A Set Theory Prover Within *Theorema*

Talk given at the CAL 01 Workshop
Las Palmas, Gran Canaria
February 20, 2001.

Wolfgang.Windsteiger@RISC.Uni-Linz.ac.at
RISC Institute
Hagenberg, Austria.

1 Introduction (from the extended abstract)

In this paper we report on the implementation of a set theory prover within the *Theorema* system. Particular emphasis has been put on the integration of set theory proving into the PCS heuristic for proving in predicate logic that has been invented by B. Buchberger and that has already been successfully applied to proofs in the area of elementary calculus using *Theorema*. It is shown how prove ("P"), compute ("C"), and solve ("S") phases contribute to natural style proofs of theorems involving notions of set theory. Opposed to an immediate approach of reducing set constructions to predicate logic "by definition" we provide special inference rules for all kinds of formulae involving sets in order to produce "elegant proofs". For instance, the proof problem of showing $A \subseteq B$ reduces by definition of "$\subseteq$" to showing $\forall_x x \in A \Rightarrow x \in B$, which we could then pass to the predicate logic prover. Instead, a human mathematician would probably just *assume* $x_0 \in A$ and *show* $x_0 \in B$ for some arbitrary x_0. This is the type of reasoning employed in the "P" phase by the *Theorema Set Theory PCS Prover*.

In addition, we try to incorporate the capabilities of the semantics of the *Theorema* language in order to deal with finitary set constructions during the compute (simplify) phase. The semantics of set operations available in *Theorema* computations not only allows computations with finite sets (such as {1, 2, 3}) but also with finite versions of the set quantifier (such as $\{x \in A \mid P_x\}$ where A is a finite set and P_x is some formula depending on x). We present an approach to *combining proving and computing* by doing semantic based computations during the "C" phase of proving in order to simplify proof situations.

Other constructions in set theory, however, directly lead to "solve problems". For example, proving $8 \in \{n^2 - 1 \mid n \in \mathbb{N}\}$ amounts to solving the equation $8 = n^2 - 1$ (over the integers). We present an approach to *combining proving and solving* by reducing typical "solve patterns" to "easier prove problems" using known solving techniques available in Mathematica.

It will be demonstrated in numerous examples, that the approach of viewing each proof as a sequence of *alternating proof phases* ("PCS") gives quite promising results in many applications. In this version of the paper, we give one example of a fully automatically generated proof by *Theorema*, for more examples see the full paper.

2 The P–C–S Paradigm for *Theorema* Provers

Proving is seen (mainly) as a sequence of alternating phases of

- Proving
- Computing
- Solving

It is a *heuristic method* that aims on *automated proofs in natural style*.

2.1 Proving

Problem Specification (Proving):

Given: a collection of formulae A (the assumptions),
a formula G (the goal, the conclusion).

Find: P, such that:
if G is a consequence of A then P should contain a
convincing argument (logically correct, sound, understandable),
why G necessarily follows from A.

We call the P to be found in a proving problem a *proof of G (from A).*

2.1.1 Example

Goal: $\{1\} \subseteq \mathbb{N}$.
Assumptions: Definition of $\subseteq$, $\mathbb{N}$, $\{\ldots\}$, and 1.

Proof: Take $a \in \{1\}$, which forces $a = 1$, hence $a \in \mathbb{N}$.

2.2 Computing

Problem Specification (Computing, Simplifying):

Given: a collection of formulae A (the assumptions),
a term T or a formula F.

Find: a term $\tilde{T}$ or a formula $\tilde{F}$, such that:

(a) $T = \tilde{T}$ is a consequence of A and
$\tilde{T}$ is (in some sense) simpler than T.

or

(b) $F \Leftrightarrow \tilde{F}$ is a consequence of A and
$\tilde{F}$ is (in some sense) simpler than F.

We name the *process* of finding $\tilde{T}$ or $\tilde{F}$ *computing* (or *computation* or *simplification*) *w.r.t. A*, and we call $\tilde{T}$ and $\tilde{F}$ respectively the *result of the computation w.r.t. A*.

2.2.1 Example

Compute : $\{x \in \mathbb{N} \mid x \geq 3\} \bigcap \{x \in \mathbb{I} \mid x < 6\}$.

Result : {3, 4, 5}.

Compute : $4 \in \{x \in \mathbb{N} \mid x \geq 3\} \bigcap \{x \in \mathbb{I} \mid x < 6\}$ . .

Result : $4 \in \{3, 4, 5\}$ or True.

Compute : $\frac{\sin(2\,x)\cos(x)}{\cos(x)^3}$.

