain double negations. Only applies in UN88 when REDUCE-DOUBLE-NEG is T (in UN88 otherwise, it is implicitly set to ALWAYS; in UN90 it is implicitly set to CONST-FLEX). When TPS mates two flexible literals f and g, it adds (f . g) as a dpair. Because it may really have needed (g . f), we allow imitation terms to contain double negations even if REDUCE-DOUBLE-NEG is T. The options are as follows: ALWAYS always allows double negations to be used. CONST forbids them for dpairs of the form (f . G), where G is a constant, but allows them otherwise. FLEX forbids them for (f . g) if g was created by a double negation in the first place (this prevents endless cycles), but allows them otherwise. CONST-FLEX forbids them in the two cases for CONST and FLEX, but allows them otherwise. NEVER forbids them outright. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

CONST–FLEX

ETA-RULE If T, eta rule is permitted in the unification package. This can be T or NIL for the UN88 procedure, but it can only be T for the UN90 procedure. (In fact, UN90 ignores this flag.) It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

T

IMITATION-FIRST Controls whether imitations are considered before projections during unification procedure UN88. No effect in UN90. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

T

LEIBNIZ-SUB-CHECK When T, check substitutions which are made for Leibniz variables, to ensure that they are relevant in their first argument. When NIL, don't do this. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

NIL

MAX-DUP-PATHS Any universal jform which has more than MAX-DUP-PATHS paths below it cannot get duplicated during search process. It takes values of type INTEGER+-OR-INFINITY and belongs to subjects

TRANSMIT, MS93-1, MS92-9, MS88, MS89, MS91-6, MS91-7, MS90-9, MS90-3, MATING-SEARCH. The default value is:

INFINITY

MAX-SEARCH-DEPTH If non nil, search to depth MAX-SEARCH-DEPTH, else search to arbitrary depth. Takes precedence over all other flags that may control the search depth in a unification tree (i.e. no tree is ever generated to a greater depth, although other flags may cause the unification search to stop temporarily at a shallower depth. Used in all search procedures, and in UN88 and UN90. See flag MAX-UTREE-DEPTH also. It takes values of type NULL-OR-POSINTEGER and belongs to subjects TRANSMIT, IMPORTANT, UNIFICATION. The default value is:

NIL

MAX-UTREE-DEPTH If non-NIL, maximum depth to which unification tree is to be generated. Used only in UN88 procedures. This variable is incremented during mating-search to allow unification tree to grow to greater depth as the search progresses. The unification tree is, however, never searched or generated to a depth greater than MAX-SEARCH-DEPTH provided it is non NIL and a positive integer. One can also consider this variable to be the initial value to which unification trees are generated during mating-search. It takes values of type NULL-OR-POSINTEGER and belongs to subjects TRANSMIT, IMPORTANT, UNIFICATION. The default value is:

5

MIN-QUICK-DEPTH The minimum depth to which a unification tree should be generated when unification tree is searched only to non branching depth. Setting this flag to 1 has the effect of generating the tree to non branching depth. Applicable only to UN88. MIN-QUICK-DEPTH is used only in the process of checking whether two literals are potential mates. It is used to construct the connection graph. See flag MAX-SEARCH-DEPTH also. See MAX-SUBSTS-QUICK for a different way to achieve a similar effect. It takes values of type NULL-OR-POSINTEGER and belongs to subjects TRANSMIT, UNIFICATION. The default value is:

3

MS-DIR The director to be used in mating search. It takes values of type SYMBOL and belongs to subjects TRANSMIT, MS91-6, MS89, MS88, MATING-SEARCH. The default value is:

QUASI−TPS1

MS90-3-QUICK If T, do MS88 quick unification on dpairs in MS90-3. If NIL, don't. It takes values of type BOOLEAN and belongs to subjects TRANSMIT, UNIFICATION, MS92-9, MS93-1, MS91-7, MS90-9, MS90-3. The default value is:

NIL

PRUNING If T, the unification routine will prune the tree as it goes. Only works for BREADTH-FIRST and BEST-FIRST unification, and only then in MS88. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

NIL

REDUCE-DOUBLE-NEG If T double negations are eliminated during lambda contraction at a unification node. This only applies in UN88. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

T

RIGID-PATH-CK If T, apply rigid-path checking when doing unification. If NIL, switch to original unification. Both UN90 and UN88 unification procedures are affected by the flag. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `UNIFICATION`. The default value is:

T

STOP-AT-TSN If T the unification algorithm terminates at a terminal success node. Otherwise, it continues generating the tree. This only applies to UN88. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

T

SUBSUMPTION-CHECK Limited subsumption check should be done during unification when this flag is set. Applies for procedures UN88 and UN90, although it is much more useful in UN88 (UN90 does not generate as many subsumed nodes, and so subsumption-checking tends to be a waste of time). See also SUBSUMPTION-NODES and SUBSUMPTION-DEPTH. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

NIL

SUBSUMPTION-DEPTH Subsumption checking takes a lot of time, compared to unification, which means that checking a new node may take more time than it could possibly save, particularly if the node is almost at the maximum depth for the unification tree. In the unification tree, new nodes at depth SUBSUMPTION-DEPTH or deeper will not be subsumption-checked; other new nodes will be. Having SUBSUMPTION-DEPTH INFINITY means that all new nodes are subsumption-checked; SUBSUMPTION-DEPTH 0 is just a slower way of turning subsumption-checking off altogether. (You should use SUBSUMPTION-CHECK NIL to do that!) This flag only applies when SUBSUMPTION-CHECK is T. See also SUBSUMPTION-NODES. It takes values of type INTEGER+-OR-INFINITY and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

INFINITY

SUBSUMPTION-NODES When SUBSUMPTION-CHECK is T, this flag determines which other nodes should be examined to see if they subsume the new node being considered. The values are as follows, arranged in order with the quickest first: PATH-NODES checks only those nodes on the path from the root to the new node. LEAF-NODES checks only the leaf nodes in the tree. LP-NODES checks leaf nodes and those on the path to the new node. ALL-NODES checks every node in the tree. Some nodes will always be excluded from subsumption checking, regardless of the value of this flag. In particular, two nodes representing different sets of connections will not be compared. This flag only applies to the UN88 procedure; in UN90, if subsumption-checking is used at all, it is implicitly set to ALL-NODES. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

LP–NODES

TOTAL-NUM-OF-DUPS Max number of duplications allowed at any time during a search using path-focused duplication. Compare NUM-OF-DUPS. This flag will be ignored if set to NIL. THE IMPLEMENTATION OF THIS IS BUGGY; setting it to NIL is safest. It takes values of type NULL-OR-POSINTEGER and belongs to subjects `TRANSMIT`, `MS90-3`, `MATING-SEARCH`, `IMPORTANT`. The default value is:

NIL

UNI-SEARCH-HEURISTIC Search strategy used to select the next node in the unification tree. BREADTH-FIRST and DEPTH-FIRST are exactly as they sound; BEST-FIRST takes whichever leaf node has the fewest free variables (and is not already terminal). All of these options work for UN90 (ms90-*, ms91-7, ms92-*); BREADTH-FIRST and BEST-FIRST are the only options for UN88 (ms88, ms89, ms91-6, mtree). It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

BREADTH–FIRST

UNIF-COUNTER If this flag is non-zero, PP* will be called to print out information about the current unification tree at regular intervals. This flag determines the length of the intervals, measured by the number of calls to the unification procedure. The amount of information is determined by the setting of UNIF-COUNTER-OUTPUT. If the flag is set to 0, this feature will be turned off. This flag only applies in UN88 unification. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

0

UNIF-COUNTER-OUTPUT See UNIF-COUNTER and UNIF-TRIGGER for the use of this flag. Settings are: 0: Print the entire tree in flat format with details. (PALL) 1: Print the entire tree in flat format without details. (PALL) 2: Print the tree in tree format with subs. (UTREE*) 3: Print the tree in tree format without subs. (UTREE*) 4: Print just the subs and details in flat format. (UTREE) 5: Print just the subs in flat format. (UTREE) 6: Print full details of the last node. (P and PP*) 7: Print some details of the last node. (P and PP) 8: Print the last node and its properties only. 9: Print the statistics for the tree so far. (STATS) 10: Print the average values for STATS, after a mating is found. This flag only applies in UN88 unification. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

0

UNIF-TRIGGER If this flag is non-NIL, PP* will be called to print out information about the current unification tree after certain events (compare UNIF-COUNTER). Settings are: NIL: Print nothing. UTREE-END: Printout whenever a tree has come to an end (either failure or success; NB UNIF-COUNTER-OUTPUT 6 or 7 will not work with this setting.) UTREE-END1: As UTREE-END, but also gives output when quick unification ends a tree without completing it. UTREE-BEGIN: Printout the root node when unification is first called. PROPS-CHANGE: Printout whenever the properties of a node are different from those of its parent. (Best used with UNIF-COUNTER-OUTPUT 6 or 7.) The amount of information is determined by the setting of UNIF-COUNTER-OUTPUT. If the flag is set to NIL, this feature will be turned off. This flag only applies in UN88 unification. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

NIL

UNIFY-VERBOSE Takes values SILENT=NIL, MIN, MED or MAX=T, and governs the amount of output relating to the unification process. It takes values of type VERBOSE and belongs to subjects `TRANSMIT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `MATING-SEARCH`, `UNIFICATION`. The default value is:

MED

28.33 Tactics

DEFAULT-TACTIC The default tactic for ETREE-NAT and USE-TACTIC. See the help messages for these commands for more information. It takes values of type TACTIC-EXP and belongs to subjects `TRANSMIT`, `TACTICS`. The default value is:

(IDTAC)

TACMODE The default mode for tactics. It takes values of type TACTIC-MODE and belongs to subjects `TRANSMIT`, `TACTICS`. The default value is:

INTERACTIVE

TACTIC-VERBOSE Determines which of the three levels of verbosity will be used: MAX – prints the message returned by each tactic called, even if it fails. MED – prints messages only when tactic succeeds. MIN – prints nothing. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `TACTICS`. The default value is:

MED

TACUSE The default use for tactics. It takes values of type TACTIC-USE and belongs to subjects `TRANSMIT`, `TACTICS`. The default value is:

NAT–DED

28.34 suggestions

GO-INSTRUCTIONS A list of instructions for GO to decide what to do with suggestions. It is a list of pairs (priority action), action being among DO, ASK, SHOW, FORGET. The default setting ((0 DO) (5 ASK) (9 SHOW) (100 FORGET)) means do suggestions of priority 0, ask me about doing suggestions of priority 5 or less, otherwise just show me suggestions of priority 9 or less and then quit. It takes values of type GO-INSTRUCT and belongs to subjects `SUGGESTS`. The default value is:

((0 DO) (5 ASK) (9 SHOW) (100 FORGET))

QUIETLY-USE-DEFAULTS If T, GO will fill in arguments with their defaults without asking for confirmation. If NIL, the command will be executed like any other command issued at the top level. It takes values of type BOOLEAN and belongs to subjects `SUGGESTS`. The default value is:

