MATHDIALOG LOGIC COMMANDS REFERENCE GUIDE v1.04

BY CARLOS E. FREITES

Remember that the goal is not just to prove theorems to the machine, the goal is to allow the machine to help another users understand your proof when they consult it. That is, in Mathdialog, formalization is very important by itself but it is also a means to able the machine to help users understand mathematics and the logical principles in which mathematics are founded. So, some logic commands may be logically redundant but not necessarily pedagogically redundant.

Proof: Is a sequence of logic commands written in the Mathdialog Command Window and sent one by one using the SEND COMMAND button or by the SHIFT-ENTER keys. Each sent command generate a copy of itself preceded by U, also a goal formula preceded by GL, or a hypothesis preceded by H, all of them written in the Blackboard Window. The logic command will always be written in the Blackboard, the former two will be written depending on the specific logic command sent. A proof will always start with a formula preceded by GL that will be our theorem thesis and a formula preceded by H, our theorem hypothesis.

Goal Formula: Is the formula, preceded by GL, we want to prove in a given stage inside a proof.

This manual will use the symbol | that doesn't belong to Mathdialog and is used to represent syntactical alternatives. For example PAIR_AXIOM[BG(TERM,{term_list}),CHECK|DEF_OF] means that PAIR_AXIOM[BG(TERM,{term_list}),CHECK] and PAIR_AXIOM[BG(TERM,{term_list}),DEF_OF] are both syntactically valid.

Example:
THEOR[POWER_P01;SET(B),BG(A,POWER(B));SUBSET(A,B)] Theorem written by the user in the Command Window and sent using the SEND COMMAND button or by the SHIFT-ENTER keys.

Written in the Blackboard by the system:

GL1 - SUBSET(A,B) The first goal formula is the thesis of our theorem.
H1 - SET(B) AND BG(A,POWER(B)) The first hypothesis is the one in our theorem.
U - BY_DEF_OBE[POWER(B)] The first logic command sent by the user.
H1 - FA(SET(z):BG(z,POWER(B)) <==> SUBSET(z,B)) Hypothesis generated by the previous logic command.
U - SUBST_UQV[FA(SET(z):BG(z,POWER(B)) <==> SUBSET(z,B)),A] Second logic command sent by the user.
H1 - BG(A,POWER(B)) <==> SUBSET(A,B) Hypothesis generated by the previous logic command.
U - PROP_CONS[SUBSET(A,B)] Third and last command command sent by the user. This is the last command because its argument match the thesis of our theorem (our main goal formula) and was successfully checked.
HD - SUBSET(A,B) “HD“ not followed by digits means that this is the thesis of our theorem so QED.

INDEX

AND_RED
Acronym of: AND REDuction
AND_RED[ ]

ASSUME
ASSUME[]

ATOMIC_OF
Acronym of: ATOMIC formula OF
ATOMIC_OF[Formula, Atomic Formula]

BY_CASES
Acronym of: BY CASES
BY_CASES[Formula]

BY_DEF_OBC
Acronym of: BY DEFinition of Object: Comprehension
BY_DEF_OBC[BG(Term1, Term2)]

BY_DEF_OBE
Acronym of: BY DEFinition of Object: Existential
BY_DEF_OBE[TERM]

BY_DEF_OF
Acronym of: BY DEFinition of this atomic formula
BY_DEF_OF[Atomic Formula]

BY_INDUCTION_ON
Acronym of: BY INDUCTION ON this variable
BY_INDUCTION_ON[EmptyString | BG(Variable,NATUR)]

BY_FUN_DEF
Acronym of: BY FUNction DEFinition
BY_FUN_DEF[G]

BY_THEOR
Acronym of: BY THEORem
BY_THEOR[Formula;Optional TermList;Optional AtomicFormulaList]

CONTRD_PROOF
Acronym of: CONTRaDiction PROOF
CONTRD_PROOF[Formula1, Formula2]

DEF_OF
Acronym of: DEFinition OF
DEF_OF[Atomic Formula]

EQ_RED_H
Acronym of: Existential Quantifier REDuction from Hypothesis
EQ_RED_H[Formula,Variable]

EQ_RED
Acronym of: Existential Quantifier REDduction
EQ_RED[Term]

EQUAL_EQUIV
Acronym of: EQUAL EQUIValence
EQUAL_EQUIV[Formula]

HYP_TOO
Acronym of: This is a HYPothesis TOO
HYP_TOO[Atomic Formula]

IF_RED
Acronym of: IF REDduction
IF_RED[ ]

IFF_RED_IF
Acronym of: IFF REDuction IF first
IFF_RED_IF[ ]

IP
Acronym of: start a Intermediate Proof
IP[Formula]

OR_RED
Acronym of: OR REDuction
OR_RED[Formula$_i$]

PAIR_AXIOM
Acronym of: by the PAIR AXIOM
PAIR_AXIOM[BG(TERM,{term_list}),CHECK|DEF_OF]

PROP_CONS
Acronym of: by PROPositional CONSequence
PROP_CONS[Formula]

QUANT_NEG
Acronym of: QUANTifier NEGation
QUANT_NEG[Formula]

SUBST_UQV
Acronym of: SUBSTitute Universal Quantified Variable
SUBST_UQV[Formula1,Term1]

UQ_RED
Acronym of: Universal Quantifier REDuction
UQ_RED[ ]

AND_RED

Acronym of: AND REDuction
AND_RED[ ]

Can be used when the goal formula Gformula has the form:

Formula$_1$ AND Formula$_2$ AND … AND Formula$_n$

and you want to prove each Formula$_i$ one by one.

The command sets

Formula$_1$

as the new goal formula. When this one is proved, the command sets

Formula$_2$

as the new goal formula. And so on until

Formula$_n$

is proven, then the command sets Gformula as a new hypothesis.

Example:
THEOR[ORD_PAIR_P01_AUX01;SET(x);EQ({x,x},{x})]

GL1 - EQ({x,x},{x})
H1 - SET(x)
U - BY_THEOR[EQ({x,x},{x}) <==> FA(BG(z,{x,x}):BG(z,{x})) AND FA(BG(z,{x}):BG(z,{x,x}));
{x},{x,x};]A02_EXTENSIONALITY_AXIOM
H1 - EQ({x,x},{x}) <==> FA(BG(z,{x,x}):BG(z,{x})) AND FA(BG(z,{x}):BG(z,{x,x}))
U - IP[FA(BG(z,{x,x}):BG(z,{x})) AND FA(BG(z,{x}):BG(z,{x,x}))]
GL1.1 - FA(BG(z,{x,x}):BG(z,{x})) AND FA(BG(z,{x}):BG(z,{x,x}))
U – AND_RED[]
GL1.1,1 – FA(BG(z,{x,x}):BG(z,{x})) First goal formula of AND_RED
U - UQ_RED[]
H1.1,1 - SET(z)
GL1.1,2 - BG(z,{x,x}) ==> BG(z,{x})
U - IF_RED[]
GL1.1,3 - BG(z,{x})
H1.1,3 - BG(z,{x,x})
U - PAIR_AXIOM[BG(z,{x,x}),DEF_OF]

