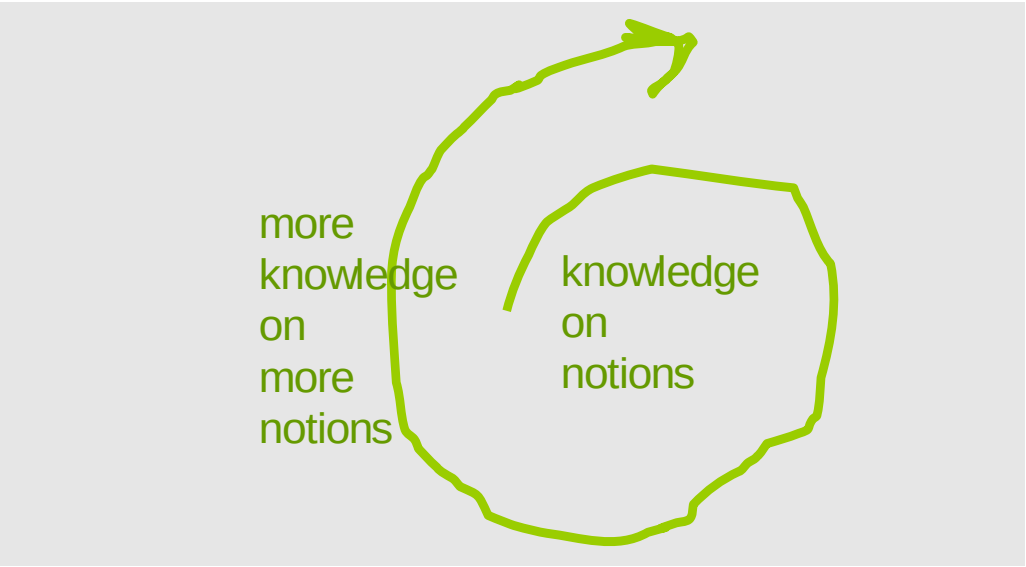
THƎOREM∀:
A System for Formal Mathematics

Bruno Buchberger
RISC, Austria

Acknowledgement to *Theorema* Group:
T. Jebelean, W. Windsteiger, T. Kutsia, M. Rosenkranz, F. Piroi; and Ph.D. students.

The Goal of Theorema

Support Mathematical Theory Exploration



Mathematical Theory Exploration

Starting from a body of knowledge on some concepts (e.g. +, * on numbers and functions)

- invent (axioms, definitions for) new concepts (operations: predicates, functions) (e.g. limit)
- invent and prove properties of notions

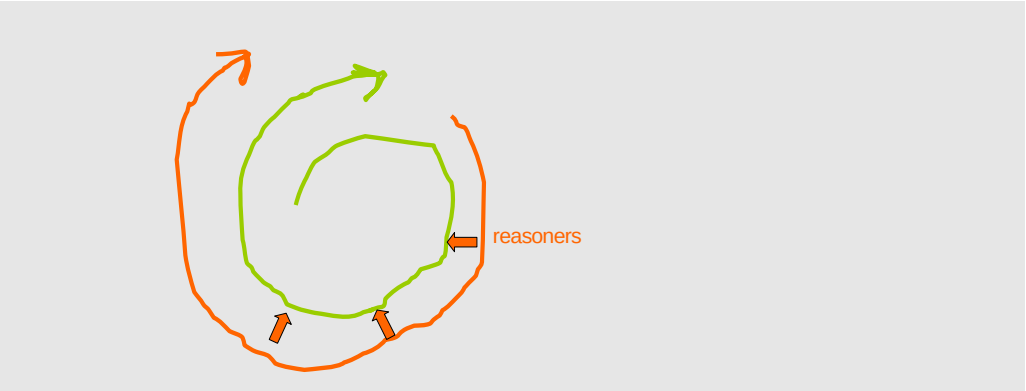
- invent problems about notions
- invent methods (algorithms) for problems and prove their correctness
- compute (apply algorithms to data)
- organize, store, and retrieve knowledge

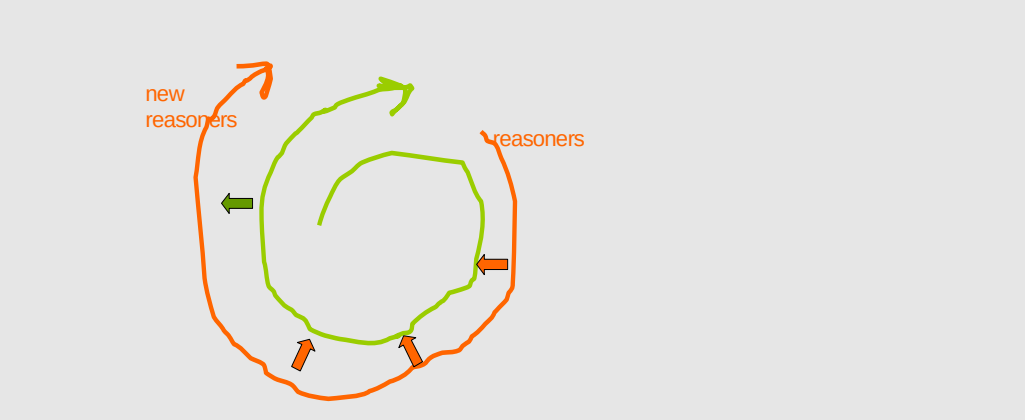
Current computer algebra systems support some of these aspects.

Current reasoning systems support some other of these aspects.

Goal: Support all these aspects.
Calculemus, MKM etc. are research networks that aim at creating integrated systems.
Group members: MIZAR, OMEGA, EDIN, ISABELLE, NUPRL, ...
Theorema is one of the (founding) groups of these networks.

A Never Ending Endeavor on Two Floors





Theorema : A Coherent Frame for Mathematical Theory Exploration

One logic frame: a version of predicate logic.

One software technologic frame: implementation language ("meta language") is Mathematica:

A. *Theorema* reasoners are programmed in Mathematica (in the future: in *Theorema* itself)

B. Mathematica as a front-end (flexible syntax, hierarchical notebooks, graphics etc.)

Example: Syntax and Manipulating Mathematical Texts

Initialize Theorema

```
Needs["Theorema`"];
```

Predicate Logic Formulae, Labels, and Keywords

```
Definition["limit:", any[f, a],
    limit[f, a] ⟺ ∀_{ϵ>0} ∃_{N} ∀_{n≥N} |f[n] − a| < ϵ]
```

Note that logic "application" is denoted by brackets in *Theorema*!

```
Definition["limit:"]
```

$$\bullet\text{def}\Big[\text{limit:}, \bullet\text{range}[\bullet\text{simpleRange}[f], \bullet\text{simpleRange}[a]], \text{True}, \bullet\text{flist}\Big[\bullet\text{lf}\Big[, \text{limit}[f, a] \Leftrightarrow \forall_{\epsilon>0} \exists_{N} \forall_{n\geq N} (|f[n]-a|<\epsilon)\Big]\Big]\Big]$$

```
Definition["limit:"] // InputForm
```

```
•def["limit:", •range[•simpleRange[
   •var[f]], •simpleRange[•var[a]]], True,
 •flist[•lf["", ™Iff[limit[•var[f],
     •var[a]], ™ForAll[•range[
      •simpleRange[•var[∈]]],
     ™Greater[•var[∈], 0],
     ™Exists[•range[•simpleRange[
        •var[™N]]], True, ™ForAll[
       •range[•simpleRange[•var[n]]],
       ™GreaterEqual[•var[n], •var[™N]],
       ™Less[™BracketingBar[™Minus[
          •var[f][•var[n]], •var[a]]],
        •var[∈]]]]]]]]]
```

The identifier spaces of *Theorema* and Mathematica are kept disjoint!

Users can program their own syntax!

```
Proposition["limit of sum", any[f, a, g, b],
    (limit[f, a] ∧ limit[g, b])  ⇒  limit[f + g, a + b]]
```

```
Definition["+:", any[f, g, x],
    (f + g)[x] = f[x] + g[x]]
```

```
Lemma["|+|", any[x, y, a, b, δ, ϵ],
    (|(x + y) − (a + b)| < (δ + ϵ))  ⟸  (|x − a| < δ ∧ |y − b| < ϵ)]
```

```
Lemma["max", any[m, M1, M2],
    m ≥ max[M1, M2]  ⇒  (m ≥ M1 ∧ m ≥ M2)]
```

■ Combining Formulae to "Theories"

```
Theory["limit",
    Definition["limit:"]
    Definition["+:"]
    Lemma["|+|"]
    Lemma["max"]
]
```

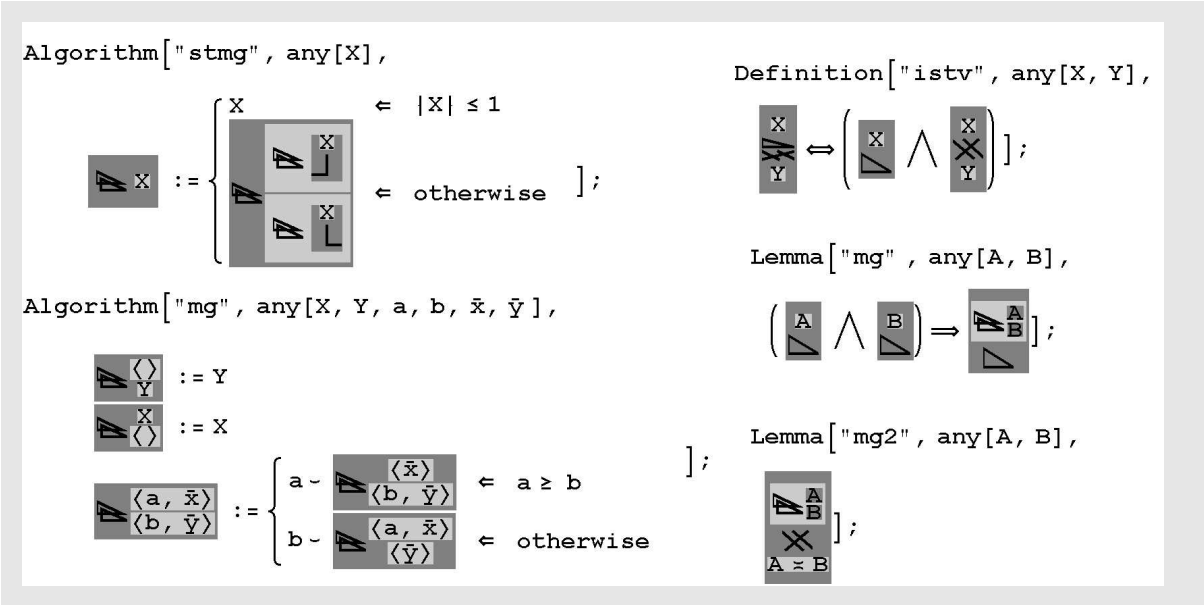
```
Theory["limit"]
```

```
•thr[limit, •range[], True, •flist[
    •def[limit:, •range[•simpleRange[f], •simpleRange[a]], True, •flist[•lf[, limit[f, a] ⇔ ∀_{ϵ>0} ∃_N ∀_{n≥N} (|f[n] − a| < ϵ)]]],
    •def[+:, •range[•simpleRange[f], •simpleRange[g], •simpleRange[x]], True, •flist[•lf[, (f + g)[x] = f[x] + g[x]]]],
    •lma[|+|, •range[•simpleRange[x], •simpleRange[y], •simpleRange[a], •simpleRange[b], •simpleRange[δ],
        •simpleRange[ϵ]], True, •flist[•lf[, |(x + y) − (a + b)| < δ + ϵ ⇐ (|x − a| < δ ∧ |y − b| < ϵ)]]],
    •lma[max, •range[•simpleRange[m], •simpleRange[M1], •simpleRange[M2]], True,
      •flist[•lf[, m ≥ max[M1, M2] ⇒ m ≥ M1 ∧ m ≥ M2]]]]
```

```
Theory["Test",
  Theory["limit"]
  Lemma["max"]
]
```

■ Logicographic Symbols (Invent Arbitrary New Symbols and Two-dimensional Nesting)

Logicographic Symbols: In *Theorema*, users can invent, design, declare, and use their own mathematical symobls:



Example: Computing (Within Predicate Logic)

Programs are Predicate Logic Formulae

Definition["addition", any[m, n],

$m + 0 = m$ " +0:"

$m + n^{+} = (m + n)^{+}$ " + .:"]

Definition["addition"] // InputForm

```
•def["addition", •range[•simpleRange[
   •var[m]], •simpleRange[•var[n]]], True,
 •flist[•lf[" +0:", ™Equal[™Plus[•var[m],
     0], •var[m]]], •lf[" + .:",
   ™Equal[™Plus[•var[m], ™SuperPlus[
      •var[n]]], ™SuperPlus[™Plus[•var[m],
      •var[n]]]]]]]
```

Compute[$0^{+++} + 0^{++}$, using → Definition["addition"]]

5

```
Compute[0^+++ + 0^++, using → ⟨⟩]
```

$((0^+)^+)^+ + (0^+)^+$

```
Definition["multiplication", any[m, n],
    m ∗ 0 = 0                    " ∗0:"
    m ∗ n^+ = (m ∗ n) + m        " ∗ .:"]
```

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

$((0^+)^+)^+ * (0^+)^+$

No hidden knowledge is used!

```
Compute[0^+++ ∗ 0^++++, using → ⟨Definition["addition"], Definition["multiplication"]⟩]
```

$$\left(\left(\left(\left(\left(\left(\left(\left(\left(\left(0^+\right)^+\right)^+\right)^+\right)^+\right)^+\right)^+\right)^+\right)^+\right)^+\right)^+$$

```
Compute[0^+++ ∗ 0^++++, using → ⟨Definition["multiplication"]⟩]
```

$(((0 + ((0^+)^+)^+) + ((0^+)^+)^+) + ((0^+)^+)^+) + ((0^+)^+)^+$

■ Using the Theory Construct

```
Theory["arithmetic",
  Definition["addition"]
  Definition["multiplication"]]
```

```
Compute[0^+++ ∗ 0^++++, using → Theory["arithmetic"]]
```

$$\left(\left(\left(\left(\left(\left(\left(\left(\left(\left(0^+\right)^+\right)^+\right)^+\right)^+\right)^+\right)^+\right)^+\right)^+\right)^+\right)^+$$

■ Defining Hidden Knowledge

```
Use[Theory["arithmetic"]]
```

$$\text{Compute}\left[0^{+++} * 0^{++++}\right]$$

$$\left(\left(\left(\left(\left(\left(\left(\left(\left(0^{+}\right)^{+}\right)^{+}\right)^{+}\right)^{+}\right)^{+}\right)^{+}\right)^{+}\right)^{+}\right)^{+}$$

```
Use[⟨⟩]
```

$$\text{Compute}\left[0^{+++} * 0^{++++}\right]$$

$$\left(\left(0^{+}\right)^{+}\right)^{+} * \left(\left(\left(0^{+}\right)^{+}\right)^{+}\right)^{+}$$

■ Avoiding to Write "Compute" All the Time

"*Theorema* Computing Sessions" possible like in Mathematica, Maple, etc. Not discussed here.

■ Using Built-in Knowledge Explicitly

```
Compute[3 * 4, using → Theory["arithmetic"]]
```

$$3 * 4$$

```
Compute[3 * 4, using → Theory["arithmetic"]] // InputForm
```

```
™Times[3, 4]
```