Result : $\frac{2\sin(x)}{\cos(x)}$ or $2\tan(x)$.

2.3 Solving

Problem Specification (Solving):

Given: a collection of formulae A (the assumptions),
a formula G of the form $\underset{\overline{x}}{\exists} P_{\overline{x}}$.

Find: a proof of G by giving S, such that
$S \subseteq \{\langle y\rangle | P_{\overline{x}\to y}$ can be derived from $A\}$ and $|S| \geq 1$.

2.3.1 Example

Goal: $\underset{x,y\in\mathbb{R}}{\exists} \underbrace{x^2 + y^2 = 1}_{P_{x,y}}$.

Assumptions: Axioms of the domain of real numbers.

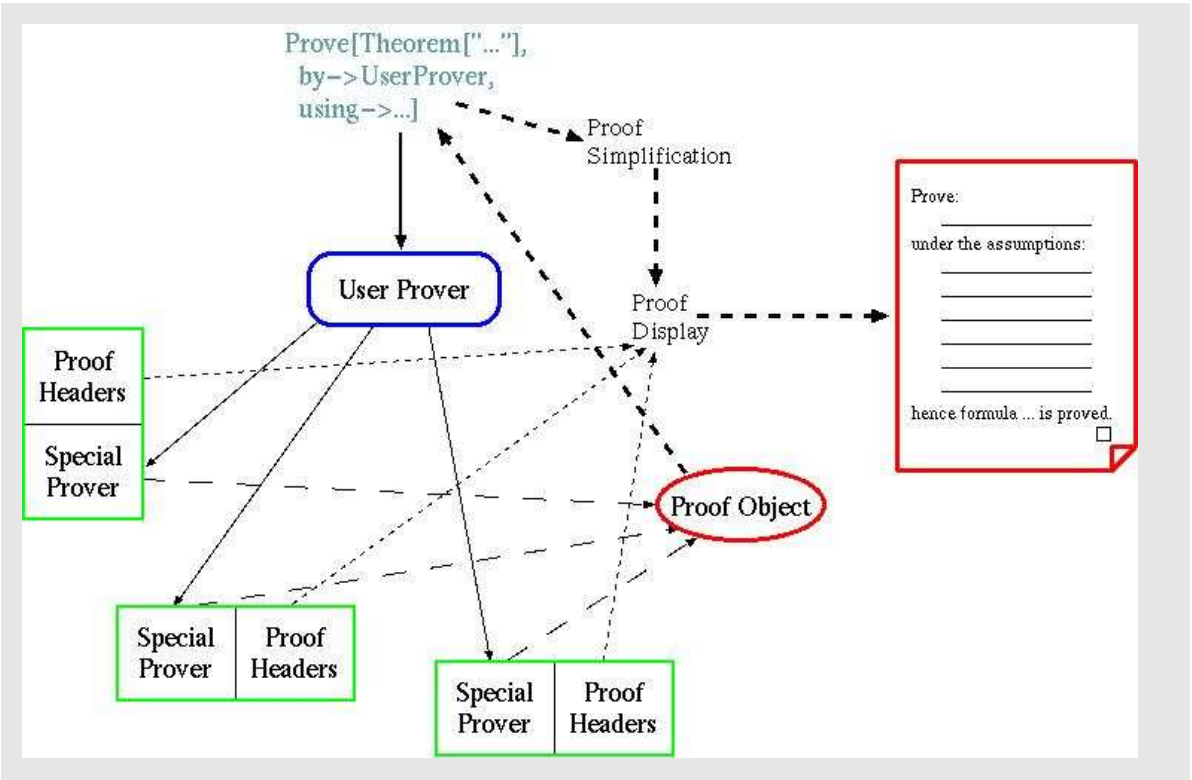
$S = \{\langle 0, 1\rangle\}$ (since $P_{x,y\to 0,1}$ gives $1 = 1$) or

$S = \left\{\langle 0, 1\rangle, \langle 1, 0\rangle, \left\langle \frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right\rangle, \left\langle \frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}\right\rangle\right\}$, or

$S = \{\langle \cos(t), \sin(t)\rangle \mid 0 \leq t < 2\,\pi\}$
(since $P_{x,y\to\cos(t),\sin(t)}$ gives $\cos(t)^2 + \sin(t)^2 = 1$, which holds for any $0 \leq t < 2\,\pi$).

3 Provers in *Theorema*

In general, *a Theorema Prover* is structured as follows:



4 The Set Theory PCS Prover

User Prover: **SetTheoryPCSProver** combines:

Special Provers (set theory specific):

4.1

STP: The **S**et **T**heory **P**rove module handles set theoretic expressions in the goal.