H1.1,3 - EQ(z,x) OR EQ(z,x)
U - PAIR_AXIOM[BG(z,{x}),BY_DEF_OF]
H1.1,3 - BG(z,{x}) <==> EQ(z,x)
U - BY_CASES[EQ(z,x) OR EQ(z,x)]
GL1.1,3,1 - EQ(z,x) ==> BG(z,{x})
U - IF_RED[]
GL1.1,3,2 - BG(z,{x})
H1.1,3,2 - EQ(z,x)
U - PROP_CONS[BG(z,{x})]
GL1.1,4 - EQ(z,x) ==> BG(z,{x})
HD1.1,3,1 - BG(z,{x})
HD1.1,3 - EQ(z,x) ==> BG(z,{x})
U - PROP_CONS[EQ(z,x) ==> BG(z,{x})]
GL1.2 - FA(BG(z,{x}):BG(z,{x,x})) Second goal formula of AND_RED
HD1.1,3 - EQ(z,x) ==> BG(z,{x})
HD1.1,2 - BG(z,{x})
HD1.1,1 - BG(z,{x,x}) ==> BG(z,{x})
HD1.1 - FA(BG(z,{x,x}):BG(z,{x}))
U - UQ_RED[]
H1.2 - SET(z)
GL1.3 - BG(z,{x}) ==> BG(z,{x,x})
U - IF_RED[]
GL1.4 - BG(z,{x,x})
H1.4 - BG(z,{x})
U - PAIR_AXIOM[BG(z,{x}),DEF_OF]
H1.4 - EQ(z,x)
U - PAIR_AXIOM[BG(z,{x,x}),BY_DEF_OF]
H1.4 - BG(z,{x,x}) <==> EQ(z,x) OR EQ(z,x)
U - PROP_CONS[BG(z,{x,x})]
GL1 - EQ({x,x},{x})
HD1.3 - BG(z,{x,x})
HD1.2 - BG(z,{x}) ==> BG(z,{x,x})
HD1.1 - FA(BG(z,{x}):BG(z,{x,x}))
HD1 - FA(BG(z,{x,x}):BG(z,{x})) AND FA(BG(z,{x}):BG(z,{x,x})) Original Gformula set as hypothesis
U - PROP_CONS[EQ({x,x},{x})]
HD - EQ({x,x},{x})

ASSUME

ASSUME[]

It is used to assume the Mathdialog NUCLEUS axioms. Normal users don't have access to it.

ATOMIC_OF

Acronym of: ATOMIC formula OF
ATOMIC_OF[Formula, Atomic Formula]

Can be used when there is a hypothesis (Formula) that defines an Atomic Formula and you need it in your hypothesis too.

Example:
THEOR[EMPTY_P01;SET(X);SUBSET(EMPTY,X)]

GL1 - SUBSET(EMPTY,X)
H1 - SET(X)
U - IP[FA(BG(x,EMPTY):BG(x,X))]
GL1.1 - FA(BG(x,EMPTY):BG(x,X))
U - UQ_RED[]
H1.1 - SET(x)
GL1.2 - BG(x,EMPTY) ==> BG(x,X)
U - BY_THEOR[NOTBG(x,EMPTY)]
H1.2 - NOTBG(x,EMPTY)
U - DEF_OF[NOTBG(x,EMPTY)]
H1.2 - NOT(BG(x,EMPTY))
U - PROP_CONS[BG(x,EMPTY) ==> BG(x,X)]
GL1 - SUBSET(EMPTY,X)
HD1.1 - BG(x,EMPTY) ==> BG(x,X)
HD1 - **FA(BG(x,EMPTY):BG(x,X))** Hypothesis that defines SUBSET(EMPTY)
U - ATOMIC_OF[FA(BG(x,EMPTY):BG(x,X)),SUBSET(EMPTY,X)]
HD - SUBSET(EMPTY,X) Taken as hypothesis by ATOMIC_OF and match the goal formula so QED.

BY_CASES

Acronym of: BY CASES
BY_CASES[Formula]

Proves the current goal formula Gformula by cases using the available hypothesis Formula.
Formula must has the form:

Formula$_1$ OR Formula$_2$ OR … OR Formula$_n$

The command sets

Formula$_1$ ==> Gformula

as the new goal formula. When this one is proven, sets

Formula$_2$ ==> Gformula

as the new goal formula. And so on until

Formula$_n$ ==> Gformula

is proven, then the command sets Gformula as a new hypothesis.

Example:
THEOR[PAIR_P02;BG(x,A),BG(y,A);BG({x,y},POWER(A))]

GL1 - BG({x,y},POWER(A))
H1 - BG(x,A) AND BG(y,A)
U - BY_DEF_OBE[POWER(A)]
H1 - FA(SET(z):BG(z,POWER(A)) <==> SUBSET(z,A))
U - SUBST_UQV[FA(SET(z):BG(z,POWER(A)) <==> SUBSET(z,A)),{x,y}]
H1 - BG({x,y},POWER(A)) <==> SUBSET({x,y},A)
U - IP[SUBSET({x,y},A)]
GL1.1 - SUBSET({x,y},A)
U - BY_DEF_OF[SUBSET({x,y},A)]
H1.1 - SUBSET({x,y},A) <==> FA(BG(z,{x,y}):BG(z,A))
U - IP[FA(BG(z,{x,y}):BG(z,A))]
GL1.1.1 - FA(BG(z,{x,y}):BG(z,A))
U - UQ_RED[]
H1.1.1 - SET(z)
GL1.1.2 - BG(z,{x,y}) ==> BG(z,A)
U - IF_RED[]
GL1.1.3 - BG(z,A) Goal formula to prove BY_CASES
H1.1.3 - BG(z,{x,y})
U - PAIR_AXIOM[BG(z,{x,y}),DEF_OF]
H1.1.3 - EQ(z,y) OR EQ(z,x)
U - BY_CASES[EQ(z,y) OR EQ(z,x)]
GL1.1.3,1 - EQ(z,y) ==> BG(z,A) First goal formula of BY_CASES
U - IF_RED[]
GL1.1.3,2 - BG(z,A)
H1.1.3,2 - EQ(z,y)
U - EQUAL_EQUIV[BG(z,A)] This proves the first goal formula of BY_CASES
GL1.1.4 - EQ(z,x) ==> BG(z,A) Second goal formula of BY_CASES
HD1.1.3,1 - BG(z,A)
HD1.1.3 - EQ(z,y) ==> BG(z,A)
U - IF_RED[]
GL1.1.5 - BG(z,A)
H1.1.5 - EQ(z,x)
U - EQUAL_EQUIV[BG(z,A)]
GL1.1 - SUBSET({x,y},A)
HD1.1.4 - BG(z,A) Original goal formula GL1.1.3 is now assumed as hypothesis

HD1.1.3 - EQ(z,x) ==> BG(z,A)
HD1.1.2 - BG(z,A)
HD1.1.1 - BG(z,{x,y}) ==> BG(z,A)
HD1.1 - FA(BG(z,{x,y}):BG(z,A))
U - PROP_CONS[SUBSET({x,y},A)]
GL1 - BG({x,y},POWER(A))
HD1 - SUBSET({x,y},A)
U - PROP_CONS[BG({x,y},POWER(A))]
HD - BG({x,y},POWER(A))

BY_DEF_OBC

Acronym of: BY DEFinition of Object: Comprehension
BY_DEF_OBC[BG(Term1, Term2)]

Can be used when you need to make explicit as hypothesis the meaning of a formula like BG(Term1, Term2) where Term2 is like {BG(x,Term):Formula} (explicit or by definition of Term2) or like {BG(x3,Term3),...,BG(xn,Termn):Formula}.

