# *Theorema*: Theorem Proving for the Masses Using *Mathematica*

*Bruno Buchberger*

RISC
Researsch Institute for Symbolic Computation
A–4232 Castle of Hagenberg, Austria
Bruno.Buchberger@RISC.Uni–Linz.ac.at

June 1998

## ■ What is Mathematics?

## ■ A "Definition"

Mathematics is
knowledge derivation and problem solving
by reasoning

## ■ Knowledge Acquisition and Problem Solving

Knowledge derivation and problem solving are only two sides of the same coin.

**Example**:

For deriving the knowledge

$$\forall_{F}\ \exists_{G}\quad \text{Ideal[F]} = \text{Ideal[G]}\ \wedge\ \text{is-Groebner-basis[G]}$$

we need to solve the problem of constructing Groebner bases.

**Example**:

For solving the problem of constructing Groebner bases we have to derive new "algorithmically useful" knowledge, namely

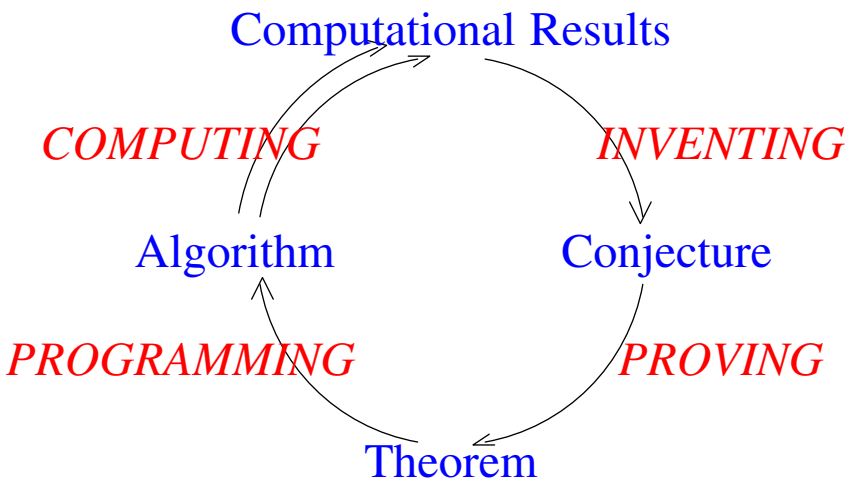
$$\text{is-Groebner-basis[G]} \Longleftrightarrow \forall_{f,g\in G}\ \text{normal-form[S-polynomial[f, g], G]} = 0.$$

**Bibligraphic Data:** Invited keynote at the Worldwide *Mathematica* Conference, Chicago, USA,
June 18 – 20, 1998

**Acknowledgements:** Partially supported by the Austrian Science Fundation (FWF), SFB project P1302 "Proving in general domains".

## ■ How Do we "Do" Mathematics?

The "Creativity Spiral" of Mathematics:




## ■ Mathematical Software Systems and Theorema

### ■ Present Math Software Systems (*Mathematica*)

Tremendous progress in supporting the programming and computing phase.

Almost no support in the proving phase.

The inventing phase is essentially "human". However, the better the computer–support for the other phases, the more likely is "invention".

### ■ Present Automated Theorem Proving Systems

Tremendous progress in proving.

No support of the other phases.

The proofs generated are not meant to be read by humans ("black box proving").

### ■ Theorema

Tries to close the gap: Support also the proving phase without giving up support in the other phases.

Natural approach: Add proving to *Mathematica*.

## ■ Distinctive Features of Theorema

- integration of existing math system (*Mathematica* 3.0)
- imitation of human proof style
- natural language formulation of proofs
- usual mathematical syntax
- emphasis on generating the routine parts of proofs
- combination with functor programming: multi−method proving (a special prover for each functor)
- failure report
- integration into math textbook generation and math teaching

## ■ The Role of *Mathematica* in *Theorema*:

- *Mathematica* ⊂ higher−order equational logic. Hence, *Mathematica* can be used as a "logic internal programming language":

**Example**:

$$\text{Groebner-basis}[F] \Longleftrightarrow \underset{f,g \in F}{\forall} \text{NF}[\text{SP}[f, g], F] = 0$$

This formula is both a theorem and an algorithm!

- *Mathematica* is also our implementation language for the provers.
- Typesetting and extensibile syntax facility.
- Notebook, nested cells and hyperlink facilities.

## ■ Theorema Computing Sessions

## ■ What is a Theorema Computing Session?

Theorema computing session = *Mathematica* session with slightly different syntax.

Theorema algorithmic formulae = predicate logic formulae with bounded quantifiers

```
InitializeTheorema[]
```

```
ComputingSession[]
```

## ■ Computing with Bounded Quantifiers

$$\forall_{i=-10,\ldots,10}\ i \leq 0$$

False

$$\exists_{i=-10,\ldots,10}\ i \leq 0$$

True

## ■ Computing with Defined Functions

any[x] :
 (f[x] := x + 3)

f[4]

7

any[n] :
 fac[1] := 1
 fac[n] := n fac[n − 1]

fac[1]

1

fac[6]

720

any[x] :
 s[x] := −1 ⇐ x < 0
 s[0] := 0
 s[x] := 1

s[5]

1

s[0]

0

s[−4]

−1

## ■ Computing with Numbers

2 + 3

5

3 ≤ 2

False

$$\sum_{i=1,\ldots,3} i$$

6

$$\sum_{i=1,\ldots,\prod_{j=1,\ldots,4} j} i$$

300

$$\max_{i=-5,\ldots,3} ((-i)^2)$$

25

$3 \shortmid 9$

True

any[n] :

$$\left(\text{is-prime}[n] := \left(\forall_{d=1,\ldots,n} (d \shortmid n) \Longrightarrow ((d = 1) \vee (d = n))\right)\right)$$

is-prime[7]