Built–in["Mathematica decimal arithmetic",
™Times → Times
™Plus → Plus]

```
Compute[3 * 4, using → ⟨Built–in["Mathematica decimal arithmetic"]⟩]
```

```
12
```

```
Compute[3 + 4 * 5!, using → ⟨Built–in["Mathematica decimal arithmetic"]⟩]
```

```
3 + 4 * 5!
```

```
?? Built–in
```

```
Built–in[name] returns the collection of built–in definitions addressed by `name`.
```

```
Built–in[AllOperators] =
  •bui[AllOperators, •range[], True, •flist[•lf[^, ^ = ^], •lf[*, * = *], •lf[–, – = –], •lf[–, – = –], •lf[+, + = +], •lf[+, + = +],
      •lf[=, = = ==], •lf[<, < = <], •lf[>, > = >], •lf['', ' = '], •lf[/, / = /], •lf[*=, *= = *=], •lf[–=, –= = –=],
      •lf[+=, += = +=], •lf[/=, /= = /=], •lf[:=, := = :=], •lf[SuperMinus, ™SuperMinus = SuperMinus],
      •lf[SuperPlus, ™SuperPlus = SuperPlus], •lf[⇓, ⇓ = ⇓], •lf[⇑, ⇑ = ⇑], •lf[⇕, ⇕ = ⇕], •lf[↓, ↓ = ↓], •lf[↓, ↓ = ↓],
      •lf[↕, ↕ = ↕], •lf[↧, ↧ = ↧], •lf[↧, ↧ = ↧], •lf[↓, ↓ = ↓], •lf[↓, ↓ = ↓], •lf[↓, ↓ = ↓], •lf[1, 1 = 1], •lf[1, 1 = 1],
      •lf[↑, ↑ = ↑], •lf[←, ← = ←], •lf[←, ← = ←], •lf[←, ← = ←], •lf[↓↑, ↓↑ = ↓↑], •lf[↑, ↑ = ↑],
      •lf[↓, ↓ = ↓], •lf[↓, ↓ = ↓], •lf[↓, ↓ = ↓], •lf[↓, ↓ = ↓], •lf[↑, ↑ = ↑], •lf[↑, ↑ = ↑], •lf[↑, ↑ = ↑], •lf[↓, ↓ = ↓], •lf[↑, ↑ = ↑],
      •lf[√, √ = √], •lf[↑, ↑ = ↑], •lf[↑, ↑ = ↑], •lf[↑, ↑ = ↑], •lf[↕, ↕ = ↕], •lf[↕, ↕ = ↕], •lf[↑, ↑ = ↑],
      •lf[D, D = D], •lf[d, d = d], •lf[∇, ∇ = ∇], •lf[□, □ = □], •lf[∘, ∘ = ∘], •lf[⊙, ⊙ = ⊙], •lf[×, × = ×], •lf[\, \ = \],
      •lf[⋄, ⋄ = ⋄], •lf[∧, ∧ = ∧], •lf[∨, ∨ = ∨], •lf[⊗, ⊗ = ⊗], •lf[·, · = ·], •lf[*, * = *], •lf[II, II = II],
      •lf[II, II = II], •lf[≀, ≀ = ≀], •lf[⌒, ⌒ = ⌒], •lf[⌣, ⌣ = ⌣], •lf[⊖, ⊖ = ⊖], •lf[⊕, ⊕ = ⊕], •lf[∓, ∓ = ∓],
      •lf[∓, ∓ = ∓], •lf[±, ± = ±], •lf[±, ± = ±], •lf[⨅, ⨅ = ⨅], •lf[⨆, ⨆ = ⨆], •lf[⊎, ⊎ = ⊎], •lf[≡, ≡ = ≡],
      •lf[≍, ≍ = ≍], •lf[≐, ≐ = ≐], •lf[∥, ∥ = ∥], •lf[≈, ≈ = ≈], •lf[≒, ≒ = ≒], •lf[≥, ≥ = ≥], •lf[≧, ≧ = ≧],
      •lf[≥, ≥ = ≥], •lf[≫, ≫ = ≫], •lf[≧, ≧ = ≧], •lf[≥, ≥ = ≥], •lf[≥, ≥ = ≥], •lf[≎, ≎ = ≎],
      •lf[≏, ≏ = ≏], •lf[◁, ◁ = ◁], •lf[◁, ◁ = ◁], •lf[⊴, ⊴ = ⊴], •lf[≤, ≤ = ≤], •lf[≦, ≦ = ≦], •lf[≦, ≦ = ≦],
      •lf[≤, ≤ = ≤], •lf[≪, ≪ = ≪], •lf[≤, ≤ = ≤], •lf[≤, ≤ = ≤], •lf[≫, ≫ = ≫], •lf[≪, ≪ = ≪],
      •lf[<, < = <], •lf[≤, ≤ = ≤], •lf[≤, ≤ = ≤], •lf[≤, ≤ = ≤], •lf[∷, ∷ = ∷], •lf[∝, ∝ = ∝], •lf[⇋, ⇋ = ⇋],
      •lf[▷, ▷ = ▷], •lf[⧐, ⧐ = ⧐], •lf[≥, ≥ = ≥], •lf[>, > = >], •lf[≥, ≥ = ≥], •lf[≥, ≥ = ≥],
      •lf[≳, ≳ = ≳], •lf[~, ~ = ~], •lf[≃, ≃ = ≃], •lf[≅, ≅ = ≅], •lf[≈, ≈ = ≈], •lf[⊥, ⊥ = ⊥], •lf[∣, ∣ = ∣],
      •lf[⇐, ⇐ = ⇐], •lf[:⇔, :⇔ = :⇔], •lf[⇔, ⇔ = ⇔], •lf[↽, ↽ = ↽], •lf[↼, ↼ = ↼], •lf[↼, ↼ = ↼],
      •lf[↼, ↼ = ↼], •lf[↽, ↽ = ↽], •lf[↽, ↽ = ↽], •lf[→, → = →], •lf[→, → = →], •lf[⇄, ⇄ = ⇄],
      •lf[↦, ↦ = ↦], •lf[↦, ↦ = ↦], •lf[→, → = →], •lf[→, → = →], •lf[←, ← = ←], •lf[→, → = →],
      •lf[↖, ↖ = ↖], •lf[↗, ↗ = ↗], •lf[∈, ∈ = ∈], •lf[∋, ∋ = ∋], •lf[⊏, ⊏ = ⊏], •lf[⊑, ⊑ = ⊑], •lf[⊐, ⊐ = ⊐],
      •lf[⊒, ⊒ = ⊒], •lf[⊂, ⊂ = ⊂], •lf[⊆, ⊆ = ⊆], •lf[⊃, ⊃ = ⊃], •lf[⊇, ⊇ = ⊇], •lf[¬, ¬ = ¬], •lf[∧, ∧ = ∧],
      •lf[∨, ∨ = ∨], •lf[∨!, ⋁! = ⋁!], •lf[∨×, ⋁× = ⋁×], •lf[⇒, ⇒ = ⇒], •lf[⊃, ⊃ = ⊃], •lf[⊣, ⊣ = ⊣],
      •lf[⊨, ⊨ = ⊨], •lf[⊤, ⊤ = ⊤], •lf[⊣, ⊣ = ⊣], •lf[⊢, ⊢ = ⊢], •lf[:, : = :], •lf[∵, ∵ = ∵], •lf[∴, ∴ = ∴]]]

Built–in[Connectives] := •bui[Connectives, •range[], True,
    •flist[•lf[=, = = ==], •lf[<, < = <], •lf[>, > = >], •lf[:=, := = :=], •lf[OrUnique, ⋁! = ⋁!],
      •lf[Xor, ⋁× = ⋁×], •lf[||||, ™CaseDistinction = CaseDistinction], •lf[≥, ≥ = ≥], •lf[≤, ≤ = ≤], •lf[⇐, ⇐ = ⇐],
      •lf[:⇔, :⇔ = :⇔], •lf[⇔, ⇔ = ⇔], •lf[¬, ¬ = ¬], •lf[∧, ∧ = ∧], •lf[∨, ∨ = ∨], •lf[⇒, ⇒ = ⇒]]]

Built–in[Containers] =
```

```
  •bui[Containers, •range[], True, •flist[•lf[Contains, ™Contains = Contains], •lf[⌢, ⌢ = ⌢], •lf[⌣, ⌣ = ⌣],
      •lf[⋉, ⋉ = ⋉], •lf[⇤, ™ReplaceElement = ReplaceElement], •lf[↦, ™InsertElement = InsertElement],
      •lf[→⇥, ™DeleteElement = DeleteElement], •lf[∈, ∈ = ∈]]]

Built–in[Mathematica decimal arithmetic] =
  •bui[Mathematica decimal arithmetic, •range[], True, •flist[•lf[™Plus, + = +], •lf[™Times, ∗ = ∗]]]

Built–in[Number Domains] := •bui[Number Domains, •range[], True,
    •flist[•lf[ℂ, ℂ = ℂ], •lf[𝕀, 𝕀 = 𝕀], •lf[ℕ, ℕ = ℕ], •lf[ℙ, ℙ = ℙ], •lf[ℚ, ℚ = ℚ], •lf[ℝ, ℝ = ℝ]]]

Built–in[Numbers] :=
  •bui[Numbers, •range[], True, •flist[•lf[!, ™Factorial = Factorial], •lf[^, ^ = ^], •lf[∗, ∗ = ∗], •lf[−, − = −],
      •lf[+, + = +], •lf[=, = = ==], •lf[<, < = <], •lf[>, > = >], •lf[(), ™Binomial = Binomial],
      •lf[:=, := = :=], •lf[inf, ™Infimum = Infimum], •lf[IsInteger, ™IsInteger = IsInteger],
      •lf[IsNatural, ™IsNatural = IsNatural], •lf[IsPrime, ™IsPrime = IsPrime], •lf[IsRational, ™IsRational = IsRational],
      •lf[max, ™Maximum = Maximum], •lf[min, ™Minimum = Minimum], •lf[sup, ™Supremum = Supremum],
      •lf[SuperMinus, ™SuperMinus = SuperMinus], •lf[SuperPlus, ™SuperPlus = SuperPlus],
      •lf[||, ™BracketingBar = BracketingBar], •lf[√, √ = √], •lf[÷, / = /], •lf[≥, ≥ = ≥], •lf[≤, ≤ = ≤], •lf[ı, ı = ı]]]

Built–in[Operators] = •bui[Operators, •range[], True,
    •flist[•lf[^, ^ = ^], •lf[∗, ∗ = ∗], •lf[−, − = −], •lf[−, − = −], •lf[+, + = +], •lf[+, + = +], •lf[=, = = ==],
      •lf[<, < = <], •lf[>, > = >], •lf[/, / = /], •lf[:=, := = :=], •lf[SuperMinus, ™SuperMinus = SuperMinus],
      •lf[SuperPlus, ™SuperPlus = SuperPlus], •lf[√, √ = √], •lf[\, \ = \], •lf[⌢, ⌢ = ⌢], •lf[⌣, ⌣ = ⌣], •lf[⋉, ⋉ = ⋉],
      •lf[≥, ≥ = ≥], •lf[≤, ≤ = ≤], •lf[ı, ı = ı], •lf[⇐, ⇐ = ⇐], •lf[:⇔, :⇔ = :⇔], •lf[⇔, ⇔ = ⇔],
      •lf[∈, ∈ = ∈], •lf[∋, ∋ = ∋], •lf[⊂, ⊂ = ⊂], •lf[⊆, ⊆ = ⊆], •lf[⊃, ⊃ = ⊃], •lf[⊇, ⊇ = ⊇],
      •lf[¬, ¬ = ¬], •lf[∧, ∧ = ∧], •lf[∨, ∨ = ∨], •lf[∨!, ⋁! = ∨!], •lf[∨×, ⋁× = ∨×], •lf[⇒, ⇒ = ⇒]]]

Built–in[Quantifiers] := •bui[Quantifiers, •range[], True,
    •flist[•lf[#, ™NumberOf = NumberOf], •lf[|, ™SequenceOf = SequenceOf], •lf[inf, ™Infimum = Infimum],
      •lf[inf, ™InfimumOf = InfimumOf], •lf[max, ™Maximum = Maximum], •lf[max, ™MaximumOf = MaximumOf],
      •lf[min, ™Minimum = Minimum], •lf[min, ™MinimumOf = MinimumOf], •lf[sup, ™Supremum = Supremum],
      •lf[sup, ™SupremumOf = SupremumOf], •lf[where, ™WithLocalValues = WithLocalValues], •lf[λ, ™Lambda = Lambda],
      •lf[μ, ™TheFirst = TheFirst], •lf[Ω, ™OperationOf = OperationOf], •lf[•domainRange, •domainRange = ••domainRange],
      •lf[•integerRange, •integerRange = ••integerRange], •lf[•predRange, •predRange = ••predRange],
      •lf[•range, •range = ••range], •lf[•setRange, •setRange = ••setRange], •lf[•simpleRange, •simpleRange = ••simpleRange],
      •lf[⟨|⟩, ™ConcatenationOf = ConcatenationOf], •lf[⟨|⟩, ™TupleOf = TupleOf], •lf[∏, ™ProductOf = ProductOf],
      •lf[∑, ™SumOf = SumOf], •lf[⋂, ™IntersectionOf = IntersectionOf], •lf[⋃, ™UnionOf = UnionOf],
      •lf[∃, ™Exists = Exists], •lf[∃!, ™ExistsUnique = ExistsUnique], •lf[∀, ™ForAll = ForAll],
      •lf[∄, ™NotExists = NotExists], •lf[∋, ™SuchThat = SuchThat], •lf[{|}, ™SetOf = SetOf]]]

Built–in[Sets] := •bui[Sets, •range[], True, •flist[•lf[^, ^ = ^], •lf[=, = = ==], •lf[|, ™SequenceOf = SequenceOf],
      •lf[{}, ™Set = TheSet], •lf[:=, := = :=], •lf[æ, ™AnElement = æ], •lf[IsSet, ™IsSet = IsSet], •lf[𝒫, ™Powerset = 𝒫],
      •lf[•integerRange, •integerRange = ••integerRange], •lf[•range, •range = ••range], •lf[•setRange, •setRange = ••setRange],
      •lf[||, ™BracketingBar = BracketingBar], •lf[×, × = ×], •lf[\, \ = \], •lf[⋂, ™IntersectionOf = IntersectionOf],
      •lf[⋂, ™Intersection = ⋂Intersection], •lf[⋃, ™UnionOf = UnionOf], •lf[⋃, ™Union = ⋃Union], •lf[∈, ∈ = ∈],
      •lf[∋, ∋ = ∋], •lf[⊂, ⊂ = ⊂], •lf[⊆, ⊆ = ⊆], •lf[⊃, ⊃ = ⊃], •lf[⊇, ⊇ = ⊇], •lf[{|}, ™SetOf = SetOf]]]

Built–in[Tuples] := •bui[Tuples, •range[], True,
    •flist[•lf[=, = = ==], •lf[\_, ™Subscript = Subscript], •lf[:=, := = :=], •lf[IsTuple, ™IsTuple = IsTuple],
      •lf[⟨⟩, ™Tuple = AngleBracket], •lf[||, ™BracketingBar = BracketingBar], •lf[⌢, ⌢ = ⌢], •lf[⌣, ⌣ = ⌣],
      •lf[⋉, ⋉ = ⋉], •lf[⇤, ™ReplaceElement = ReplaceElement], •lf[↦, ™InsertElement = InsertElement],
      •lf[→⇥, ™DeleteElement = DeleteElement], •lf[∈, ∈ = ∈], •lf[∋, ∋ = ∋]]]

Built–in[{Theorema`Language`Semantics`FormalText`Private`name_String
      Theorema`Language`Semantics`FormalText`Private`b: ({{_, _, _} ..} | {{_•bui} ..})}] :=
  (SetCellTag[EnvironmentLabel[Theorema`Language`Semantics`FormalText`Private`name, •bui]]; DefineBuiltin[
      Theorema`Language`Semantics`FormalText`Private`name, Theorema`Language`Semantics`FormalText`Private`b];)
```

```
Built–in[Theorema`Language`Semantics`FormalText`Private`name_String] :=
  (PrintMessage[Theorema::noEnvironment, •bui, Theorema`Language`Semantics`FormalText`Private`name];
    EmptyEnvironment[•bui])

Built–in[Theorema`Language`Semantics`UserLanguage`Private`s_] :=
  (PrintMessage[Theorema::noEnvironment, •bui, Theorema`Language`Semantics`UserLanguage`Private`s];
    EmptyEnvironment[•bui])
```

```
Compute[3 + 4 * 5!, using → ⟨Built–in["Numbers"]⟩]
```

```
483
```

Thus, if one wants, the full power of the Mathematica algorithm library can be used from within *Theorema* programs.

■ Sequence Variables

(For the metamathematics of sequence variables see [Kutsia, Buchberger, MKM 2004])

Sequence variables allow flexible arity operations.

```
Definition["reverse", any[x, x̄],
    r[⟨⟩] = ⟨⟩                    "reverse empty"
    r[⟨x, x̄⟩] = r[⟨x̄⟩] ⌢ x        "reverse general"]
```

```
Compute[r[⟨⟩], using → ⟨Definition["reverse"]⟩]
```

```
⟨⟩
```

```
Compute[r[⟨3, 2, 4, 2, 1⟩], using → ⟨Definition["reverse"]⟩]
```

```
((((⟨⟩ ⌢ 1) ⌢ 2) ⌢ 4) ⌢ 2) ⌢ 3
```

```
Definition["compose", any[x, x̄],
    x ⌣ ⟨x̄⟩ = ⟨x, x̄⟩        "prepend"
    ⟨x̄⟩ ⌢ x = ⟨x̄, x⟩        "append"
    ⟨x̄⟩ ⌢ ⟨ȳ⟩ = ⟨x̄, ȳ⟩     "concatenate"]
```

```
Compute[r[⟨3, 2, 4, 2, 1⟩], using → ⟨Definition["compose"], Definition["reverse"]⟩]
```

```
⟨1, 2, 4, 2, 3⟩
```

Sequence Variables Allow Complicated Pattern Matching

```
Definition["first lines of a propositional prover", any[g, g1, g2, ā, b, b1, b2, b̄, c̄, d̄, k],
    prove[g, ⟨ā, g, b̄⟩] = T                                  "goal is among assumptions"
    prove[g, ⟨ā, b, c̄, N[b], d̄⟩] = T                          "contradicting assumptions"
    prove[A[g1, g2], k] = and[prove[g1, k], prove[g2, k]]    "prove conjunction"
    prove[g, ⟨ā, A[b1, b2], c̄⟩] = prove[g, ⟨ā, b1, b2, c̄⟩]    "use conjunction"
]
```

```
Use[⟨Definition["first lines of a propositional prover"]⟩]
```

```
Compute[prove[X < Y, ⟨F[U, V], G[U], X < Y, F[V, U]⟩]]
```

```
T
```

```
Compute[prove[X < Y, ⟨F[U, V], G[U], X > Y, F[V, U]⟩]]
```

```
prove[X < Y, ⟨F[U, V], G[U], X > Y, F[V, U]⟩]
```

```
Compute[prove[X < Y, ⟨F[U, V], G[U], X > Y, N[G[U]], F[V, U]⟩]]
```

```
T
```

```
Compute[prove[X < Y, ⟨F[U, V], G[U], X > Y, N[G[A]], F[V, U]⟩]]
```

```
prove[X < Y, ⟨F[U, V], G[U], X > Y, N[G[A]], F[V, U]⟩]
```

```
Compute[prove[A[X < Y, X > Y], ⟨F[U, V], G[U], X > Y, N[G[A]], F[V, U]⟩]]
```

```
and[prove[X < Y, ⟨F[U, V], G[U], X > Y, N[G[A]], F[V, U]⟩], T]
```