If Term2 is like {BG(x,Term):Formula,}, the command sets

BG(Term1, Term2) <==> BG(Term1,Term) AND Formula

as a new hypothesis.

I the other case, the command sets

BG(Term1, Term2) <==> BG(x3,Term3) AND … AND BG(xn,Termn) AND EQ(Term1,(x3,...,xn)) AND Formula

as a new hypothesis.

Examples:

U - BY_DEF_OBC[BG(x,DOMAIN(F,A,B))]
H1 - BG(x,DOMAIN(F,A,B)) <==> BG(x,A) AND TE(BG(y,B):BG((x,y),F))

U - BY_DEF_OBC[BG(x,{BG(t,b):FA(BG(y,A):BG(t,y))})]
H1 - BG(x,{BG(t,b):FA(BG(y,A):BG(t,y))}) <==> BG(x,b) AND FA(BG(y,A):BG(x,y))

U - BY_DEF_OBC[BG((n1 * n2),NATUR)]
H1 - BG((n1 * n2),NATUR) <==> BG((n1 * n2),REALP) AND
BG((n1 * n2),INTERSECT(ALL_IND_SET_CF(REALS,+,*,0,1,REALP,0-)))

Notice that the constants NATUR, INTEG, REALS, REALP, etc are not explicitly defined in Mathdialog. The user must use this Logic Command.

BY_DEF_OBE

Acronym of: BY DEFinition of Object: Existential
BY_DEF_OBE[TERM]

Can be used when you need to use an existential definition. If TERM correspond to a uniqueness existential definition and TE!(BG(x,Y)|SET(x):Formula(x)) is the main formula of the theorem that justify that definition, then:

In the case TE!(BG(x,Y):Formula(x)), the command sets

BG(TERM,Y) AND Formula(TERM)

as a new hypothesis

In the case TE!(SET(x):Formula(x)), the command sets

Formula(TERM)

as a new hypothesis.

For the next example we will need this theorem:

THEOR[CART_PROD_EXIST;SET(A),SET(B);
TE!(SET(P):FA(SET(p): BG(p,P) <==> TE(BG(x,A):TE(BG(y,B):EQ(p,(x,y)))))))]

That is the justification of the existential definition:

DEF_OBE[CART_PROD;SET(A),SET(B);CART_PROD(A,B) = CART_PROD_EXIST]

Notice that this definition is totally tied to the theorem: Its hypotheses must literary or exactly match, even the variables names.

Example:

U - BY_DEF_OBE[CART_PROD(A,B)]
H1 - FA(SET(x):FA(SET(y):BG((x,y),CART_PROD(A,B)) <==> BG(x,A) AND BG(y,B)))

BY_DEF_OF

Acronym of: BY DEFinition of this atomic formula
BY_DEF_OF[Atomic Formula]

Can be used when you need the definition of the predicate Atomic Formula.

The command sets the definition of Atomic Formula as a new hypothesis.

Example:

U - BY_DEF_OF[NOTBG(x,I(s))]
H1 - NOTBG(x,I(s)) <==> NOT(BG(x,I(s)))

NOTE: The definition of NOTBG(x,A) is

DEF_PRED[NOTBG;SET(x),SET(A);NOTBG(x,A) <==> NOT(BG(x,A))]

BY_INDUCTION_ON

Acronym of: BY INDUCTION ON this variable
BY_INDUCTION_ON[EmptyString | BG(Variable,NATUR)]

Can be used to prove the current goal formula by induction.

If the current goal formula has the form:

FA(BG(Variable,NATUR):Formula(Variable))

The command sets

Formula(1)

as the new goal formula. When it is proved, sets

Formula(Variable) ==> Formula(Variable+1)

as the new goal formula. When it is proven, the command sets the original goal formula

FA(BG(Variable,NATUR):Formula(Variable))

as a new hypothesis.

If the current goal formula doesn't has the form:

FA(BG(Variable,NATUR):Formula(Variable))

but has the form:

Formula(Variable)

where over its argument Variable there is a hypothesis with form BG(Variable,NATUR)

sets Formula(1)

as the new goal formula. When it is proven, sets

Formula(Variable) ==> Formula(Variable+1)

as the new goal formula. When it is proven, the command sets the original goal formula

Formula(Variable)

as a new hypothesis.

Exmple:

.
.
.
GL1 - BG((n1 + n2),NATUR)
HD1.1 - BG(n2,NATUR)
HD1 - BG((n2 + 1),NATUR)
U - BY_INDUCTION_ON[BG(n2,NATUR)]
GL1,1 - BG((n1 + 1),NATUR) First goal formula of BY_INDUCTION_ON
U - PROP_CONS[BG((n1 + 1),NATUR)] This proves the first goal formula of BY_INDUCTION_ON
GL2 - BG((n1 + n2),NATUR) ==> BG((n1 + (n2 + 1)),NATUR) Second goal formula of BY_INDUCTION_ON
HD1 - BG((n1 + 1),NATUR) First goal of BY_INDUCTION_ON set as proven hypothesis
U – IF_RED[]
GL3 - BG((n1 + (n2 + 1)),NATUR)
H3 - BG((n1 + n2),NATUR)
.
.
.
U - EQUAL_EQUIV[BG((n1 + (n2 + 1)),NATUR)] This proves the second goal formula of BY_INDUCTION_ON
HD2 - BG((n1 + (n2 + 1)),NATUR)
HD1 - BG((n1 + n2),NATUR) ==> BG((n1 + (n2 + 1)),NATUR) This proves the second goal formula of BY_INDUCTION_ON
HD - BG((n1 + n2),NATUR) This is what BY_INDUCTION_ON has proved
.

.
.

BY_FUN_DEF

Acronym of: BY FUNction DEFinition
BY_FUN_DEF[G]

Can be uses to make explicit the setting of a variable G to the term with the form FUN(BG(a,C):TERM(a)) by using LET in the theorem's hypothesis.

The command sets the formula

BG(G,FUNCS_IN_TO_SET(C,REM(BG(a,C):TERM(a)))) AND FA(BG(a,C):EQ(G(a),TERM(a)) AND BG((a,TERM(a)),G))

as a new hypothesis.

Example:
THEOR[INVGR_P2;GROUP(G,@,e),BG(z,G):LET(I,FUN(BG(x,G):INVGR(G,@,e,x))); BG(I(z),DOM_PROJ1(@,G,G,G))]

GL1 - BG(I(z),DOM_PROJ1(@,G,G,G))
H1 - GROUP(G,@,e) AND BG(z,G) AND LET(I,FUN(BG(x,G):INVGR(G,@,e,x))) AND EQ(I,FUN(BG(x,G):INVGR(G,@,e,x)))
U - BY_FUN_DEF[I]
H1 - BG(I,FUNCS_IN_TO_SET(G,REM(BG(x,G):INVGR(G,@,e,x)))) AND FA(BG(x,G):EQ(I(x),INVGR(G,@,e,x)) AND BG((x,INVGR(G,@,e,x)),I))
U - PROP_CONS[FA(BG(x,G):EQ(I(x),INVGR(G,@,e,x)) AND BG((x,INVGR(G,@,e,x)),I))]
H1 - FA(BG(x,G):EQ(I(x),INVGR(G,@,e,x)) AND BG((x,INVGR(G,@,e,x)),I))
.
.
.
[the proof continue in the BY_THEOR logic command example]

BY_THEOR

Acronym of: BY THEORem
BY_THEOR[Formula;Optional TermList;Optional AtomicFormulaList]

Can be used when you want to use a formula Formula that is the main thesis of a theorem which hypotheses are a subset of the available hypotheses.