True

$$\#_{i=10,\ldots,20} (\text{is-prime}[i])$$

4

$$\mu_{i=10,\ldots,20} (\text{is-prime}[i])$$

11

## ■ Computing with Tuples

$17 \frown \langle 2, 3, 4, 2\rangle$

$\langle 17, 2, 3, 4, 2\rangle$

$\langle 17, 2, 3, 4, 2\rangle \asymp \langle 3, 4, 2\rangle$

$\langle 17, 2, 3, 4, 2, 3, 4, 2\rangle$

$\langle 2, 3, 1, 4, 5\rangle_3$

1

$$\left\langle i_{i=1,\ldots,10} \mid \right\rangle$$

$\langle 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\rangle$

$$\left\langle \text{is-prime}[i]_{i=1,\ldots,10} \mid \right\rangle$$

$\langle$True, True, True, False, True, False, True, False, False, False$\rangle$

any[X, Y] :

$X \oplus Y := \left\langle (X_i + Y_i) \mid_{i=1,\ldots,|X|} \right\rangle$

$X \preceq Y :=$ where$\left[n = |X|, \forall_{i=1,\ldots,n} (X_i \preceq Y_i)\right]$

$\langle 1, 2, 3\rangle \oplus \langle 2, 3, 4\rangle$

$\langle 3, 5, 7\rangle$

$\langle 1, 2, 3\rangle \preceq \langle 2, 3, 4\rangle$

True

$\langle 1, 2, 3\rangle \preceq \langle 2, 1, 4\rangle$

False

any[x, y, $\overline{Z}$] :

is-sorted[$\langle\rangle$] := True

is-sorted[$\langle x\rangle$] := True

is-sorted[$\langle x, y, \overline{Z}\rangle$] := $x \preceq y \bigwedge$ is-sorted[$\langle y, \overline{Z}\rangle$]

is-sorted[$\langle\rangle$]

True

is-sorted[$\langle 5\rangle$]

True

is-sorted[$\langle 5, 6, 8, 8, 9\rangle$]

True

is-sorted[$\langle 5, 6, 5, 8, 9\rangle$]

False

## ■ Computing with Sets

$\left\{ (2\,i) \mid_{i\in\{1,2,3\}} \right\}$

{2, 4, 6}

$\left\{ i \mid_{i=1,\ldots,20} \text{is-prime}[i] \right\}$

{1, 2, 3, 5, 7, 11, 13, 17, 19}

{2, 3, 1, 2, 3} = {1, 2, 3, 3, 3}

True

{2, 3, 1, 2, 3}

{1, 2, 3}

{1, 2, 3, 3, 3}

{1, 2, 3}

{1, 2, 1, 3} = {2, 1, 4, 3}

False

2 ∈ {2, 1, 2, 4}

True

any[is-set[A, B]] :

$A \ominus B := \left\{ x \underset{x \in A}{|} (x \notin B) \right\}$

$A \subseteq B := \underset{x \in A}{\forall} x \in B$

{2, 3, 1, 3} ⊖ {3, 5}

{1, 2}

2 ⊖ 3

2 ⊖ 3

is-set[2]

False

{3, 2, 5} ⊆ {1, 2, 3, 4, 5}

True

is-set[{2, 3, 2}]

True

is-tuple[{2, 3, 2}]

False

is-set[⟨2, 3, 2⟩]

False

is-tuple[⟨2, 3, 2⟩]

True

|⟨2, 3, 2⟩|

3

|{2, 3, 2}|

2

|−3|

3

æ[{3}]

3

an-element[{3, 2, 1, 4}]

1

any[is-set[A]] :
 ℙ[A] := {{}} ⇐ (A = {})
 ℙ[A] := where[a = an-element[A], P = ℙ[A ∖ {a}],
 P⋃ {({a} ∪ B) | }]
 B∈P

ℙ[{}]

{{}}

ℙ[{3}]

{{}, {3}}

ℙ[{1, 3}]

{{}, {1}, {3}, {1, 3}}

ℙ[{1, 3, 4, 8}]

{{}, {1}, {3}, {4}, {8}, {1, 3}, {1, 4}, {1, 8}, {3, 4}, {3, 8}, {4, 8}, {1, 3, 4}, {1, 3, 8}, {1, 4, 8}, {3, 4, 8}, {1, 3, 4, 8}}

EndComputingSession[]

## ■ Theorema Computing in Proving Sessions

**Definition**["is prime", any[n],
 is-prime[n] = (∀ (d ∣ n) ⟹ ((d = 1) ∨ (d = n)))]
 d=1,…,n

?? is-prime

Global`is-prime

Compute[is-prime[7], KB → Definition["is prime"]]

True

Clear[SuperPlus]

**Definition**["addition", any[m, n],
 m + 0 = m "+: base case"
 m + n⁺ = (m + n)⁺ "+: ind case" ]

```
Compute[0^++ + 0^++, KB → Definition["addition"]]
```

$$((((0^{+})^{+})^{+})^{+})^{+}$$

## ■ Theorema Proving in Proving Sessions

```
Proposition["zero from left", any[n],
 0 + n = n   "ZL"]

Prove[Proposition["zero from left"],
 KB → Definition["addition"],
 Prover → NNEqIndProver ]
```

Note: The proof note books generated by the calls to "Prove", are given in the appendix.

```
$TmaTraceLevel = 50;

Theory["addition",
 Proposition["zero from left"]
 Definition["addition"]
]

Compute[0 + 0^++, KB → Definition["addition"]]
```

addition_+: ind case

addition_+: ind case

addition_+: base case

$$(0^{+})^{+}$$

```
Compute[0 + 0^++, KB → Theory["addition"]]
```

addition_zero from left_ZL

$$(0^{+})^{+}$$

```
$TmaTraceLevel = 0;
```

This simple example, in fact, shows the essence of mathematics:

More mathematical knowledge gives more efficient computation!