Bounded Quantifiers

```
Use[⟨⟩]
```

```
Use[⟨Built–in["Quantifiers"], Built–in["Connectives"], Built–in["Numbers"], Built–in["Tuples"], Built–in["Sets"]⟩]
```

$$\text{Compute}\left[\underset{i=-10,\ldots,10}{\forall} i \leq 0\right]$$

False

$$\text{Compute}\left[\underset{i=-10,\ldots,10}{\exists} i \leq 0\right]$$

True

$$\text{Compute}\left[\left\{(2\,i) \underset{i\in\{1,2,3\}}{|}\right\}\right]$$

{2, 4, 6}

$$\text{Compute}\left[\left\{i \underset{i=1,\ldots,20}{|} \text{is–prime}[i]\right\}\right]$$

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

Compute[{2, 3, 1, 2, 3} = {1, 2, 3, 3, 3}]

True

Compute[{2, 3, 1, 2, 3}]

{1, 2, 3}

Compute[{1, 2, 3, 3, 3}]

{1, 2, 3}

Compute[{1, 2, 1, 3} = {2, 1, 4, 3}]

False

Definition["some quantifier programs", any[n, X, Y, A, B],

$$\mathrm{IP}[n] \Leftrightarrow \left(\underset{d=1,\ldots,n}{\forall} (d \mid n) \Longrightarrow ((d = 1) \vee (d = n))\right)$$

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

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

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

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

$$\text{Ꝑ}[A] := \begin{cases} \{\{\}\} & \Leftarrow \quad A = \{\} \\ \text{where}\left[a = \text{æ}[A],\ P = \text{Ꝑ}[A \ominus \{a\}],\ P \bigcup \left\{(\{a\} \bigcup B) \underset{B \in P}{|}\right\}\right] & \Leftarrow \quad \text{otherwise} \end{cases}$$

]

Principle: Predicate logic definitions and (proved) propositions can also be read as programs!

```
Use[⟨⟩]
```

```
Use[⟨Built–in["Quantifiers"], Built–in["Connectives"], Built–in["Numbers"],
    Built–in["Tuples"], Built–in["Sets"], Definition["some quantifier programs"]⟩]
```

```
Compute[IP[1]]
```

```
True
```

```
Compute[IP[28]]
```

```
False
```

```
Compute[⟨1, 1, 1, 1, 1⟩ ⊕ ⟨3, 3, 3, 3, 3⟩]
```

```
⟨4, 4, 4, 4, 4⟩
```

```
Compute[⟨1, 1, 1, 1, 1⟩ ≼ ⟨3, 3, 3, 3, 3⟩]
```

```
True
```

```
Compute[⟨1, 1, 8, 1, 1⟩ ≼ ⟨3, 3, 3, 3, 3⟩]
```

```
False
```

```
Compute[{1, 2, 3} ∖ {2, 3, 4, 5}]
```

```
{1}
```

```
Compute[{3, 2, 33} ⊆ {2, 3, 4, 5}]
```

```
False
```

```
Compute[℘[{}]]
```

```
{{}}
```

```
Compute[℘[{3}]]
```

```
{{}, {3}}
```

```
Compute[℘[{1, 3}]]
```

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

```
Compute[℘[{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}}
```

■ Functors

■ Some Internal Preparations

The following two Mathematica functions will be needed in the definition of arithmetic modulo the prime p:

```
Clear[PFInverse, PFQuotient]
PFInverse[x_, p_] := ExtendedGCD[x, p][[2, 1]]
PFQuotient[x_, y_, p_] := Mod[x PFInverse[y, p], p]
```

```
PFQuotient[1, 3, 5]
```

```
2
```

In the current version of *Theorema*, the following Mathematica function definition must be entered

```
|⟨x___?IsSequenceFree⟩| := Length[{x}]
```

for completing a definition in the *Theorema* package.

Now we make a couple of Mathematica functions available in *Theorema*, including the two new ones from above,

```
Built–in["Mathematica general functions",

  is–even → EvenQ
  is–odd → OddQ
  mod → Mod
  gcd → GCD
  lcm → LCM
  sign → Sign
  maximum → Max
  quotient → Quotient
  pf–inverse → PFInverse
  pf–quotient → PFQuotient
]
```

```
General::spell1 : Possible spelling error: new symbol name "mod" is similar to existing symbol "Mod".
```

```
General::spell1 : Possible spelling error: new symbol name "gcd" is similar to existing symbol "GCD".
```

```
General::spell1 : Possible spelling error: new symbol name "lcm" is similar to existing symbol "LCM".
```

```
General::stop : Further output of General::spell1 will be suppressed during this calculation.
```

The following function 'pairs' computes a list consisting of all pairs of elements in a given list:

Definition["Theorema general functions", any[a, x, $\overline{x}$],

pairs[⟨⟩] = ⟨⟩

pairs[⟨x, $\overline{x}$⟩] = $\left\langle \langle x, \langle \overline{x} \rangle_i \rangle \underset{i=1,\ldots,|\langle \overline{x} \rangle|}{|} \right\rangle \asymp$ pairs[⟨$\overline{x}$⟩]]

For example,

```
Use[⟨Built–in["Numbers"], Built–in["Tuples"], Built–in["Connectives"],
    Built–in["Quantifiers"], Definition["Theorema general functions"]⟩]
```

```
Compute[ pairs[⟨⟩]]
```

⟨⟩

```
Compute[ pairs[⟨1⟩]]
```

⟨⟩

```
Compute[ pairs[⟨1, 2, 3, 4, 5⟩]]
```

⟨⟨1, 2⟩, ⟨1, 3⟩, ⟨1, 4⟩, ⟨1, 5⟩, ⟨2, 3⟩, ⟨2, 4⟩, ⟨2, 5⟩, ⟨3, 4⟩, ⟨3, 5⟩, ⟨4, 5⟩⟩

```
Compute[ pairs[⟨1, 2, 1, 1, 2⟩]]
```

⟨⟨1, 2⟩, ⟨1, 1⟩, ⟨1, 1⟩, ⟨1, 2⟩, ⟨2, 1⟩, ⟨2, 1⟩, ⟨2, 2⟩, ⟨1, 1⟩, ⟨1, 2⟩, ⟨1, 2⟩⟩

■ What are Functors?

Functors are functions that map domains into domains.

A domain is a carrier together with operations on the carrier.

How can we code all the "enormous" information in a domain into one object?

In "ordinary" algebra: A domain D is a tuple (carrier, operation1, operation2, ...).

In *Theorema* (adopting the notion of "interpretation" from model theory):

D maps operators into operations.

D[+] is a (binary) operation.

D[+][x,y] is the operation D[+] applied to x and y. (We have Currying in *Theorema*.)

Syntax: D[+][x,y], in *Theorema*, can be written in the form $x \underset{D}{+} y$.

(*Theorema* functors are like SML functors but slightly more general: Carrier of D may be defined by arbitrary characteristic function $\underset{D}{\chi}$.)

A functor can now easily take entire domains D, ... as an arguments: For example functor F that describes the Cartesian product of two domains D and E:

$$F[D, E][+][\ \langle a, b\rangle,\ \langle x, y\rangle\ ] := \langle\ D[+][a, x],\quad E[+][b, y]\ \rangle$$

or

$$F[D, E] := \underset{N}{æ}\ \underset{a,b,x,y}{\forall}\ N[+][\ \langle a, b\rangle,\ \langle x, y\rangle\ ] := \langle\ D[+][a, x],\quad E[+][b, y]\ \rangle$$

or, with some syntax,

$$F[D, E] := \underset{N}{æ}\ \underset{a,b,x,y}{\forall}\ \langle a, b\rangle \underset{N}{+} \langle x, y\rangle := \left\langle\ a \underset{D}{+} x,\quad b \underset{E}{+} y\ \right\rangle$$

or, with some more syntax,

$$\begin{aligned} F[D, E] := \ & \text{Functor}\Big[\ N,\quad \text{any}[a, b, x, y], \\ & \langle a, b\rangle \underset{N}{+} \langle x, y\rangle := \left\langle\ a \underset{D}{+} x,\quad b \underset{E}{+} y\ \right\rangle \\ & \langle a, b\rangle \underset{N}{*} \langle x, y\rangle := \left\langle\ a \underset{D}{*} x,\quad b \underset{E}{*} y\ \right\rangle\ \Big] \\ & \ldots. \end{aligned}$$

or, even,

$$\begin{aligned} F[D, E] := \ & \text{Functor}\Big[\ N,\quad \text{any}[a, b, x, y, O], \\ & \langle a, b\rangle \underset{N}{O} \langle x, y\rangle := \left\langle\ a \underset{D}{O} x,\quad b \underset{E}{O} y\ \right\rangle\Big] \\ & \ldots. \end{aligned}$$

This is the form implemented in *Theorema*.

■ Why Are Functors Useful?

Computational Aspect: Economy of programming. No code repetition for building up "towers of domains".

Logical Aspect: Economy of proving. No proof repetition when building up "towers of domains".

If D is a monoid then Cartesian[D,D] is a monoid.

If D is a Groebner ring then Poly[D] is a Groebner ring.

If D is a partition then Equ[D] is an equivalence.

....

$$\text{is–monoid}[D] \Leftrightarrow \underset{\in_D[x],\in_D[y],\in_D[z]}{\underset{x,y,z}{\forall}} \begin{pmatrix} x \underset{D}{\circ} (y \underset{D}{\circ} z) = (x \underset{D}{\circ} y) \underset{D}{\circ} y \\ x \underset{D}{\circ} \underset{D}{1} = x \end{pmatrix}$$

is–monoid defines a "category"

The three fundamental notions for structuring mathematics: domains, categories, functors.

"Conservation Theorems":

If domain D is in category A

then F[D], the result of applying functor F to D, is in category B.

Theorema provides some tools for organizing such proofs.

■ Example: Rational Numbers

▫ Definition

Definition["rational numbers field", any[],

rational–numbers–field[] = Functor[N, any[x, y],

$\mathbb{s} = \langle\rangle$

$\underset{N}{\in}[x] \Leftrightarrow$ is–rational[x]

$\underset{N}{0} = 0$

$\underset{N}{1} = 1$

$x \underset{N}{+} y = x + y$

$\underset{N}{-} x = -x$

$x \underset{N}{-} y = x - y$

$x \underset{N}{*} y = x * y$

$\underset{N}{\text{inverse}}[x] = 1/x$

$x \underset{N}{/} y = x / y$

]]

Construct Domain

```
Definition["ℚ0", ℚ0 = rational–numbers–field[]]
```

```
Theory["T",
  Definition["Theorema general functions"]
  Definition["rational numbers field"]
  Definition["ℚ0"]
]
```

```
Use[⟨Built–in["Quantifiers"], Built–in["Connectives"], Built–in["Numbers"],
    Built–in["Tuples"], Built–in["Sets"], Built–in["Mathematica general functions"], Theory["T"]⟩]
```

Apply for Computation

$$\text{Compute}\left[\underset{\mathbb{Q}0}{\in}[2.0005]\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{Q}0}{\in}[2/5]\right]$$

True

$$\text{Compute}\left[\underset{\mathbb{Q}0}{0}\right]$$

0

$$\text{Compute}\left[\underset{\mathbb{Q}0}{1}\right]$$

1

$$\text{Compute}\left[5/2 \underset{\mathbb{Q}0}{+} 3\right]$$

$$\frac{11}{2}$$

$\text{Compute}\left[\underset{\mathbb{Q}0}{-}\,-2/3\right]$

$\frac{2}{3}$

$\text{Compute}\left[3\,\underset{\mathbb{Q}0}{-}\,-2/3\right]$

$\frac{11}{3}$

$\text{Compute}\left[5\,\underset{\mathbb{Q}0}{*}\,3/4\right]$

$\frac{15}{4}$

$\text{Compute}\left[\underset{\mathbb{Q}0}{\text{inverse}}[3/4]\right]$

$\frac{4}{3}$

$\text{Compute}\left[7/5\,\underset{\mathbb{Q}0}{/}\,(-2/3)\right]$

$\frac{-21}{10}$

$\text{Compute}\left[7/5\,\underset{\mathbb{Q}}{/}\,(-2/3)\right]$

$\frac{7}{5}\underset{\mathbb{Q}}{/}\left(\frac{-2}{3}\right)$

Example: Prime Fields

Definition

```
Definition["prime field", any[p],
  prime–field[p] = Functor[N, any[x, y],

      𝕊 = ⟨⟩
      ──────────────────────────

      ∈_N[x] ⇔ ⋀ { is–integer[x]
                 { 0 ≤ x
                 { x < p
      0_N = 0
      1_N = 1
      x +_N y = mod[x + y, p]
      −_N x = mod[−x, p]
      x −_N y = mod[x − y, p]
      x *_N y = mod[x * y, p]
      inverse_N[x] = pf–inverse[x, p]
      x /_N y = pf–quotient[x, y, p]

      ]]
```

Remark on the operations in the functor: inverse_N and $/_N$ refer to the function pf–inverse and pf–quotient, respectively, which we linked to the Mathematica function PFInverse and PFQuotient by the 'Built-in' construct. PFInverse and PFQuotient were defined above in Mathematica.

Construct Domain

```
Definition["𝔽5", 𝔽5 = prime–field[5]]
```

```
Theory["T",
  Definition["Theorema general functions"]
  Definition["prime field"]                ]
  Definition["𝔽5"]
```

```
Use[⟨Built–in["Quantifiers"], Built–in["Connectives"], Built–in["Numbers"],
    Built–in["Tuples"], Built–in["Sets"], Built–in["Mathematica general functions"], Theory["T"]⟩]
```

Apply for Computation

$$\text{Compute}\left[\underset{\mathbb{F}5}{\in}[0]\right]$$

True

$$\text{Compute}\left[\underset{\mathbb{F}5}{\in}[4]\right]$$

True

$$\text{Compute}\left[\underset{\mathbb{F}5}{\in}[5]\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{F}5}{\in}[-1]\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{F}5}{0}\right]$$

0

$$\text{Compute}\left[\underset{\mathbb{F}5}{1}\right]$$

1

$$\text{Compute}\left[4 \underset{\mathbb{F}5}{+} 1\right]$$

0

$\text{Compute}\left[\underset{\mathbb{F}5}{-}(2)\right]$

3

$\text{Compute}\left[1 \underset{\mathbb{F}5}{-} (2)\right]$

4

$\text{Compute}\left[4 \underset{\mathbb{F}5}{*} 3\right]$

2

$\text{Compute}\left[\underset{\mathbb{F}5}{\text{inverse}}[3]\right]$

2

$\text{Compute}\left[4 \underset{\mathbb{F}5}{/} 3\right]$

3

Example: Power Products as Tupels

Definition

Definition["tuples division semigroup lexical", any[k, D],

tuples–division–semigroup–lexical[k]

(∗ the division semigroup in tuples representation of length k with lexicographic ordering ∗) =

Functor[N, any[m, $\overline{m}$, n, $\overline{n}$, x, y, i],

$$\$ = \langle\rangle$$

$$\underset{N}{\in}[x] \Leftrightarrow \bigwedge \begin{cases} \text{is–tuple}[x] \\ |x| = k \\ \underset{i=1,\ldots,k}{\forall} \bigwedge \begin{cases} \text{is–integer}[x_i] \\ x_i \geq 0 \end{cases} \end{cases}$$

$$\underset{N}{1} = \left\langle 0 \underset{i=1,\ldots,k}{|} \right\rangle$$

$$x \underset{N}{*} y = \left\langle x_i + y_i \underset{i=1,\ldots,k}{|} \right\rangle$$

$$x \underset{N}{/} y = \left\langle x_i - y_i \underset{i=1,\ldots,k}{|} \right\rangle$$

$$\left(x \underset{N}{\mid} y\right) \Longleftrightarrow \underset{i=1,\ldots,k}{\forall} x_i \leq y_i$$

$$\underset{N}{\text{lcm}}[x, y] \ (* \text{ the least common multiple of } x \text{ and } y \ *) = \left\langle \text{maximum}[x_i, y_i] \underset{i=1,\ldots,k}{|} \right\rangle$$

$$\langle\rangle \underset{N}{>} \langle\rangle \Leftrightarrow \text{False}$$

$$\langle m, \overline{m}\rangle \underset{N}{>} \langle n, \overline{n}\rangle \Leftrightarrow \begin{cases} \text{True} & \Leftarrow \ m > n \\ \langle\overline{m}\rangle \underset{N}{>} \langle\overline{n}\rangle & \Leftarrow \ (m = n) \\ \text{False} & \Leftarrow \ \text{otherwise} \end{cases}$$

$$\underset{N}{\deg}[x] \ (* \text{ the degree of } x \ *) = \sum_{i=1,\ldots,k} x_i$$

]]

Note: '$\underset{N}{\mid}$' must be typed '\ [VerticalBar]'.