Sets Formula as a new hypothesis.

The collection of all formulated theorems is named general context. BY_THEOR looks the formula F in its argument in the general context for a theorem whose thesis match with F. Then verify that the theorem's hypotheses is a subset of the available hypotheses. Sometimes BY_THEOR cannot make this check for all its hypotheses, we must help it by telling it some atomic formulas are in the available hypotheses. This is done in AtomicFormulaList, it must be a list of atomic formulas separated by commas. Also some times, BY_THEOR can not match the thesis of a general context theorem M with F, if F has terms that are not variables in the position that M has variables. We must help it by telling what terms must be treated as variables. This is done in TermList, it must be a list of terms separated by commas.

Examples:

.
.
.
[This proof is the continuation from the BY_FUN_DEF Logic Command example]

U - SUBST_UQV[FA(BG(x,G):EQ(I(x),INVGR(G,@,e,x)) AND BG((x,INVGR(G,@,e,x)),I)),z]
GL1.1 - EQ(I(z),INVGR(G,@,e,z)) AND BG((z,INVGR(G,@,e,z)),I)
GL1.2 - BG(z,G)
U - PROP_CONS[BG(z,G)]
GL1 - BG(I(z),DOM_PROJ1(@,G,G,G))
HD1.1 - BG(z,G)
HD1 - EQ(I(z),INVGR(G,@,e,z)) AND BG((z,INVGR(G,@,e,z)),I)
U - BY_THEOR[BG(INVGR(G,@,e,z),G)]INV_GROUP_UNIQUENESS_EXIST_DEF_1
H1 - BG(INVGR(G,@,e,z),G)
U - BY_THEOR[BG(INVGR(G,@,e,z),DOM_PROJ1(@,G,G,G));INVGR(G,@,e,z);]BIN_OPER_P01
H1 - BG(INVGR(G,@,e,z),DOM_PROJ1(@,G,G,G))
U - EQUAL_EQUIV[BG(I(z),DOM_PROJ1(@,G,G,G))]
HD – BG(I(z),DOM_PROJ1(@,G,G,G))

THEOR[BIN_UNION_P04;**SET(C)**,SET(D);
FA(SET(z):BG(z,C) OR BG(z,D) <==> BG(z,BIN_UNION(C,D)))]

GL1 - FA(SET(z):BG(z,C) OR BG(z,D) <==> BG(z,BIN_UNION(C,D)))
H1 - SET(C) AND SET(D)
U - UQ_RED[]
H1 - SET(z)
GL2 - BG(z,C) OR BG(z,D) <==> BG(z,BIN_UNION(C,D))
U - IFF_RED_IF[]
GL2,1 - BG(z,C) OR BG(z,D) ==> BG(z,BIN_UNION(C,D))
U - IF_RED[]
GL2,2 - BG(z,BIN_UNION(C,D))
H2,2 - BG(z,C) OR BG(z,D)
U - BY_CASES[BG(z,C) OR BG(z,D)]
GL2,2,1 - BG(z,C) ==> BG(z,BIN_UNION(C,D))
U - IF_RED[]
GL2,2,2 - BG(z,BIN_UNION(C,D))
H2,2,2 - BG(z,C)
U - BY_THEOR[BG(z,BIN_UNION(C,D))]BIN_UNION_P02
GL2,3 - BG(z,D) ==> BG(z,BIN_UNION(C,D))
HD2,2,1 - BG(z,BIN_UNION(C,D))
HD2,2 - BG(z,C) ==> BG(z,BIN_UNION(C,D))
U - IF_RED[]
GL2,4 - BG(z,BIN_UNION(C,D))
H2,4 - BG(z,D)
U - BY_THEOR[BG(z,BIN_UNION(C,D))]BIN_UNION_P03
GL3 - BG(z,BIN_UNION(C,D)) ==> BG(z,C) OR BG(z,D)
HD2,3 - BG(z,BIN_UNION(C,D))
HD2,2 - BG(z,D) ==> BG(z,BIN_UNION(C,D))
HD2,1 - BG(z,BIN_UNION(C,D))
HD2 - BG(z,C) OR BG(z,D) ==> BG(z,BIN_UNION(C,D))
U - IF_RED[]
GL4 - BG(z,C) OR BG(z,D)
H4 - BG(z,BIN_UNION(C,D))
U - BY_THEOR[BG(z,C) OR BG(z,D);;SET(C)]BIN_UNION_P01

*Here, even without **SET(C)**, Mathdialog finds the theorem THEOR[BIN_UNION_P01; SET(C),SET(D),BG(z,BIN_UNION(C,D)); BG(z,C) OR BG(z,D)] but whitout **SET(C)**, it fail to realize SET(C) is in the proof hypotheses so we tell it 'hey! believe me SET(C) is in the proof, so go ahead". This look trivial and that may happen frequently, that is not the case: use BY_THEOR without optional arguments and if it does not works, use the optional arguments.*

HD3 - BG(z,C) OR BG(z,D)
HD2 - BG(z,BIN_UNION(C,D)) ==> BG(z,C) OR BG(z,D)
HD1 - BG(z,C) OR BG(z,D) <==> BG(z,BIN_UNION(C,D))
HD - FA(SET(z):BG(z,C) OR BG(z,D) <==> BG(z,BIN_UNION(C,D)))

THEOR[BIN_OPER_P05; BIN_OPER(F,G),BG(x,G),BG(y,G);BG((x F y),DOM_PROJ2(F,G,G,G))]

GL1 - BG((x F y),DOM_PROJ2(F,G,G,G))
H1 - BIN_OPER(F,G) AND BG(x,G) AND BG(y,G)
U - BY_THEOR[BG((x F y),G)]BIN_OPER_P03
H1 - BG((x F y),G)
U - BY_THEOR[BG((x F y),DOM_PROJ2(F,G,G,G));(x F y);]BIN_OPER_P02
Here, Mathdialog cannot match the BIN_OPER_P02's thesis BG(z,DOM_PROJ2(F,A,A,A)). In this case BG((x F y),DOM_PROJ2(F,G,G,G)) has the term (x F y) in the position that BG(z,DOM_PROJ2(F,A,A,A)) has the variable z. So we have to tell to Mathdialog that (x F y) must be taken as a variable. As in the last example, this not too frequent, so it is better to try BY_THEOR without optional arguments first.
HD - BG((x F y),DOM_PROJ2(F,G,G,G))

IMPORTANT

Notice that the labels INV_GROUP_UNIQUENESS_EXIST_DEF_1 and BIN_OPER_P01 are displayed by the system, not by the user and are the names of the used theorem in each case.