## ■ Example: Theorema Propositional Proving

```
Proposition["Iterated Implication",
 (P ⟹ (Q ⟹ R)) ⟺ ((P ∧ Q) ⟹ R)   "ii"]

Prove[Proposition["Iterated Implication"], KB → {}, Prover -> PredicateProver]

Proposition["Peirce Implication",
   ((A ⟹ B) ⟹ A) ⟹ A   "p"]

Prove[Proposition["Peirce Implication"], KB → {}, Prover -> PredicateProver]
```

## ■ Example: Theorema Predicate Logic Proving

## ■ Quantifier distributivity

```
$TmaDefaultProver = PredicateProver;
```

**Proposition**["Quantifier Distributivity",
$\left(\forall_{x} (P[x] \Longrightarrow Q)\right) \Longleftrightarrow \left(\left(\exists_{x} P[x]\right) \Longrightarrow Q\right)$ "∀d"]

**Prove**[Proposition["Quantifier Distributivity"], KB → {},
Prover → PredicateProver]

PNDLevel controls the number of rules used!

**Prove**[Proposition["Quantifier Distributivity"], KB → {},
Prover → PredicateProver, ProverOptions → {PNDLevel −> 1}]

## ■ The Limit of a Sum

In the spirit of "structured programming", we build the definition of limit up in a hierarchy of definitions, one quantifier at a time:

**Definition**["2−ary limit",
$\forall_{f,a}$ limit[f, a] $\Longleftrightarrow \forall_{\epsilon}$ limit[f, a, ϵ] "l2:"]

**Definition**["3−ary limit",
$\forall_{f,a,\epsilon}$ limit[f, a, ϵ] $\Longleftrightarrow \exists_{N}$ limit[f, a, ϵ, N] "l3:"]

**Definition**["4−ary limit",
$\forall_{f,a,\epsilon,N}$ limit[f, a, ϵ, N] $\Longleftrightarrow \forall_{n} (n \geq N) \Longrightarrow (|f[n] - a| < \epsilon)$ "l4:"]

Auxiliary knowledge:

**Definition**["sum of sequences", any[f, g, x],
$(f \oplus g)[x] = f[x] + g[x]$ "⊕:"]

**Proposition**["distance of sum", any[x, y, a, b, δ, ϵ],
$(|x - a| < \delta \wedge |y - b| < \epsilon) \Longrightarrow (|(x + y) - (a + b)| < \delta + \epsilon)$ "|++|"]

**Proposition**["max", any[m, M, N],
$(m \geq \max[M, N]) \Longleftrightarrow (m \geq M \wedge m \geq N)$ "max:"]

**Theory**["4−ary limit",
Definition["4−ary limit"]
Definition["sum of sequences"]
Proposition["distance of sum"]
Proposition["max"]]

Ultimately, we want to prove

**Proposition**["2−ary limit of sum", any[f, a, g, b],
(limit[f, a] ∧ limit[g, b]) "l2+"
⟹ limit[f ⊕ g, a + b]]

The decomposition of the limit definition suggests building up the proposition (and its proof) in a hierarchy of propositions (and proofs):

**Proposition**["4−ary limit of sum",
$\forall_{f,a,\delta,M,g,b,\epsilon,N}$ ((limit[f, a, $\delta$, M] $\wedge$ limit[g, b, $\epsilon$, N]) "l4+"
$\Longrightarrow$ limit[f $\oplus$ g, a + b, $\delta + \epsilon$, max[M, N]])]

**Prove**[Proposition["4−ary limit of sum"], KB → Theory["4−ary limit"]]

**Theory**["3−ary limit",
Definition["3−ary limit"]
Proposition["4−ary limit of sum"]]

**Proposition**["3−ary limit of sum",
$\forall_{f,a,\delta,g,b,\epsilon}$ ((limit[f, a, $\delta$] $\wedge$ limit[g, b, $\epsilon$]) "l3+"
$\Longrightarrow$ limit[f $\oplus$ g, a + b, $\delta + \epsilon$])]

**Prove**[Proposition["3−ary limit of sum"], KB → Theory["3−ary limit"], Prover → PredicateProver,
ProverOptions → {PNDSearchDepth → 60}]

**Proposition**["3−ary limit of sum: variant",
$\forall_{f,a,g,b,\epsilon}$ ((limit[f, a, $\epsilon/2$] $\wedge$ limit[g, b, $\epsilon/2$]) "l3+"
$\Longrightarrow$ limit[f $\oplus$ g, a + b, $\epsilon$])]

**Theory**["2−ary limit",
Definition["2−ary limit"]
Proposition["3−ary limit of sum: variant"]]

**Proposition**["2−ary limit of sum",
$\forall_{f,a,g,b}$ ((limit[f, a] $\wedge$ limit[g, b]) "l2+"
$\Longrightarrow$ limit[f $\oplus$ g, a + b])]

**Prove**[Proposition["2−ary limit of sum"], KB → Theory["2−ary limit"], Prover → PredicateProver]

## ■ Example: Theorema Natural Number Induction Proving

## ■ Definition of + and *

**Definition**["addition", any[m, n],
$m + 0 = m$ " +0"
$m + n^+ = (m + n)^+$ " +."]

**Definition**["multiplication", any[m, n],
$m\,0 = 0$ " ∗0"
$m * n^+ = (m\,n) + m$ " ∗."]

## ■ Proof of Some Properties of + and *

**Proposition**["addition from left", any[m, n],
$0 + n = n$ "0+"
$m^+ + n = (m + n)^+$ ".+"]

```
Prove[Proposition["addition from left"],
 KB → Definition["addition"],
 Prover → NNEqIndProver ]

Proposition["associativity and commutativity of addition", any[m, n, p],
 m + (n + p) = (m + n) + p    "a+",
 m + n = n + m                "c+"]

KnowledgeBase["addition left/right",
 Definition["addition"]
 Proposition["addition from left"]]

Prove[Proposition["associativity and commutativity of addition"],
 KB → KnowledgeBase["addition left/right"],
 Prover → NNEqIndProver ]

KnowledgeBase["addition and multiplication",
 Definition["addition"]
 Definition["multiplication"]]

Proposition["zero multiplication from left", any[n],
 0 n = 0    "0∗"]

Prove[Proposition["zero multiplication from left"],
 KB → KnowledgeBase["addition and multiplication"],
 Prover → NNEqIndProver ]

Proposition["successor multiplication from left", any[m, n],
 m⁺ n = (m n) + n    ".∗"]

KnowledgeBase["addition and multiplication",

 Definition["addition"]
 Proposition["addition from left"]
 Proposition["associativity and commutativity of addition"]]
 Definition["multiplication"]
 Proposition["zero multiplication from left"]

Prove[Proposition["successor multiplication from left"],
 KB → KnowledgeBase["addition and multiplication"],
 Prover → NNEqIndProver ]

Proposition["commutativity of multiplication", any[m, n],
 m n = n m    "c∗"]

Theory["add/mult left/right",

 Definition["addition"]
 Definition["multiplication"]
 Proposition["addition from left"]                    ]
 Proposition["zero multiplication from left"]
 Proposition["successor multiplication from left"]

Prove[Proposition["commutativity of multiplication"],
 KB → Theory["add/mult left/right"],
 Prover → NNEqIndProver ]

Proposition["associativity of multiplication", any[m, n, p],
 m (n p) = (m n) p    "a∗"]
```