4.2

STKBR: The **S**et **T**heory **K**nowledge**B**ase **R**ewriting module infers new knowledge from set theoretic expressions in the assumptions.

4.3

STC: The **S**et **T**heory **C**ompute module performs (set theoretic) simplifications in the goal and in the assumptions.

4.4

STS: The **S**et **T**heory **S**olve module tries to solve existential goals.

In the context of set theory, such goals appear naturally e.g. in goals of the form $y \in \{T_x \mid_x P_x\}$, which immediately lead to $\exists_x y = T_x \wedge P_x$.

Other special provers involved from general PCS proving (see Buchberger):

4.5

Basic natural deduction.

4.6

Recognition of terminal proof cases.

4.7

Rewriting with respect to formulae in the knowledge base.

5 System Demonstration

5.1 General Outline of the Case Study

We investigate interactions of the following notions:

- Relations as sets of pairs.
- Equivalence relations and equivalence classes.
- The set of all equivalence classes in a set.
- Partitions of a set.
- The relation induced by a partition of a set.

5.2 Computing

The semantics of the *Theorema* language allows computations on certain expressions.

Definition["reflexivity", any[Rel, S],
$\text{reflexive}_S[\text{Rel}] :\Leftrightarrow \forall_{x \in S} \langle x, x\rangle \in \text{Rel}$]

Definition["symmetry", any[Rel, S],
$\text{symmetric}_S[\text{Rel}] :\Leftrightarrow \forall_{x,y \in S} (\langle x, y\rangle \in \text{Rel} \Rightarrow \langle y, x\rangle \in \text{Rel})$]