Construct Domain

Definition["𝕋3",

𝕋3 = tuples–division–semigroup–lexical[3]]

```
Theory["T",
  Definition["Theorema general functions"]
  Definition["tuples division semigroup lexical"]
  Definition["𝕋3"]]
```

```
Use[⟨Built–in["Quantifiers"], Built–in["Connectives"], Built–in["Numbers"],
    Built–in["Tuples"], Built–in["Sets"], Built–in["Mathematica general functions"], Theory["T"]⟩]
```

Apply for Computation

$$\text{Compute}\left[\underset{\mathbb{T}3}{\in}[\langle 1, 0, 2\rangle]\right]$$

True

$$\text{Compute}\left[\underset{\mathbb{T}3}{\in}[\langle 1, 0, -2\rangle]\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{T}3}{\in}[\langle 1, 0, 2, 4\rangle]\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{T}3}{1}\right]$$

$\langle 0, 0, 0\rangle$

$$\text{Compute}\left[\langle 1, 2, 4\rangle \underset{\mathbb{T}3}{*} \langle 2, 0, 2\rangle\right]$$

$\langle 3, 2, 6\rangle$

$$\text{Compute}\left[\langle 0, 0, 0\rangle \underset{\mathbb{T}3}{|} \langle 0, 0, 0\rangle\right]$$

True

$$\text{Compute}\left[\langle 0, 4, 1\rangle \underset{\mathbb{Z}3}{\mid} \langle 0, 5, 3\rangle\right]$$

True

$$\text{Compute}\left[\langle 0, 4, 4\rangle \underset{\mathbb{Z}3}{\mid} \langle 0, 5, 3\rangle\right]$$

False

$$\text{Compute}\left[\langle 0, 5, 1\rangle \underset{\mathbb{Z}3}{\mid} \langle 0, 5, 0\rangle\right]$$

False

$$\text{Compute}\left[\langle 55, 2, 4\rangle \underset{\mathbb{Z}3}{/} \langle 2, 0, 2\rangle\right]$$

$\langle 53, 2, 2\rangle$

$$\text{Compute}\left[\underset{\mathbb{Z}3}{\text{lcm}}[\langle 55, 2, 4\rangle, \langle 2, 0, 2\rangle]\right]$$

$\langle 55, 2, 4\rangle$

$$\text{Compute}\left[\underset{\mathbb{Z}3}{\text{lcm}}[\langle 55, 2, 4\rangle, \langle 2, 20, 9\rangle]\right]$$

$\langle 55, 20, 9\rangle$

$$\text{Compute}\left[\langle 55, 2, 4\rangle \underset{\mathbb{Z}3}{>} \langle 2, 0, 2\rangle\right]$$

True

$$\text{Compute}\left[\langle 55, 2, 4\rangle \underset{\mathbb{Z}3}{>} \langle 55, 2, 4\rangle\right]$$

False

$$\text{Compute}\left[\langle 55, 2, 4\rangle \underset{\mathbb{T}3}{>} \langle 55, 3, 4\rangle\right]$$

False

$$\text{Compute}\left[\langle 55, 3, 4\rangle \underset{\mathbb{T}3}{>} \langle 55, 3, 3\rangle\right]$$

True

$$\text{Compute}\left[\langle 55, 3, 4\rangle \underset{\mathbb{T}3}{>} \langle 55, 3, 2342\rangle\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{T}3}{\text{deg}}[\langle 55, 3, 4\rangle]\right]$$

62

Example: Polynomials Over a Coefficient Domain and a Power Product Domain

Definition

```
Definition["reduction polynomials", any[C, T],
  reduction–polynomials[C, T]
    (* polynomials as tuples of pairs of coefficients in C and terms in T with reduction operations *) =
```

Functor$\Big[$N, any[c, d, i, m, $\overline{m}$, n, $\overline{n}$, p, q, s, t],

$$\frac{\$ = \langle\rangle}{\underset{N}{\in}[p] \Leftrightarrow \left(\bigwedge\left\{\begin{array}{l} \text{is–tuple}[p] \\ \underset{j=1,\ldots,|p|}{\forall}\left(\bigwedge\left\{\begin{array}{l} \text{is–tuple}[p_j] \\ |p_j| = 2 \\ \underset{C}{\in}[(p_j)_1] \\ \underset{T}{\in}[(p_j)_2] \\ (p_j)_1 \neq \underset{C}{0} \end{array}\right.\right) \\ \underset{j=1,\ldots,|p|-1}{\forall} (p_j)_2 \underset{T}{>} (p_{j+1})_2 \end{array}\right.\right)}$$

$$\underset{N}{1} = \left\langle\left\langle \underset{C}{1}, \underset{T}{1}\right\rangle\right\rangle$$

$$\underset{N}{0} = \langle\rangle$$

$$\langle\rangle \underset{N}{+} q = q$$

$$p \underset{N}{+} \langle\rangle = p$$

$$\langle\langle c, s\rangle, \overline{m}\rangle \underset{N}{+} \langle\langle d, t\rangle, \overline{n}\rangle = \begin{cases} \langle c, s\rangle \frown \left(\langle\overline{m}\rangle \underset{N}{+} \langle\langle d, t\rangle, \overline{n}\rangle\right) & \Leftarrow \quad s \underset{T}{>} t \\ \langle d, t\rangle \frown \left(\langle\langle c, s\rangle, \overline{m}\rangle \underset{N}{+} \langle\overline{n}\rangle\right) & \Leftarrow \quad t \underset{T}{>} s \\ \left\langle c \underset{C}{+} d, s\right\rangle \frown \left(\langle\overline{m}\rangle \underset{N}{+} \langle\overline{n}\rangle\right) & \Leftarrow \quad c \neq \underset{C}{-} d \\ \langle\overline{m}\rangle \underset{N}{+} \langle\overline{n}\rangle & \Leftarrow \quad \text{otherwise} \end{cases}$$

$$\underset{N}{-}\langle\rangle = \langle\rangle$$

$$\underset{N}{-}\langle\langle c, s\rangle, \overline{m}\rangle = \left\langle \underset{C}{-} c, s\right\rangle \frown \left(\underset{N}{-}\langle\overline{m}\rangle\right)$$

$$p \underset{N}{-} q = p \underset{N}{+} \left(\underset{N}{-} q\right)$$

$$\langle\rangle \underset{N}{*} q = \langle\rangle$$

$$p \underset{N}{*} \langle\rangle = \langle\rangle$$

$$\langle\langle c, s\rangle, \overline{m}\rangle \underset{N}{*} \langle\langle d, t\rangle, \overline{n}\rangle = \left(\left\langle\left\langle c \underset{C}{*} d, s \underset{T}{*} t\right\rangle\right\rangle \underset{N}{+} \langle\langle c, s\rangle\rangle \underset{N}{*} \langle\overline{n}\rangle\right) \underset{N}{+} \langle\overline{m}\rangle \underset{N}{*} \langle\langle d, t\rangle, \overline{n}\rangle$$

$$\left(\langle\rangle \underset{N}{>} p\right) \Leftrightarrow \text{False}$$

$$\left(\langle\langle c, s\rangle, \overline{m}\rangle \underset{N}{>} \langle\rangle\right) \Leftrightarrow \text{True}$$

$$\left(\langle\langle c, s\rangle, \overline{m}\rangle \underset{N}{>} \langle\langle d, t\rangle, \overline{n}\rangle\right) \Leftrightarrow \left(\bigvee\left\{\begin{array}{l} s \underset{T}{>} t \\ \bigwedge\left\{\begin{array}{l} s = t \\ c \underset{C}{>} d \end{array}\right. \\ \bigwedge\left\{\begin{array}{l} s = t \\ c = d \\ \langle\overline{m}\rangle \underset{N}{>} \langle\overline{n}\rangle \end{array}\right. \end{array}\right.\right)$$

rdm[$\langle\rangle$, p] = 0

Some comments on the operations in the functor:

The element predicate $\underset{N}{\in}$ expressed the fact that polynomials must be tuples of monomials that are ordered by $\underset{T}{>}$. Monomials are tuples of length two with the first element being a non-zero coefficient taken from C and the second element being a "power product" taken from T.

Note that monomials must not have zero coefficients in this representation. Accordingly, the third clause in the definition of $\underset{N}{+}$ in the definition checks for $c \neq \underset{C}{-} d$ in order to avoid monomials with zeros in the sum of two polynomials!

The ordering $\underset{N}{>}$ introduced on the polynomials is only partial even if the ordering $\underset{T}{>}$ in the domain of power products is total.

The reduction multiplier $\underset{N}{\text{rdm}}[\langle\langle c, s\rangle, \overline{m}\rangle, \langle\langle d, t\rangle, \overline{n}\rangle]$[of two non-zero polynomials is the singleton polynomial $\left\langle\left\langle \underset{C}{\text{rdm}}[c, d], s \underset{T}{/} t\right\rangle\right\rangle$ that consists of the only monomial $\left\langle \underset{C}{\text{rdm}}[c, d], s \underset{T}{/} t\right\rangle$ in case $t \underset{T}{|} s$ (t divides s in T). In all other cases we define it to be $\underset{n}{0}$ so that in these cases no reduction takes place and, again, reducibility of p w.r.t. q can in all cases be decided by checking whether or not $\underset{N}{\text{rdm}}[p, q] = \underset{N}{0}$.

The least common reducible $\underset{N}{\text{lcrd}}[\langle\langle c, s\rangle, \overline{m}\rangle, \langle\langle d, t\rangle, \overline{n}\rangle]$ is the singletion polynomial $\left\langle\left\langle \underset{C}{\text{lcrd}}[c, d], \underset{T}{\text{lcm}}[s, t]\right\rangle\right\rangle$ that consists of the only monomial $\left\langle \underset{C}{\text{lcrd}}[c, d], \underset{T}{\text{lcm}}[s, t]\right\rangle$, where we refer to the $\underset{C}{\text{lcrd}}[c, d]$ in the coefficient domain C and to the $\underset{T}{\text{lcm}}[s, t]$ (the least common multiple) in the power product domain T. Of course, this reference to the least common multiple is the essence of Groebner bases construction in polynomial rings!

Attention: The quantifiers in the functor must not have bound variables that are identical to the bound variables in other functors. Hence, in this functor, we use 'j' as a bound variable in the definition of $\underset{N}{\in}$ and not 'i' because 'i' was used in the functor 'tuples-division-semigroup-lexical'. This is a bug in the current implementation of functors in *Theorema*. It will be eliminated in the next version of *Theorema*.

Construction of Poly Domain in 3 Indeterminates over the Rationals

```
Definition["ℙ",
    𝕋03 = tuples–division–semigroup–lexical[3]
    ℚ0 = rational–numbers–field[]
    ℙ = reduction–polynomials[ℚ0, 𝕋03]]
```

```
General::spell1 : Possible spelling error: new symbol name "𝕋03" is similar to existing symbol "𝕋3".
```

```
Theory["T",
    Definition["Theorema general functions"]
    Definition["rational numbers field"]
    Definition["tuples division semigroup lexical"]
    Definition["reduction polynomials"]
    Definition["ℙ"]
```

```
Use[⟨Built–in["Quantifiers"], Built–in["Connectives"], Built–in["Numbers"],
  Built–in["Tuples"], Built–in["Sets"], Built–in["Mathematica general functions"], Theory["T"]⟩]
```

Element Operation

```
Compute[∈_ℙ[⟨⟩]]
```

```
True
```

```
Compute[∈_ℙ[1_ℙ]]
```

```
True
```

```
Compute[∈_ℙ[⟨⟨2, ⟨3, 0, 4⟩⟩⟩]]
```

```
True
```

```
Compute[∈_ℙ[⟨⟨2, ⟨3, 0, 4⟩⟩, ⟨2/3, ⟨5, 2, 2⟩⟩⟩]]
```

```
False
```

```
Compute[∈_ℙ[⟨⟨2, ⟨3, 11, 4⟩⟩, ⟨2/3, ⟨5, 2, 2⟩⟩⟩]]
```

```
False
```

```
Compute[∈_ℙ[⟨⟨2, ⟨6, 0, 4⟩⟩, ⟨2/3, ⟨5, 2, 2⟩⟩⟩]]
```

```
True
```

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[\langle\langle 2, \langle 5, 11, 4\rangle\rangle, \langle 2/3, \langle 5, 2, 2\rangle\rangle, \langle 17, \langle 5, 1, 3\rangle\rangle\rangle]\right]$$

True

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 4, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 0, 0\rangle\rangle\rangle]\right]$$

True

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 4, 0\rangle\rangle, \langle -5, \langle 0, 3, 0\rangle\rangle, \langle -5, \langle 0, 2, 0\rangle\rangle\rangle]\right]$$

True

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 0, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 4, 0\rangle\rangle, \langle -5, \langle 0, 3, 0\rangle\rangle, \langle -5, \langle 0, 2, 0\rangle\rangle\rangle]\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[2]\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 4, 0\rangle\rangle, \langle -5, \langle 0, 3, 0\rangle\rangle, \langle -5, \langle 0, 2, 0\rangle\rangle\rangle]\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2.3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 4, 0\rangle\rangle, \langle -5, \langle 0, 3, 0\rangle\rangle, \langle -5, \langle 0, 2, 0\rangle\rangle\rangle]\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, -2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 4, 0\rangle\rangle, \langle -5, \langle 0, 3, 0\rangle\rangle, \langle -5, \langle 0, 2, 0\rangle\rangle\rangle]\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle, 3\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 4, 0\rangle\rangle, \langle -5, \langle 0, 3, 0\rangle\rangle, \langle -5, \langle 0, 2, 0\rangle\rangle\rangle]\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 3, 3\rangle\rangle, \langle -5, \langle 0, 4, 0\rangle\rangle, \langle -5, \langle 0, 3, 0\rangle\rangle, \langle -5, \langle 0, 2, 0\rangle\rangle\rangle]\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 4, 0\rangle\rangle, \langle -5, \langle 0, 3, 0\rangle\rangle, \langle -5, \langle 0, 3, 0\rangle\rangle\rangle]\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 4, 0\rangle\rangle, \langle -5, \langle 0, 3, 0\rangle\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle\rangle]\right]$$

True

Constants

$$\text{Compute}\left[\underset{\mathbb{P}}{1}\right]$$

$$\langle\langle 1, \langle 0, 0, 0\rangle\rangle\rangle$$

$$\text{Compute}\left[\underset{\mathbb{P}}{0}\right]$$

$$\langle\rangle$$

Addition, Subtraction

$$\text{Compute}\left[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 0, 0\rangle\rangle\rangle \underset{\mathbb{P}}{+} \langle\langle 5, \langle 3, 0, 3\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle 1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle\rangle\right]$$

$$\left\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 5, \langle 3, 0, 3\rangle\rangle, \left\langle\frac{4}{3}, \langle 2, 2, 2\rangle\right\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle, \langle -5, \langle 0, 0, 0\rangle\rangle\right\rangle$$

$$\text{Compute}\Big[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 0, 0\rangle\rangle\rangle \underset{\mathbb{P}}{-} \langle\langle 5, \langle 3, 0, 3\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle 1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle\rangle\Big]$$

$$\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle -5, \langle 3, 0, 3\rangle\rangle, \langle -1, \langle 2, 1, 3\rangle\rangle, \langle 5, \langle 0, 1, 0\rangle\rangle, \langle -5, \langle 0, 0, 0\rangle\rangle\rangle$$

Multiplication

$$\text{Compute}\Big[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 0, 0\rangle\rangle\rangle \underset{\mathbb{P}}{*} \underset{\mathbb{P}}{0}\Big]$$

$$\langle\rangle$$

$$\text{Compute}\Big[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 0, 0\rangle\rangle\rangle \underset{\mathbb{P}}{*} \underset{\mathbb{P}}{1}\Big]$$

$$\left\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \left\langle\frac{2}{3}, \langle 2, 2, 2\rangle\right\rangle, \left\langle\frac{-1}{2}, \langle 2, 1, 3\rangle\right\rangle, \langle -5, \langle 0, 0, 0\rangle\rangle\right\rangle$$

$$\text{Compute}\Big[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 0, 0\rangle\rangle\rangle \underset{\mathbb{P}}{*} \langle\langle -5, \langle 2, 2, 2\rangle\rangle\rangle\Big]$$

$$\left\langle\langle -10, \langle 5, 2, 6\rangle\rangle, \left\langle\frac{-10}{3}, \langle 4, 4, 4\rangle\right\rangle, \left\langle\frac{5}{2}, \langle 4, 3, 5\rangle\right\rangle, \langle 25, \langle 2, 2, 2\rangle\rangle\right\rangle$$

$$\text{Compute}\Big[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 0, 0\rangle\rangle\rangle \underset{\mathbb{P}}{*} \langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 0, 0\rangle\rangle\rangle\Big]$$

$$\left\langle\langle 4, \langle 6, 0, 8\rangle\rangle, \left\langle\frac{8}{3}, \langle 5, 2, 6\rangle\right\rangle, \langle -2, \langle 5, 1, 7\rangle\rangle, \left\langle\frac{4}{9}, \langle 4, 4, 4\rangle\right\rangle, \left\langle\frac{-2}{3}, \langle 4, 3, 5\rangle\right\rangle, \left\langle\frac{1}{4}, \langle 4, 2, 6\rangle\right\rangle, \langle -20, \langle 3, 0, 4\rangle\rangle, \left\langle\frac{-20}{3}, \langle 2, 2, 2\rangle\right\rangle, \langle 5, \langle 2, 1, 3\rangle\rangle, \langle 25, \langle 0, 0, 0\rangle\rangle\right\rangle$$

Ordering

$$\text{Compute}\Big[\underset{\mathbb{P}}{1} \underset{\mathbb{P}}{>} \underset{\mathbb{P}}{0}\Big]$$

True

$$\text{Compute}\left[1 \underset{\mathbb{P}}{>} 1\right]$$

$$1 \underset{\text{rational-numbers-field}[]}{>} 1$$

$$\text{Compute}\left[0 \underset{\mathbb{P}}{>} 0\right]$$

False

$$\text{Compute}\left[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 0, 0\rangle\rangle\rangle \underset{\mathbb{P}}{>} \langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 0, 0\rangle\rangle\rangle\right]$$

False

$$\text{Compute}\left[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle\rangle \underset{\mathbb{P}}{>} \langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 0, 0\rangle\rangle\rangle\right]$$

True

$$\text{Compute}\left[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle\rangle \underset{\mathbb{P}}{>} \langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 0, 1, 3\rangle\rangle, \langle -5, \langle 0, 0, 0\rangle\rangle\rangle\right]$$

True

$$\text{Compute}\left[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle\rangle \underset{\mathbb{P}}{>} \langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 3, 2\rangle\rangle, \langle -1/2, \langle 0, 1, 3\rangle\rangle, \langle -5, \langle 0, 0, 0\rangle\rangle\rangle\right]$$