## ■ Power

**Definition**["power", any[m, n],
pow[m, 0] = $0^+$ "p0"
pow[m, $n^+$] = pow[m, n] m "p."]

**Proposition**["addition of powers", any[n, p, m],
pow[m, n + p] = pow[m, n] pow[m, p] "p"]

**Theory**["preparation for power",
Proposition["addition from left"]
Proposition["multiplication from left"]
Proposition["commutativity of multiplication"]
Proposition["associativity of multiplication"]
Definition["power"]]

**Prove**[Proposition["addition of powers"],
KB −> Theory["preparation for power"],
Prover −> NNEqIndProver, ProverOptions −> {ConstOrder −> {{•[SuperPlus], •[Plus], •[Times], pow}}} ]

## ■ Example: Theorema Sequence Induction Proving

## ■ Length of Concatenation

**Definition**["concatenation, length", any[x, $\overline{x}$, $\overline{y}$],
$(\langle \overline{x} \rangle \asymp \langle \overline{y} \rangle) = \langle \overline{x}, \overline{y} \rangle$ "$\asymp$"
$|\langle\,\rangle| = 0$ "||"
$|\langle x, \overline{y} \rangle| = |\langle \overline{y} \rangle|^+$ "|…|"]

**Definition**["addition", any[x, y],
0 + x = x "+0"
$x^+ + y = (x + y)^+$ "++"]

**Theory**["prerequisites for |≍|",
Definition["concatenation, length"]
Definition["addition"]]

**Proposition**["|≍|", any[$\overline{x}$],
$|\langle \overline{x} \rangle \asymp \langle \overline{y} \rangle| = |\langle \overline{x} \rangle| + |\langle \overline{y} \rangle|$]

**Prove**[Proposition["|≍|"], KB → Theory["prerequisites for |≍|"], Prover → TupleEqIndProver]

## ■ Reversion of Concatenation

**Definition**["concatenation and reversion", any[x, $\overline{x}$, $\overline{y}$],
$(x \frown \langle \overline{y} \rangle) = \langle x, \overline{y} \rangle$ "$\frown$"
$(\langle \overline{y} \rangle \frown x) = \langle \overline{y}, x \rangle$ "$\frown$"
$(\langle \overline{x} \rangle \asymp \langle \overline{y} \rangle) = \langle \overline{x}, \overline{y} \rangle$ "$\asymp$"
$\underline{\langle\,\rangle} = \langle\,\rangle$ "rev.b"
$\underline{\langle x, \overline{y} \rangle} = \underline{\langle \overline{y} \rangle} \frown x$ "rev.s"]

**Proposition**["concatenation and append", any[x, y, z, X, Y],
$(X \asymp \langle\rangle) = X$ "$\asymp\langle\rangle$"
$((X \asymp Y) \frown z) = (X \asymp (Y \frown z))$ "$\asymp\frown$"]

**Theory**["prerequisites for reversion of concatenation",
Definition["concatenation and reversion"]
Proposition["properties of append etc"]]

**Theorem**["reversion of concatenation", any[$\overline{x}$, $\overline{y}$],
$(\underline{\langle\overline{x}\rangle \asymp \langle\overline{y}\rangle}) = (\underline{\langle\overline{y}\rangle \asymp \langle\overline{x}\rangle})$ "_$\asymp$"]

**Prove**[Theorem["reversion of concatenation"], KB → Theory["prerequisites for reversion of concatenation"],
Prover → TupleEqIndProver,
ProverOptions → {ConstOrder → {{•[Tuple], •[Cup], •[Cap], •[CupCap],
UnderBar}}, TermOrder → RPLexOrder}]

## ■ Reverse of Reverse

**Proposition**["prepend to reverse", any[x, Y],
$(x \smile \underline{Y}) = \underline{(Y \frown x)}$ "$\smile$_"]

**Theory**["prerequisites for reverse of reverse",
Definition["concatenation and reversion"]
Proposition["prepend to reverse"]]

**Proposition**["reversion of reversion", any[$\overline{x}$],
$\underline{\underline{\langle\overline{x}\rangle}} = \langle\overline{x}\rangle$ "_r"]

**Prove**[Proposition["reversion of reversion"], KB → Theory["prerequisites for reverse of reverse"],
Prover → TupleEqIndProver,
ProverOptions → {ConstOrder → {{•[SuperPlus], •[BracketingBar]}, {•[Tuple], •[Cup], •[Cap], •[CupCap],
UnderBar}}}]

## ■ Conclusion

- Proving is at the heart of mathematics.
- Mathematics is at the heart of science and technology.
- A slight improvement in the quality of proving will have drastic effects in all areas of science and technology.
- We think that theorem proving along the lines of *Theorema* will be a practical tool within one generation and will drastically improve the quality and efficiency of inventing, teaching, and applying mathematics.

# ■ Appendix: Proof Examples

## ■ Proof Example: Zero Plus

Prove:

(ZL) $\underset{n}{\forall}\,(0 + n = n)$,

under the assumptions:

(+: base case) $\forall_{m} (m + 0 = m)$,

(+: ind case) $\forall_{m,n} (m + n^{+} = (m + n)^{+})$.

There are 2 alternatives for proceeding with the proof.

Alternative 1: using Simplifier

Sorry, proving (ZL) fails.

Alternative 2: using NIP

We prove (ZL) by induction on $n$.