T

RESOLVE-CONFLICT If T, always the first of several suggestions is chosen, if NIL, the user will be asked. It takes values of type BOOLEAN and belongs to subjects `SUGGESTS`. The default value is:

T

28.35 Searchlists

TEST-EASIER-IF-HIGH The list of flags that, if set to high numbers, make mating-search easier. Used by SCALE-UP. "Easier" in this context means "more likely to succeed eventually, although possibly taking longer about it". Compare TEST-FASTER-IF-HIGH; the list is somewhat debatable, which is why you're allowed to change it. It takes values of type TPS-FLAGLIST and belongs to subjects `TEST-TOP`. The default value is:

(MAX–SEARCH–DEPTH SEARCH–TIME–LIMIT NUM–OF–DUPS MAX–UTREE–DEPTH MAX–MATES MAX–SEARCH–LIMIT)

TEST-EASIER-IF-LOW The list of flags that, if set to low numbers, make mating-search easier. Used by SCALE-UP. "Easier" in this context means "more likely to succeed eventually, although possibly taking longer about it". Compare TEST-FASTER-IF-LOW; the list is somewhat debatable, which is why you're allowed to change it. It takes values of type TPS-FLAGLIST and belongs to subjects `TEST-TOP`. The default value is:

(MIN–QUICK–DEPTH)

TEST-EASIER-IF-NIL The list of flags that, if set to NIL, make mating-search easier. Used by SCALE-UP. "Easier" in this context means "more likely to succeed eventually, although possibly taking longer about it". Compare TEST-FASTER-IF-NIL; the list is somewhat debatable, which is why you're allowed to change it. It takes values of type TPSFLAGLIST and belongs to subjects `TEST-TOP`. The default value is:

()

TEST-EASIER-IF-T The list of flags that, if set to T, make mating-search easier. Used by SCALE-UP. "Easier" in this context means "more likely to succeed eventually, although possibly taking longer about it". Compare TEST-FASTER-IF-T; the list is somewhat debatable, which is why you're allowed to change it. It takes values of type TPSFLAGLIST and belongs to subjects `TEST-TOP`. The default value is:

(ETA–RULE MIN–QUANTIFIER–SCOPE MS–SPLIT)

TEST-FASTER-IF-HIGH The list of flags that, if set to high numbers, make mating-search faster. Used by SCALE-DOWN. "Faster" in this context means "if it succeeds at all, it does so more quickly". Compare TEST-EASIER-IF-HIGH; the list is somewhat debatable, which is why you're allowed to change it. It takes values of type TPSFLAGLIST and belongs to subjects `TEST-TOP`. The default value is:

(MIN–QUICK–DEPTH)

TEST-FASTER-IF-LOW The list of flags that, if set to low numbers, make mating-search faster. Used by SCALE-DOWN. "Faster" in this context means "if it succeeds at all, it does so more quickly". Compare TEST-EASIER-IF-LOW; the list is somewhat debatable, which is why you're allowed to change it. It takes values of type TPSFLAGLIST and belongs to subjects `TEST-TOP`. The default value is:

(MAX–SEARCH–DEPTH SEARCH–TIME–LIMIT NUM–OF–DUPS
MAX–UTREE–DEPTH MAX–MATES MAX–SEARCH–LIMIT)

TEST-FASTER-IF-NIL The list of flags that, if set to NIL, make mating-search run faster. Used by SCALE-DOWN. "Faster" in this context means "if it succeeds at all, it does so more quickly". Compare TEST-EASIER-IF-NIL; the list is somewhat debatable, which is why you're allowed to change it. It takes values of type TPSFLAGLIST and belongs to subjects `TEST-TOP`. The default value is:

()

TEST-FASTER-IF-T The list of flags that, if set to T, make mating-search faster. Used by SCALE-DOWN. "Faster" in this context means "if it succeeds at all, it does so more quickly". Compare TEST-EASIER-IF-T; the list is somewhat debatable, which is why you're allowed to change it. It takes values of type TPSFLAGLIST and belongs to subjects `TEST-TOP`. The default value is:

(MIN–QUANTIFIER–SCOPE MS–SPLIT)

TEST-FIX-UNIF-DEPTHS If T, then LEAST-SEARCH-DEPTH will be used to fix the unification depths MAX-UTREE-DEPTH and MAX-SEARCH-DEPTH as soon as a search in the TEST top level is successful, and these will not be varied again. Destructively alters the search list, by changing the range of these two flags to a single element. It takes values of type BOOLEAN and belongs to subjects `TEST-TOP`. The default value is:

T

TEST-INCREASE-TIME After each unsuccessful search in the test top level, the value of TEST-INITIAL-TIME-LIMIT will be increased by this proportion. (So, e.g., setting this flag to 10 will result in a 10% increase on each attempt; setting it to 100 will double TEST-INITIAL-TIME-LIMIT every time around.) NOTE: After the first successful search, this flag will be set to zero. The change will be permanent, in order to allow CONTINUE to work properly. It takes values of type INTEGER+ and belongs to subjects `TEST-TOP`. The default value is:

0

TEST-INITIAL-TIME-LIMIT The time limit to be used for each individual search. This limit will be increased if it is found to be insufficient. See also the flags TEST-INCREASE-TIME and TEST-REDUCE-TIME. The time referred to will be internal time without counting garbage collection, if possible (see the flag EXCLUDING-GC-TIME). It takes values of type POSINTEGER and belongs to subjects `TEST-TOP`. The default value is:

30

TEST-MAX-SEARCH-VALUES The maximum number of values that will be put in the range of any flag in an automatically-generated searchlist. (In a manually-generated list, you can have as large a range as you like.) It takes values of type POSINTEGER and belongs to subjects `TEST-TOP`. The default value is:

10

TEST-NEXT-SEARCH-FN The name of a function which should take a searchlist and the time taken for the previous attempt as arguments, and should set the flags in the list appropriately for the next search. This function should also return T in *finished-flag* if all settings have been tried. The only values defined so far are: EXHAUSTIVE-SEARCH, which tries all combinations of flags in a searchlist, varying one flag through its entire range before trying the next flag. BREADTH-FIRST-SEARCH, which also tries all combinations of flags, but varies each flag a little at a time. PRESS-DOWN, which is used by the PRESS-DOWN command. PRESS-DOWN-2, which behaves like breadth-first search except that if varying a flag makes the search faster, that flag is then prevented from returning above its original value (the range of each flag is assumed to be ordered; if the range is (A B C D), and setting it to C results in a faster search, it will never again be set to A or B). PUSH-UP, which is used by the PUSH-UP command. PUSH-UP-2, which is like breadth-first search but terminates once a successful mode is discovered; it is used for relaxing an unsuccessful mode until it is successful. It takes values of type SYMBOL and belongs to subjects `TEST-TOP`. The default value is:

EXHAUSTIVE–SEARCH

TEST-REDUCE-TIME If T, then TEST-INITIAL-TIME-LIMIT will be reduced every time a faster combination of flags is found. If NIL, then it won't be. It takes values of type BOOLEAN and belongs to subjects `TEST-TOP`. The default value is:

T

TEST-VERBOSE If NIL, suppresses a lot of the output of the test top level. It takes values of type BOOLEAN and belongs to subjects `TEST-TOP`. The default value is:

T

TESTWIN-HEIGHT Contains the initial height of the testwindow. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `TEST-TOP`. The default value is:

24

TESTWIN-WIDTH Contains the initial width of the testwindow. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `TEST-TOP`. The default value is:

80

28.36 Vpforms

ALLOW-NONLEAF-CONNS The value of this flag is a list of symbols. If ALL is in the list, then the jform contains literals for each node (except LAMBDA rewrites).

If REWRITES is in the list, then the jform contains literals for each rewrite node (except LAMBDA's).

If the name of an etree node is in the list, then the jform contains literals for the specified node.

NOTE: This flag affects the way jforms are generated. Consequently, different search procedures may (or may not) be affected by it. It takes values of type SYMBOLLIST and belongs to subjects `TRANSMIT`, `MATING-SEARCH`. The default value is:

()

DISSOLVE DISSOLVE is set to a list of connections which are used to perform dissolution when forming the jform from the etree. If the list of connections is NIL the jform is constructed as usual. (See Murray, Rosenthal, Dissolution: Making Paths Vanish, JACM 40, 3, July 1993, pp. 504-535) It takes values of type MATINGPAIRLIST and belongs to subjects `TRANSMIT`, `MATING-SEARCH`. The default value is:

()

LIT-NAME Prefix for labels associated with literals. It takes values of type SYMBOL and belongs to subjects `JFORMS`. The default value is:

LIT

MATE-UP-TO-NNF If MATE-UP-TO-NNF is T, then literals represent the negation normal form of formulas or their negation. This allows connections between formulas that are only equal up to negation normal form. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MATING-SEARCH`. The default value is:

T

ORDER-COMPONENTS When T or PATHNUM, the components of a jform node will be rearranged in order of the number of paths which lie below them (go through them). When T-REVERSED or PATHNUM-REVERSED, the components of a jform node will be rearranged in reverse order of the number of paths which lie below them (go through them). When NIL or COMMON, then the jform of the current eproof will not be modified by the mating search; When REVERSE, the order of the components in the jform of current eproof will be reversed; When PREFER-RIGID2, the order of the components in the jform of the current eproof will be sorted in terms of the number of rigid literals in a jform before beginning the mating search. When PREFER-RIGID3, the components in the jform of the current eproof will be sorted as for

PREFER-RIGID2, but with preference given to literals that arise from DUAL rewriting.