Definition["Equivalence class", any[x, A, R],
$\text{class}_{A,R}[x] := \{a \in A \mid \langle a, x\rangle \in R\}$]

```
Compute[reflexive_{i | i=0,...,10}[{⟨i, j⟩ | i,j=0,...,10 (Mod[i, 5] = Mod[j, 5])}], using → ⟨Definition["reflexivity"]⟩,
  built-in → ⟨Built-in["Sets"][SetOf], Built-in["Quantifiers"], Built-in["Connectives"]⟩]
```

```
⟨0, 0⟩ ∈ {⟨0, 0⟩, ⟨0, 5⟩, ⟨0, 10⟩, ⟨1, 1⟩, ⟨1, 6⟩, ⟨2, 2⟩, ⟨2, 7⟩, ⟨3, 3⟩, ⟨3, 8⟩, ⟨4, 4⟩, ⟨4, 9⟩, ⟨5, 0⟩, ⟨5, 5⟩,
      ⟨5, 10⟩, ⟨6, 1⟩, ⟨6, 6⟩, ⟨7, 2⟩, ⟨7, 7⟩, ⟨8, 3⟩, ⟨8, 8⟩, ⟨9, 4⟩, ⟨9, 9⟩, ⟨10, 0⟩, ⟨10, 5⟩, ⟨10, 10⟩} ∧
  ⟨1, 1⟩ ∈ {⟨0, 0⟩, ⟨0, 5⟩, ⟨0, 10⟩, ⟨1, 1⟩, ⟨1, 6⟩, ⟨2, 2⟩, ⟨2, 7⟩, ⟨3, 3⟩, ⟨3, 8⟩, ⟨4, 4⟩, ⟨4, 9⟩, ⟨5, 0⟩,
      ⟨5, 5⟩, ⟨5, 10⟩, ⟨6, 1⟩, ⟨6, 6⟩, ⟨7, 2⟩, ⟨7, 7⟩, ⟨8, 3⟩, ⟨8, 8⟩, ⟨9, 4⟩, ⟨9, 9⟩, ⟨10, 0⟩, ⟨10, 5⟩, ⟨10, 10⟩} ∧
  ⟨2, 2⟩ ∈ {⟨0, 0⟩, ⟨0, 5⟩, ⟨0, 10⟩, ⟨1, 1⟩, ⟨1, 6⟩, ⟨2, 2⟩, ⟨2, 7⟩, ⟨3, 3⟩, ⟨3, 8⟩, ⟨4, 4⟩, ⟨4, 9⟩, ⟨5, 0⟩,
      ⟨5, 5⟩, ⟨5, 10⟩, ⟨6, 1⟩, ⟨6, 6⟩, ⟨7, 2⟩, ⟨7, 7⟩, ⟨8, 3⟩, ⟨8, 8⟩, ⟨9, 4⟩, ⟨9, 9⟩, ⟨10, 0⟩, ⟨10, 5⟩, ⟨10, 10⟩} ∧
  ⟨3, 3⟩ ∈ {⟨0, 0⟩, ⟨0, 5⟩, ⟨0, 10⟩, ⟨1, 1⟩, ⟨1, 6⟩, ⟨2, 2⟩, ⟨2, 7⟩, ⟨3, 3⟩, ⟨3, 8⟩, ⟨4, 4⟩, ⟨4, 9⟩, ⟨5, 0⟩,
      ⟨5, 5⟩, ⟨5, 10⟩, ⟨6, 1⟩, ⟨6, 6⟩, ⟨7, 2⟩, ⟨7, 7⟩, ⟨8, 3⟩, ⟨8, 8⟩, ⟨9, 4⟩, ⟨9, 9⟩, ⟨10, 0⟩, ⟨10, 5⟩, ⟨10, 10⟩} ∧
  ⟨4, 4⟩ ∈ {⟨0, 0⟩, ⟨0, 5⟩, ⟨0, 10⟩, ⟨1, 1⟩, ⟨1, 6⟩, ⟨2, 2⟩, ⟨2, 7⟩, ⟨3, 3⟩, ⟨3, 8⟩, ⟨4, 4⟩, ⟨4, 9⟩, ⟨5, 0⟩,
      ⟨5, 5⟩, ⟨5, 10⟩, ⟨6, 1⟩, ⟨6, 6⟩, ⟨7, 2⟩, ⟨7, 7⟩, ⟨8, 3⟩, ⟨8, 8⟩, ⟨9, 4⟩, ⟨9, 9⟩, ⟨10, 0⟩, ⟨10, 5⟩, ⟨10, 10⟩} ∧
  ⟨5, 5⟩ ∈ {⟨0, 0⟩, ⟨0, 5⟩, ⟨0, 10⟩, ⟨1, 1⟩, ⟨1, 6⟩, ⟨2, 2⟩, ⟨2, 7⟩, ⟨3, 3⟩, ⟨3, 8⟩, ⟨4, 4⟩, ⟨4, 9⟩, ⟨5, 0⟩,
      ⟨5, 5⟩, ⟨5, 10⟩, ⟨6, 1⟩, ⟨6, 6⟩, ⟨7, 2⟩, ⟨7, 7⟩, ⟨8, 3⟩, ⟨8, 8⟩, ⟨9, 4⟩, ⟨9, 9⟩, ⟨10, 0⟩, ⟨10, 5⟩, ⟨10, 10⟩} ∧
  ⟨6, 6⟩ ∈ {⟨0, 0⟩, ⟨0, 5⟩, ⟨0, 10⟩, ⟨1, 1⟩, ⟨1, 6⟩, ⟨2, 2⟩, ⟨2, 7⟩, ⟨3, 3⟩, ⟨3, 8⟩, ⟨4, 4⟩, ⟨4, 9⟩, ⟨5, 0⟩,
      ⟨5, 5⟩, ⟨5, 10⟩, ⟨6, 1⟩, ⟨6, 6⟩, ⟨7, 2⟩, ⟨7, 7⟩, ⟨8, 3⟩, ⟨8, 8⟩, ⟨9, 4⟩, ⟨9, 9⟩, ⟨10, 0⟩, ⟨10, 5⟩, ⟨10, 10⟩} ∧