Induction Base:

(ZL.IB) $0 + 0 = 0$.

A proof by simplification of (ZL.IB) works.

Simplification of the lhs term:

$0 + 0 =$ by (+: base case)

$0 \rfloor$

Simplification of the rhs term:

$0 \rfloor$

Induction Step:

Induction Hypothesis:

(ZL.IH) $0 + n_1 = n_1$

Induction Conclusion:

(ZL.IS) $0 + n_1{}^{+} = n_1{}^{+}$.

A proof by simplification of (ZL.IS) works.

Simplification of the lhs term:

$0 + n_1{}^{+} =$ by (+: ind case)

$(0 + n_1)^{+} =$ by (ZL.IH)

$n_1{}^{+} \rfloor$

Simplification of the rhs term:

$n_1{}^{+} \rfloor$

## ■ Proof Example: Iterated Implication

Prove:

(ii) $(P \Rightarrow (Q \Rightarrow R)) \Leftrightarrow P \wedge Q \Rightarrow R$,

with no assumptions.

We prove (ii) in both directions.

Direction from left to right:

We assume

(ii.LA) $P \Rightarrow (Q \Rightarrow R)$

and show

(ii.RC) $P \wedge Q \Rightarrow R$.

We prove (ii.RC) by the deduction rule.

We assume

(ii.RC.A) $P \wedge Q$

and show

(ii.RC.C) $R$.

From (ii.RC.A.1) and (ii.LA) we obtain by modus ponens

(ii.LA.C) $Q \Rightarrow R$.

From (ii.RC.A.2) and (ii.LA.C) we obtain by modus ponens

(ii.LA.C.C) $R$.

Formula (ii.RC.C) is true because it is identical to (ii.LA.C.C).

Direction from right to left:

We assume

(ii.RA) $P \wedge Q \Rightarrow R$

and show

(ii.LC) $P \Rightarrow (Q \Rightarrow R)$.

We prove (ii.LC) by the deduction rule.

We assume

(ii.LC.A) $P$

and show

(ii.LC.C) $Q \Rightarrow R$.

We prove (ii.LC.C) by the deduction rule.

We assume

(ii.LC.C.A) $Q$

and show

(ii.LC.C.C) $R$.

Because the conclusion of (ii.RA) is identical to the goal (ii.LC.C.C) we try two alternatives:

Alternative proof 1:

For Proving (ii.LC.C.C), by (ii.RA), it suffices to prove

(ii.RA.A) $P \wedge Q$.

We prove the individual conjunctive parts of (ii.RA.A):

Proof of (ii.RA.A.1) P:

Formula (ii.RA.A.1) is true because it is identical to (ii.LC.A).

Proof of (ii.RA.A.2) Q:

Formula (ii.RA.A.2) is true because it is identical to (ii.LC.C.A).

Alternative proof 2:

Formula (ii.RA) is transformed into

(ii.RA') $\neg (P \wedge Q) \vee R$.

Pending proof of (ii.LC.C.C).

## ■ Proof Example: Peirce Implication

Prove:

(p) $((A \Rightarrow B) \Rightarrow A) \Rightarrow A$,

with no assumptions.

We prove (p) by the deduction rule.

We assume

(p.A) $(A \Rightarrow B) \Rightarrow A$

and show

(p.C) A.

Because the RHS of (p.A) is identical to the goal (p.C) we try two alternatives:

Alternative proof 1:

For Proving (p.C), by (p.A), it suffices to prove

(p.A.A) $A \Rightarrow B$.

Proving (p.A.A) by the deduction rule fails.

We assume

(p.A.A.A) A

and show

(p.A.A.C) B.

Proving (p.A.A.C) by contradiction fails.

We assume

(p.A.A.C.pos) $\neg B$,

and show a contradiction.

Sorry, proving (a contradiction) fails.

Alternative proof 2:

Formula (p.A) is transformed into

(p.A') $\neg (A \Rightarrow B) \vee A$.

We prove (p.C) by case distinction using (p.A').

Case (p.A'.1) $\neg A \Rightarrow B$:

Formula (p.A'.1) is expanded into

(p.A'.1') $A \wedge \neg B$.

Formula (p.C) is true because it is identical to (p.A'.1'.1).

Case (p.A'.2) A:

Formula (p.C) is true because it is identical to (p.A'.2).

## ■ Proof Example: Distributiviy of ForAll

Prove:

(∀d) $\left(\underset{x}{\forall} (P[x] \Rightarrow Q)\right) \Leftrightarrow \underset{x}{\exists} (P[x]) \Rightarrow Q$,

with no assumptions.

We prove (∀d) in both directions.

Direction from left to right:

We assume

(∀d.LA) $\underset{x}{\forall}\,(P[x] \Rightarrow Q)$

and show

(∀d.RC) $\underset{x}{\exists}\,(P[x]) \Rightarrow Q$.

We prove (∀d.RC) by the deduction rule.

We assume

(∀d.RC.A) $\underset{x}{\exists}\,(P[x])$

and show

(∀d.RC.C) $Q$.

By (∀d.RC.A) we can take appropriate values such that:

(∀d.RC.A') $P[x_0]$.

From (∀d.RC.A') and an appropriate instance of (∀d.LA) we obtain by modus ponens:

(∀d.LA.C) $Q$.

Formula (∀d.RC.C) is true because it is identical to (∀d.LA.C).

Direction from right to left:

We assume

(∀d.RA) $\underset{x}{\exists}\,(P[x]) \Rightarrow Q$

and show

(∀d.LC) $\underset{x}{\forall}\,(P[x] \Rightarrow Q)$.

For proving (∀d.LC) we take all variables arbitrary but fixed and prove:

(∀d.LC') $P[x_1] \Rightarrow Q$.

We prove (∀d.LC') by the deduction rule.

We assume

(∀d.LC'.A) $P[x_1]$

and show

(∀d.LC'.C) Q.

Because the conclusion of (∀d.RA) is identical to the goal (∀d.LC'.C) we try two alternatives:

Alternative proof 1:

For Proving (∀d.LC'.C), by (∀d.RA), it suffices to prove

(∀d.RA.A) $\underset{x}{\exists}\,(P[x])$.