(PREFER-RIGID1 is still available; it is an obsolete version of PREFER-RIGID2.) It takes values of type ORDERCOM and belongs to subjects `TRANSMIT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `IMPORTANT`, `JFORMS`, `MATING-SEARCH`. The default value is:

T

PRINT-LIT-NAME If the value of this flag is true, labels (instead of wffs associated with literal, or neg-literal) are printed inside the editor. It takes values of type BOOLEAN and belongs to subjects `JFORMS`. The default value is:

T

PRINTVPDFLAG If T, vertical path diagrams are written into the VPD-FILENAME whenever wffs are written into the PRINTEDTFILE. In particular PRINTEDTFLAG must be T, for the automatic writing to take place. It takes values of type BOOLEAN and belongs to subjects `EDITOR`, `JFORMS`. The default value is:

NIL

TEXFORMAT HPD for a horizontal path diagram (p.d.) of the positive wff. VPD for a vertical p.d. of the negated wff. VPP (or anything else) for a vertical p.d. of the positive wff. It takes values of type SYMBOL and belongs to subjects `JFORMS`. The default value is:

VPP

VPD-BRIEF The default value for BRIEF when printing VP diagrams in a file. Currently the options are: T = no atom values will show in VP diagram A = atom values but no labels will appear in VP diagram NIL = atom values and labels will show in VP diagram LT = atom values and labels and a legend will show in VP diagram L = labels but no atom values will show in VP diagram, and a legend will show both B = boxed labels and atoms will show in the VP diagram. BT = boxed labels will show in the diagram, and the atom values will be listed below. B and BT only work in TeX format (i.e. with the VPT command). It takes values of type VPFORMAT and belongs to subjects `JFORMS`. The default value is:

L

VPD-FILENAME Default filename when printing VP diagrams in a file. It takes values of type FILESPEC and belongs to subjects `JFORMS`. The default value is:

```
"vpd.vpf"
```

VPD-LIT-NAME Prefix for labels associated with literals when VP diagrams are created automatically within the editor. It takes values of type SYMBOL and belongs to subjects `JFORMS`. The default value is:

V

VPD-PTYPES If T, print types when printing VP diagrams in a file. It takes values of type BOOLEAN and belongs to subjects `JFORMS`. The default value is:

T

VPD-STYLE The default value for STYLE when printing VP diagrams in a file. It takes values of type VPSTYLE and belongs to subjects `JFORMS`. The default value is:

GENERIC

VPD-VPFPAGE The default value for the width of the page when printing VP diagrams in a file. It takes values of type POSINTEGER and belongs to subjects `JFORMS`. The default value is:

78

VPFORM-LABELS In the editor, a value of T for this flag will suppress printing of labels in vpforms; if it is NIL, labels and atom values will be printed. If this flag is set the default value for argument BRIEF will be A. Unless one decides to override the default value, labels will not be printed. This flag has no effect on the editor command VPD, and on the wffop DISPLAY-VPD. To suppress labels when using these commands, please set the flag VPD-BRIEF to A. It takes values of type BOOLEAN and belongs to subjects `JFORMS`. The default value is:

NIL

VPFORM-TEX-MAGNIFICATION The magnification factor to use for TeX files containing vpforms. This has two possible settings: if it is lower than 10, then it is used in the form \magnification=\magstepN Roughly, 0 = 10pt, 1 = 12pt, 2 = 14pt, 3 = 17pt, 5 = 25pt.

Otherwise, it is used in the form \magnificationN, in which case 1000 corresponds to "normal size" (12pt), 800 is 80% , 1200 is 120% , and so on. It takes values of type INTEGER+ and belongs to subjects `JFORMS`. The default value is:

1000

VPFORM-TEX-NEST Maximal number of boxes to nest in path diagrams for TeX. 0 means not to break into boxes. It takes values of type INTEGER+ and belongs to subjects `JFORMS`. The default value is:

4

VPFORM-TEX-PREAMBLE The string to be put at the beginning of a TeX file containing vpforms. It takes values of type STRING and belongs to subjects `JFORMS`. The default value is:

""

VPW-HEIGHT Contains the intial height of the vpform window; there is no need to update this if the window is resized after being opened. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `JFORMS`. The default value is:

25

VPW-WIDTH Contains the current width of the vpform window; should be updated by the user if the window is resized after being opened. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `JFORMS`. The default value is:

120

28.37 Semantics

MAX-BINDER-COMPUTATION The maximum number of elements TPS is willing to consider when interpreting binders (quantifiers and lambdas) in a model. This depends on the size of domains and the nesting of binders in the formula. It takes values of type INTEGER+ and belongs to subjects `MS04-2`, `SEMANTIC-BOUNDS`. The default value is:

1048576

MAX-DOMAIN-SIZE The maximum size of semantic domains TPS will consider. It does not make sense to set this to any value other than a size such a domain may have. For example, the default value 2^16 is 65536. Assuming every base type is of size 2, the next reasonable value would be 2^32, which is over 4 billion. Consequently, the value of this flag should not be changed until TPS is either considering models other than standard models based on powers of 2 or computing power increases tremendously. It takes values of type INTEGER+ and belongs to subjects `MS04-2`, `SEMANTIC-BOUNDS`. The default value is:

65536

28.38 Printing

REWRITING-RELATION-SYMBOL Contains the symbol that is printed between lines obtained by rewriting from immediately preceding lines. It takes values of type SYMBOL and belongs to subjects `S-EQN`. The default value is:

=

VERBOSE-REWRITE-JUSTIFICATION When set to T, justification of lines obtained by rewriting in the REWRITING top level will indicate the rewriting theory used to obtain the transformation. It takes values of type BOOLEAN and belongs to subjects `S-EQN`. The default value is:

T

28.39 Applying Rules

APP*-REWRITE-DEPTH The maximal rewrite depth of an app* application. It takes values of type NULL-OR-POSINTEGER and belongs to subjects `S-EQN`. The default value is:

50

REWRITING-AUTO-DEPTH The maximal depth of a search tree when applying AUTO. For the SIMPLE search procedure, the number corresponds to the maximal rewrite depth, whereas for BIDIR and BIDIR-SORTED the maximal search depth is twice the specified number. It takes values of type POSINTEGER and belongs to subjects `S-EQN`. The default value is:

5

REWRITING-AUTO-GLOBAL-SORT When NIL, BIDIR-SORTED will choose the next wff to be rewritten from the successors of the current wff. When T, it will choose the next wff from all unexplored wffs obtained so far from the initial or the target wff, respectively. See the flag REWRITING-AUTO-SEARCH-TYPE. It takes values of type BOOLEAN and belongs to subjects `S-EQN`. The default value is:

NIL

REWRITING-AUTO-MAX-WFF-SIZE The maximal size of a wff to be rewritten when applying AUTO. It takes values of type POSINTEGER and belongs to subjects `S-EQN`. The default value is:

15

REWRITING-AUTO-MIN-DEPTH The minimal depth of a search tree needed by AUTO to find a derivation. The value should be less or equal to that of REWRITING-AUTO-DEPTH, otherwise no search will be performed. It takes values of type INTEGER+ and belongs to subjects `S-EQN`. The default value is:

0

REWRITING-AUTO-SEARCH-TYPE The search procedure to use with AUTO. Currently defined are SIMPLE, BIDIR and BIDIR-SORTED. BIDIR-SORTED will try to rewrite shorter wffs first. When this is not needed, use BIDIR. The precise behaviour of BIDIR-SORTED depends on the flag REWRITING-AUTO-GLOBAL-SORT. It takes values of type AUTO-SEARCHTYPE and belongs to subjects `S-EQN`. The default value is:

BIDIR–SORTED

REWRITING-AUTO-SUBSTS List of terms to substitute for any free variables which may be introduced during rewriting by AUTO. If NIL, the list will be generated automatically from atomic subwffs of the source and the target wff. It takes values of type GWFFLIST and belongs to subjects `S-EQN`. The default value is:

()

REWRITING-AUTO-TABLE-SIZE The maximal size of a search table used by AUTO. Note that while the SIMPLE search procedure uses only one table of that size, BIDIR and BIDIR-SORTED use two. It takes values of type POSINTEGER and belongs to subjects `S-EQN`. The default value is:

10000

28.40 Propositional Rules

RULEP-MAINFN The main function used for RULEP. Defaults to RULEP-DELUXE, in which case RULEP will find a minimal subset of the support lines which suffices to justify the planned line. If set to RULEP-SIMPLE, RULEP will merely check that the planned line follows from the support lines that are specified by the user. It takes values of type RULEP-MAINFN-TYPE and belongs to subjects `RULES-MOD`. The default value is:

RULEP–DELUXE

28.41 Wff Editor

EDPPWFFLAG If T, wffs are always pretty-printed in the formula editor. It takes values of type BOOLEAN and belongs to subjects `PRINTING`, `EDITOR`. The default value is:

NIL

EDPRINTDEPTH The depth to which wffs are printed in the formula editor. It takes values of type INTEGER+ and belongs to subjects `PRINTING`, `EDITOR`. The default value is:

24

EDWIN-CURRENT If T, the Current Edwff window is opened to display the current wff being edited when the editor is started. It takes values of type BOOLEAN and belongs to subjects `PRINTING`, `EDITOR`. The default value is:

T

EDWIN-CURRENT-HEIGHT Controls the initial height of the Current Edwff window. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `EDITOR`. The default value is:

3

EDWIN-CURRENT-WIDTH Controls the initial width of the Current Edwff window. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `EDITOR`. The default value is:

80

EDWIN-TOP If T, the Top Edwff window is opened to display the entire wff being edited when the editor is started. It takes values of type BOOLEAN and belongs to subjects `PRINTING`, `EDITOR`. The default value is:

T

EDWIN-TOP-HEIGHT Controls the initial height of the Top Edwff window. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `EDITOR`. The default value is:

3

EDWIN-TOP-WIDTH Controls the initial width of the Top Edwff window. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `EDITOR`. The default value is:

80

EDWIN-VPFORM If T, the Current Vpform window is opened to display the vpform of the current wff being edited when the editor is started. This flag is ignored in ETPS, where the Vpform window is never opened. It takes values of type BOOLEAN and belongs to subjects `PRINTING`, `EDITOR`. The default value is:

NIL

EDWIN-VPFORM-HEIGHT Controls the initial height of the Current Vpform window. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `EDITOR`. The default value is:

30

EDWIN-VPFORM-WIDTH Controls the initial width of the Current Vpform window. It takes values of type POSINTEGER and belongs to subjects `WINDOW-PROPS`, `EDITOR`. The default value is:

60

28.42 wff Primitives

META-BDVAR-NAME The prefix for names of bound meta variables. It takes values of type SYMBOL and belongs to subjects `INTERNAL-NAMES`. The default value is:

BD

META-VAR-NAME The prefix for names of meta variables. It takes values of type SYMBOL and belongs to subjects `INTERNAL-NAMES`. The default value is:

MV

REN-VAR-FN The value of this flag is a function to be called when a variable must be renamed automatically. It has three possible settings: REN-VAR-X1 is the standard renaming function. It renames y to y^1, then to y^2, and so on. If there is another variable y, of a different type, it makes no difference. REN-VAR-X11 is much like REN-VAR-X1, except it will avoid creating two variables of the same name at different types (so it tends to produce higher exponents than REN-VAR-X1). REN-VAR-XA renames alphabetically, turning y into ya, then yba, and so on. It takes values of type SYMBOL and belongs to subjects `WFF-PRIMS`. The default value is:

REN–VAR–X1

RENAME-ALL-BD-VARS When T, all bound variables inside a definition will be renamed before instantiation. It takes values of type BOOLEAN and belongs to subjects `WFF-PRIMS`. The default value is:

NIL

28.43 Wff Parsing

BASE-TYPE If not NIL, it should be the 'default' type for individual variables in a logic system. Typically I (for iota). It takes values of type SYMBOL and belongs to subjects `PARSING`. The default value is:

I

FIRST-ORDER-MODE-PARSE If T, every letter by itself is a symbol for the parser, with the exception of keywords like FORALL, AND etc., which can be in mixed case. If NIL, symbols must be separated by spaces (or brackets, dots, etc.). It takes values of type BOOLEAN and belongs to subjects `PARSING`. The default value is:

NIL

LOWERCASERAISE If T, lower case characters will be raised to upper case, when read. Has no effect in first-order mode. It takes values of type BOOLEAN and belongs to subjects `PARSING`. The default value is:

NIL

TYPE-IOTA-MODE If T, type variables are always assumed to be iota. It takes values of type BOOLEAN and belongs to subjects `PARSING`. The default value is:

T

UNTYPED-LAMBDA-CALCULUS Takes values T or NIL. To set it to T if you want to use the editor to deal with untyped lambda-calculus. It takes values of type BOOLEAN and belongs to subjects `EDITOR`. The default value is:

NIL

28.44 Basic Abbreviations

REWRITE-EQUALITIES One of the following: NONE: do not rewrite equalities ONLY-EXT: rewrite only those equalities that can be rewritten using extensionality. LEIBNIZ: rewrite all equalities using the Leibniz definition. ALL: rewrite all equalities, to an equivalence for those of type OOO, to the extensional form [lambda f(AB) lambda g(AB) forall x(B) f x = g x] for those of type O(AB)(AB), and to the Leibniz form [lambda x(A) lambda y(A) forall q(OA). q x implies q y] for those of type OAA. LAZY2: As for ALL, but keeping a duplicate leaf as in the LAZY2 setting of the flag REWRITE-DEFNS. PARITY1: Uses the parity to determine whether equalities should be rewritten as the setting LEIBNIZ or as the setting ALL. For example, using PARITY1 when trying to prove the wff A(OI) = B(OI) implies C the equality is expaned using Leibniz, and when trying to prove the wff D implies A(OI) = B(OI) the equality is expanded using extensionality. The heuristic is that we often use the substitutivity property when we use an equation and use extensionality to show an equation. It takes values of type REWRITE-DEFNS and belongs to subjects `TRANSMIT`, `MS98-1`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `IMPORTANT`, `WFF-PRIMS`, `MATING-SEARCH`. The default value is:

ALL

28.45 Lambda-Calculus

LAMBDA-CONV BETA-ETA-TOGETHER means that BETA and ETA rules are used together; BETA-ETA-SEPARATE means BETA and ETA rules are used separately; BETA-ONLY means that only BETA rule is allowed. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `TACTICS`, `ETR-NAT`, `ETREES`. The default value is:

BETA–ETA–TOGETHER

28.46 Primitive Substitutions

BAD-VAR-CONNECTED-PRUNE When generating set constraints, prune those which do not have bad variables (selected variables the set variable cannot depend upon) shared between the literals in the constraints. For example, if p cannot depend on x or y, the constraints

p 0 -> A x p x -> A 0 p x -> A x, B y p y -> A x, B y

would be pruned while the constraints

p x -> A x p x -> A x y, B y

would not be pruned. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `IMPORTANT`, `MATING-SEARCH`. The default value is:

T

DELAY-SETVARS If T, first solve the rigid part of the jform, then try to solve the flexible parts using setvar constraints. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `PRIMSUBS`. The default value is:

NIL

INCLUDE-COINDUCTION-PRINCIPLE When solving co-closure set-variable constraints we include in the lemma a higher-order statement that we have the greatest solution.

For example, suppose we want a set N such that

X 0 and forall z [X [f z] implies [X z]]

If include-coinduction-principle is set to T, then the lemma will include a conjunct of the form

forall p . [p 0] and [forall z [p [f z] implies [p z]]] implies forall x [p x implies N x]. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `IMPORTANT`, `PRIMSUBS`, `MATING-SEARCH`. The default value is:

T

INCLUDE-INDUCTION-PRINCIPLE When solving closure set-variable constraints we include in the lemma a higher-order statement that we have the least solution.

For example, suppose we want a set N such that

N 0 and forall n [N n implies [N [S n]]]

If include-induction-principle is set to T, then the lemma will include a conjunct of the form

forall p . p 0 and [forall n [p n implies [p [S n]]]] implies forall x [N x implies p x]. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `IMPORTANT`, `PRIMSUBS`, `MATING-SEARCH`. The default value is:

T

MAX-CONSTRAINT-SIZE Maximum number of literals allowed in a single constraint It takes values of type INTEGER+-OR-INFINITY and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `IMPORTANT`, `MATING-SEARCH`. The default value is:

3

MAX-NUM-CONSTRAINTS Maximum number of combined constraints in each constraint set. It takes values of type INTEGER+-OR-INFINITY and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `IMPORTANT`, `MATING-SEARCH`. The default value is:

2

MAX-PRIM-DEPTH Maximum depth to which primsubs with quantifiers are generated. The types of the quantified variables range over the values in PRIM-BDTYPES. With PRIMSUB-METHOD PR89 : This flag is ignored. Primsubs of the form "exists x . literal" and "forall x . literal" will be generated. With PRIMSUB-METHOD PR93 : At depth 1, a single quantifier is introduced, as in PR89. At depth N>1, we have (N-1) quantifiers ranging over a formula containing (N-1) conjunctions disjunctions of (N-2) disjunctions conjunctions. With PRIMSUB-METHOD PR95 : At depth 1, as in PR89. At depth N>1, we have (N-1) quantifiers ranging over a formula with between MIN-PRIM-LITS and MAX-PRIM-LITS literals, with all combinations of connectives between them. With PRIMSUB-METHOD PR97 : At depth N>0, we have (N-1) quantifiers ranging over each subformula taken from the etree which contains between MIN-PRIM-LITS and MAX-PRIM-LITS literals. You can see these subformulas by doing ETP from the MATE top level. With PRIMSUB-METHOD PR97A : As in PR97, but all substitutions are in negation normal form. With PRIMSUB-METHOD PR97B : The substitutions from PR97A and PR95 are interleaved. The order is determined firstly by the number of literals, then by the number of quantifiers, and lastly with PR97 substs taking precedence over PR95. With PRIMSUB-METHOD PR97C : If set to N, all primsubs will have < N quantifiers. With PRIMSUB-METHOD PR00 : This is ignored. It takes values of type POSINTEGER and belongs to subjects `TRANSMIT`, `IMPORTANT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `PRIMSUBS`. The default value is:

1

MAX-PRIM-LITS Maximum no. of literals allowed in a primsub. Does not apply for PRIMSUB-METHOD PR89 or PR93. See the help message for MIN-PRIM-DEPTH, which explains how primsubs are generated. It takes values of type POSINTEGER and belongs to subjects `TRANSMIT`, `IMPORTANT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `PRIMSUBS`. The default value is:

4

MIN-PRIM-DEPTH Minimum depth at which primsubs with quantifiers are generated. The types of the quantified variables range over the values in PRIM-BDTYPES. With PRIMSUB-METHOD PR89 : This flag is ignored. Primsubs of the form "exists x . literal" and "forall x . literal" will be generated. With PRIMSUB-METHOD PR93 : At depth 1, a single quantifier is introduced, as in PR89. At depth N>1, we have (N-1) quantifiers ranging over a formula containing (N-1) conjunctions disjunctions of (N-2) disjunctions conjunctions. With PRIMSUB-METHOD PR95 : At depth 1, as in PR89. At depth N>1, we have (N-1) quantifiers ranging over a formula with between MIN-PRIM-LITS and MAX-PRIM-LITS literals, with all combinations of connectives between them. With PRIMSUB-METHOD PR97 : At depth N>0, we have (N-1) quantifiers ranging over each subformula taken from the etree which contains between MIN-PRIM-LITS and MAX-PRIM-LITS literals. You can see these subformulas by doing NAME-PRIM from the MATE top level. With PRIMSUB-METHOD PR97A : As in PR97, but all substitutions are in negation normal form. With PRIMSUB-METHOD PR97B : The substitutions from PR97A and PR95 are interleaved. The order is determined firstly by the number of literals, then by the number of quantifiers, and lastly with PR97 substs taking precedence over PR95. With PRIMSUB-METHOD PR97C : If set to N, the number of quantifiers in any primsub will be >= N-1. With PRIMSUB-METHOD PR00 : The value is ignored. It takes values of type POSINTEGER and belongs to subjects `TRANSMIT`, `IMPORTANT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `PRIMSUBS`. The default value is:

1

MIN-PRIM-LITS Minimum no. of literals allowed in a primsub. Does not apply for PRIMSUB-METHOD PR89 or PR93. See the help message for MIN-PRIM-DEPTH, which explains how primsubs are generated. It takes values of type POSINTEGER and belongs to subjects `TRANSMIT`, `IMPORTANT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `PRIMSUBS`. The default value is:

2

NEG-PRIM-SUB When T, one of the primitive substitutions will introduce negation. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `PRIMSUBS`. The default value is:

NIL

PR00-ALLOW-SUBNODE-CONNS If T, we allow connections between nodes and their subnodes. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `PRIMSUBS`. The default value is:

T

PR00-MAX-SUBSTS-VAR The setting for MAX-SUBSTS-VAR when generating set variable instantiations by unification using PRIMSUB-METHOD PR00. It takes values of type NULL-OR-INTEGER and belongs to subjects `TRANSMIT`, `UNIFICATION`, `PRIMSUBS`. The default value is:

4

PR00-NUM-ITERATIONS Number of times to iterate the PR00 Set Substitution process. It takes values of type POSINTEGER and belongs to subjects `TRANSMIT`, `IMPORTANT`, `PRIMSUBS`. The default value is:

1

PR00-REQUIRE-ARG-DEPS If T, do not consider set subsitutions which do not depend on some argument. For example, do not consider P –> lambda x y PHI where neither x nor y is free in PHI. This often rules out many setsubs generated by unification. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `PRIMSUBS`. The default value is:

NIL

PR97C-MAX-ABBREVS The maximum number of abbreviations that may appear in a PR97C primsub. It takes values of type POSINTEGER and belongs to subjects `TRANSMIT`, `IMPORTANT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `PRIMSUBS`. The default value is:

1

PR97C-PRENEX If T, PR97C generates substitutions in prenex normal form. If NIL, it doesn't. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `IMPORTANT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `PRIMSUBS`. The default value is:

T

PRIM-BDTYPES List of types of quantified variables used to construct primitive substitutions. This list will always be used when constructing primitive substitutions interactively, but see the flag PRIM-BDTYPES-AUTO for more information on the types that will be used by automatic search procedures. It takes values of type TYPESYMLIST-NIL and belongs to subjects `TRANSMIT`, `IMPORTANT`, `PRIMSUBS`. The default value is:

("I")

PRIM-BDTYPES-AUTO Has five possible values: REPLACE, REPLACE-SUB, APPEND, APPEND-SUB and IGNORE. Determines how the procedures that use primitive substitutions handle the flag PRIM-BDTYPES, as follows: REPLACE – the value of PRIM-BDTYPES will be changed to an automatically-generated list of all the primitive types used in the gwff to be proven. REPLACE-SUB – as for replace, except that the list will be of all the subtypes of the types that appear in the gwff. APPEND – the same list is calculated as for REPLACE, but instead of replacing the current setting of PRIM-BDTYPES it will be appended to it. APPEND-SUB – the same list is calculated as for APPEND, but instead of replacing the current setting of PRIM-BDTYPES it will be appended to it. IGNORE – no list will be generated, and the user's setting of PRIM-BDTYPES will be left intact. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `IMPORTANT`, `PRIMSUBS`. The default value is:

REPLACE

PRIM-PREFIX Prefix for weak labels associated with primitive substitutions. It takes values of type SYMBOL and belongs to subjects `PRIMSUBS`. The default value is:

PRIM

PRIMSUB-METHOD Takes one of the values PR89, PR93, PR95, PR97, PR97A, PR97B. This determines how primsubs will be generated, in conjunction with MAX-PRIM-DEPTH, MIN-PRIM-DEPTH, MAX-PRIM-LITS and MIN-PRIM-LITS. With PRIMSUB-METHOD PR89 : Primsubs of the form "exists x . literal" and "forall x . literal" will be generated. With PRIMSUB-METHOD PR93 : For all integers from MIN-PRIM-DEPTH to MAX-PRIM-DEPTH: At depth 1, a single quantifier is introduced, as in PR89. At depth N>1, we have (N-1) quantifiers ranging over a formula containing (N-1) conjunctions disjunctions of (N-2) disjunctions conjunctions. With PRIMSUB-METHOD PR95 : For all integers from MIN-PRIM-DEPTH to MAX-PRIM-DEPTH: At depth 1, as in PR89. At depth N>1, we have (N-1) quantifiers ranging over a formula with between MIN-PRIM-LITS and MAX-PRIM-LITS literals, with all combinations of connectives between them. With PRIMSUB-METHOD PR97 : For all integers from MIN-PRIM-DEPTH to MAX-PRIM-DEPTH: At depth N>0, we have (N-1) quantifiers ranging over each subformula taken from the etree which contains between MIN-PRIM-LITS and MAX-PRIM-LITS literals. You can see these subformulas by doing NAME-PRIM from the MATE top level. (Note: both the instantiated and uninstantiated versions of each definition are used.) With PRIMSUB-METHOD PR97A : As in PR97, but all substitutions are in negation normal form. With PRIMSUB-METHOD PR97B : The substitutions from PR97A and PR95 are interleaved. The order is determined firstly by the number of literals, then by the number of quantifiers, and lastly with PR97 substs taking precedence over PR95. With PRIMSUB-METHOD PR97C : Using the connectives

AND and OR, and the quantifiers EXISTS and FORALL (ranging over variables of types PRIM-BDTYPES), and also using any abbreviations or equalities that occur in the gwff to be proven, primsubs are built up using the bounds given by MIN- and MAX-PRIM-LITS and MIN- and MAX-PRIM-DEPTH. See also PR97C-PRENEX and PR97C-MAX-ABBREVS. With PRIMSUB-METHOD PR00 : This uses higher order unification to determine set substitutions that solve part of the mating search in advance. PR00 only works with DEFAULT-MS MS98-1 and SKOLEM-DEFAULT NIL. PR00 can be controlled using the flags PR00-MAX-SUBSTS-VAR, PR00-REQUIRE-ARG-DEPS, PR00-NUM-ITERATIONS. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `IMPORTANT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `PRIMSUBS`. The default value is:

PR93

WHICH-CONSTRAINTS Which kinds of set constraints should be generated and solved.

. MAX: Constraints for p of the form Psi | p t ==> Gamma(p) solved using maximal solution. . MIN: Constraints for p of the form Psi | Gamma(p) ==> p t solved using minimal solution. . PR00: Generates instantiated ftrees and connections by mating nonleaves. It takes values of type SYMBOLLIST and belongs to subjects `TRANSMIT`, `PRIMSUBS`, `IMPORTANT`, `MATING-SEARCH`. The default value is:

(MAX MIN)

28.47 Miscellaneous

REWRITE-EQUIVS This chooses one of the two ways of constructing an etree from an equivalence A EQUIV B: 1 chooses the option with the fewest vertical paths (positive: A AND B OR A AND B negative: A IMPLIES B AND B IMPLIES A) 2 chooses the option with the fewest horizontal paths (negative: A AND B OR A AND B positive: A IMPLIES B AND B IMPLIES A) 3 behaves as for 2 except for the first equivalence it finds, when it behaves as for 1. (This means that a gwff which is a quantified equivalence will produce an etree which can be split.) 4 always chooses A IMPLIES B AND B IMPLIES A 5 always chooses A AND B OR A AND B Any other setting will behave like 1.

This does not work with MIN-QUANTIFIER-SCOPE T; in that case, etrees will be constructed as in case 1, regardless of the setting of this flag. It takes values of type POSINTEGER and belongs to subjects `TRANSMIT`, `MS98-1`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `IMPORTANT`, `MATING-SEARCH`. The default value is:

1

28.48 RuleP

RULEP-WFFEQ The wffop used for testing whether two wffs are equal when checking RULEP and propositional mating search. It takes values of type SYMBOL and belongs to subjects `TRANSMIT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `MATING-SEARCH`, `JFORMS`. The default value is:

WFFEQ–AB

28.49 Skolemizing

NAME-SKOLEM-FN Name of the functions which names a Skolem function. It takes values of type SYMBOL and belongs to subjects `WFF-PRIMS`. The default value is:

NAME–SKOLEM–CAP

28.50 Quantifiers

UI-HERBRAND-LIMIT Maximum number of times to apply ui-herbrand-tac to the same universally-quantified formula. It takes values of type POSINTEGER and belongs to subjects `TACTICS`. The default value is:

3

28.51 Auxiliary

USE-RULEP When true, indicates that RuleP should be used when possible in translating from expansion proof to natural deduction proof. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `TACTICS`, `ETR-NAT`, `MATING-SEARCH`. The default value is:

T

USE-SYMSIMP When true, indicates that symmetric simplification should be used when possible in translating from expansion proof to natural deduction proof. Consult Pfenning's thesis for a description of symmetric simplification. It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `MS93-1`, `MS92-9`, `MS91-7`, `MS91-6`, `MS90-9`, `MS90-3`, `MS89`, `MS88`, `TACTICS`, `ETR-NAT`, `MATING-SEARCH`. The default value is:

T

28.52 Events

ADVICE-ASKED-ENABLED If NIL, recording events of type ADVICE-ASKED is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

ADVICE-FILE The file recording advice. It takes values of type FILESPEC and belongs to subjects `EVENTS`. The default value is:

"etps3.advice"

COMMAND-ENABLED If NIL, recording events of type COMMAND is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

COMMAND-FILE The file recording commands. It takes values of type FILESPEC and belongs to subjects `EVENTS`. The default value is:

"/home/pa01/etps3.command"

DONE-EXC-ENABLED If NIL, recording events of type DONE-EXC is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

ERROR-ENABLED If NIL, recording events of type ERROR is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

ERROR-FILE The file recording the events of errors. It takes values of type FILESPEC and belongs to subjects `EVENTS`. The default value is:

"etps3.error"

EVENT-CYCLE The indivisible unit in number of inputs. When WRITE-WHEN for an EVENT is 'n', the event info will be written every n * event-cycle inputs. n=0 means don't write. It takes values of type INTEGER+ and belongs to subjects `EVENTS`. The default value is:

5

EVENTS-ENABLED If nil, all events are disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

INPUT-ERROR-ENABLED If NIL, recording events of type INPUT-ERROR is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

INPUT-ERROR-FILE The file recording illegal inputs caught by TPS. It takes values of type FILESPEC and belongs to subjects `EVENTS`. The default value is:

"etps3.ierror"

PROOF-ACTION-ENABLED If NIL, recording events of type PROOF-ACTION is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

PROOF-FILE The file recording started and completed proofs. It takes values of type FILESPEC and belongs to subjects `EVENTS`. The default value is:

"/home/pa01/etps3.proof"

QUIET-EVENTS If T, no message will be given when events are written. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

RULE-ERROR-ENABLED If NIL, recording events of type RULE-ERROR is disabled. It takes values of type BOOLEAN and belongs to subjects `EVENTS`. The default value is:

T

RULE-ERROR-FILE The file recording illegal rules caught by TPS. It takes values of type FILESPEC and belongs to subjects `EVENTS`. The default value is:

"etps3.rerror"

SCORE-FILE The file recording completed exercises. It takes values of type FILESPEC and belongs to subjects `EVENTS`. The default value is:

"etps3.scores"

USER-PASSWD-FILE The file recording user id's and passwords for a class using ETPS over the web. It takes values of type FILESPEC and belongs to subjects `EVENTS`. The default value is:

"user-passwd"

28.53 Grader

CAL-PERCENTAGE The program calculates percentage based on total scores if the value of this variable is T. It takes values of type BOOLEAN and belongs to subjects `GR-MISC`. The default value is:

NIL

COURSE-NAME Name of the course. Also used as a suffix for various files which are created or modified by the grading package. It takes values of type STRING and belongs to subjects `GR-MISC`. The default value is:

"course"

DEFAULT-PENALTY-FN Default penalty function for late exercises. The default is no-penalty which doesn't take any points off. It takes values of type FUNCTION and belongs to subjects `GR-MISC`. The default value is:

NO–PENALTY

DROP-MIN When calculating totals, the program drops the minimum scores on each of the items in this list. It takes values of type CONSP1 and belongs to subjects `GR-MISC`. The default value is:

NIL

DUE-DATE-FLAG If this flag is nil, the user is not prompted for due dates (in the command ETPS-GRADE) and it's assumed that all exercises were submitted in time. It takes values of type BOOLEAN and belongs to subjects `GR-MISC`. The default value is:

T

ETPS-FILE Name of the file which contains ETPS records. It takes values of type FILESPEC and belongs to subjects `GR-FILENAMES`. The default value is:

""

GRADE-DIR Name of the directory in which the grader files are to be found, or "" for the directory from which grader was started. This name should end with a backslash, as in : "/usr/teacher/course-grades/". When this flag is changed, all of the other filenames will change with it. Note that in old versions of CMU lisp, the "" option will not work properly. It takes values of type STRING and belongs to subjects `GR-FILENAMES`. The default value is:

""

GRADE-FILE Name of the GRADE-FILE. It takes values of type FILESPEC and belongs to subjects `GR-FILENAMES`. The default value is:

""

LETTER-GRADE-FILE Name of the file which will contain letter grades. It takes values of type FILESPEC and belongs to subjects `GR-FILENAMES`. The default value is:

""

LETTER-GRADE-FLAG The program creates a separate file containing letter grades if the value of this variable is true. It takes values of type BOOLEAN and belongs to subjects `GR-MISC`. The default value is:

T

NEW-ITEM The list of new items to be calculated when calculating totals. See the manual for more details. It takes values of type CONSP1 and belongs to subjects `GR-MISC`. The default value is:

NIL

OLD-GRADE-FILE Name of the back-up GRADE-FILE. It takes values of type FILESPEC and belongs to subjects `GR-FILENAMES`. The default value is:

""

OLD-TOTALS-GRADE-FILE Name of the back-up TOTALS-GRADE-FILE . It takes values of type FILESPEC and belongs to subjects `GR-FILENAMES`. The default value is:

""

PATCH-FILE Name of the file containing changes to the grader core image. It takes values of type FILESPEC and belongs to subjects `GR-FILENAMES`. The default value is:

"grader.patch"

PRINT-N-DIGITS The number of digits to be printed after the decimal. It takes values of type INTEGER+ and belongs to subjects `GR-MISC`. The default value is:

0

STATISTICAL-OPTIONS List of statistical data to be calculated. Currently the program can calculate mean, median , standard deviation. The default is (-mean- -median- -sdev-). It takes values of type CONSP1 and belongs to subjects `GR-MISC`. The default value is:

(–MEAN– –MEDIAN– –SDEV–)

TOTALS-GRADE-FILE Name of the file which will contain totals. It takes values of type FILESPEC and belongs to subjects `GR-FILENAMES`. The default value is:

""

28.54 Maintenance

COMPILED-EXTENSION The extension of compiled files in TPS3. It takes values of type STRING and belongs to subjects `MAINTAIN`. The default value is:

"fasl"

EXPERTFLAG If T, arbitrary Lisp expression may be evaluated on top levels. It takes values of type BOOLEAN and belongs to subjects `MAINTAIN`. The default value is:

NIL

GOODMODES A name for a pair MODES and GWFFS where MODES is a list of modes and GWFFS is a list of theorems. Every theorem in GWFFS should be provable using some mode in MODES. To check this, or to use these modes to try to prove a new theorem, one can use TEST-INIT and TPS-TEST.