False

Reduction Multiplier

$$\text{Compute}\left[\underset{\mathbb{P}}{\text{rdm}}\left[\underset{\mathbb{P}}{0}, \langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle\rangle\right]\right]$$

$$\langle\rangle$$

$$\text{Compute}\left[\underset{\mathbb{P}}{\text{rdm}}\left[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle\rangle, \underset{\mathbb{P}}{0}\right]\right]$$

$$\langle\rangle$$

$$\text{Compute}\left[\underset{\mathbb{P}}{\text{rdm}}[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle\rangle, \langle\langle 5, \langle 3, 0, 3\rangle\rangle, \langle 2/3, \langle 2, 0, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 0\rangle\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle\rangle]\right]$$

$$\left\langle\left\langle \frac{2}{5}, \langle 0, 0, 1\rangle\right\rangle\right\rangle$$

$$\text{Compute}\left[\underset{\mathbb{P}}{\text{rdm}}[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle\rangle, \langle\langle 5, \langle 4, 0, 2\rangle\rangle, \langle 2/3, \langle 2, 0, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 0\rangle\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle\rangle]\right]$$

$$\langle\rangle$$

Least Common Reducible

$$\text{Compute}\left[\underset{\mathbb{P}}{\text{lcrd}}[\langle\langle 2, \langle 3, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle\rangle, \langle\langle 5, \langle 4, 0, 2\rangle\rangle, \langle 2/3, \langle 2, 0, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 0\rangle\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle\rangle]\right]$$

$$\langle\langle 1, \langle 4, 0, 4\rangle\rangle\rangle$$

$$\text{Compute}\left[\underset{\mathbb{P}}{\text{lcrd}}[\langle\langle 2, \langle 4, 0, 4\rangle\rangle, \langle 2/3, \langle 2, 2, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 3\rangle\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle\rangle, \langle\langle 5, \langle 4, 0, 2\rangle\rangle, \langle 2/3, \langle 2, 0, 2\rangle\rangle, \langle -1/2, \langle 2, 1, 0\rangle\rangle, \langle -5, \langle 0, 1, 0\rangle\rangle\rangle]\right]$$

$$\langle\langle 1, \langle 4, 0, 4\rangle\rangle\rangle$$

```
Compute[lcrd_ℙ[⟨⟨2, ⟨4, 0, 4⟩⟩, ⟨2/3, ⟨2, 2, 2⟩⟩, ⟨−1/2, ⟨2, 1, 3⟩⟩, ⟨−5, ⟨0, 1, 0⟩⟩⟩,
    ⟨⟨5, ⟨4, 3, 5⟩⟩, ⟨2/3, ⟨2, 0, 2⟩⟩, ⟨−1/2, ⟨2, 1, 0⟩⟩, ⟨−5, ⟨0, 1, 0⟩⟩⟩⟩]]
```

```
⟨⟨1, ⟨4, 3, 5⟩⟩⟩
```

Construction of Poly Domain in 2 Indeterminates over the Prime Field Modulo 5

```
Definition["ℙ",
    𝕋2 = tuples–division–semigroup–lexical[2]
    𝔽5 = prime–field[5]
    ℙ = reduction–polynomials[𝔽5, 𝕋2]]
```

```
Theory["T",
    Definition["Theorema general functions"]
    Definition["prime field"]
    Definition["tuples division semigroup lexical"]
    Definition["reduction polynomials"]
    Definition["ℙ"]]
```

```
Use[⟨Built–in["Quantifiers"], Built–in["Connectives"], Built–in["Numbers"],
    Built–in["Tuples"], Built–in["Sets"], Built–in["Mathematica general functions"], Theory["T"]⟩]
```

Apply for Computation

```
Compute[ϵ_ℙ[⟨⟩]]
```

```
True
```

```
Compute[ϵ_ℙ[1_ℙ]]
```

```
True
```

```
Compute[ϵ_ℙ[⟨⟨2, ⟨3, 0, 4⟩⟩⟩]]
```

```
False
```

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[\langle\langle 2, \langle 3, 0\rangle\rangle\rangle]\right]$$

True

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[\langle\langle 7, \langle 3, 0\rangle\rangle\rangle]\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[\langle\langle 2, \langle 3, 7\rangle\rangle,\ \langle 2/3, \langle 5, 2\rangle\rangle\rangle]\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[\langle\langle 2, \langle 3, 7\rangle\rangle,\ \langle 3, \langle 5, 2\rangle\rangle\rangle]\right]$$

False

$$\text{Compute}\left[\underset{\mathbb{P}}{\in}[\langle\langle 2, \langle 33, 7\rangle\rangle,\ \langle 3, \langle 5, 2\rangle\rangle\rangle]\right]$$

True

$$\text{Compute}\left[\underset{\mathbb{P}}{1}\right]$$

$\langle\langle 1, \langle 0, 0\rangle\rangle\rangle$

$$\text{Compute}\left[\underset{\mathbb{P}}{0}\right]$$

$\langle\rangle$

$$\text{Compute}\left[\langle\langle 2, \langle 3, 2\rangle\rangle,\ \langle 3, \langle 2, 2\rangle\rangle,\ \langle 1, \langle 2, 1\rangle\rangle,\ \langle 3, \langle 0, 0\rangle\rangle\rangle \underset{\mathbb{P}}{+} \langle\langle 4, \langle 7, 0\rangle\rangle,\ \langle 3, \langle 2, 2\rangle\rangle,\ \langle 4, \langle 2, 1\rangle\rangle,\ \langle 2, \langle 0, 1\rangle\rangle\rangle\right]$$

$\langle\langle 4, \langle 7, 0\rangle\rangle, \langle 2, \langle 3, 2\rangle\rangle, \langle 1, \langle 2, 2\rangle\rangle, \langle 2, \langle 0, 1\rangle\rangle, \langle 3, \langle 0, 0\rangle\rangle\rangle$

$$\text{Compute}\Big[\langle\langle 2, \langle 3, 2\rangle\rangle, \langle 3, \langle 2, 2\rangle\rangle, \langle 1, \langle 2, 1\rangle\rangle, \langle 3, \langle 0, 0\rangle\rangle\rangle \underset{\mathbb{P}}{\overline{\ }} \langle\langle 4, \langle 7, 0\rangle\rangle, \langle 3, \langle 2, 2\rangle\rangle, \langle 4, \langle 2, 1\rangle\rangle, \langle 2, \langle 0, 1\rangle\rangle\rangle\Big]$$

$$\langle\langle 1, \langle 7, 0\rangle\rangle, \langle 2, \langle 3, 2\rangle\rangle, \langle 2, \langle 2, 1\rangle\rangle, \langle 3, \langle 0, 1\rangle\rangle, \langle 3, \langle 0, 0\rangle\rangle\rangle$$

$$\text{Compute}\Big[\langle\langle 2, \langle 3, 2\rangle\rangle, \langle 3, \langle 2, 2\rangle\rangle, \langle 1, \langle 2, 1\rangle\rangle, \langle 3, \langle 0, 0\rangle\rangle\rangle \underset{\mathbb{P}}{*} \underset{\mathbb{P}}{1}\Big]$$

$$\langle\langle 2, \langle 3, 2\rangle\rangle, \langle 3, \langle 2, 2\rangle\rangle, \langle 1, \langle 2, 1\rangle\rangle, \langle 3, \langle 0, 0\rangle\rangle\rangle$$

$$\text{Compute}\Big[\langle\langle 2, \langle 3, 2\rangle\rangle, \langle 3, \langle 2, 2\rangle\rangle, \langle 1, \langle 2, 1\rangle\rangle, \langle 3, \langle 0, 0\rangle\rangle\rangle \underset{\mathbb{P}}{*} \langle\langle 2, \langle 3, 2\rangle\rangle, \langle 3, \langle 2, 2\rangle\rangle, \langle 1, \langle 2, 1\rangle\rangle, \langle 3, \langle 0, 0\rangle\rangle\rangle\Big]$$

$$\langle\langle 4, \langle 6, 4\rangle\rangle, \langle 2, \langle 5, 4\rangle\rangle, \langle 4, \langle 5, 3\rangle\rangle, \langle 4, \langle 4, 4\rangle\rangle, \langle 1, \langle 4, 3\rangle\rangle, \langle 1, \langle 4, 2\rangle\rangle, \langle 2, \langle 3, 2\rangle\rangle, \langle 3, \langle 2, 2\rangle\rangle, \langle 1, \langle 2, 1\rangle\rangle, \langle 4, \langle 0, 0\rangle\rangle\rangle$$

$$\text{Compute}\Big[\underset{\mathbb{P}}{0} \underset{\mathbb{P}}{>} \underset{\mathbb{P}}{0}\Big]$$

False

$$\text{Compute}\Big[\langle\langle 2, \langle 3, 2\rangle\rangle, \langle 3, \langle 2, 2\rangle\rangle, \langle 1, \langle 2, 1\rangle\rangle, \langle 3, \langle 0, 0\rangle\rangle\rangle \underset{\mathbb{P}}{>} \langle\langle 2, \langle 3, 2\rangle\rangle, \langle 3, \langle 2, 2\rangle\rangle, \langle 1, \langle 1, 1\rangle\rangle, \langle 3, \langle 0, 0\rangle\rangle\rangle\Big]$$

True

$$\text{Compute}\Big[\langle\langle 2, \langle 3, 2\rangle\rangle, \langle 3, \langle 2, 2\rangle\rangle, \langle 1, \langle 2, 1\rangle\rangle, \langle 3, \langle 0, 0\rangle\rangle\rangle \underset{\mathbb{P}}{>} \langle\langle 2, \langle 3, 2\rangle\rangle, \langle 3, \langle 3, 1\rangle\rangle, \langle 1, \langle 1, 1\rangle\rangle, \langle 3, \langle 0, 0\rangle\rangle\rangle\Big]$$

False

$$\text{Compute}\Big[\underset{\mathbb{P}}{\text{rdm}}[\langle\langle 2, \langle 3, 2\rangle\rangle, \langle 3, \langle 2, 2\rangle\rangle, \langle 1, \langle 2, 1\rangle\rangle, \langle 3, \langle 0, 0\rangle\rangle\rangle, \langle\langle 3, \langle 2, 1\rangle\rangle, \langle 3, \langle 1, 2\rangle\rangle, \langle 1, \langle 1, 1\rangle\rangle, \langle 3, \langle 0, 0\rangle\rangle\rangle]\Big]$$

$$\langle\langle 4, \langle 1, 1\rangle\rangle\rangle$$

$$\text{Compute}\Big[\underset{\mathbb{P}}{\text{rdm}}[\langle\langle 2, \langle 3, 2\rangle\rangle, \langle 3, \langle 2, 2\rangle\rangle, \langle 1, \langle 2, 1\rangle\rangle, \langle 3, \langle 0, 0\rangle\rangle\rangle, \langle\langle 3, \langle 2, 3\rangle\rangle, \langle 3, \langle 1, 2\rangle\rangle, \langle 1, \langle 1, 1\rangle\rangle, \langle 3, \langle 0, 0\rangle\rangle\rangle]\Big]$$

$$\langle\rangle$$

```
Compute[lcrd_ℙ[⟨⟨2, ⟨3, 2⟩⟩, ⟨3, ⟨2, 2⟩⟩, ⟨1, ⟨2, 1⟩⟩, ⟨3, ⟨0, 0⟩⟩⟩, ⟨⟨3, ⟨2, 3⟩⟩, ⟨3, ⟨1, 2⟩⟩, ⟨1, ⟨1, 1⟩⟩, ⟨3, ⟨0, 0⟩⟩⟩⟩]]
```

```
⟨⟨1, ⟨3, 3⟩⟩⟩
```

Example: Induction Proofs

Initialize *Theorema* Provers

```
Needs["Theorema`"];
```

```
Needs["Theorema`Provers`UserProvers`PCS`"]
```

```
SetGlobals[TraceLevel → 0, DebugLevel → 0, SearchDepth → 100, Prover → PCS, FormatMetas → "Subscripted"]
```

```
SetOptions[Prove, transformBy → ProofSimplifier, TransformerOptions → {branches → Proved}];
```

```
$RecursionLimit = 1024;
```

```
$TmaSolveLevel = 3;
```

```
$NDDebug = False;
```

```
Off[General::spell]
```

```
Off[General::spell1]
```

```
SetGlobals[LabelStyle → "Numeric", Prover → NNEqIndProver];
Needs["Theorema`Provers`Cascade`Cascade`"]; Needs["Theorema`Provers`Cascade`ConjectureGenerator`"]
```

```
Needs["Theorema`Provers`UserProvers`GroebnerBasesProver`"]; $TmaProofPresentation = 100;
```

```
Theorema`Provers`PCS`Solver`Private`$TmaSolveLevel = 3
```

```
3
```

```
SetOptions[Prove, transformBy → ProofSimplifier, TransformerOptions → {branches → Proved}];
$RecursionLimit = 1024; Off[General::spell]; Off[General::spell1]; $TmaQRTarget = {"kb", "goal"};
```

Inductive Definition of Addition

```
Definition["addition", any[m, n],
    m + 0 = m                  " +0:"
    m + n⁺ = (m + n)⁺          " + .:"]
```

Proof of Left Zero and Left Induction: Some Principles of *Theorema* Proving Style

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

```
Prove[Proposition["left zero"],
    using → ⟨Definition["addition"]⟩,
    by → NNEqIndProver,

    ProverOptions → {TermOrder → LeftToRight},
  transformBy → ProofSimplifier, TransformerOptions → {branches → {Proved}}];
```

```
Prove[Proposition["left zero"],
    using → ⟨Definition["addition"]⟩,
    by → NNEqIndProver,

    ProverOptions → {TermOrder → LeftToRight},
  transformBy → ProofSimplifier, TransformerOptions → {branches → {Proved, Failed}}];
```

```
Proposition["left induction", any[m, n],
    m⁺ + n = (m + n)⁺   ". +"]
```

```
Prove[Proposition["left induction"],
    using → ⟨Definition["addition"]⟩,
    by → NNEqIndProver,

    ProverOptions → {TermOrder → LeftToRight},
  transformBy → ProofSimplifier, TransformerOptions → {branches → {Proved}}];
```

```
Prove[Proposition["left induction"],
    using → ⟨Definition["addition"], Proposition["left zero"]⟩,
    by → NNEqIndProver,

    ProverOptions → {TermOrder → LeftToRight},
  transformBy → ProofSimplifier, TransformerOptions → {branches → {Proved}}];
```

Proof of Commutativity and Associativity: The Role of Lemmata and Sequence of Exploration

```
Proposition["commutativity of addition", any[m, n],
    m + n = n + m   " + = " ]
```

```
Prove[Proposition["commutativity of addition"],
    using → ⟨Definition["addition"]⟩,
    by → NNEqIndProver,

    ProverOptions → {TermOrder → LeftToRight},
  transformBy → ProofSimplifier, TransformerOptions → {branches → {Proved, Failed}}];
```

```
Prove[Proposition["commutativity of addition"],
    using → ⟨Definition["addition"], Proposition["left zero"], Proposition["left induction"]⟩,
    by → NNEqIndProver,

    ProverOptions → {TermOrder → LeftToRight},
  transformBy → ProofSimplifier, TransformerOptions → {branches → {Proved, Failed}}];
```

```
Prove[Proposition["commutativity of addition"],
    using → ⟨Definition["addition"], Proposition["left zero"], Proposition["left induction"]⟩,
    by → NNEqIndProver,

    ProverOptions → {TermOrder → LeftToRight},
  transformBy → ProofSimplifier, TransformerOptions → {branches → {Proved}}];
```

```
Proposition["associativity of addition", any[m, n, p],
    m + (n + p) = (m + n) + p   "+"]
```

```
Prove[Proposition["associativity of addition"],
    using → ⟨Definition["addition"], Proposition["left zero"], Proposition["left induction"]⟩,
    by → NNEqIndProver,

    ProverOptions → {TermOrder → LeftToRight},
  transformBy → ProofSimplifier, TransformerOptions → {branches → {Proved}}];
```

```
Prove[Proposition["associativity of addition"],
    using → ⟨Definition["addition"]⟩,
    by → NNEqIndProver,

    ProverOptions → {TermOrder → LeftToRight},
  transformBy → ProofSimplifier, TransformerOptions → {branches → {Proved, Failed}}];
```

Multiplication: Subproofs Using Only Knowledge on Addition

```
Prove[Proposition["Associativity of Multiplication"],
    using → {Propositions["Some Equational Properties of Multiplication"],
    Propositions["Equational Properties of Addition"]},
    by → NNEqIndProver] // Timing
```

```
{79.31 Second, - ProofObject -}
```

See notebooks on associativity of *.

Use Propositions Proved for the Next Proofs

Theorema tool "Expand Knowledge Base":

```
Propositions["equational properties of addition", any[m, n, p],
    0 + n = n                    "0+ "
    m^+ + n = (m + n)^+          "^++ "
    m + n = n + m                " + "
    (m + n) + p = m + (n + p)    "( + ) + "
]
```

```
Prove[Propositions["equational properties of addition"],
    using → Definition["addition"],
  by → NNEqIndProver, proof–object → "SuccessBranch", ExpandKnowledge → True]
```

```
{- ProofObject -, - ProofObject -, - ProofObject -, - ProofObject -}
```

Obtain Ideas for Lemmata from Failing Proofs

Theorema tool "Cascade":

```
Prove[Proposition["commutativity of addition"],
    using → Definition["addition"],
  by → Cascade[NNEqIndProver, ConjectureGenerator], ProverOptions → {TermOrder → LeftToRight}];
```

This idea (in a slight modification) is also an essential ingredient for algorithm synthesis in *Theorema*: see "lazy thinking" method.

Lift Proved Knowledge to the Level of Special Provers

Special Provers S: Correct only in special theories T:

if S derives goal G from knowledge K,

then G must be a logical consequence of T ⋃ K.

Example: associativity of *, this time with special prover (simplifier) for 0, $\square^+$, + terms.

```
Prove[Proposition["Associativity of Multiplication"],
    using → {Propositions["Some Equational Properties of Multiplication"],
    Propositions["Equational Properties of Addition"]},
    built–in –> {Property[+ → {Associative, Commutative}]},
    by → NNEqIndProver, proof–object → "SuccessBranch"] // Timing
```

```
{15.93 Second, - ProofObject -}
```

In *Theorema,* we implement "many" special provers (e.g. PCS method, Groebner bases, Zeilberger-Paule, Collins).