Formula (∀d.RA.A) is proved because (∀d.LC'.A) is an instance of it.

Alternative proof 2:

Formula (∀d.RA) is transformed into

(∀d.RA') $\neg\,\underset{x}{\exists}\,(P[x]) \bigvee Q$.

Pending proof of (∀d.LC'.C).

## ■ Proof Example: Limit–4

Prove:

(l4+) $\underset{a,b,f,g,M,N,\delta,\epsilon}{\forall}$ (limit[f, a, δ, M] ∧ limit[g, b, ϵ, N] ⇒ limit[f ⊕ g, a + b, δ + ϵ, max[M, N]]),

under the assumptions:

(l4:) $\underset{a,f,N,\epsilon}{\forall}\left(\text{limit[f, a, ϵ, N]} \Leftrightarrow \underset{n}{\forall}\,(n \geq N \Rightarrow |f[n] - a| < \epsilon)\right)$,

(⊕:) $\underset{f,g,x}{\forall}\,((f \oplus g)[x] = f[x] + g[x])$,

(max:) $\underset{m,M,N}{\forall}\,(m \geq \max[M, N] \Leftrightarrow (m \geq M \wedge m \geq N))$,

(|++|) $\underset{a,b,x,y,\delta,\epsilon}{\forall}\,(|x - a| < \delta \wedge |y - b| < \epsilon \Rightarrow |(x + y) - (a + b)| < \delta + \epsilon)$.

For proving (l4+) we take all variables arbitrary but fixed and prove:

(l4+') $\text{limit}[f_0, a_0, \delta_0, M_0] \wedge \text{limit}[g_0, b_0, \epsilon_0, N_0] \Rightarrow \text{limit}[f_0 \oplus g_0, a_0 + b_0, \delta_0 + \epsilon_0, \max[M_0, N_0]]$.

We prove (l4+') by the deduction rule.

We assume

(l4+'.A) $\text{limit}[f_0, a_0, \delta_0, M_0] \wedge \text{limit}[g_0, b_0, \epsilon_0, N_0]$

and show

(l4+'.C) $\text{limit}[f_0 \oplus g_0, a_0 + b_0, \delta_0 + \epsilon_0, \max[M_0, N_0]]$.

From (l4+'.A.1) and an appropriate instance of (l4:) we obtain by modus ponens:

(l4:.C) $\underset{n}{\forall}\ (n \geq M_0 \Rightarrow |f_0[n] - a_0| < \delta_0)$.

From (l4+'.A.2) and an appropriate instance of (l4:) we obtain by modus ponens:

(l4:.C) $\underset{n}{\forall}\ (n \geq N_0 \Rightarrow |g_0[n] - b_0| < \epsilon_0)$.

Because the RHS of (l4:) is identical to the goal (l4+'.C) we try two alternatives:

Alternative proof 1:

For Proving (l4+'.C), by (l4:), it suffices to prove

(l4:.A) $\underset{n}{\forall}\ (n \geq \max[M_0, N_0] \Rightarrow |(f_0 \oplus g_0)[n] - (a_0 + b_0)| < \delta_0 + \epsilon_0)$.

For proving (l4:.A) we take all variables arbitrary but fixed and prove:

(l4:.A') $n_0 \geq \max[M_0, N_0] \Rightarrow |(f_0 \oplus g_0)[n_0] - (a_0 + b_0)| < \delta_0 + \epsilon_0$.

We prove (l4:.A') by the deduction rule.

We assume

(l4:.A'.A) $n_0 \geq \max[M_0, N_0]$

and show

(l4:.A'.C) $|(f_0 \oplus g_0)[n_0] - (a_0 + b_0)| < \delta_0 + \epsilon_0$.

Using (⊕:), the goal (l4:.A'.C) is transformed into:

(l4:.A'.C') $|(f_0[n_0] + g_0[n_0]) - (a_0 + b_0)| < \delta_0 + \epsilon_0$.

From (l4:.A'.A) and an appropriate instance of (max:) we obtain by modus ponens:

(max:.C) $n_0 \geq M_0 \wedge n_0 \geq N_0$.

From (max:.C.1) and an appropriate instance of (l4:.C) we obtain by modus ponens:

(l4:.C.C) $|f_0[n_0] - a_0| < \delta_0$.

From (max:.C.2) and an appropriate instance of (l4:.C) we obtain by modus ponens:

(l4:.C.C) $|g_0[n_0] - b_0| < \epsilon_0$.

Because the RHS of (|++|) is identical to the goal (l4:.A'.C') we try two alternatives:

Alternative proof 1:

For Proving (l4:.A'.C'), by (|++|), it suffices to prove

(|++|.A) $|f_0[n_0] - a_0| < \delta_0 \wedge |g_0[n_0] - b_0| < \epsilon_0$.

We prove the individual conjunctive parts of (|++|.A):

Proof of (|++|.A.1) $|f_0[n_0] - a_0| < \delta_0$:

Formula (|++|.A.1) is true because it is identical to (l4:.C.C).

Proof of (|++|.A.2) $|g_0[n_0] - b_0| < \epsilon_0$:

Formula (|++|.A.2) is true because it is identical to (l4:.C.C).

Alternative proof 2:

Alternative proof 2:

## ■ Proof Example: Limit–3

Prove:

(l3+) $\underset{a,b,f,g,\mathtt{d},\mathtt{e}}{\forall} (\text{limit}[f, a, \mathtt{d}] \wedge \text{limit}[g, b, \mathtt{e}] \Rightarrow \text{limit}[f \oplus g, a + b, \mathtt{d} + \mathtt{e}])$,

under the assumptions:

(l3:) $\underset{a,f,\mathtt{e}}{\forall} \left(\text{limit}[f, a, \mathtt{e}] \Leftrightarrow \underset{N}{\exists} (\text{limit}[f, a, \mathtt{e}, N])\right)$,

(l4+) $\underset{a,b,f,g,M,N,\mathtt{d},\mathtt{e}}{\forall} (\text{limit}[f, a, \mathtt{d}, M] \wedge \text{limit}[g, b, \mathtt{e}, N] \Rightarrow \text{limit}[f \oplus g, a + b, \mathtt{d} + \mathtt{e}, \max[M, N]])$.