  ⟨7, 7⟩ ∈ {⟨0, 0⟩, ⟨0, 5⟩, ⟨0, 10⟩, ⟨1, 1⟩, ⟨1, 6⟩, ⟨2, 2⟩, ⟨2, 7⟩, ⟨3, 3⟩, ⟨3, 8⟩, ⟨4, 4⟩, ⟨4, 9⟩, ⟨5, 0⟩,
      ⟨5, 5⟩, ⟨5, 10⟩, ⟨6, 1⟩, ⟨6, 6⟩, ⟨7, 2⟩, ⟨7, 7⟩, ⟨8, 3⟩, ⟨8, 8⟩, ⟨9, 4⟩, ⟨9, 9⟩, ⟨10, 0⟩, ⟨10, 5⟩, ⟨10, 10⟩} ∧
  ⟨8, 8⟩ ∈ {⟨0, 0⟩, ⟨0, 5⟩, ⟨0, 10⟩, ⟨1, 1⟩, ⟨1, 6⟩, ⟨2, 2⟩, ⟨2, 7⟩, ⟨3, 3⟩, ⟨3, 8⟩, ⟨4, 4⟩, ⟨4, 9⟩, ⟨5, 0⟩,
      ⟨5, 5⟩, ⟨5, 10⟩, ⟨6, 1⟩, ⟨6, 6⟩, ⟨7, 2⟩, ⟨7, 7⟩, ⟨8, 3⟩, ⟨8, 8⟩, ⟨9, 4⟩, ⟨9, 9⟩, ⟨10, 0⟩, ⟨10, 5⟩, ⟨10, 10⟩} ∧
  ⟨9, 9⟩ ∈ {⟨0, 0⟩, ⟨0, 5⟩, ⟨0, 10⟩, ⟨1, 1⟩, ⟨1, 6⟩, ⟨2, 2⟩, ⟨2, 7⟩, ⟨3, 3⟩, ⟨3, 8⟩, ⟨4, 4⟩, ⟨4, 9⟩, ⟨5, 0⟩,
      ⟨5, 5⟩, ⟨5, 10⟩, ⟨6, 1⟩, ⟨6, 6⟩, ⟨7, 2⟩, ⟨7, 7⟩, ⟨8, 3⟩, ⟨8, 8⟩, ⟨9, 4⟩, ⟨9, 9⟩, ⟨10, 0⟩, ⟨10, 5⟩, ⟨10, 10⟩} ∧
  ⟨10, 10⟩ ∈ {⟨0, 0⟩, ⟨0, 5⟩, ⟨0, 10⟩, ⟨1, 1⟩, ⟨1, 6⟩, ⟨2, 2⟩, ⟨2, 7⟩, ⟨3, 3⟩, ⟨3, 8⟩, ⟨4, 4⟩, ⟨4, 9⟩, ⟨5, 0⟩,
      ⟨5, 5⟩, ⟨5, 10⟩, ⟨6, 1⟩, ⟨6, 6⟩, ⟨7, 2⟩, ⟨7, 7⟩, ⟨8, 3⟩, ⟨8, 8⟩, ⟨9, 4⟩, ⟨9, 9⟩, ⟨10, 0⟩, ⟨10, 5⟩, ⟨10, 10⟩}
```

```
Compute[reflexive_{i | i=0,...,10}[{⟨i, j⟩ | i,j=0,...,10 (Mod[i, 5] = Mod[j, 5])}], using → ⟨Definition["reflexivity"]⟩,
  built-in → ⟨Built-in["Sets"], Built-in["Quantifiers"], Built-in["Connectives"]⟩]
```

```
True
```

```
Compute[symmetric_{i | i=0,...,10}[{⟨i, j⟩ | i,j=0,...,10 (Mod[i, 5] = Mod[j, 5])}],
  using → ⟨Definition["symmetry"]⟩, built-in →
    ⟨Built-in["Sets"], Built-in["Quantifiers"], Built-in["Connectives"], Built-in["Tuples"][AngleBracket]⟩]
```

```
True
```

```
Compute[class_{i | i=0,...,10},{⟨i,j⟩ | i,j=0,...,10 (Mod[i,5]=Mod[j,5])}[4],
  using → ⟨Definition["Equivalence class"]⟩, built-in → ⟨Built-in["Sets"], Built-in["Quantifiers"]⟩]
```

```
{4, 9}
```

```
Compute[class_{i | i=0,...,10},{⟨i,j⟩ | i,j=0,...,10 (Mod[i,5]=Mod[j,5])}[4] = {9, 4},
  using → ⟨Definition["Equivalence class"]⟩, built-in → ⟨Built-in["Sets"], Built-in["Quantifiers"]⟩]
```

```
True
```

```
Clear[™Subscript]
```