SEE ALSO: MODES-GWFFS, TEST-INIT, TPS-TEST, ADD-GOODMODES, REMOVE-GOODMODES It takes values of type MODES-GWFFS and belongs to subjects `MAINTAIN`. The default value is:

EMPTYGOODMODES

INIT-DIALOGUE If T, the value of INIT-DIALOGUE-FN will be called on startup after the INI file has been read and the terminal is initialized. It takes values of type BOOLEAN and belongs to subjects `MAINTAIN`. The default value is:

NIL

INIT-DIALOGUE-FN The value of this flag is a function of no arguments, which will be called after the INI file has been read, if the flag INIT-DIALOGUE is T. It may be used to set the terminal type correctly, load some libraries, if the user wishes, or even decide between expert and non-expert modes. The default function does nothing; the function INIT-DEFINE-MY-DEFAULT-MODE defines a mode called MY-DEFAULT-MODE containing the state of all the system's flags at the point immediately after the INI file is read. It takes values of type ANYTHING and belongs to subjects `MAINTAIN`. The default value is:

INIT−DIALOGUE−DEFAULT−FN

JAVA-COMM How to start the Tps java interface.

An example for Unix is cd /home/theorem/tps/java ; java TpsStart

An example for Windows is java -classpath C:\TPS\java\TpsStart It takes values of type STRING and belongs to subjects `MAINTAIN`. The default value is:

""

LISP-IMPLEMENTATION-TYPE Tells what Common Lisp we are running on. Initialized when TPS starts up. Can't be changed. It takes values of type STRING and belongs to subjects `SYSTEM`. The default value is:

""

LOAD-WARN-P If T, library files will be checked while building the library master index; also, warning messages will be printed when redefining TPS-objects while loading a file or fetching library objects. It takes values of type BOOLEAN and belongs to subjects `MAINTAIN`. The default value is:

T

MACHINE-INSTANCE Tells what particular machine we are running on. Initialized when TPS starts up. Can't be changed. It takes values of type STRING and belongs to subjects `SYSTEM`. The default value is:

""

MACHINE-TYPE Tells what hardware that we are running on. Initialized when TPS starts up. Can't be changed. It takes values of type STRING and belongs to subjects `SYSTEM`. The default value is:

""

NEWS-DIR The directory with the NEWS and NOTE files. It takes values of type DIRSPEC and belongs to subjects `MAINTAIN`. The default value is:

""

READ-LLOAD-SOURCES-P If T while LLoading, one can later Ledit compiled functions. It takes values of type BOOLEAN and belongs to subjects `MAINTAIN`. The default value is:

T

SAVE-FILE The name of the file in which to save the core-image for TPS3. It takes values of type FILESPEC and belongs to subjects `MAINTAIN`. The default value is:

"tps3.exe"

SHORT-SITE-NAME Tells what site we are running at. Initialized when TPS starts up. Can't be changed. It takes values of type STRING and belongs to subjects `SYSTEM`. The default value is:

""

SOURCE-EXTENSION The extensions (:type) of source files in TPS3. It takes values of type STRING and belongs to subjects `MAINTAIN`. The default value is:

```
"lisp"
```

SOURCE-PATH A list of pathnames with source files for TPS3. It takes values of type DIRSPECLIST and belongs to subjects `MAINTAIN`. The default value is:

```
()
```

TEST-MODIFY A string which will be evaluated in exactly the same way as an alias. May contain any valid lisp commands, and will be evaluated after setting the mode during tps-test. So, for example, setting it to "(set-flag 'skolem-default nil) (when search-time-limit (setq search-time-limit (* 2 search-time-limit))) (when max-search-limit (setq max-search-limit (* 2 max-search-limit)))" would make tps-test changed SKOLEM-DEFAULT to NIL and double the time limits before each search. It takes values of type STRING and belongs to subjects `MAINTAIN`. The default value is:

```
""
```

TEST-THEOREMS A list of pairs; the first of each pair is the name of a theorem; the second is the name of a mode. If the mode name is NIL, TPS will attempt to choose a mode from the list of best modes in the library. This flag is used by the command TPS-TEST, and can be set automatically by the command TEST-INIT.

The default setting is a sample list of two standard TPS exercises, both to be run in mode ML (also standard in TPS). If you set this flag yourself, beware of unexported symbols — which is to say, make sure that the symbols you use are all in the USER package (this is particularly necessary if you are using library theorems which are not yet loaded into TPS, or they may end up interned in the wrong package). If in doubt, put "USER::" before all symbols, thus:

(setq test-theorems '((cl-user::thm30 . cl-user::mode-thm30) (cl-user::x2112 . cl-user::ml)))

You can use the flag TEST-MODIFY to alter modes on the fly as TPS-TEST runs. See the help messages for TEST-INIT and TEST-MODIFY for more information. It takes values of type SYMBOLPAIRLIST and belongs to subjects `MAINTAIN`. The default value is:

```
((X2106  ML) (X2108  ML))
```

28.55 Rules object

BUILD-MATCH If T, <rule>-MATCH functions for use with SUGGEST will be built. It takes values of type BOOLEAN and belongs to subjects `RULES-PACK`. The default value is:

T

HLINE-JUSTIFICATION The justification for hlines, if TREAT-HLINES-AS-DLINES is NIL. It takes values of type STRING and belongs to subjects `RULES-OBJECT`. The default value is:

"Hyp"

TREAT-HLINES-AS-DLINES If T, hlines may have multiple hypotheses and a justification, if NIL, hlines can only have one hypothesis (itself) and 'Hyps' as justification. It takes values of type BOOLEAN and belongs to subjects `RULES-OBJECT`. The default value is:

T

28.56 Unclassified

MAX-SUBSTS-PROJ The total number of projection substitutions allowed for any given variable. See also MAX-SUBSTS-VAR and MAX-SUBSTS-PROJ-TOTAL. This applies to higher-order unification (UN88 or UN90) only. It takes values of type NULL-OR-INTEGER and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

NIL

MAX-SUBSTS-PROJ-TOTAL The total number of projection substitutions allowed for any given dpairset. See also MAX-SUBSTS-VAR and MAX-SUBSTS-PROJ. This applies to higher-order unification (UN88 or UN90) only. It takes values of type NULL-OR-INTEGER and belongs to subjects `TRANSMIT`, `UNIFICATION`. The default value is:

NIL

MAX-SUBSTS-QUICK When NIL, quick unification is governed by the MIN-QUICK-DEPTH flag, and only minimal amounts of MAX-SUBSTS checking are done during quick unification. When MIN-SUBSTS-QUICK is a positive integer, quick unification (i.e. partial unification of a possible connection) is considered as a special case of normal unification, with MAX-SUBSTS-VAR temporarily equal to the value of MAX-SUBSTS-QUICK. When MIN-SUBSTS-QUICK is 0, quick unification goes down as far as it can until it is forced to either branch or violate MAX-SUBSTS-VAR. (This is almost equivalent to MAX-SUBSTS-QUICK NIL and MIN-QUICK-DEPTH 1.)

Note: non-NIL values of MAX-SUBSTS-QUICK only take effect if MAX-SUBSTS-VAR is also non-NIL. In this case, other flags will also be affected, as follows: APPLY-MATCH will be ignored (the matching routine that is used will be a variant of APPLY-MATCH-ALL-FRDPAIRS)

COUNTSUBS-FIRST and STOP-AT-TSN will be T. SUBSUMPTION-CHECK, UNIF-COUNTER and UNIF-TRIGGER will be NIL. UNI-SEARCH-HEURISTIC will be BREADTH-FIRST. MIN-QUICK-DEPTH and MAX-UTREE-DEPTH will be ignored. It takes values of type NULL-OR-INTEGER and belongs to subjects `TRANSMIT`, `MS98-1`, `IMPORTANT`, `UNIFICATION`. The default value is:

NIL

MAX-SUBSTS-VAR The maximum number of substitutions allowed for any given free variable in a dpairset. This is cumulative (i.e. if an old variable f is replaced by h1, which is in turn replaced by h2, that counts as two substitutions for f). Only projections or imitations are counted; eliminating substitutions are not. See also MAX-SUBSTS-PROJ and MAX-SUBSTS-PROJ-TOTAL. This applies to higher-order unification (UN88 or UN90) only. It takes values of type NULL-OR-INTEGER and belongs to subjects `TRANSMIT`, `MS98-1`, `IMPORTANT`, `UNIFICATION`. The default value is:

NIL

NUM-OF-DUPS Max number of duplications allowed on any path in search procedures using path-focused duplication. This flag may be set to 0. It takes values of type INTEGER+ and belongs to subjects `TRANSMIT`, `MS98-1`, `MS93-1`, `MS92-9`, `MS91-7`, `MS90-9`, `MS90-3`, `MATING-SEARCH`, `IMPORTANT`. The default value is:

2

PRIMSUB-VAR-SELECT If T, primsubs will only be applied to those variables which occur both negatively and positively as the head variable of some leaves in the current eproof. If NIL, primsubs will be applied to any variable which occurs either negatively or positively or both, anywhere It takes values of type BOOLEAN and belongs to subjects `TRANSMIT`, `PRIMSUBS`. The default value is:

T

28.57 Library

ADD-SUBDIRECTORIES When restoring the library index, search the directories in DEFAULT-LIB-DIR and BACKUP-LIB-DIR for subdirectories which also contain library files, and add these to the flags. This flag only works for Allegro, CMU, Kyoto and Lucid Common Lisps. It takes values of type BOOLEAN and belongs to subjects `LIBRARY`. The default value is:

T

BACKUP-LIB-DIR The list of all backup directories of library files. These should be directories to which the user has read access. No attempt will be made to write to a directory on this list. See also DEFAULT-LIB-DIR and SHOW-ALL-LIBOBJECTS. It takes values of type DIRSPECLIST and belongs to subjects `LIBRARY`. The default value is:

```
()
```

DEFAULT-LIB-DIR The list of writeable directories containing library files. All of the directories in this list ought to be library directories to which the user has write access. See also BACKUP-LIB-DIR and SHOW-ALL-LIBOBJECTS. It takes values of type DIRSPECLIST and belongs to subjects `LIBRARY`. The default value is:

```
()
```