How do we prove the correctness of these methods?

On the object level.
How lift the correctness to the metalevel?

■ Use Accumulated Ideas of Mathematics: "Schemes"

This idea (in a slight modification) is the second essential ingredient for algorithm synthesis in *Theorema*: see "lazy thinking" method.

A Typical Definition Scheme:

$$\forall_{P,Q}\left(\text{alternating–quantification}[Q, P] \Leftrightarrow \forall_{f}\left(Q[f] \Leftrightarrow \forall_{x}\exists_{y}\forall_{z} P[f, x, y, z]\right)\right).$$

Two Typical Proposition Scheme:

$$\forall_{f,g,h}\left(\text{is–commutative}[f] \Leftrightarrow \forall_{x,y}(f[x, y] = f[y, x])\right).$$

$$\forall_{f,g,h}\left(\text{is–homomorphic}[f, g, h] \Leftrightarrow \forall_{x,y}(h[f[x, y]] = g[h[x], h[y]])\right).$$

Two Typical Problems Schemes:

$$\forall_{A,P,Q}\left(\text{explicit–problem}[A, P, Q] \Leftrightarrow \forall_{x}\begin{matrix} P[A[x]] \\ Q[x, A[x]] \end{matrix}\right).$$

$$\forall_{A,P,Q} \left(\text{canonical–simplifier}[A, Q] \Leftrightarrow \forall_{x} \begin{array}{l} Q[x, A[x]] \\ Q[x, y] \Rightarrow (A[x] = A[y]) \end{array} \right).$$

Two Typical Algorithm Schemes:

$$\forall_{F,c,s,g,h1,h2} \left(\text{divide–and–conquer}[F, c, s, g, h1, h2] \Leftrightarrow \forall_{x} \left(F[x] = \begin{cases} s[x] & \Leftarrow \quad c[x] \\ g[F[h1[x]], F[h2[x]]] & \Leftarrow \quad \text{otherwise} \end{cases} \right) \right)$$

$$\forall_{G,lc,df}$$

$$\text{critical–pair–completion}[G, lc, df] \Leftrightarrow$$

$$\forall_{F} (G[F] = G[F, \text{pairs}[F]])$$

$$\forall_{F} (G[F, \langle\rangle] = F)$$

$$\forall_{F,g1,g2,\bar{p}} \left(G[F, \langle\langle g1, g2\rangle, \bar{p}\rangle] = \right.$$

$$\text{where}\left[f = lc[g1, g2], h1 = trd[rd[f, g1], F], h2 = trd[rd[f, g2], F], \right.$$

$$\left. \left. \begin{cases} G[F, \langle\bar{p}\rangle] & \Leftarrow \quad h1 = h2 \\ G\left[F \frown df[h1, h2], \langle\bar{p}\rangle \asymp \left\langle \langle F_k, df[h1, h2]\rangle \mid_{k=1,\ldots,|F|} \right\rangle\right] & \Leftarrow \quad \text{otherwise} \end{cases} \right] \right)$$

■ Example: Set Theory Prover

$$\textbf{Definition}\left[\text{"reflexivity"}, \text{any}[\sim, A], \text{is–reflexive}_A[\sim] :\Leftrightarrow \forall_{x \in A} x \sim x\right]$$

$$\textbf{Definition}\left[\text{"symmetry"}, \text{any}[\sim, A], \text{is–symmetric}_A[\sim] :\Leftrightarrow \forall_{x,y \in A} (x \sim y \Rightarrow y \sim x)\right]$$

$$\textbf{Definition}\left[\text{"transitivity"}, \text{any}[\sim, A], \text{is–transitive}_A[\sim] :\Leftrightarrow \forall_{x,y,z \in A} (x \sim y \wedge y \sim z \Rightarrow x \sim z)\right]$$

$$\textbf{Definition}\left[\text{"equivalence"}, \text{any}[\sim, A], \text{is–equivalence}_A[\sim] :\Leftrightarrow \bigwedge \begin{cases} \text{is–reflexive}_A[\sim] \\ \text{is–symmetric}_A[\sim] \\ \text{is–transitive}_A[\sim] \end{cases}\right]$$

Definition["class", any[x, ~ , A],
  $\text{class}_{A,\sim}[x] := \{a \in A \mid a \sim x \wedge x \in A\}$]

Proposition["equal classes", any[x ∈ A, y ∈ A, ~ , A], with[is–equivalence$_A$[~]],
  $x \sim y \Rightarrow (\text{class}_{A,\sim}[x] = \text{class}_{A,\sim}[y])$]

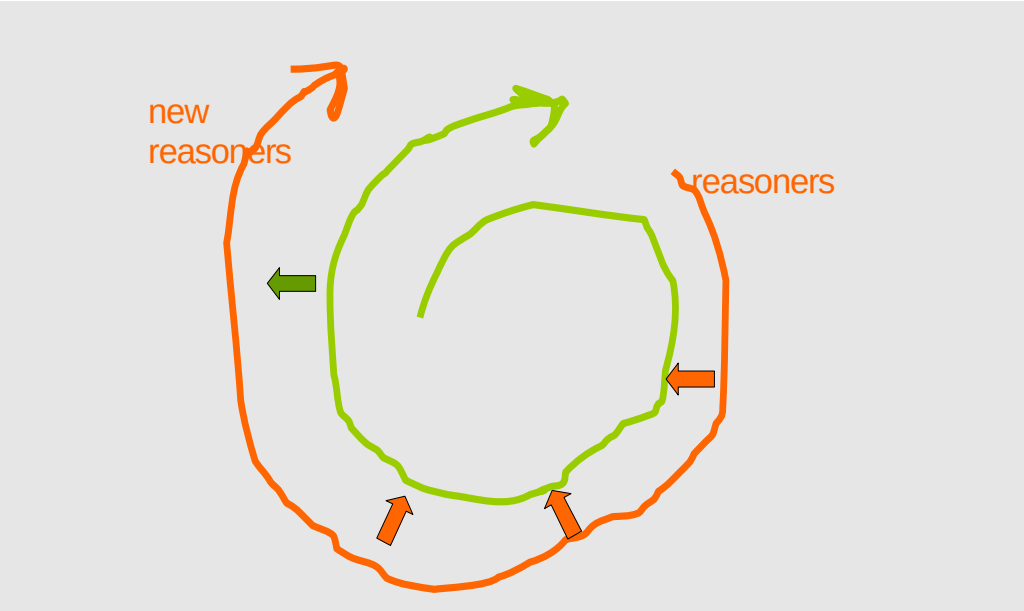
Prove[Proposition["equal classes"],
  using → ⟨Definition["equivalence"], Definition["transitivity"], Definition["symmetry"], Definition["class"]⟩,
  by → SetTheoryPCSProver,
  transformBy → ProofSimplifier, TransformerOptions → {branches → Proved, steps → Useful}, ShowOptions → {},
  ProverOptions → {GRWTarget → {"goal", "kb"}, AllowIntroduceQuantifiers → True,
    UseCyclicRules → True, DisableProver → {STC, PND}, ApplyBuiltIns → {}}, SearchDepth → 50]

- ProofObject -

(Proving needs approximately 1 minute.)

Example: The PCS Prover for Analysis

Principle of Lifting Knowledge to Inferencing



In this case, the "green" knowledge which we use for designing a special prover is an algorithm for solving systems of inequalities over the real numbers.

■ A Rough Sketch of the Method

A heuristic proof method (BB 2000) for predicate logic.

Generates "natural" proofs.

For formulae with alternating quantifiers.

Main idea: Reduce proof of such formula on (real) functions to
to the solution of inequalities on the reals (objects).

Structure:

1. Use predicate logic rules (except "simplification" by =, ⇒, ⇔)
2. Simplify (using "semantic unification")
3. Solve inequalities.

■ An Example

$$\textbf{Definition}\Big[\text{"limit:"}, \text{any}[f, a],\quad \text{limit}[f, a] \Longleftrightarrow \underset{\epsilon>0}{\forall\epsilon}\ \underset{N}{\exists N}\ \underset{n\geq N}{\forall n}\ |f[n]-a|<\epsilon\Big]$$

$$\textbf{Proposition}[\text{"limit of sum"}, \text{any}[f, a, g, b],\quad (\text{limit}[f, a] \wedge \text{limit}[g, b]) \Rightarrow \text{limit}[f+g, a+b]]$$

$$\textbf{Definition}[\text{"+:"}, \text{any}[f, g, x],\quad (f+g)[x] = f[x]+g[x]]$$

$$\textbf{Lemma}[\text{"|+|"}, \text{any}[x, y, a, b, \delta, \epsilon],\quad (|(x+y)-(a+b)|<(\delta+\epsilon)) \Longleftarrow (|x-a|<\delta \wedge |y-b|<\epsilon)]$$

$$\textbf{Lemma}[\text{"max"}, \text{any}[m, M1, M2],\quad m \geq \max[M1, M2] \Rightarrow (m \geq M1 \wedge m \geq M2)]$$

$$\textbf{Theory}\Big[\text{"limit"},\ \textbf{Definition}[\text{"limit:"}],\ \textbf{Definition}[\text{"+:"}],\ \textbf{Lemma}[\text{"|+|"}],\ \textbf{Lemma}[\text{"max"}]\Big]$$

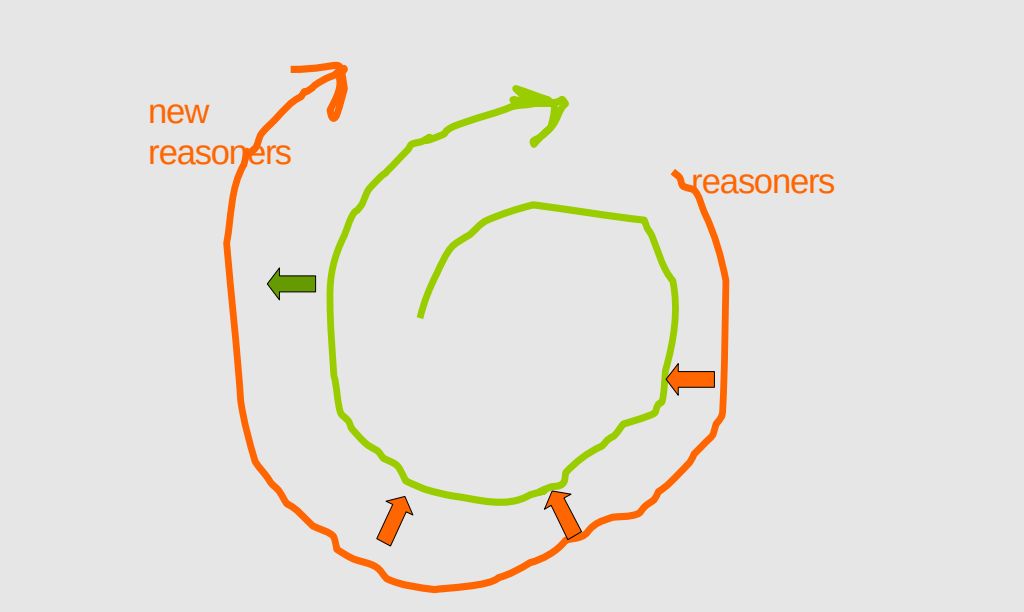
```
Prove[Proposition["limit of sum"], using → Theory["limit"], by → PCS]
```

```
- ProofObject -
```

Proof contains interesting algorithmic and didactic information!

Example: The Groebner Bases Prover for Geometrical Theorems

Principle of Lifting Knowledge to Inferencing



In this case, the "green" knowledge which we use for designing a special prover is an algorithm for solving systems of inequalities over the real numbers.

The Method

Each automated proof generated also contains an explanation of the method of reducing the proof problem to the problem of constructing a couple of Groebner bases.

Formula["Test", any[x, y],

$$((x^2 y - 3x) \neq 0) \vee ((x y + x + y) \neq 0) \vee (((x^2 y + 3x = 0) \vee ((-2x^2 + -7xy + x^2 y + x^3 y + -2y^2 + -2xy^2 + 2x^2 y^2) = 0)) \wedge ((x^2 + -xy + x^2 y + -2y^2 + -2xy^2) = 0))$$ "B1"

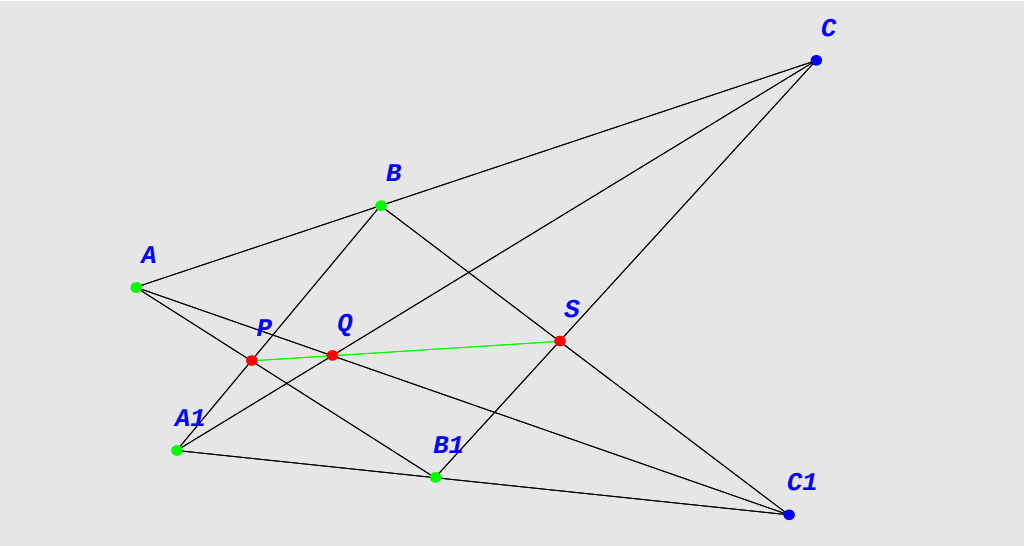
]

```
Prove[Formula["Test"], using → {}, by → GroebnerBasesProver]
```

```
- ProofObject -
```

■ Example Geo *Theorema* Proving: Pappus Theorem

- What does the theorem say geometrically?



- Textbook formulation:

Let A,B, C and A1,B1, C1 be on two lines and P = AB1 ⋂ A1B, Q = AC1 ⋂ A1C, S = BC1 ⋂ B1C. Then P, Q, and S are collinear.

- Input to the system:

```
Proposition["Pappus", any[A, B, A1, B1, C, C1, P, Q, S],
  point[A, B, A1, B1] ∧ pon[C, line[A, B]] ∧ pon[C1, line[A1, B1]] ∧ inter[P, line[A, B1], line[A1, B]] ∧
    inter[Q, line[A, C1], line[A1, C]] ∧ inter[S, line[B, C1], line[B1, C]] ⇒ collinear[P, Q, S]]
```

- Input to the system:

```
Prove[Proposition["Pappus"], by → GeometryProver,
  ProverOptions → {Method –> "GroebnerProver", Refutation → True}]
```

```
- ProofObject -
```

- Notebook generated automatically by the proving algorithm based on Groebner basis algorithm:

Prove:

(Proposition (Pappus))

$$\forall_{A,B,A1,B1,C,C1,P,Q,S} (\text{point}[A, B, A1, B1] \wedge \text{pon}[C, \text{line}[A, B]] \wedge \text{pon}[C1, \text{line}[A1, B1]] \wedge \text{inter}[P, \text{line}[A, B1], \text{line}[A1, B]] \wedge \text{inter}[Q, \text{line}[A, C1], \text{line}[A1, C]] \wedge \text{inter}[S, \text{line}[B, C1], \text{line}[B1, C]] \Rightarrow \text{collinear}[P, Q, S])$$

with no assumptions.

To prove the above statement we shall use the Gröbner basis method. First we have to transform the problem into algebraic form.

Algebraic Form:

To transform the geometric problem into algebraic form we have to chose first an orthogonal coordinate system.

Let's have the origin in point A, and points {} and $\{B, C\}$ on the two axes.