5.3 Computing ⊆ Proving

Goal: **Make use of knowledge provided by the semantics of the language during proving!**

5.3.1 Reflexivity

```
Proposition["R reflexive on finite set",
  reflexive_{i | i=0,...,10}[{⟨i, j⟩ | i,j=0,...,10 (Mod[i, 5] = Mod[j, 5])}]]
```

```
Prove[Proposition["R reflexive on finite set"],
  using → Definition["reflexivity"],
  built-in → Built-in["Quantifiers"],
  by → SetTheoryPCSProver, ProverOptions → {TransformRanges → False,
    SimplifyFormula → False, DisableProver → {STP, PND}}]
```

Prove:

(Proposition (R reflexive on finite set)) $\text{reflexive}_{\{i \mid i=0,\ldots,10\}}\left[\left\{\langle i, j\rangle \mid \substack{i=0,\ldots,10 \\ j=0,\ldots,10} \text{Mod}[i, 5] = \text{Mod}[j, 5]\right\}\right]$,

under the assumption:

(Definition (reflexivity)) $\underset{Rel,S}{\forall}\left(\text{reflexive}_S[Rel] :\Leftrightarrow \underset{x\in S}{\forall}(\langle x, x\rangle \in Rel)\right)$.

Using available computation rules we simplify (Proposition (R reflexive on finite set)):

(1)

$\text{reflexive}_{\{0,1,2,3,4,5,6,7,8,9,10\}}[\{\langle 0, 0\rangle, \langle 0, 5\rangle, \langle 0, 10\rangle, \langle 1, 1\rangle, \langle 1, 6\rangle, \langle 2, 2\rangle, \langle 2, 7\rangle, \langle 3, 3\rangle, \langle 3, 8\rangle, \langle 4, 4\rangle, \langle 4, 9\rangle, \langle 5, 0\rangle, \langle 5, 5\rangle, \langle 5, 10\rangle, \langle 6, 1\rangle, \langle 6, 6\rangle, \langle 7, 2\rangle, \langle 7, 7\rangle, \langle 8, 3\rangle, \langle 8, 8\rangle, \langle 9, 4\rangle, \langle 9, 9\rangle, \langle 10, 0\rangle, \langle 10, 5\rangle, \langle 10, 10\rangle\}]$.

Formula (1), using (Definition (reflexivity)), is implied by:

(2) $\underset{x\in\{0,1,2,3,4,5,6,7,8,9,10\}}{\forall}(\langle x, x\rangle \in \{\langle 0, 0\rangle, \langle 0, 5\rangle, \langle 0, 10\rangle, \langle 1, 1\rangle, \langle 1, 6\rangle, \langle 2, 2\rangle, \langle 2, 7\rangle, \langle 3, 3\rangle, \langle 3, 8\rangle, \langle 4, 4\rangle, \langle 4, 9\rangle, \langle 5, 0\rangle, \langle 5, 5\rangle, \langle 5, 10\rangle, \langle 6, 1\rangle, \langle 6, 6\rangle, \langle 7, 2\rangle, \langle 7, 7\rangle, \langle 8, 3\rangle, \langle 8, 8\rangle, \langle 9, 4\rangle, \langle 9, 9\rangle, \langle 10, 0\rangle, \langle 10, 5\rangle, \langle 10, 10\rangle\})$.

Using available computation rules we simplify (2):

(3) True.

Formula (3) is true because it is the constant True.

□

5.3.2 Symmetry

```
Proposition["R symmetric on finite set",
  symmetric_{i | i=0,...,10}[{⟨i, j⟩ | i,j=0,...,10 (Mod[i, 5] = Mod[j, 5])}]]
```

```
Prove[Proposition["R symmetric on finite set"],
  using → Definition["symmetry"],
  built-in → ⟨Built-in["Quantifiers"], Built-in["Tuples"]⟩,
  by → SetTheoryPCSProver, ProverOptions → {TransformRanges → False,
    SimplifyFormula → False, DisableProver → {STP, STKBR, PND}}]
```

Prove:

(Proposition (R symmetric on finite set)) $\text{symmetric}_{\{i \mid i=0,\ldots,10\}}\left[\left\{\langle i, j\rangle \mid \substack{i=0,\ldots,10 \\ j=0,\ldots,10} \text{Mod}[i, 5] = \text{Mod}[j, 5]\right\}\right]$,

under the assumption:

(Definition (symmetry)) $\forall_{Rel,S}\left(\text{symmetric}_S[Rel] :\Leftrightarrow \forall_{\substack{x\in S \\ y\in S}} (\langle x, y\rangle \in Rel \Rightarrow \langle y, x\rangle \in Rel)\right)$.

Using available computation rules we simplify (Proposition (R symmetric on finite set)):

(1)

$\text{symmetric}_{\{0,1,2,3,4,5,6,7,8,9,10\}}[\{\langle 0, 0\rangle, \langle 0, 5\rangle, \langle 0, 10\rangle, \langle 1, 1\rangle, \langle 1, 6\rangle, \langle 2, 2\rangle, \langle 2, 7\rangle, \langle 3, 3\rangle, \langle 3, 8\rangle, \langle 4, 4\rangle, \langle 4, 9\rangle, \langle 5, 0\rangle, \langle 5, 5\rangle, \langle 5, 10\rangle, \langle 6, 1\rangle, \langle 6, 6\rangle, \langle 7, 2\rangle, \langle 7, 7\rangle, \langle 8, 3\rangle, \langle 8, 8\rangle, \langle 9, 4\rangle, \langle 9, 9\rangle, \langle 10, 0\rangle, \langle 10, 5\rangle, \langle 10, 10\rangle\}]$.