DEFAULT-LIBFILE-TYPE The default value for the extension of library files. It takes values of type STRING and belongs to subjects `LIBRARY`. The default value is:

```
"lib"
```

DEFAULT-LIBINDEX-TYPE The default value for the extension of library index files. It takes values of type STRING and belongs to subjects `LIBRARY`. The default value is:

```
"rec"
```

LIB-BESTMODE-FILE Name of the file containing best modes for the theorems in the library. It takes values of type FILESPEC and belongs to subjects `LIBRARY`. The default value is:

```
"bestmodes.rec"
```

LIB-KEYWORD-FILE Name of the file containing acceptable keywords for the library. It takes values of type FILESPEC and belongs to subjects `LIBRARY`. The default value is:

```
"keywords.rec"
```

LIB-MASTERINDEX-FILE Name of the file containing index of entries in the library. It takes values of type FILESPEC and belongs to subjects `LIBRARY`. The default value is:

```
"libindex.rec"
```

RECORDFLAGS List of flags to be saved when using the mateop DATEREC. It takes values of type TPSFLAGLIST and belongs to subjects `MATING-SEARCH`, `LIBRARY`. The default value is:

```
()
```

REMOVE-TRAILING-DIR If T, the parts of the directory specification that are the same for all library files will be removed before printing. If NIL, the full directory will be printed. It takes values of type BOOLEAN and belongs to subjects `LIBRARY`. The default value is:

T

SHOW-ALL-LIBOBJECTS When loading an object, if there are multiple objects of that name and type, when NIL then accept the first object found (searching DEFAULT-LIB-DIR and then BACKUP-LIB-DIR in order). When T, show a list of all the objects and ask the user to choose. It takes values of type BOOLEAN and belongs to subjects `LIBRARY`. The default value is:

T

28.58 Editing

AUTO-KEYWORDS If T, keywords will automatically be generated and attached to the library object. However, setting auto-keywords to T requires expanding all definitions, which can take an enormous amount of time when definitions are deeply nested. It takes values of type BOOLEAN and belongs to subjects `LIBRARY`. The default value is:

NIL

AUTO-LIB-DIR A writeable directory containing library files, used for automatic library insertion. See the LIBRARY command INSERT-TPTP and INSERT-TPTP* It takes values of type DIRSPEC and belongs to subjects `LIBRARY`. The default value is:

""

28.59 Library Classification

CLASS-DIRECTION Suppose A is a class with child class B. If the value of CLASS-DIRECTION is Up, we think of B as depending on A (eg, A could be GROUPS and B could be FIELDS). If the value of CLASS-DIRECTION is Down, we think of A as depending on B (eg, B could be GROUPS and A could be FIELDS).

The value of this flag affects the behavior of CLASSIFY-ITEM and FETCH-CLASS*.

See Also: CLASSIFY-ITEM, FETCH-CLASS*, FETCH-UP, FETCH-DOWN It takes values of type UPDOWN and belongs to subjects `LIBRARY`. The default value is:

Down

CLASS-SCHEME The classification scheme used to organize the library interface. A classification scheme is a way of organizing library items into a tree (actually a directed acyclic graph) of classes. Each class can have classes as children. Each class has associated libitems.

See Also: CREATE-CLASS-SCHEME, PSCHEMES, PCLASS-SCHEME-TREE, PCLASS-TREE, CREATE-LIBCLASS, CLASSIFY-CLASS, CLASSIFY-ITEM, FETCH-LIBCLASS, FETCH-LIBCLASS*, FETCH-UP, FETCH-DOWN, GOTO-CLASS, ROOT-CLASS It takes values of type SYMBOL and belongs to subjects `LIBRARY`. The default value is:

LIBDIR

28.60 Bugs

DEFAULT-BUG-DIR If USE-DEFAULT-BUG-DIR is T, this is the default value for the directory where bugs generated by BUG-SAVE will be stored, and the first directory that will be searched by BUG-RESTORE. If USE-DEFAULT-BUG-DIR is NIL, this flag is ignored, and bugs will be saved like normal library objects, in the directories listed in DEFAULT-LIB-DIR. It takes values of type DIRSPEC and belongs to subjects `LIBRARY`. The default value is:

""

USE-DEFAULT-BUG-DIR Determines whether or not to use the directory given by DEFAULT-BUG-DIR for saving. If T, bugs are saved to and restored from DEFAULT-BUG-DIR, otherwise they aren't. See DEFAULT-BUG-DIR. It takes values of type BOOLEAN and belongs to subjects `LIBRARY`. The default value is:

T

Chapter 29

Modes

The internal name of this category is FLAG-MODE.
A mode can be defined using DEFMODE. Allowable properties are: `FLAG-SETTINGS`, `MHELP`.

29.1 Collecting Help

SCRIBE-DOC Mode used for producing documentation in Scribe. The settings of the flags are:

ALLSCOPEFLAG NIL

ATOMVALFLAG NIL

DISPLAYWFF NIL

FIRST-ORDER-PRINT-MODE NIL

FLUSHLEFTFLAG NIL

LEFTMARGIN 0

LOCALLEFTFLAG NIL

PPWFFLAG NIL

PRINTDEPTH 0

PRINTTYPES T

RIGHTMARGIN 70

SCOPE NIL

STYLE SCRIBE

SCRIBE-DOC-FIRST-ORDER Mode used for producing documentation in Scribe in first-order mode. The settings of the flags are:

ALLSCOPEFLAG NIL

ATOMVALFLAG NIL

DISPLAYWFF NIL

FIRST-ORDER-PRINT-MODE T

FLUSHLEFTFLAG NIL

LEFTMARGIN 0

LOCALLEFTFLAG NIL

PPWFFLAG NIL

PRINTDEPTH 0

PRINTTYPES NIL

RIGHTMARGIN 70

SCOPE NIL

STYLE SCRIBE

29.2 OTL Object

RULES Set flags so that the rules package can be run successfully. The settings of the flags are:

FIRST-ORDER-MODE-PARSE NIL

MAKE-WFFOPS-LABELS T

SCRIBE-OTL Mode used for printing proofs in Scribe. The settings of the flags are:

ALLSCOPEFLAG NIL

ATOMVALFLAG NIL

DISPLAYWFF NIL

FLUSHLEFTFLAG NIL

LEFTMARGIN 0

LOCALLEFTFLAG NIL

PPWFFLAG T

PRINTDEPTH 0

RIGHTMARGIN 70

SCOPE NIL

STYLE SCRIBE

TEX-1-OTL mode used for printing proofs in tex. The settings of the flags are:

ALLSCOPEFLAG NIL

ATOMVALFLAG NIL

DISPLAYWFF NIL

FLUSHLEFTFLAG NIL

LEFTMARGIN 0

LOCALLEFTFLAG NIL

PPWFFLAG T

PRINTDEPTH 0

RIGHTMARGIN 85

SCOPE NIL

STYLE TEX-1

TEX-OTL mode used for printing proofs in tex. The settings of the flags are:

ALLSCOPEFLAG NIL

ATOMVALFLAG NIL

DISPLAYWFF NIL

FLUSHLEFTFLAG NIL

LEFTMARGIN 0

LOCALLEFTFLAG NIL

PPWFFLAG T

PRINTDEPTH 0

RIGHTMARGIN 70

SCOPE NIL

STYLE TEX

29.3 Printing

RE-READ Used when writing out wffs to a file in such a way that they may be read back in and parsed correctly in higher-order mode. The settings of the flags are:

PRINT-META NIL

ATOMVALFLAG NIL

DISPLAYWFF NIL

FIRST-ORDER-PRINT-MODE NIL

LEFTMARGIN 1

PPWFFLAG T

PRINTDEPTH 0

PRINTTYPES T

PRINTTYPES-ALL T

RIGHTMARGIN 78

SCOPE NIL

STYLE GENERIC-STRING

29.4 Recording

SCRIBE-EDWFF Mode used for writing formulas from the editor. The settings of the flags are:

ALLSCOPEFLAG NIL

ATOMVALFLAG NIL

DISPLAYWFF T

FIRST-ORDER-PRINT-MODE NIL

FLUSHLEFTFLAG NIL

LEFTMARGIN 0

LOCALLEFTFLAG NIL

PPWFFLAG T

PRINTDEPTH 0

PRINTTYPES T

RIGHTMARGIN 70

SCOPE NIL

STYLE SCRIBE

SCRIBE-MATEWFF Mode used for writing formulas from mating search. The settings of the flags are:

ALLSCOPEFLAG NIL

ATOMVALFLAG NIL

DISPLAYWFF T

FIRST-ORDER-PRINT-MODE NIL

FLUSHLEFTFLAG NIL

LEFTMARGIN 0

LOCALLEFTFLAG NIL

PPWFFLAG T

PRINTDEPTH 0

PRINTTYPES T

RIGHTMARGIN 70

SCOPE NIL

STYLE SCRIBE

29.5 Expansion Trees

NAIVE Sets flags so all definitions and equalities will be rewritten, skolemizing will be done using SK1, but equalities will be rewritten using the Leibniz definition. The settings of the flags are:

SKOLEM-DEFAULT SK1

REWRITE-DEFNS '(LAZY1)

REWRITE-EQUALITIES 'LEIBNIZ

REMOVE-LEIBNIZ T

MIN-QUANTIFIER-SCOPE NIL

USE-RULEP NIL

USE-SYMSIMP NIL

29.6 MS91-6 and MS91-7 search procedures

MS91-DEEP Generates one new option set at a time and accepts it, irrespective of its weight. Does not generate option sets with ordinary duplications (i.e. duplications not used by a primsub), nor sets with multiple primsubs for the same variable; will instead generate recursive substitutions (i.e. will substitute for the expansion variables introduced by the first lot of substitutions). Does not set the PRIMSUBS flags. The settings of the flags are:

MS91-WEIGHT-LIMIT-RANGE INFINITY

NEW-OPTION-SET-LIMIT 1

WEIGHT-A-COEFFICIENT 0

WEIGHT-B-COEFFICIENT 1

WEIGHT-C-COEFFICIENT 0

WEIGHT-B-FN ALL-PENALTIES-FN

RECONSIDER-FN INF-WEIGHT

PENALTY-FOR-EACH-PRIMSUB 3

PENALTY-FOR-MULTIPLE-PRIMSUBS 5

PENALTY-FOR-MULTIPLE-SUBS INFINITY

PENALTY-FOR-ORDINARY-DUP INFINITY

OPTIONS-GENERATE-FN ADD-OPTIONS-ORIGINAL

OPTIONS-GENERATE-ARG 75

OPTIONS-GENERATE-UPDATE IDENT-ARG

MS91-NODUPS Generates one new option set at a time and accepts it, irrespective of its weight. Does not generate option sets with ordinary duplications (i.e. duplications not used by a primsub). Does not set the PRIMSUBS flags. The settings of the flags are:

MS91-WEIGHT-LIMIT-RANGE INFINITY

NEW-OPTION-SET-LIMIT 1

WEIGHT-A-COEFFICIENT 0

WEIGHT-B-COEFFICIENT 1

WEIGHT-C-COEFFICIENT 0

WEIGHT-B-FN ALL-PENALTIES-FN

RECONSIDER-FN INF-WEIGHT

PENALTY-FOR-EACH-PRIMSUB 3

PENALTY-FOR-MULTIPLE-PRIMSUBS 5

PENALTY-FOR-MULTIPLE-SUBS 5

PENALTY-FOR-ORDINARY-DUP INFINITY

OPTIONS-GENERATE-FN ADD-OPTIONS-ORIGINAL

OPTIONS-GENERATE-ARG 75

OPTIONS-GENERATE-UPDATE IDENT-ARG

MS91-ORIGINAL The original flag settings. Does not set the PRIMSUBS flags. The settings of the flags are:

MS91-WEIGHT-LIMIT-RANGE 3

NEW-OPTION-SET-LIMIT 5

WEIGHT-A-COEFFICIENT 1

WEIGHT-B-COEFFICIENT 1

WEIGHT-C-COEFFICIENT 1

WEIGHT-A-FN EXPANSION-LEVEL-WEIGHT-A

WEIGHT-B-FN SIMPLE-WEIGHT-B-FN

WEIGHT-C-FN OPTION-SET-NUM-LEAVES

RECONSIDER-FN INF-WEIGHT

PENALTY-FOR-EACH-PRIMSUB 3

PENALTY-FOR-MULTIPLE-PRIMSUBS 5

PENALTY-FOR-MULTIPLE-SUBS 5

OPTIONS-GENERATE-FN ADD-OPTIONS-ORIGINAL

OPTIONS-GENERATE-ARG 75

OPTIONS-GENERATE-UPDATE IDENT-ARG

MS91-SIMPLEST Generates option sets in the simplest possible order, in batches of five. Does not set the PRIMSUBS flags. The settings of the flags are:

MS91-WEIGHT-LIMIT-RANGE 1

WEIGHT-A-COEFFICIENT 0

WEIGHT-B-COEFFICIENT 1

WEIGHT-C-COEFFICIENT 0

WEIGHT-B-FN SIMPLEST-WEIGHT-B-FN

RECONSIDER-FN INF-WEIGHT

NEW-OPTION-SET-LIMIT 5

OPTIONS-GENERATE-FN ADD-OPTIONS-ORIGINAL

OPTIONS-GENERATE-ARG 75

OPTIONS-GENERATE-UPDATE IDENT-ARG

29.7 wff Primitives

FIRST-ORDER Puts parser and printer into first-order mode. The settings of the flags are:

FIRST-ORDER-MODE-PARSE T

TYPE-IOTA-MODE T

FIRST-ORDER-PRINT-MODE T

PRINTTYPES NIL

HIGHER-ORDER Puts parser and printer into higher-order mode. The settings of the flags are:

FIRST-ORDER-MODE-PARSE NIL

FIRST-ORDER-PRINT-MODE NIL

PRINTTYPES T

29.8 Maintenance

QUIET Turn off all output that can be turned off, without affecting search at all. Should make most other modes run a bit faster. The settings of the flags are:

PRINTLINEFLAG NIL

UNIFY-VERBOSE SILENT

MATING-VERBOSE SILENT

ETREE-NAT-VERBOSE NIL

MS98-VERBOSE NIL

TACTIC-VERBOSE MIN

OPTIONS-VERBOSE NIL

LOAD-WARN-P NIL

29.9 Unclassified

MATH-LOGIC-2-MODE Mode to be used for Math Logic II. The settings of the flags are:

FIRST-ORDER-MODE-PARSE NIL

TYPE-IOTA-MODE T

FIRST-ORDER-PRINT-MODE NIL

PRINTTYPES T

TREAT-HLINES-AS-DLINES T

DEFAULT-WFFEQ WFFEQ-AB

ML Puts parser and printer into higher-order mode for Lisp package ML. The settings of the flags are:

FIRST-ORDER-MODE-PARSE NIL

FIRST-ORDER-PRINT-MODE NIL

TYPE-IOTA-MODE T

BASE-TYPE I

PRINTTYPES T

MSV-OFF Turn off all of the MAX-SUBSTS-* routines. The settings of the flags are:

MAX-SUBSTS-VAR NIL

MAX-SUBSTS-PROJ-TOTAL NIL

MAX-SUBSTS-PROJ NIL

MAX-SUBSTS-QUICK NIL

APPLY-MATCH 'APPLY-MATCH-ALL-FRDPAIRS

MSV-ON Turn on the MAX-SUBSTS-* routines and increase the unification depths to infinity. The settings of the flags are:

MAX-SUBSTS-VAR 5

MAX-SUBSTS-PROJ-TOTAL NIL

MAX-SUBSTS-PROJ NIL

MAX-SUBSTS-QUICK 5

APPLY-MATCH 'APPLY-MATCH-ALL-FRDPAIRS

MAX-UTREE-DEPTH NIL

MAX-SEARCH-DEPTH NIL

MIN-QUICK-DEPTH NIL

Chapter 30

Grader Commands

The internal name of this category is GEXPR.
A Grader Command can be defined using DEFGEXPR. Allowable properties are: `ARGTYPES`, `ARGNAMES`, `ARGHELP`, `MAINFNS`, `PRINT-COMMAND`, `DONT-RESTORE`, `MHELP`.

30.1 Getting Out and Help

GR-EXIT Leave GRADING PACKAGE, and exit TPS.

GR-LEAVE Leave GRADING PACKAGE to the next enclosing top level.

LEAVE Leave GRADING PACKAGE to the next enclosing top level.

30.2 Variables

CHG-VARS Change the values of various variables.

GR-REVIEW Enter REVIEW to change VARIABLES.

30.3 The Grade-File

CREATE-GRADEFILE Create a new grade file.

30.4 Manual Grades

ALTER-GRADE Change the existing grades of some students.

INSERT-GRADES Insert one or more grades in the grade file.

LATE-EXERCISES Use this command to keep track of students who submit late assignments.

MODIFY-GRADE Change the existing grades of some students.

RESUME-INSERT-GRADES Resume entering grades from a previously interrupted session.

30.5 Automatic Grades

DUE-DATES Assign due-dates to exercises.

ETPS-GRADE Copy grades from ETPS record file to GRADE FILE.

30.6 The Class List

ADD-STUDENTS Insert students in the grade file.

DELETE-STUDENT Delete some students from the grade file.

30.7 Making the Output Convenient

ALIASES Assign actual names to exercises. The teacher may use short names for the assignments (to obtain a display which can fit on paper), and use this function to keep track of their actual names.

CHANGE-SEQUENCE change the sequence of assignments

COMMENT To insert comments in the grade file.

30.8 Generating Values

STATISTICS Compute statistical data.

30.9 Displaying Information

DISPLAY Display student-grades on the terminal.

INFO-EXERCISES Display aliases, penalty-fns, statistical data, weight, and due-dates for the exercises on the terminal.

NUMBER-OF-STUDENTS Use this command to find the number of students in the grade-file

30.10 Totaling

CALCULATE-GRADE Compute totals.

CHANGE-WEIGHT Change existing weighting factors.

PENALTY-FNS Assign penalty functions for various exercises.

30.11 Sorting

SORT-FN Sort the grades.

30.12 Letter-Grades

LETTER-GRADE Assign letter grades.

Chapter 31

Events

The internal name of this category is EVENT.
An event can be defined using DEFEVENT. Allowable properties are: `EVENT-ARGS`, `TEMPLATE`, `TEMPLATE-NAMES`, `WRITE-WHEN`, `WRITE-FILE`, `SIGNAL-HOOK`, `WRITE-HOOK`, `MHELP`.

31.1 MS88 search procedure

ADDED-CONN Event which is signalled whenever a connection is added to a mating.

CONSIDERED-CONN Event which is signalled whenever a connection is considered.

DUPE Event which is signalled whenever a variable duplication is done in a mating.

DUPE-VAR Event which is signalled whenever a variable is duplicated.

INCOMP-MATING Event which is signalled whenever an incompatible mating is found.

MATE-SUBSUMED-TEST Event which is signalled whenever a mating is tested for subsumption.

MATE-SUBSUMED-TRUE Event which is signalled whenever a mating is subsumed by an incompatible mating.

MATING-CHANGED Event which is signalled whenever a different mating is considered.

PRIMSUB Event which is signalled whenever a primitive substitution is applied to an expansion tree.

REMOVED-CONN Event which is signalled whenever a connection is removed from a mating.

START-TIME Event which is signalled whenever a mating should have its run time started, such as when it becomes the active mating.

STOP-TIME Event which is signalled whenever a mating should have its run time stopped, such as when it is no longer the active mating.

UNIF-SUBSUMED-TEST Event which is signalled whenever a set of disagreement pairs unification is tested for subsumption.

UNIF-SUBSUMED-TRUE Event which is signalled whenever a set of disagreement pairs is found to be subsumed by an ununifiable set.

31.2 Events

ADVICE-ASKED Event of user asking for advice.

COMMAND Event of user issuing a command.

DONE-EXC The event of completing an exercise.

ERROR The event of a Lisp Error.

INPUT-ERROR Event of illegal input caught by TPS.

PROOF-ACTION The event of completing any proof.

RULE-ERROR Event of illegal rule applications caught by TPS.

Chapter 32

Styles

The internal name of this category is DEVICE-STYLE.
A style can be defined using DEFSTYLE. Allowable properties are: `PRINT-SYMBOL`, `PRINT-SPACE-P`, `TERPRI-HEURISTICS`, `PRINT-TYPESYM`, `PRINT-TYPE-CHAR`, `PRINT-INDENT`, `PRINT-TAB`, `PRINT-NEXTPAR`, `PRINT-LINE`, `MARGIN-CORRECT`, `DISPLAY-PREFIX`, `DISPLAY-POSTFIX`, `BEGIN-ENVIRONMENT`, `END-ENVIRONMENT`, `TEXT-PREFIX`, `TEXT-POSTFIX`, `CHAR-CAT`, `MHELP`.

32.1 Review

GENERIC GENERIC stands for any terminal without special characters.

32.2 Concept

CONCEPT CONCEPT stands for any terminal without special characters.

CONCEPT-S CONCEPT-S stands for any CONCEPT terminal with special characters.

32.3 Printing

GENERIC-STRING GENERIC-STRING stands for re-readable string format. It is used in conjunction with the RE-READ mode.

ISTYLE ISTYLE stands for tps running with an interface.

SCRIBE SCRIBE stands for a file to be processed by SCRIBE before printing.

32.4 SAIL characters

SAIL SAIL stands for a file (or terminal) with SAIL characters.

32.5 TeX

TEX TEX stands for an output style to be run through TeX (or LaTeX, if the flag LATEX-EMULATION is set).

TEX-1 TEX-1 stands for an output style to be run through TeX (or LaTeX, if the flag LATEX-EMULATION is set).

32.6 X Windows

XTERM XTERM stands for a terminal running xterm with normal font vtsingle and bold font vtsymbold.

Index