In the case of BIN_OPER_P01, that is the normal case, the theorem is:

THEOR[BIN_OPER_P01;BG(x,A),BIN_OPER(F,A); BG(x,DOM_PROJ1(F,A,A,A))]

INV_GROUP_UNIQUENESS_EXIST_DEF_1 is a special case because involves an existential definition. The theorem

THEOR[INV_GROUP_UNIQUENESS;GROUP(G,@,e),BG(x,G);TE!(BG(i,G):EQ(e,(i @ x)))]

formulate the existence of a unique set under the given hypotheses. To name that set, the following sentence was used:

DEF_OBE[INVGR;GROUP(G,@,e),BG(x,G);INVGR(G,@,e,x) = INV_GROUP_UNIQUENESS]

In the moment that definition was saved, two theorem were automatically build and saved:

THEOR[INV_GROUP_UNIQUENESS_EXIST_DEF_1;GROUP(G,@,e),BG(x,G);
BG(INVGR(G,@,e,x),G)

THEOR[INV_GROUP_UNIQUENESS_EXIST_DEF_0;GROUP(G,@,e),BG(x,G);
EQ(e,(INVGR(G,@,e,x) @ x))

CONTRD_PROOF

Acronym of: CONTRaDiction PROOF
CONTRD_PROOF[Formula1, Formula2]

Can be used to prove by contradiction the goal formula GFormula. Formula1 must be a negation of the GFormula. Formula2 must be a hypothesis (or a very simply propositional consequence of the available hypothesis).

The command sets

GFormula <==> NOT(Formula1)

as the new goal formula. When it is proved, sets

Formula1

as a new hypothesis and sets

Formula2 ==> BG(EMPTY,EMPTY) AND NOT(BG(EMPTY,EMPTY))

as the new goal formula. Notice that this is a contradiction because

BG(EMPTY,EMPTY) AND NOT(BG(EMPTY,EMPTY))

is a falsehood. When it is proved, sets the original goal formula

Gformula

as a new hypothesis

Example:
THEOR[SUBSET_P02;SUBSET(A,B),NOTBG(x,B);NOTBG(x,A)]

GL1 - NOTBG(x,A)
H1 - SUBSET(A,B) AND NOTBG(x,B)
U - CONTRD_PROOF[BG(x,A),NOTBG(x,B)]
GL1,1 - NOTBG(x,A) <==> NOT(BG(x,A))
U - BY_DEF_OF[NOTBG(x,A)] That is, by definition BG(x,A) is the negation of NOTBG(x,A)
GL2 - NOTBG(x,B) ==> BG(EMPTY,EMPTY) AND NOT(BG(EMPTY,EMPTY))
H2 – BG(x,A) The negation of our goal formula is taken as hypothesis
HD1 - NOTBG(x,A) <==> NOT(BG(x,A))
U - BY_THEOR[BG(x,B)]SUBSET_DIRECT_0
H2 - BG(x,B)
U - BY_DEF_OF[NOTBG(x,B)]
H2 - NOTBG(x,B) <==> NOT(BG(x,B))
U - **PROP_CONS[NOTBG(x,B) ==> BG(EMPTY,EMPTY) AND NOT(BG(EMPTY,EMPTY))]**
This proves our goal formula, the contradiction
HD1 - NOTBG(x,B) ==> BG(EMPTY,EMPTY) AND NOT(BG(EMPTY,EMPTY))
HD - NOTBG(x,A) The original goal formula is taken as hypothesis

DEF_OF

Acronym of: DEFinition OF
DEF_OF[Atomic Formula]

Can be used when the definition of an atomic formula (Atomic Formula) available in the current hypothesis is needed.

The command sets the definition of Atomic Formula as a new hypothesis.

Example:
THEOR[EMPTY_P01;SET(X);SUBSET(EMPTY,X)]

GL1 - SUBSET(EMPTY,X)
H1 - SET(X)
U - IP[FA(BG(x,EMPTY):BG(x,X))]
GL1.1 - FA(BG(x,EMPTY):BG(x,X))
U - UQ_RED[]
H1.1 - SET(x)
GL1.2 - BG(x,EMPTY) ==> BG(x,X)
U - BY_THEOR[NOTBG(x,EMPTY)]A01_EXISTENCE_AXIOM
H1.2 - NOTBG(x,EMPTY)
U - DEF_OF[NOTBG(x,EMPTY)]
H1.2 - NOT(BG(x,EMPTY))
U - PROP_CONS[BG(x,EMPTY) ==> BG(x,X)]
GL1 - SUBSET(EMPTY,X)
HD1.1 - BG(x,EMPTY) ==> BG(x,X)
HD1 - FA(BG(x,EMPTY):BG(x,X))
U - ATOMIC_OF[FA(BG(x,EMPTY):BG(x,X)),SUBSET(EMPTY,X)]
HD - SUBSET(EMPTY,X)

EQ_RED_H

Acronym of: Existential Quantifier REDuction from Hypothesis
EQ_RED_H[Formula,Variable]

Can be used to instantiate a free variable using a existential quantified formula available in the current hypotheses. Formula must be an available existential quantifier of the available hypothesis [that is Formula must has the form TE(SET(x) | BG(x,TERM):Formula2(x))] and Variable a free variable.

First, the command replace Variable by the existential quantifier bound variable and and eliminate it as follow:

SET(Variable) | BG(Variable,TERM) AND Formula2(Variable) and sets this formula a new hypothesis. If Formula is an uniqueness existential quantifier, then the following formula is set as a new hypothesis:

SET(Variable) | BG(Variable,TERM) AND Formula2(Variable) AND
TE(SET(fv) | BG(fv,TERM):Formula2(fv)) ==> EQ(Variable,fv))

where fv is a new free variable.

Example:
THEOR[INV_GROUP_UNIQUENESS;GROUP(G,@,e),BG(x,G);TE!(BG(i,G):EQ(e,(i @ x)))]

GL1 - TE!(BG(i,G):EQ(e,(i @ x)))
H1 - GROUP(G,@,e) AND BG(x,G)
U - BY_THEOR[TE(BG(y,G):EQ(e,(y @ x)))]GROUP_DIRECT_0
H1 - TE(BG(y,G):EQ(e,(y @ x)))
U - EQ_RED_H[TE(BG(y,G):EQ(e,(y @ x))),x']
H1 - BG(x',G) AND EQ(e,(x' @ x))
U - EQ_RED[x']
GL1,1 - BG(x',G)
U - PROP_CONS[BG(x',G)]
GL1,1 - EQ(e,(x' @ x))
U - PROP_CONS[EQ(e,(x' @ x))]
GL2 - EQ(x',a)
H2 - BG(a,G) AND EQ(e,(a @ x))
HD1 - EQ(e,(x' @ x))
U - EQUAL_EQUIV[EQ((x' @ x),(a @ x))]
H2 - EQ((x' @ x),(a @ x))
U - BY_THEOR[EQ((x' @ x),(a @ x)) ==> EQ(x',a)]GROUP_P04
H2 - EQ((x' @ x),(a @ x)) ==> EQ(x',a)
U - PROP_CONS[EQ(x',a)]
HD1 - EQ(x',a)
HD - TE!(BG(i,G):EQ(e,(i @ x)))

EQ_RED

Acronym of: Existential Quantifier REDduction
EQ_RED[Term]

Can be used to eliminate the existential quantifier if the current goal formula has that form. That is, if the current goal formula has the form TE(SET(x) | BG(x,TERM):Formula(x)). Term must be a valid term in the current context. It first replace Term for the quantifier bounded variable in SET(x) | BG(x,TERM), the result

BG(Term,TERM)

is set as the new goal formula. If SET(x) | BG(x,TERM) is SET(x) instead of BG(Term,TERM), then this step is omitted. When this one is proved,

Formula(Term)

is set as the new goal formula. When this one is proved, the original goal formula

TE(SET(x) | BG(x,TERM):Formula(x))

is set a new hypothesis.