Formula (1), using (Definition (symmetry)), is implied by:

(2)

$$\forall_{\substack{x\in\{0,1,2,3,4,5,6,7,8,9,10\}\\ y\in\{0,1,2,3,4,5,6,7,8,9,10\}}} (\langle x, y\rangle \in$$

$$\{\langle 0, 0\rangle, \langle 0, 5\rangle, \langle 0, 10\rangle, \langle 1, 1\rangle, \langle 1, 6\rangle, \langle 2, 2\rangle, \langle 2, 7\rangle, \langle 3, 3\rangle, \langle 3, 8\rangle, \langle 4, 4\rangle, \langle 4, 9\rangle, \langle 5, 0\rangle, \langle 5, 5\rangle,$$
$$\langle 5, 10\rangle, \langle 6, 1\rangle, \langle 6, 6\rangle, \langle 7, 2\rangle, \langle 7, 7\rangle, \langle 8, 3\rangle, \langle 8, 8\rangle, \langle 9, 4\rangle, \langle 9, 9\rangle, \langle 10, 0\rangle, \langle 10, 5\rangle, \langle 10, 10\rangle\} \Rightarrow$$
$$\langle y, x\rangle \in \{\langle 0, 0\rangle, \langle 0, 5\rangle, \langle 0, 10\rangle, \langle 1, 1\rangle, \langle 1, 6\rangle, \langle 2, 2\rangle, \langle 2, 7\rangle, \langle 3, 3\rangle, \langle 3, 8\rangle, \langle 4, 4\rangle, \langle 4, 9\rangle, \langle 5, 0\rangle,$$
$$\langle 5, 5\rangle, \langle 5, 10\rangle, \langle 6, 1\rangle, \langle 6, 6\rangle, \langle 7, 2\rangle, \langle 7, 7\rangle, \langle 8, 3\rangle, \langle 8, 8\rangle, \langle 9, 4\rangle, \langle 9, 9\rangle, \langle 10, 0\rangle, \langle 10, 5\rangle, \langle 10, 10\rangle\})$$

Using available computation rules we simplify (2):

(3) True.

Formula (3) is true because it is the constant True.

□

5.4 Computing ⊂ Proving

5.4.1 Reflexivity

$$\text{Compute}\left[\text{reflexive}_{\mathbb{N}}\left[\left\{\langle \text{i, j}\rangle \underset{\text{i,j}\in\mathbb{N}}{|} (\text{Mod[i, 5]} = \text{Mod[j, 5]})\right\}\right], \text{using} \to \langle \text{Definition["reflexivity"]}\rangle, \text{built-in} \to \right.$$
$$\left.\langle \text{Built-in["Sets"], Built-in["Quantifiers"], Built-in["Connectives"], Built-in["Tuples"][AngleBracket]}\rangle\right]$$

$$\forall_{x\in\mathbb{N}}\left(\langle x, x\rangle \in \left\{\langle i, j\rangle \underset{\substack{i\in\mathbb{N}\\ j\in\mathbb{N}}}{|} \text{Mod}[i, 5] = \text{Mod}[j, 5]\right\}\right)$$

5.4.2 Symmetry

$$\text{Compute}\left[\text{symmetric}_{\mathbb{N}}\left[\left\{\langle \text{i, j}\rangle \underset{\text{i,j}\in\mathbb{N}}{|} (\text{Mod[i, 5]} = \text{Mod[j, 5]})\right\}\right], \text{using} \to \langle \text{Definition["symmetry"]}\rangle, \text{built-in} \to \right.$$
$$\left.\langle \text{Built-in["Sets"], Built-in["Quantifiers"], Built-in["Connectives"], Built-in["Tuples"][AngleBracket]}\rangle\right]$$

$$\forall_{\substack{x\in\mathbb{N}\\ y\in\mathbb{N}}}\left(\langle x, y\rangle \in \left\{\langle i, j\rangle \underset{\substack{i\in\mathbb{N}\\ j\in\mathbb{N}}}{|} \text{Mod}[i, 5] = \text{Mod}[j, 5]\right\} \Rightarrow \langle y, x\rangle \in \left\{\langle i, j\rangle \underset{\substack{i\in\mathbb{N}\\ j\in\mathbb{N}}}{|} \text{Mod}[i, 5] = \text{Mod}[j, 5]\right\}\right)$$

5.4.3 Representation of Equivalence Class

$$\text{Compute}\left[\text{class}_{\mathbb{N},\{\langle i,j\rangle \mid i,j\in\mathbb{N}\ \wedge\ (\text{Mod}[i,5]=\text{Mod}[j,5])\}}[4] = \{4+5\,k \mid k\in\mathbb{N}\},\ \text{using} \to \langle\text{Definition["Equivalence class"]}\rangle,\ \text{built-in} \to \langle\text{Built-in["Sets"]},\ \text{Built-in["Quantifiers"]}\rangle\right]$$

$$\{a \mid a\in\mathbb{N}\ \wedge\ \langle a,4\rangle \in \{\langle i,j\rangle \mid i\in\mathbb{N}\ \wedge\ j\in\mathbb{N}\ \wedge\ \text{Mod}[i,5]=\text{Mod}[j,5]\}\} = \{4+5*k \mid k\in\mathbb{N}\}$$