Using this coordinate system we have the following points:

$$\{\{A, 0, 0\}, \{B, 0, u_1\}, \{A1, u_2, u_3\}, \{B1, u_4, u_5\}, \{C, 0, u_6\}, \{C1, u_7, x_1\}, \{P, x_2, x_3\}, \{Q, x_4, x_5\}, \{S, x_6, x_7\}\}$$

The algebraic form of the given construction is:

(1)

$$\forall_{x_1,x_2,x_3,x_4,x_5,x_6,x_7} (u_3 u_4 + -u_2 u_5 + -u_3 u_7 + u_5 u_7 + u_2 x_1 + -u_4 x_1 == 0 \wedge u_5 x_2 + -u_4 x_3 == 0 \wedge -u_1 u_2 + u_1 x_2 + -u_3 x_2 + u_2 x_3 == 0 \wedge x_1 x_4 + -u_7 x_5 == 0 \wedge -u_2 u_6 + -u_3 x_4 + u_6 x_4 + u_2 x_5 == 0 \wedge u_1 u_7 + -u_1 x_6 + x_1 x_6 + -u_7 x_7 == 0 \wedge -u_4 u_6 + -u_5 x_6 + u_6 x_6 + u_4 x_7 == 0 \Rightarrow x_3 x_4 + -x_2 x_5 + -x_3 x_6 + x_5 x_6 + x_2 x_7 + -x_4 x_7 == 0)$$

This problem is equivalent to:

(2)

$$\neg \left(\exists_{x_1,x_2,x_3,x_4,x_5,x_6,x_7} (u_3 u_4 + -u_2 u_5 + -u_3 u_7 + u_5 u_7 + u_2 x_1 + -u_4 x_1 == 0 \wedge u_5 x_2 + -u_4 x_3 == 0 \wedge -u_1 u_2 + u_1 x_2 + -u_3 x_2 + u_2 x_3 == 0 \wedge x_1 x_4 + -u_7 x_5 == 0 \wedge -u_2 u_6 + -u_3 x_4 + u_6 x_4 + u_2 x_5 == 0 \wedge u_1 u_7 + -u_1 x_6 + x_1 x_6 + -u_7 x_7 == 0 \wedge -u_4 u_6 + -u_5 x_6 + u_6 x_6 + u_4 x_7 == 0 \wedge x_3 x_4 + -x_2 x_5 + -x_3 x_6 + x_5 x_6 + x_2 x_7 + -x_4 x_7 \neq 0) \right)$$

To remove the last inequality, we use the well-known Rabinowitsch trick. Let v_0 be a new variable. Then the problem becomes:

(3)

$$\neg\left(\underset{x_1,x_2,x_3,x_4,x_5,x_6,x_7,v_0}{\exists}\left(u_3\,u_4+-u_2\,u_5+-u_3\,u_7+u_5\,u_7+u_2\,x_1+-u_4\,x_1==0\wedge\right.\right.$$
$$u_5\,x_2+-u_4\,x_3==0\wedge-u_1\,u_2+u_1\,x_2+-u_3\,x_2+u_2\,x_3==0\wedge$$
$$x_1\,x_4+-u_7\,x_5==0\wedge-u_2\,u_6+-u_3\,x_4+u_6\,x_4+u_2\,x_5==0\wedge$$
$$u_1\,u_7+-u_1\,x_6+x_1\,x_6+-u_7\,x_7==0\wedge-u_4\,u_6+-u_5\,x_6+u_6\,x_6+u_4\,x_7==0\wedge$$
$$\left.\left.1+-v_0\,(x_3\,x_4+-x_2\,x_5+-x_3\,x_6+x_5\,x_6+x_2\,x_7+-x_4\,x_7)==0\right)\right)$$

To prove this statement we have to compute the Gröbner bases of the above polynomials.

The polynomials of the Gröbner bases are: {1}

As the obtained Gröbner bases is {1} the statement is generically true.

Statistics:

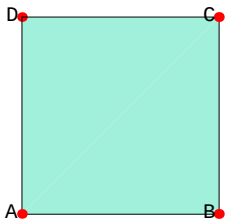
The length of the Gröbner bases is 1.

Time needed to compute the Gröbner bases: 0.42 Seconds.

■ Example Origami Proving: Trisection an Angle (jointly with T. Ida)

We give an example of trisecting an angle. This example shows a non-trivial use of Axiom (O6). The method of construction is due to H. Abe as described in [Geretschläger 2002, Fushimi 1980]. In the following we will explain the construction of trisecting a given angle using our computational origami system.

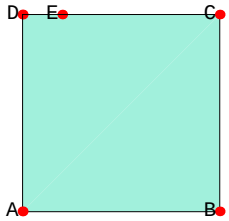
```
NewOrigami[Square[10, MarkPoints → {"A", "B", "C", "D"}], FigureCaption → "Step "];
```



Step 1

We then introduce an arbitrary point, say E at the coordinate (2, 10), assuming that point A is at (0, 0).

```
PutPoint[{"E", Point[2, 10]}];
```

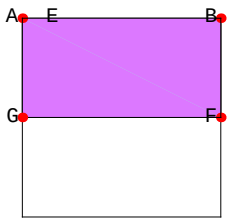


Step 2

Our problem is to trisect an angle ∠EAB. The method consists of the following seven steps (steps 3-9) of folds and unfolds.

Step 3: We make a fold to bring point A to D, to obtain the perpendicular bisector of segment AD. This is the application of (O2). The points F and G, which will be the terminal points of the crease, are automatically generated by the system.

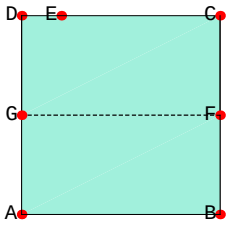
```
OFold[A, D];
```



Step 3

Step 4: We unfold the origami and obtain the crease FG.

```
Unfold[];
```

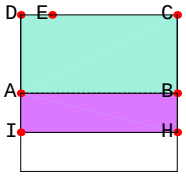


Step 4

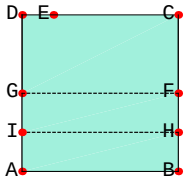
Steps 5 and 6: Likewise we obtain the crease HI.

```
OFold[A, G];
```

```
Unfold[];
```



Step 5



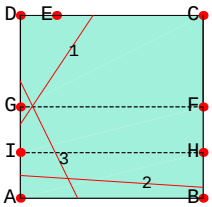
Step 6

Step 7: This step is the crucial step of the construction. We apply (O6). We try to superpose G and the line that is the extension of the segments AE, and to superpose A and the line that is the extension of the segment HI, simultaneously. There are three possible fold lines to realize this superposition. The system responds with the query of "Specify the line number" together with the lines on the origami image.

```
OFold[G, AE, A, HI];
```

Which line(1,2,3)?

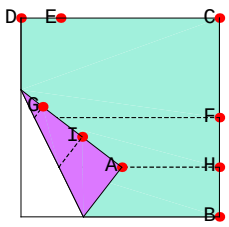
Specify the line number.



Step 7

Step 8: We reply to the query by the call of OFold[Along→3,Move→ A, ...], which tells the system that we choose the line number 3. This gives the fold line that we are primarily interested in. However, readers can easily see that the other two fold lines are also solutions (which trisect different angles).

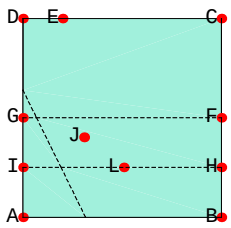
```
OFold[Along → 3, Move → A, MarkCrease → False];
```



Step 8

Step 9: We will copy the points A and I on the other face that is below the face that A and I are on, and unfold the origami. The copied points appear as L and J (the names are automatically generated).

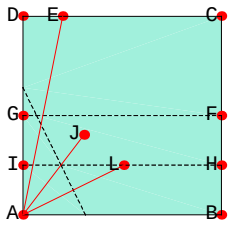
```
CopyPoint[{A, I}]; Unfold[];
```



Step 9

Step 10: Now we see that the segments AJ and AL trisect the angle ∠EAB.

```
ShowOrigamiSegment[{{{A, E}}, {{A, J}}, {{A, L}}}];
```



Step 10

Although it is not obvious to see that equational solving is performed, our system solves a set of polynomial equations, up to the third degree, at each step. In the case of steps 7 and 8, the system solves a cubic equation. This explains why we have (at most) 3 possible fold lines at step 7.

Proof of the correctness of the trisection method by Groebner Bases Method:

We now prove the following theorem with our system.

Theorem: The origami construction in section 3 trisects an angle.

∡EAB /3= ∡JAB/2 = ∡LAB.

Prove:

(Formula (TrisectingAngle): (1))

$$\forall_{a5,a6,a7,a8,b5,b6,b7,b8,c5,c6,c7,c8,r,x1,x10,x2,x3,x4,x5,x6,x7,x8,x9,y1,y10,y2,y3,y4,y5,y6,y7,y8,y9,\eta1,\eta2,\eta3,\eta4,\eta5,\mu1,\mu2,\mu3,\mu4} ((c8 = 0) \wedge$$

$$(a5 * r = 0) \wedge (c5 + \tfrac{1}{2} * b5 * r = 0) \wedge (-1 * r + x4 = 0) \wedge (x5 = 0) \wedge (-1 * r + x7 = 0) \wedge$$
$$(x8 = 0) \wedge (b5 * (r + (-1) * x1) + a5 * y1 = 0) \wedge (c5 + \tfrac{1}{2} * a5 * (r + x1) + \tfrac{1}{2} * b5 * y1 = 0) \wedge$$
$$(-1 * b7 * x10 + a7 * y10 = 0) \wedge (c7 + \tfrac{1}{2} * a7 * x10 + \tfrac{1}{2} * b7 * y10 = 0) \wedge (b6 * (r + (-1) * x2) + a6 * y2 = 0) \wedge$$
$$(c6 + \tfrac{1}{2} * a6 * (r + x2) + \tfrac{1}{2} * b6 * y2 = 0) \wedge (c8 + a8 * x3 + b8 * y3 = 0) \wedge$$
$$(c5 + a5 * x4 + b5 * y4 = 0) \wedge (-1 * b6 * x5 + a6 * y5 = 0) \wedge (c5 + a5 * x5 + b5 * y5 = 0) \wedge$$
$$(c6 + \tfrac{1}{2} * a6 * x5 + \tfrac{1}{2} * b6 * y5 = 0) \wedge (b7 * (x5 + (-1) * x6) + a7 * (-1 * y5 + y6) = 0) \wedge$$
$$(c7 + \tfrac{1}{2} * a7 * (x5 + x6) + \tfrac{1}{2} * b7 * (y5 + y6) = 0) \wedge (c6 + a6 * x7 + b6 * y7 = 0) \wedge (c6 + a6 * x8 + b6 * y8 = 0) \wedge$$
$$(b7 * (x8 + (-1) * x9) + a7 * (-1 * y8 + y9) = 0) \wedge (c7 + \tfrac{1}{2} * a7 * (x8 + x9) + \tfrac{1}{2} * b7 * (y8 + y9) = 0) \wedge$$
$$(-1 + r * \eta1 = 0) \wedge (-1 + (a5^2 + b5^2) * \eta2 = 0) \wedge (-1 + (a6^2 + b6^2) * \eta3 = 0) \wedge$$
$$(-1 + (a7^2 + b7^2) * \eta4 = 0) \wedge (-1 + (a8^2 + b8^2) * \eta5 = 0) \wedge (c8 + a8 * \mu1 + b8 * \mu2 = 0) \wedge$$
$$(b7 * (x5 + (-1) * \mu1) + a7 * (-1 * y5 + \mu2) = 0) \wedge (c7 + \tfrac{1}{2} * a7 * (x5 + \mu1) + \tfrac{1}{2} * b7 * (y5 + \mu2) = 0) \wedge$$
$$(-1 * b7 * \mu3 + a7 * \mu4 = 0) \wedge (c6 + a6 * \mu3 + b6 * \mu4 = 0) \wedge (c7 + \tfrac{1}{2} * a7 * \mu3 + \tfrac{1}{2} * b7 * \mu4 = 0) \Rightarrow$$
$$(-3 * x10^2 * x3 * y10 + x3 * y10^3 + x10^3 * y3 + (-3) * x10 * y10^2 * y3 = 0) \wedge$$
$$(-2 * x10 * x9 * y10 + x10^2 * y9 + (-1) * y10^2 * y9 = 0))$$

with no assumptions.

Proved.

The Theorem is proved by the Groebner Bases method.

The formula in the scope of the universal quantifier is transformed into an equivalent formula that is a conjunction of disjunctions of equalities and negated equalities. The universal quantifier can then be distributed over the individual parts of the conjunction. By this, we obtain:

Independent proof problems:

(Formula (TrisectingAngle): (1).1)

$$\forall_{a5,a6,a7,a8,b5,b6,b7,b8,c5,c6,c7,c8,r,x1,x10,x2,x3,x4,x5,x6,x7,x8,x9,y1,y10,y2,y3,y4,y5,y6,y7,y8,y9,\eta1,\eta2,\eta3,\eta4,\eta5,\mu1,\mu2,\mu3,\mu4}$$

$$((-2 * x10 * x9 * y10 + x10^2 * y9 + (-y10^2 * y9) = 0) \vee c8 \neq 0 \vee x5 \neq 0 \vee$$
$$x8 \neq 0 \vee -1 + r * \eta1 \neq 0 \vee -1 + a5^2 * \eta2 + b5^2 * \eta2 \neq 0 \vee -1 + a6^2 * \eta3 + b6^2 * \eta3 \neq 0 \vee$$
$$-1 + a7^2 * \eta4 + b7^2 * \eta4 \neq 0 \vee -1 + a8^2 * \eta5 + b8^2 * \eta5 \neq 0 \vee c5 + \tfrac{1}{2} * b5 * r \neq 0 \vee (-r) + x4 \neq 0 \vee$$
$$(-r) + x7 \neq 0 \vee b5 * r + (-b5 * x1) + a5 * y1 \neq 0 \vee b6 * r + (-b6 * x2) + a6 * y2 \neq 0 \vee$$
$$b7 * x5 + (-b7 * x6) + (-a7 * y5) + a7 * y6 \neq 0 \vee b7 * x5 + (-a7 * y5) + (-b7 * \mu1) + a7 * \mu2 \neq 0 \vee$$
$$b7 * x8 + (-b7 * x9) + (-a7 * y8) + a7 * y9 \neq 0 \vee (-b6 * x5) + a6 * y5 \neq 0 \vee (-b7 * x10) + a7 * y10 \neq 0 \vee$$
$$(-b7 * \mu3) + a7 * \mu4 \neq 0 \vee c5 + a5 * x4 + b5 * y4 \neq 0 \vee c5 + a5 * x5 + b5 * y5 \neq 0 \vee$$
$$c5 + \tfrac{1}{2} * a5 * r + \tfrac{1}{2} * a5 * x1 + \tfrac{1}{2} * b5 * y1 \neq 0 \vee c6 + a6 * x7 + b6 * y7 \neq 0 \vee$$
$$c6 + a6 * x8 + b6 * y8 \neq 0 \vee c6 + a6 * \mu3 + b6 * \mu4 \neq 0 \vee c6 + \tfrac{1}{2} * a6 * x5 + \tfrac{1}{2} * b6 * y5 \neq 0 \vee$$
$$c6 + \tfrac{1}{2} * a6 * r + \tfrac{1}{2} * a6 * x2 + \tfrac{1}{2} * b6 * y2 \neq 0 \vee c7 + \tfrac{1}{2} * a7 * x10 + \tfrac{1}{2} * b7 * y10 \neq 0 \vee$$
$$c7 + \tfrac{1}{2} * a7 * \mu3 + \tfrac{1}{2} * b7 * \mu4 \neq 0 \vee c7 + \tfrac{1}{2} * a7 * x5 + \tfrac{1}{2} * a7 * x6 + \tfrac{1}{2} * b7 * y5 + \tfrac{1}{2} * b7 * y6 \neq 0 \vee$$
$$c7 + \tfrac{1}{2} * a7 * x5 + \tfrac{1}{2} * b7 * y5 + \tfrac{1}{2} * a7 * \mu1 + \tfrac{1}{2} * b7 * \mu2 \neq 0 \vee$$
$$c7 + \tfrac{1}{2} * a7 * x8 + \tfrac{1}{2} * a7 * x9 + \tfrac{1}{2} * b7 * y8 + \tfrac{1}{2} * b7 * y9 \neq 0 \vee$$
$$c8 + a8 * x3 + b8 * y3 \neq 0 \vee c8 + a8 * \mu1 + b8 * \mu2 \neq 0 \vee a5 * r \neq 0)$$

(Formula (TrisectingAngle): (1).2): ...

We now prove the above individual problems separately:

Proof of (Formula (TrisectingAngle): (1).1): ...

This proof problem has the following structure:

(Formula (TrisectingAngle): (1).1.structure): ...

(Formula (TrisectingAngle): (1).1.structure) is equivalent to

(Formula (TrisectingAngle): (1).1.implication): ...

(Formula (TrisectingAngle): (1).1.implication) is equivalent to

(Formula (TrisectingAngle): (1).1.not-exists): ...

By introducing the slack variable(s)

$\{\xi\}$

(Formula (TrisectingAngle): (1).1.not-exists) is transformed into the equivalent formula

(Formula (TrisectingAngle): (1).1.not-exists-slack): ...

Hence, we see that the proof problem is transformed into the question on whether or not a system of polynomial equations has a solution or not. This question can be answered by checking whether or not the (reduced) Groebner basis of

....

is exactly $\{1\}$.