In case the original goal formula is an uniqueness existential quantifier formula, that is if the current goal formula has the form TE!(SET(x) | BG(x,TERM):Formula(x)), then

TE(SET(x) | BG(x,TERM):Formula(x) ==> EQ(Term,x))

is set as the new goal formula. And when this one is proved, the original goal formula

TE!(SET(x) | BG(x,TERM):Formula(x))

is set a new hypothesis.

Example:

GL1 - TE!(BG(y,RANGE(F,A,B)):BG((x,y),F))
H1 - FUNC(F,A,B) AND BG(x,DOMAIN(F,A,B))
U - BY_THEOR[TE!(BG(y,B):BG((x,y),F))]FUNC_DIRECT_0
H1 - TE!(BG(y,B):BG((x,y),F))
U - EQ_RED_H[TE!(BG(y,B):BG((x,y),F)),b]
H1 - BG(b,B) AND BG((x,b),F) AND FA(BG(y,B):BG((x,y),F) ==> EQ(b,y))
U - EQ_RED[b]
GL1,1 – BG(b,RANGE(F,A,B)) First goal formula of EQ_RED[b]
U - BY_DEF_OBC[BG(b,RANGE(F,A,B))]
H1,1 - BG(b,RANGE(F,A,B)) <==> BG(b,B) AND TE(BG(a,A):BG((a,b),F))
U – IP[TE(BG(a,A):BG((a,b),F))]
.
.
.
GL1,1 – BG(b,RANGE(F,A,B)) First goal formula of EQ_RED[b], now with new hypotheses
HD1,1.1 - BG((x,b),F)
HD1,1 - TE(BG(a,A):BG((a,b),F))
U - PROP_CONS[BG(b,RANGE(F,A,B))]
GL1,1 – BG((x,b),F) Second goal formula of EQ_RED[b]
U – PROP_CONS[BG((x,b),F)]
GL2 – EQ(b,a) Third and last goal formula of EQ_RED[b]
H2 - BG(a,RANGE(F,A,B)) AND BG((x,a),F) Hypothesis added to the context of GL2 to prove it
HD1 – BG((x,b),F)

U - PROP_CONS[FA(BG(y,B):BG((x,y),F) ==> EQ(b,y))]
.
.
.
U – PROP_CONS[EQ(b,a)] With this, we have gotten the last goal formula of EQ_RED[b]
HD1 - EQ(b,a)
HD - TE!(BG(y,RANGE(F,A,B)):BG((x,y),F))

EQUAL_EQUIV

Acronym of: EQUAL EQUIValence
EQUAL_EQUIV[Formula]

Can be used when you want to use a formula Formula that is the equality consequence of the available hypotheses.

Sets Formula as a new hypothesis.

Example:
THEOR[PAIR_P01;BG(z,A);SUBSET({z},A)]

GL1 - SUBSET({z},A)
H1 - BG(z,A)
U - BY_DEF_OF[SUBSET({z},A)]
H1 - SUBSET({z},A) <==> FA(BG(a,{z}):BG(a,A))
U - IP[FA(BG(a,{z}):BG(a,A))]
GL1.1 - FA(BG(a,{z}):BG(a,A))
U - UQ_RED[]
H1.1 - SET(a)
GL1.2 - BG(a,{z}) ==> BG(a,A)
U - IF_RED[]
GL1.3 - BG(a,A)
H1.3 - BG(a,{z})
U - PAIR_AXIOM[BG(a,{z}),DEF_OF]
H1.3 - EQ(a,z)
U - EQUAL_EQUIV[BG(a,A)] BG(a,A) is an equality consequence by EQ(a,z) and BG(z,A)
GL1 - SUBSET({z},A)
HD1.2 - BG(a,A)
HD1.1 - BG(a,{z}) ==> BG(a,A)
HD1 - FA(BG(a,{z}):BG(a,A))
U - PROP_CONS[SUBSET({z},A)]
HD - SUBSET({z},A)

HYP_TOO

Acronym of: This is a HYPothesis TOO
HYP_TOO[Atomic Formula]

Can be used when you want to use an Atomic Formula that is an implicit hypothesis of the available ones.

Sets Atomic Formula as a new hypothesis.

Example:

GL1 - BG(x,DOM_PROJ1(F,A,A,A))
H1 - BG(x,A) AND BIN_OPER(F,A)
U - HYP_TOO[FUNC_IN(F,CART_PROD(A,A),A)]

> This is an implicit hypothesis because BIN_OPER(F,A) is a hypothesis in the current context and FUNC_IN(F,CART_PROD(A,A),A) is a hypothesis in the definition of BIN_OPER(F,A).
> Other implicit hypothesis in the current context is FUNC(F,CART_PROD(A,A),A) because it is a hypothesis in the definition of FUNC_IN(F,CART_PROD(A,A),A).
> Other implicit hypothesis in the current context is RELAT(F,CART_PROD(A,A),A) because it is a hypothesis in the definition of FUNC(F,CART_PROD(A,A),A).
> Why SUBSET(F,CART_PROD(CART_PROD(A,A),A) is even another implicit hypothesis in the current context?

H - FUNC_IN(F,CART_PROD(A,A),A)
.
.
.

IF_RED

Acronym of: IF REDduction
IF_RED[ ]

Can be used to reduce the conditional in the current goal form. That is if the current goal form has the form:

Formula1 ==> Formula2

Sets Formula1 as a new hypothesis and sets Formula2 as the new goal formula.

Example:
THEOR[BIN_UNION_P05;SUBSET(X,A),SET(B);SUBSET(X,BIN_UNION(A,B))]

GL1 - SUBSET(X,BIN_UNION(A,B))
H1 - SUBSET(X,A) AND SET(B)
U - IP[FA(BG(y,X):BG(y,BIN_UNION(A,B)))]
GL1.1 - FA(BG(y,X):BG(y,BIN_UNION(A,B)))
U - UQ_RED[]
H1.1 - SET(y)
GL1.2 - BG(y,X) ==> BG(y,BIN_UNION(A,B))
U - IF_RED[]
GL1.3 - BG(y,BIN_UNION(A,B))
H1.3 - BG(y,X)
U - BY_THEOR[BG(y,A)]SUBSET_DIRECT_0
H1.3 - BG(y,A)
U - BY_THEOR[BG(y,A) OR BG(y,B) <==>
BG(y,BIN_UNION(A,B))]BIN_UNION_P04_DIRECT_0
H1.3 - BG(y,A) OR BG(y,B) <==> BG(y,BIN_UNION(A,B))
U - PROP_CONS[BG(y,BIN_UNION(A,B))] This finishes the IF_RED command
GL1 - SUBSET(X,BIN_UNION(A,B))
HD1.2 - BG(y,BIN_UNION(A,B))
HD1.1 - BG(y,X) ==> BG(y,BIN_UNION(A,B)) This is what IF_RED has proved
HD1 - FA(BG(y,X):BG(y,BIN_UNION(A,B)))
U - BY_DEF_OF[SUBSET(X,BIN_UNION(A,B))]
H1 - SUBSET(X,BIN_UNION(A,B)) <==> FA(BG(z,X):BG(z,BIN_UNION(A,B)))
U - PROP_CONS[SUBSET(X,BIN_UNION(A,B))]
HD - SUBSET(X,BIN_UNION(A,B))

IFF_RED_IF

Acronym of: IFF REDuction IF first
IFF_RED_IF[ ]

Can be used to reduce the biconditional in the current goal form. That is if the current goal formula has the form

Formula1 <==> Formula2

the command sets

Formula1 ==> Formula2

as the new goal formula. When this one is proved, the command sets

Formula2 ==> Formula1

as the new goal formula. When this one is proved, the command sets the original goal formula

Formula1 <==> Formula2

as a new hypothesis.