For proving (l3+) we take all variables arbitrary but fixed and prove:

(l3+') $\text{limit}[f_0, a_0, \mathtt{d}_0] \wedge \text{limit}[g_0, b_0, \mathtt{e}_0] \Rightarrow \text{limit}[f_0 \oplus g_0, a_0 + b_0, \mathtt{d}_0 + \mathtt{e}_0]$.

We prove (l3+') by the deduction rule.

We assume

(l3+'.A) $\text{limit}[f_0, a_0, \mathtt{d}_0] \wedge \text{limit}[g_0, b_0, \mathtt{e}_0]$

and show

(l3+'.C) $\text{limit}[f_0 \oplus g_0, a_0 + b_0, \mathtt{d}_0 + \mathtt{e}_0]$.

From (l3+'.A.1) and an appropriate instance of (l3:) we obtain by modus ponens:

(l3:.C) $\underset{N}{\exists} (\text{limit}[f_0, a_0, \mathtt{d}_0, N])$.

By (l3:.C) we can take appropriate values such that:

(l3:.C') $\text{limit}[f_0, a_0, \mathtt{d}_0, N_0]$.

From (l3+'.A.2) and an appropriate instance of (l3:) we obtain by modus ponens:

(l3:.C) $\underset{N}{\exists} (\text{limit}[g_0, b_0, \mathtt{e}_0, N])$.

By (l3:.C) we can take appropriate values such that:

(l3:.C') $\text{limit}[g_0, b_0, \mathtt{e}_0, N_1]$.

Because the RHS of (l3:) is identical to the goal (l3+'.C) we try two alternatives:

Alternative proof 1:

For Proving (l3+'.C), by (l3:), it suffices to prove

(l3:.A) $\underset{N}{\exists}\,(\text{limit}[f_0 \oplus g_0,\ a_0 + b_0,\ \mathtt{d}_0 + \mathtt{e}_0,\ N])$.

Because the conclusion of (l4+) matches the goal, we specialize it to:

(l3:.A') $\underset{M,N}{\forall}\,(\text{limit}[f_0,\ a_0,\ \mathtt{d}_0,\ M] \wedge \text{limit}[g_0,\ b_0,\ \mathtt{e}_0,\ N] \Rightarrow \text{limit}[f_0 \oplus g_0,\ a_0 + b_0,\ \mathtt{d}_0 + \mathtt{e}_0,\ \max[M,\ N]])$.

Because a literal of the formula (l3:.A') matches the assumption (l3:.C'), we specialize it to:

(l3:.A'/1)

$\underset{N}{\forall}\,(\text{limit}[f_0,\ a_0,\ \mathtt{d}_0,\ N_0] \wedge \text{limit}[g_0,\ b_0,\ \mathtt{e}_0,\ N] \Rightarrow \text{limit}[f_0 \oplus g_0,\ a_0 + b_0,\ \mathtt{d}_0 + \mathtt{e}_0,\ \max[N_0,\ N]])$.

Because a literal of the formula (l3:.A'/1) matches the assumption (l3:.C'), we specialize it to:

(l3:.A'/1/2)

$\text{limit}[f_0,\ a_0,\ \mathtt{d}_0,\ N_0] \wedge \text{limit}[g_0,\ b_0,\ \mathtt{e}_0,\ N_1] \Rightarrow \text{limit}[f_0 \oplus g_0,\ a_0 + b_0,\ \mathtt{d}_0 + \mathtt{e}_0,\ \max[N_0,\ N_1]]$.

Because the RHS of (l3:.A'/1/2) is identical to the goal (l3:.A ) we try two alternatives:

Alternative proof 1:

For Proving (l3:.A ), by (l3:.A'/1/2), it suffices to prove

(l3:.A'/1/2.A) $\text{limit}[f_0,\ a_0,\ \mathtt{d}_0,\ N_0] \wedge \text{limit}[g_0,\ b_0,\ \mathtt{e}_0,\ N_1]$.

We prove the individual conjunctive parts of (l3:.A'/1/2.A):

Proof of (l3:.A'/1/2.A.1) $\text{limit}[f_0,\ a_0,\ \mathtt{d}_0,\ N_0]$:

Formula (l3:.A'/1/2.A.1) is true because it is identical to (l3:.C').

Proof of (l3:.A'/1/2.A.2) $\text{limit}[g_0,\ b_0,\ \mathtt{e}_0,\ N_1]$:

Formula (l3:.A'/1/2.A.2) is true because it is identical to (l3:.C').

Alternative proof 2:

Alternative proof 2:

## ■ Proof Example: Limit–2

Prove:

(l2+) $\underset{a,b,f,g}{\forall}\,(\text{limit}[f,\ a] \wedge \text{limit}[g,\ b] \Rightarrow \text{limit}[f \oplus g,\ a + b])$,

under the assumptions:

(l2:) $\underset{a,f}{\forall}\left(\text{limit}[f,\ a] \Leftrightarrow \underset{\mathtt{e}}{\forall}\,(\text{limit}[f,\ a,\ \mathtt{e}])\right)$,

(l3+) $\underset{a,b,f,g,\mathtt{e}}{\forall}\,(\text{limit}[f,\ a,\ \frac{\mathtt{e}}{2}] \wedge \text{limit}[g,\ b,\ \frac{\mathtt{e}}{2}] \Rightarrow \text{limit}[f \oplus g,\ a + b,\ \mathtt{e}])$.

For proving (l2+) we take all variables arbitrary but fixed and prove:

(l2+') $\text{limit}[f_0, a_0] \wedge \text{limit}[g_0, b_0] \Rightarrow \text{limit}[f_0 \oplus g_0, a_0 + b_0]$.

We prove (l2+') by the deduction rule.

We assume

(l2+'.A) $\text{limit}[f_0, a_0] \wedge \text{limit}[g_0, b_0]$

and show

(l2+'.C) $\text{limit}[f_0 \oplus g_0, a_0 + b_0]$.