Hence, we compute the Groebner basis for the following polynomial list:

$$\Big\{-1+(-2)\,\mathrm{x10}\,\mathrm{x9}\,\mathrm{y10}\,\xi+\mathrm{x10}^2\,\mathrm{y9}\,\xi+(-1)\,\mathrm{y10}^2\,\mathrm{y9}\,\xi,\ \mathrm{c8},\ \mathrm{x5},\ \mathrm{x8},\ -1+\mathrm{r}\,\eta1,\ -1+\mathrm{a5}^2\,\eta2+\mathrm{b5}^2\,\eta2,\ -1+\mathrm{a6}^2\,\eta3+\mathrm{b6}^2\,\eta3,$$
$$-1+\mathrm{a7}^2\,\eta4+\mathrm{b7}^2\,\eta4,\ -1+\mathrm{a8}^2\,\eta5+\mathrm{b8}^2\,\eta5,\ \mathrm{c5}+\frac{\mathrm{b5\,r}}{2},\ -\mathrm{r}+\mathrm{x4},\ -\mathrm{r}+\mathrm{x7},\ \mathrm{b5\,r}+(-1)\,\mathrm{b5\,x1}+\mathrm{a5\,y1},$$
$$\mathrm{b6\,r}+(-1)\,\mathrm{b6\,x2}+\mathrm{a6\,y2},\ \mathrm{b7\,x5}+(-1)\,\mathrm{b7\,x6}+(-1)\,\mathrm{a7\,y5}+\mathrm{a7\,y6},\ \mathrm{b7\,x5}+(-1)\,\mathrm{a7\,y5}+(-1)\,\mathrm{b7}\,\mu1+\mathrm{a7}\,\mu2,$$
$$\mathrm{b7\,x8}+(-1)\,\mathrm{b7\,x9}+(-1)\,\mathrm{a7\,y8}+\mathrm{a7\,y9},\ -\mathrm{b6\,x5}+\mathrm{a6\,y5},\ -\mathrm{b7\,x10}+\mathrm{a7\,y10},\ -\mathrm{b7}\,\mu3+\mathrm{a7}\,\mu4,\ \mathrm{c5}+\mathrm{a5\,x4}+\mathrm{b5\,y4},$$
$$\mathrm{c5}+\mathrm{a5\,x5}+\mathrm{b5\,y5},\ \mathrm{c5}+\frac{\mathrm{a5\,r}}{2}+\frac{\mathrm{a5\,x1}}{2}+\frac{\mathrm{b5\,y1}}{2},\ \mathrm{c6}+\mathrm{a6\,x7}+\mathrm{b6\,y7},\ \mathrm{c6}+\mathrm{a6\,x8}+\mathrm{b6\,y8},\ \mathrm{c6}+\mathrm{a6}\,\mu3+\mathrm{b6}\,\mu4,$$
$$\mathrm{c6}+\frac{\mathrm{a6\,x5}}{2}+\frac{\mathrm{b6\,y5}}{2},\ \mathrm{c6}+\frac{\mathrm{a6\,r}}{2}+\frac{\mathrm{a6\,x2}}{2}+\frac{\mathrm{b6\,y2}}{2},\ \mathrm{c7}+\frac{\mathrm{a7\,x10}}{2}+\frac{\mathrm{b7\,y10}}{2},\ \mathrm{c7}+\frac{\mathrm{a7}\,\mu3}{2}+\frac{\mathrm{b7}\,\mu4}{2},\ \mathrm{c7}+\frac{\mathrm{a7\,x5}}{2}+\frac{\mathrm{a7\,x6}}{2}+\frac{\mathrm{b7\,y5}}{2}+\frac{\mathrm{b7\,y6}}{2},$$
$$\mathrm{c7}+\frac{\mathrm{a7\,x5}}{2}+\frac{\mathrm{b7\,y5}}{2}+\frac{\mathrm{a7}\,\mu1}{2}+\frac{\mathrm{b7}\,\mu2}{2},\ \mathrm{c7}+\frac{\mathrm{a7\,x8}}{2}+\frac{\mathrm{a7\,x9}}{2}+\frac{\mathrm{b7\,y8}}{2}+\frac{\mathrm{b7\,y9}}{2},\ \mathrm{c8}+\mathrm{a8\,x3}+\mathrm{b8\,y3},\ \mathrm{c8}+\mathrm{a8}\,\mu1+\mathrm{b8}\,\mu2,\ \mathrm{a5\,r}\Big\}$$

The Groebner basis:

$\{1\}$

Hence, (Formula (TrisectingAngle): (1).1) is proved.

Example: Proving Combinatorial Identities

Zeilberger, Paule, ... method. Groebner bases for non-commutative polyonomials play a role.

Example: Closed form for the following expression was an open problem for many years! The solution was given in P. Paule, *Computer-Solution of Problem 94-2,* SIAM REVIEW Vol.37 (1995), 105-106 using the Paule-Schorn automated conjecture generator / prover.

Formula["SIAM series",

$$\sum_{k=1,\ldots,n} \frac{(-1)^{k+1}(4k+1)(2k)!}{2^k(2k-1)(k+1)!\,2^k k!}]$$

Simplify[Formula["SIAM series"], by → PauleSchorn–Telescope, built–in → Built–in["PauleSchorn"]]

$$\sum_{k=1,\ldots,n} \left(\frac{(-1)^{k+1} * (4*k+1) * (2*k)!}{2^k * (2*k-1) * (k+1)! * 2^k * k!}\right) = 2^{-2k}$$

2^{-2k}

Prove[Formula["SIAM series"], by → PauleSchorn–Telescope, built–in → Built–in["PauleSchorn"]]

ProofSimplifier[Null, branches → Proved]

$$\text{System`FullSimplify}\left[\frac{\frac{(-1)^{1+k}\,2^{1-2(1+k)}\,(2(1+k))!\,(2+k)}{(1+k)!\,(2+k)!\,(-1+2(1+k))} - \frac{(-1)^k\,2^{1-2k}\,(2k)!\,(1+k)}{k!\,(1+k)!\,(-1+2k)}}{\frac{-(-1)^k\,2^{-2k}\,(2k)!\,(1+4k)}{k!\,(1+k)!\,(-1+2k)}}\right]$$

1

Example: Algorithmic Algorithm Synthesis

The Algorithm Invention ("Synthesis") Problem

Given a problem specification P (in predicate logic), find an algorithm A such that

$$\forall_x P[x, A[x]].$$

Examples of specifications P:

$$\begin{aligned}
P[x, y] &\Leftrightarrow \text{is–greater}[x, y] \\
P[x, y] &\Leftrightarrow \text{is–sorted–version}[x, y] \\
P[x, y] &\Leftrightarrow \text{has–derivative}[x, y] \\
P[x, y] &\Leftrightarrow \text{are–factors–of}[x, y] \\
P[x, y] &\Leftrightarrow \text{is–Gröbner–basis}[x, y] \\
&\ldots.
\end{aligned}$$

A general algorithm S for "all" P cannot exist (cf. B. Caviness 1970, ...) but ...

There is a rich literature on algorithm synthesis methods.

The "Lazy Thinking" Method for Algorithm Synthesis [BB 2003]

Given a problem specification P

- consider various "algorithm schemes" for A
- and try to prove (automatically) $\forall_x$ **P[x, A[x]].**
- This proof will normally fail because nothing is known on the unspecified sub-algorithms in the algorithm scheme.
- From the temporary assumptions and goals in the failing proof situation (automatically) generate such specifications for the unspecified sub-algorithms that would make the proof possible.

Now, apply the method recursively to the auxiliary functions.

This is "lazy"

- we use the condensed algorithmic experience of others (in the schemes)

- we start (routine) proving before we have an algorithm
- we just wait until we fail in order to get an idea from the failure.

■ An Algorithm for Generating Sub-problems from Failing Correctness Proofs

A simple (but amazingly powerful) rule:

Collect temporary assumptions T[x0, ... A [ ], ...]
and temporary goals G[x0, ...m [A [ ]]]

and produces specification

$$\forall_{X,\ldots,Y,\ldots} \left(T[X, \ldots Y, \ldots] \Longrightarrow G[Y, \ldots m[Y]]\right).$$

Details: see papers [BB 2003] and example.

Research topic: other rules.

■ How far can we go with this method?

By this method, we have synthesized, e.g., various sorting algorithms.

Can we automatically synthesize algorithms for non-trivial problems? What is "non-trivial"?

Example of a non-trivial problem: construction of Gröbner bases.

Main algorithmic idea of Gröbner bases theory: The "S-polynomials" ("critical pairs") together with the S-polynomial theorem.

Hence, question: Can Lazy Thinking automatically invent the notion of S-polynomial and automatically deliver the S-polynomial theorem.

■ The Problem of Constructing Gröbner Bases

Find algorithm Gb such that

$$\forall_F \begin{pmatrix} \text{is–finite[Gb[F]]} \\ \text{is–Gröbner–basis[Gb[F]]} \\ \text{ideal[F] = ideal[Gb[F]].} \end{pmatrix}$$

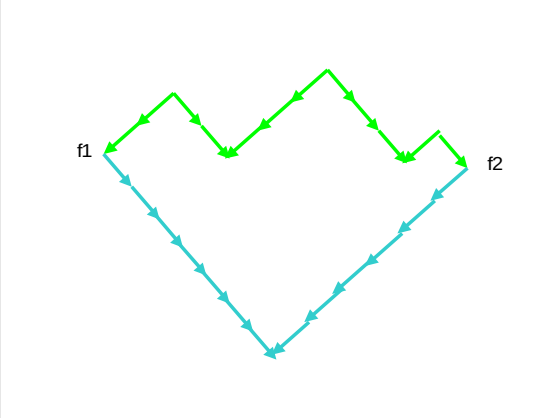
Definitions [BB 1965, 1970]:

is–Gröbner–basis[G] $\Leftrightarrow$ is–confluent[$\rightarrow_G$].

Confluence of Division $\rightarrow_G$

$$(h1 \rightarrow_G h2) \Leftrightarrow \underset{g \in G}{\exists} \left(\begin{cases} lp[g] \mid lp[h1] \\ h2 = h1 - (lm[h1] / lm[g])\, g \end{cases} \right),$$

is–confluent[$\rightarrow$] : $\Leftrightarrow \underset{f1,f2}{\forall} (f1 \leftrightarrow^* f2 \Rightarrow f1 \downarrow^* f2)$



Knowledge on the Concepts Involved

$h1 \rightarrow_G h2 \;\Rightarrow\; p \,.\, h1 \rightarrow_G p \,.\, h2$

etc.

Algorithm Scheme "Critical Pair / Completion"

A[F] = A[F, pairs[F]]

A[F, ⟨⟩] = F

A[F, ⟨⟨g1, g2⟩, $\bar{p}$⟩] =

$$\text{where}\Big[f = \mathbf{lc}[g1, g2],\ h1 = trd[rd[f, g1], F],\ h2 = trd[rd[f, g2], F],$$

$$\begin{cases} A[F, \langle \bar{p} \rangle] & \Leftarrow \quad h1 = h2 \\ A\Big[F \frown \mathbf{df}[h1, h2], \langle \bar{p} \rangle \asymp \Big\langle \langle F_k, \mathbf{df}[h1, h2] \rangle \underset{k=1,\ldots,|F|}{\mid} \Big\rangle\Big] & \Leftarrow \quad \text{otherwise} \end{cases} \Big]$$

This scheme can be tried in any domain, in which we have a reduction operation rd that depends on sets F of objects and a Noetherian relation $\succ$ which interacts with rd in the following natural way:

$$\forall_{f,g} (f \succeq \mathrm{rd}[f, g]).$$

■ The Essential Problem

The problem of synthesizing a Gröbner bases algorithm can now be also stated by asking whether starting with the proof of

$$\forall_{F} \begin{pmatrix} \text{is–finite}[\, A[F]\,] \\ \text{is–Gröbner–basis}[\, A[F]] \\ \text{ideal}[F] = \text{ideal}[\, A[F]]. \end{pmatrix}$$

we can *automatically* produce the idea that

$$\mathbf{lc}[g1, g2] = \mathbf{lcm}[\mathrm{lp}[g1], \mathrm{lp}[g2]]$$

and

$$\mathbf{df}[h1, h2] = h1 - h2$$

and prove that the idea is correct.

■ Now Start the (Automated) Correctness Proof

With current theorem proving technology, in the *Theorema* system, the proof attempt could be done automatically. (Not yet fully implemented.)

■ Details

▫ Upon Termination

It should be clear that, upon termination, the final result is a finite set (of polynomials) G that has the property

$$\forall_{g1,g2 \in G} \left(\text{where}\left[f = \mathrm{lc}[g1, g2],\ h1 = \mathrm{trd}[\mathrm{rd}[f, g1], F],\ h2 = \mathrm{trd}[\mathrm{rd}[f, g2], F],\ \bigvee \begin{Bmatrix} h1 = h2 \\ \mathrm{df}[h1, h2] \in G \end{Bmatrix}\right]\right).$$

(Termination proof has do be done extra!)

We Have to Prove

We now try to prove that, if G has this property, then

$$\begin{array}{l} \text{is–finite}[G], \\ \text{ideal}[F] = \text{ideal}[G], \\ \text{is–Gröbner–basis}[G], \\ \quad \text{i.e. is–Church–Rosser}[\ \rightarrow_G\ ]. \end{array}$$

Here, we only deal with the third, most important, property.

Using Available Knowledge

Using Newman's lemma and some elementary properties it can be shown that it is sufficient to prove

$$\text{is–Church–Rosser}[\ \rightarrow_G\ ] \Leftrightarrow \forall_{p} \forall_{f1,f2} \left(\left(\left\{ \begin{array}{l} p \rightarrow f1 \\ p \rightarrow f2 \end{array} \right. \right) \Rightarrow f1 \downarrow^* f2 \right).$$

Assumption

Let now the power product p and the polynomials f1, f2 be arbitary but fixed and assume

$$\left\{ \begin{array}{l} p \rightarrow_G f1 \\ p \rightarrow_G f2. \end{array} \right.$$

We have to find a polyonomial g such that

$$\begin{array}{l} f1 \rightarrow_G{}^* g, \\ f2 \rightarrow_G{}^* g. \end{array}$$

From the Assumption

From the assumption we know that there exist polynomials g1 and g2 in G such that

$$\begin{array}{l} \text{lp}[g1] \mid p, \\ f1 = \text{rd}[p, g1], \\ \text{lp}[g2] \mid p, \\ f2 = \text{rd}[p, g2]. \end{array}$$

From the final situation in the algorithm scheme we know that for these g1 and g2

$$\bigvee \left\{ \begin{array}{l} h1 = h2 \\ \text{df}[h1, h2] \in G, \end{array} \right.$$

where

$$h1 := trd[f1', G],\ f1' := rd[lc[g1, g2], g1],$$
$$h2 := trd[f2', G],\ f2' := rd[lc[g1, g2], g2].$$

Case h1=h2: In this case

$$lc[g1, g2] \rightarrow_{g1} rd[lc[g1, g2], g1] \rightarrow_{G}{}^{*} trd[rd[lc[g1, g2], g1], G] =$$
$$trd[rd[lc[g1, g2], g2], G] \leftarrow_{G}{}^{*} rd[lc[g1, g2], g2] \leftarrow_{g2} lc[g1, g2].$$

(Note that here we used the requirements rd[lc[g1,g2],g1]<lc[g1,g2] and rd[lc[g1,g2],g2]<lc[g1,g2].)

Hence, by elementary properties of polynomial reduction,

$$\forall_{a,q}\ (\ a\ q\ lc[g1, g2] \rightarrow_{g1} a\ q\ rd[lc[g1, g2], g1] \rightarrow_{G}{}^{*} a\ q\ trd[rd[lc[g1, g2], g1], G] =$$
$$a\ q\ trd[rd[lc[g1, g2], g2], G] \leftarrow_{G}{}^{*} a\ q\ rd[lc[g1, g2], g2] \leftarrow_{g2} a\ q\ lc[g1, g2]\ ).$$

Now we are stuck in the proof.

Use Specification Generation Algorithm

However, using the above specification generation rule, we see that we could proceed successfully with the proof if lc[g1,g2] satisfied the following requirement

$$\forall_{p,g1,g2} \left(\left(\left\{\begin{matrix} lp[g1] \mid p \\ lp[g2] \mid p \end{matrix}\right.\right) \Rightarrow \left(\exists_{a,q}\ (p = a\ q\ lc[g1, g2]\ )\right)\right), \quad \text{(lc requirement)}$$

With such an lc, we then would have

$$p \rightarrow_{g1} rd[p, g1] = a\ q\ rd[lc[g1, g2], g1] \rightarrow_{G}{}^{*} a\ q\ trd[rd[lc[g1, g2], g1], G] =$$
$$a\ q\ trd[rd[lc[g1, g2], g2], G] \leftarrow_{G}{}^{*} a\ q\ rd[lc[g1, g2], g2] = rd[p, g2] \leftarrow_{g2} p$$

and, hence,

$$f1 \rightarrow_{G}{}^{*} a\ q\ trd[rd[lc[g1, g2], g1], G],$$

$$f2 \rightarrow_{G}{}^{*} a\ q\ trd[rd[lc[g1, g2], g1], G],$$

i.e. we would have found a suitable g.

Summarizing the Specifications of the Unknown Subalgorithm lc

(lc requirement), which also could be written in the form:

$$\forall_{p,g1,g2} \left(\left(\left\{ \begin{array}{l} lp[g1] \mid p \\ lp[g2] \mid p \end{array} \right. \right) \Rightarrow (lc[g1, g2] \mid p) \right),$$

and the requirements:

$$rd[lc[g1, g2], g1] < lc[g1, g2],$$
$$rd[lc[g1, g2], g2] < lc[g1, g2],$$

which, in the case of the domain of polynomials, are equivalent to

$$lp[g1] \mid lc[g1, g2],$$
$$lp[g2] \mid lc[g1, g2].$$

A Suitable lc

$$lcp[g1, g2] = lcm[lp[g1], lp[g2]]$$

is a suitable function that satisfies the above requirements.

Heureka! The crucial function lc (the "critical pair" function) in the critical pair / completion algorithm scheme has been "automatically" synthesized!

Case h1≠h2 and, hence, df[h1,h2]∈G:

In this part of the proof we are basically stuck right at the beginning.

We can try to reduce this case to the first case, which would generate the following requirement

$$\forall_{h1,h2} (h1 \downarrow_{\{df[h1,h2]\}}{}^{*} h2) \quad \text{(df requirement)}.$$

(Looking to the knowledge base of elementary properties of polynomial reduction, it is now easy to find a function df that satifies (df requirement), namely

$$df[h1, h2] = h1 - h2,$$

because, in fact,

$$\forall_{f,g} (f \downarrow_{\{f-g\}}{}^{*} g).$$

Heureka! The function df (the "completion" function) in the critical pair / completion algorithm scheme has been "automatically" synthesized!)

Namely,

■ Conclusion

I believe that, going into the direction of systems like *Theorema*, the following is / will soon be possible:

- Computer-support, for the "working mathematician", of all aspects of doing mathematics will reach higher and higher levels including inventing, exploring, proving.
- In particular, nonalgorithmic and algorithmic mathematics will be supportable in one common logical and software-technological frame (in other words, mathematics, and computer science will reconcile).
- Mathematical software systems, in addition to providing algorithm libraries, will have to provide huge mathematical knowledge libraries. Building up and using such libraries, essentially, is a task of formal logic. (Compare International MKM Network.)
- The education of mathematicians will have to shift towards including in-depth training on the formal / logical aspects of mathematics.