Example:
THEOR[ORD_PAIR_P01_AUX05;SET(x),SET(y); EQ({x},{y}) <==> EQ(x,y)]

GL1 - EQ({x},{y}) <==> EQ(x,y)
H1 - SET(x) AND SET(y)
U - IFF_RED_IF[]
GL1,1 - EQ({x},{y}) ==> EQ(x,y) First goal formula of IFF_RED_IF[]
U - IF_RED[]
GL1,2 - EQ(x,y)
H1,2 - EQ({x},{y})
U - PAIR_AXIOM[BG(x,{x}),CHECK]
H1,2 - BG(x,{x})
U - PAIR_AXIOM[BG(y,{y}),CHECK]
H1,2 - BG(y,{y})
U - EQUAL_EQUIV[BG(x,{y})]
H1,2 - BG(x,{y})
U - PAIR_AXIOM[BG(x,{y}),DEF_OF] this proves the first goal formula of IFF_RED_IF[]
GL2 - EQ(x,y) ==> EQ({x},{y}) Second goal formula of IFF_RED_IF[]
HD1,1 - EQ(x,y)
HD1 - EQ({x},{y}) ==> EQ(x,y)
U - IF_RED[]
GL3 - EQ({x},{y})
H3 - EQ(x,y)
U – EQUAL_EQUIV[EQ({x},{y})] this proves the second goal formula of IFF_RED_IF[]
HD2 - EQ({x},{y})
HD1 - EQ(x,y) ==> EQ({x},{y})
HD - EQ({x},{y}) <==> EQ(x,y) The original and main goal formula is set as hypothesis, QED.

IP

Acronym of: start a Intermediate Proof
IP[Formula]

Can be used to introduce a new formula Formula that you want to prove. That is set Formula as the new goal formula. Formula can be any valid formula assuming the current hypothesis.

Example:
THEOR[NOEMP_P01;BG(x,A);NOEMP(A)]

GL1 - NOEMP(A)

H1 - BG(x,A)
U - IP[TE(SET(s):BG(s,A))]
GL1.1 - TE(SET(s):BG(s,A)) New goal formula
U - EQ_RED[x]
GL1.2 - BG(x,A)
U – PROP_CONS[BG(x,A)] This ends the logic command IP
GL1 - NOEMP(A)
HD1.1 - BG(x,A)
HD1 - TE(SET(s):BG(s,A)) This is the argument of IP, became in hypothesis because was proven
U - ATOMIC_OF[TE(SET(s):BG(s,A)),NOEMP(A)]
HD - NOEMP(A)

OR_RED

Acronym of: OR REDuction
OR_RED[Formula$_i$]

Can be used when the goal formula Gformula has the form:

Formula$_1$ OR Formula$_2$ OR … OR Formula$_n$

and you want prove it by proving the formula Formula$_i$

Example:

GL1 - BG({B},{A,B,C}) OR BG({C},{A,B,C}) OR BG(B,{A,B,C})
H1 - SET(A) AND SET(B) AND SET(C)
U - OR_RED[BG(B,{A,B,C})]
GL2 - BG(B,{A,B,C}) This new goal formula will imply the former
U - PAIR_AXIOM[BG(B,{A,B,C}),CHECK]
HD1 - BG(B,{A,B,C})
HD - BG({B},{A,B,C}) OR BG({C},{A,B,C}) OR BG(B,{A,B,C})

PAIR_AXIOM

Acronym of: by the PAIR AXIOM
PAIR_AXIOM[BG(TERM,{term_list}),CHECK|DEF_OF]

If CHECK is used, verify that TERM match one of the terms in term_list, if so it sets

BG(TERM,{term_list})

as a new hypothesis. term_list must has the form:

Term$_1$, Term$_2$, … , Term$_n$

If DEF_OF is used, the command set

EQ(TERM,Term$_1$) OR EQ(TERM,Term$_2$) OR … OR EQ(TERM,Term$_n$)

as a new hypothesis.

Example:
THEOR[BIN_UNION_P02;SET(B),BG(x,A);BG(x,BIN_UNION(A,B))]

GL1 - BG(x,BIN_UNION(A,B))
H1 - SET(A) AND BG(x,B)
U - BY_DEF_OBC[BG(x,BIN_UNION(A,B))]
H1 - BG(x,BIN_UNION(A,B)) <==> BG(x,UNION({A,B})) AND (BG(x,A) OR BG(x,B))
U - BY_DEF_OBE[UNION({A,B})]
H1 - FA(SET(z):BG(z,UNION({A,B})) <==> TE(BG(a,{A,B}):BG(z,a)))
U - SUBST_UQV[FA(SET(z):BG(z,UNION({A,B})) <==> TE(BG(a,{A,B}):BG(z,a))),x]
H1 - BG(x,UNION({A,B})) <==> TE(BG(a,{A,B}):BG(x,a))
U - IP[TE(BG(a,{A,B}):BG(x,a))]
GL1.1 - TE(BG(a,{A,B}):BG(x,a))
U - EQ_RED[B]
GL1.1,1 – BG(B,{A,B})
U – PAIR_AXIOM[BG(B,{A,B}),DEF_OF] [This two lines are in italic because they are not
H1.1,1 - EQ(A,{A,B}) OR EQ(B,{A,B}) part of the real proof]
U - PAIR_AXIOM[BG(B,{A,B}),CHECK]
GL1.2 - BG(x,B)
HD1.1 - BG(B,{A,B})
U - PROP_CONS[BG(x,B)]
GL1 - BG(x,BIN_UNION(A,B))
HD1.1 - BG(x,B)
HD1 - TE(BG(a,{A,B}):BG(x,a))
U - PROP_CONS[BG(x,BIN_UNION(A,B))]
HD - BG(x,BIN_UNION(A,B))

PROP_CONS

Acronym of: by PROPositional CONSequence
PROP_CONS[Formula]

Can be used when you want to use a formula Formula that is a simply propositional consequence of the available hypotheses.

Sets Formula as a new hypothesis.

Example:
THEOR[POWER_P01;SET(B),BG(A,POWER(B));SUBSET(A,B)]

GL1 - SUBSET(A,B)
H1 - SET(B) AND **BG(A,POWER(B))**
U - BY_DEF_OBE[POWER(B)]
H1 - FA(SET(z):BG(z,POWER(B)) <==> SUBSET(z,B))
U - SUBST_UQV[FA(SET(z):BG(z,POWER(B)) <==> SUBSET(z,B)),A]
H1 - BG(A,POWER(B)) <==> SUBSET(A,B)
U - PROP_CONS[SUBSET(A,B)] SUBSET(A,B) is a propositional consequence of
BG(A,POWER(B) and BG(A,POWER(B)) <==> SUBSET(A,B)
HD - SUBSET(A,B) “HD“ not followed by digits means that this is the thesis of our theorem so QED.