From (l2+'.A.1) and an appropriate instance of (l2:) we obtain by modus ponens:

(l2:.C) $\underset{\epsilon}{\forall}(\text{limit}[f_0, a_0, \epsilon])$.

From (l2+'.A.2) and an appropriate instance of (l2:) we obtain by modus ponens:

(l2:.C) $\underset{\epsilon}{\forall}(\text{limit}[g_0, b_0, \epsilon])$.

Because the RHS of (l2:) is identical to the goal (l2+'.C) we try two alternatives:

Alternative proof 1:

For Proving (l2+'.C), by (l2:), it suffices to prove

(l2:.A) $\underset{\epsilon}{\forall}(\text{limit}[f_0 \oplus g_0, a_0 + b_0, \epsilon])$.

For proving (l2:.A) we take all variables arbitrary but fixed and prove:

(l2:.A') $\text{limit}[f_0 \oplus g_0, a_0 + b_0, \epsilon_0]$.

Because the RHS of (l3+) is identical to the goal (l2:.A') we try two alternatives:

Alternative proof 1:

For Proving (l2:.A'), by (l3+), it suffices to prove

(l3+.A) $\text{limit}[f_0, a_0, \frac{\epsilon_0}{2}] \bigwedge \text{limit}[g_0, b_0, \frac{\epsilon_0}{2}]$.

We prove the individual conjunctive parts of (l3+.A):

Proof of (l3+.A.1) $\text{limit}[f_0, a_0, \frac{\epsilon_0}{2}]$:

Formula (l3+.A.1) is proved because it is an instance of (l2:.C).

Proof of (l3+.A.2) $\text{limit}[g_0, b_0, \frac{\epsilon_0}{2}]$:

Formula (l3+.A.2) is proved because it is an instance of (l2:.C).

Alternative proof 2:

Alternative proof 2:

## ■ Proof Example: Dot Plus

Prove:

(.+) $\forall_{m,n} (m^+ + n = (m + n)^+)$,

under the assumptions:

( +0) $\forall_{m} (m + 0 = m)$,

( +.) $\forall_{m,n} (m + n^+ = (m + n)^+)$.

There are 2 alternatives for proceeding with the proof.

Alternative 1: using Simplifier

Alternative 2: using NIP

We prove (.+) by induction on m.

Induction Base:

(.+.IB) $\forall_{n} (0^+ + n = (0 + n)^+)$.

There are 2 alternatives for proceeding with the proof.

Alternative 1: using Simplifier

Alternative 2: using NIP

We prove (.+.IB) by induction on n.

Induction Base:

(.+.IB.IB) $0^+ + 0 = (0 + 0)^+$.

A proof by simplification of (.+.IB.IB) works.

Simplification of the lhs term:

$0^+ + 0 =$ by ( +0)

$0^+$ ⌋

Simplification of the rhs term:

$(0 + 0)^+ =$ by ( +0)

$0^+$ ⌋

Induction Step:

Induction Hypothesis:

(.+.IB.IH) $0^+ + n_4 = (0 + n_4)^+$

Induction Conclusion:

(.+.IB.IS) $0^+ + n_4{}^+ = (0 + n_4{}^+)^+$.

A proof by simplification of (.+.IB.IS) works.

Simplification of the lhs term:

$0^+ + n_4{}^+ =$ by ( +.)

$(0^+ + n_4)^+ =$ by (.+.IB.IH)

$((0 + n_4)^+)^+$ ⌋

Simplification of the rhs term:

$(0 + n_4{}^+)^+ =$ by ( +.)

$((0 + n_4)^+)^+$ ⌋

Induction Step:

Induction Hypothesis:

(.+.IH) $\underset{n}{\forall}\,(m_1{}^+ + n = (m_1 + n)^+)$

Induction Conclusion:

(.+.IS) $\underset{n}{\forall}\,((m_1{}^+)^+ + n = (m_1{}^+ + n)^+)$.

There are 2 alternatives for proceeding with the proof.

Alternative 1: using Simplifier

We take in (.+.IS) all variables arbitrary but fixed:

(.+.IS.F) $(m_1{}^+)^+ + n_5 = (m_1{}^+ + n_5)^+$

and simplify it.

Simplification of the lhs term:

$(m_1{}^+)^+ + n_5$ ⌋

Simplification of the rhs term:

$(m_1{}^+ + n_5)^+ =$ by (.+.IH)

$((m_1 + n_5)^+)^+$ ⌋

Hence, it is sufficient to prove:

(.+.IS.S) $\underset{n}{\forall}\,((m_1{}^+)^+ + n = ((m_1 + n)^+)^+)$.

There are 2 alternatives for proceeding with the proof.

Alternative 1: using Simplifier

Alternative 2: using NIP

We prove (.+.IS.S) by induction on n.

Induction Base:

(.+.IS.S.IB) $(m_1{}^+)^+ + 0 = ((m_1 + 0)^+)^+$.

A proof by simplification of (.+.IS.S.IB) works.

Simplification of the lhs term:

$(m_1{}^+)^+ + 0 =$ by ( +0)

$(m_1{}^+)^+$ ⌋

Simplification of the rhs term:

$((m_1 + 0)^+)^+ =$ by ( +0)

$(m_1{}^+)^+$ ⌋

Induction Step:

Induction Hypothesis:

(.+.IS.S.IH) $(m_1{}^+)^+ + n_7 = ((m_1 + n_7)^+)^+$

Induction Conclusion:

(.+.IS.S.IS) $(m_1{}^+)^+ + n_7{}^+ = ((m_1 + n_7{}^+)^+)^+$.

A proof by simplification of (.+.IS.S.IS) works.

Simplification of the lhs term:

$(m_1{}^+)^+ + n_7{}^+ =$ by ( +.)

$((m_1{}^+)^+ + n_7)^+ =$ by (.+.IS.S.IH)

$(((m_1 + n_7)^+)^+)^+$ ⌋

Simplification of the rhs term:

$((m_1 + n_7{}^+)^+)^+ =$ by ( +.)

$(((m_1 + n_7)^+)^+)^+$ ⌋

Alternative 2: using NIP