QUANT_NEG

Acronym of: QUANTifier NEGation
QUANT_NEG[Formula]

Can be uses when you find useful negate a quantifier of the current hypotheses.

Formula must be a quantifier of the current hypotheses or a explicit negation of it, in other words it must has the form:

TE|FA(SET(Variable)|BG(Variable,Term):Formula(Variable))

or the form:

NOT(TE|FA(SET(Variable)|BG(Variable,Term):Formula(Variable)))

If Formula has the form:

FA(SET(Variable):Formula(Variable))

then set

NOT(TE(SET(Variable):NOT(Formula(Variable)))

as a new hypothesis.

If Formula has the form:

FA(BG(Variable,Term):Formula(Variable))

then set

NOT(TE(BG(Variable,Term):NOT(Formula(Variable)))

as a new hypothesis

If Formula has the form:

TE(SET(Variable):Formula(Variable))

then set

NOT(FA(SET(Variable):NOT(Formula(Variable)))

as a new hypothesis

If Formula has the form:

TE(BG(Variable,Term):Formula(Variable))

then set

NOT(FA(BG(Variable,Term):NOT(Formula(Variable)))

as a new hypothesis

If Formula has the form:

NOT(FA(SET(Variable):Formula(Variable)))

then set

TE(SET(Variable):NOT(Formula(Variable)))

as a new hypothesis

If Formula has the form:

NOT(FA(BG(Variable,Term):Formula(Variable))) then set

TE(BG(Variable,Term):NOT(Formula(Variable)))

as a new hypothesis

If Formula has the form:

NOT(TE(SET(Variable):Formula(Variable)))

then set

FA(SET(Variable):NOT(Formula(Variable)))

as a new hypothesis

If Formula has the form:

NOT(TE(BG(Variable,Term):Formula(Variable)))

then set

FA(BG(Variable,Term):NOT(Formula(Variable)))

as a new hypothesis

Example:
THEOR[IS_EMPTY_P01;SET(C);NOT(IS_EMPTY(C)) <==> NOEMP(C)]

GL1 - NOT(IS_EMPTY(C)) <==> NOEMP(C)
H1 - SET(C)
U - BY_DEF_OF[IS_EMPTY(C)]
H1 - IS_EMPTY(C) <==> FA(SET(z):NOT(BG(z,C)))
U - BY_DEF_OF[NOEMP(C)]
H1 - NOEMP(C) <==> TE(SET(x):BG(x,C))
U - IFF_RED_IF[]
GL1,1 - NOT(IS_EMPTY(C)) ==> NOEMP(C)
U - IF_RED[]
GL1,2 - NOEMP(C)
H1,2 - NOT(IS_EMPTY(C))
U - PROP_CONS[NOT(IS_EMPTY(C)) <==> NOT(FA(SET(z):NOT(BG(z,C))))]
H1,2 - NOT(IS_EMPTY(C)) <==> NOT(FA(SET(z):NOT(BG(z,C))))
U - PROP_CONS[NOT(FA(SET(z):NOT(BG(z,C))))]
H1,2 - NOT(FA(SET(z):NOT(BG(z,C))))
U - QUANT_NEG[NOT(FA(SET(z):NOT(BG(z,C))))]

H1,2 - TE(SET(z):BG(z,C))
U - PROP_CONS[NOEMP(C)]
GL2 - NOEMP(C) ==> NOT(IS_EMPTY(C))
HD1,1 - NOEMP(C)
HD1 - NOT(IS_EMPTY(C)) ==> NOEMP(C)
U - IF_RED[]
GL3 - NOT(IS_EMPTY(C))
H3 - NOEMP(C)
U - PROP_CONS[TE(SET(z):BG(z,C))]
H3 - TE(SET(z):BG(z,C))
U - QUANT_NEG[TE(SET(z):BG(z,C))]
H3 - NOT(FA(SET(z):NOT(BG(z,C))))
U - PROP_CONS[NOT(IS_EMPTY(C))]
HD2 - NOT(IS_EMPTY(C))
HD1 - NOEMP(C) ==> NOT(IS_EMPTY(C))
HD - NOT(IS_EMPTY(C)) <==> NOEMP(C)

SUBST_UQV

Acronym of: SUBSTitute Universal Quantified Variable
SUBST_UQV[Formula1,Term1]

Can be used when you have a universal quantified formula Formula1 in the available hypotheses and you want instantiate the term Term1. That is, Formula1 must has the form:

FA(SET(Variable)|BG(Variable,Term):Formula(Variable))

If it has the form

FA(BG(Variable,Term):Formula(Variable))

then the command sets

BG(Term1,Term)

as the new goal formula. When this one is proved, the command sets

Formula(Term1)

as a new hypothesis.

If Formula1 has the form

FA(SET(Term):Formula(Variable)),

then the command sets

Formula(Term1)

as hypothesis.

Example:
THEOR[POWER_P01;SET(B),BG(A,POWER(B));SUBSET(A,B)]

GL1 - SUBSET(A,B)
H1 - SET(B) AND BG(A,POWER(B))
U - BY_DEF_OBE[POWER(B)]
H1 - FA(SET(z):BG(z,POWER(B)) <==> SUBSET(z,B))
U - SUBST_UQV[FA(SET(z):BG(z,POWER(B)) <==> SUBSET(z,B)),A]
H1 - BG(A,POWER(B)) <==> SUBSET(A,B)
U - PROP_CONS[SUBSET(A,B)]
HD - SUBSET(A,B)

UQ_RED

Acronym of: Universal Quantifier REDuction
UQ_RED[ ]

Can be used to reduce the current goal formula that must be an universal quantifier formula [that is the current goal formula must has the form FA(SET(x) | BG(x,TERM):Formula(x))]

If the goal formula has the form:

FA(BG(x,TERM):Formula(x))

Then set

BG(x,TERM) ==> Formula(x)

as the new goal formula and set SET(x) as a new hypothesis.

If the goal formula has the form:

FA(SET(x):Formula(x))

Then set

Formula(x)

as the new goal formula and set SET(x) as a new hypothesis.

Example:
THEOR[EMPTY_P01;SET(X);SUBSET(EMPTY,X)]

GL1 – SUBSET(EMPTY,X)
H1 – SET(X)
U – IP[FA(BG(x,EMPTY):BG(x,X))]
GL1.1 – FA(BG(x,EMPTY):BG(x,X))
U – UQ_RED[]
H1.1 – SET(x)
GL1.2 - BG(x,EMPTY) ==> BG(x,X)
U – BY_THEOR[NOTBG(x,EMPTY)]A01_EXISTENCE_AXIOM
H1.2 – NOTBG(x,EMPTY)
U – DEF_OF[NOTBG(x,EMPTY)]
H1.2 – NOT(BG(x,EMPTY))
U - PROP_CONS[BG(x,EMPTY) ==> BG(x,X)] This proves the first and unique UQ_RED command goal formula
GL1 - SUBSET(EMPTY,X) Because GL1.2 has been proved, GL1 back to be the goal formula
HD1.1 - BG(x,EMPTY) ==> BG(x,X)
HD1 - FA(BG(x,EMPTY):BG(x,X)) The original goal formula when UQ_RED was sent is assumed as hypothesis
U – ATOMIC_OF[FA(BG(x,EMPTY):BG(x,X)),SUBSET(EMPTY,X)]HD - SUBSET(EMPTY,X)