5.5 Solving ⊆ Proving

Goal: **Make use of available algebraic solution techniques during proving!**

5.5.1 Reflexivity

$$\textbf{Proposition}\left["\text{R reflexive on }\mathbb{N}",\ \text{reflexive}_{\mathbb{N}}\left[\{\langle i,j\rangle \mid i,j\in\mathbb{N}\ \wedge\ (\text{Mod}[i,5]=\text{Mod}[j,5])\}\right]\right]$$

$$\text{Prove}\left[\text{Proposition["R reflexive on }\mathbb{N}\text{"]},\ \text{using} \to \text{Definition["reflexivity"]},\ \text{built-in} \to \text{Built-in["Tuples"]},\ \text{by} \to \text{SetTheoryPCSProver},\ \text{ProverOptions} \to \{\text{ApplyBuiltIns} \to \{\text{"kb"}\},\ \text{SimplifyFormula} \to \text{False},\ \text{DisableProver} \to \{\text{STC, PND}\}\}\right]$$

Prove:

(Proposition (R reflexive on $\mathbb{N}$)) $\text{reflexive}_{\mathbb{N}}\left[\{\langle i,j\rangle \mid i,j\ (i\in\mathbb{N}\wedge(\text{Mod}[i,5]=\text{Mod}[j,5]))\wedge j\in\mathbb{N}\}\right]$,

under the assumption:

(Definition (reflexivity)) $\underset{Rel,S}{\forall}\left(\text{reflexive}_S[Rel] :\Leftrightarrow \underset{x}{\forall}\,(x\in S\Rightarrow\langle x,x\rangle\in Rel)\right)$.

Formula (Proposition (R reflexive on $\mathbb{N}$)), using (Definition (reflexivity)), is implied by:

(1) $\underset{x}{\forall}\left(x\in\mathbb{N}\Rightarrow\langle x,x\rangle\in\{\langle i,j\rangle \mid i,j\ i\in\mathbb{N}\wedge j\in\mathbb{N}\wedge(\text{Mod}[i,5]=\text{Mod}[j,5])\}\right)$.

We assume

(2) $x_0\in\mathbb{N}$,

and show

(3) $\langle x_0, x_0\rangle \in \left\{\langle i, j\rangle \mid_{i,j} i \in \mathbb{N} \wedge j \in \mathbb{N} \wedge (\text{Mod}[i, 5] = \text{Mod}[j, 5])\right\}$.

In order to prove (3) we have to show

(4) $\exists_{i,j} ((i \in \mathbb{N} \wedge j \in \mathbb{N} \wedge (\text{Mod}[i, 5] = \text{Mod}[j, 5])) \wedge (\langle x_0, x_0\rangle = \langle i, j\rangle))$.

Since $i := x_0$ and $j := x_0$ solves the equational part of (4) it suffices to show

(5) $x_0 \in \mathbb{N} \wedge x_0 \in \mathbb{N} \wedge (\text{Mod}[x_0, 5] = \text{Mod}[x_0, 5])$.

We prove the individual conjunctive parts of (5):

Proof of (5.1) $x_0 \in \mathbb{N}$:

Formula (5.1) is true because it is identical to (2).

Proof of (5.2) $x_0 \in \mathbb{N}$:

Formula (5.2) is true because it is identical to (2).

Proof of (5.3) $\text{Mod}[x_0, 5] = \text{Mod}[x_0, 5]$:

It can be easily verified that formula (5.3) always holds.

□

5.5.2 Symmetry

```
Proposition["R symmetric on ℕ",
  symmetric_ℕ[{⟨i, j⟩ | i,j∈ℕ (Mod[i, 5] = Mod[j, 5])}]]
```

```
Prove[Proposition["R symmetric on ℕ"],
  using → Definition["symmetry"],
  built-in → Built-in["Tuples"],
  by → SetTheoryPCSProver, ProverOptions → {ApplyBuiltIns → {"kb"},
    SimplifyFormula → False, DisableProver → {STC, PND}}]
```

Prove:

(Proposition (R symmetric on ℕ)) $\text{symmetric}_{\mathbb{N}}\left[\left\{\langle i, j\rangle \mid_{i,j} (i \in \mathbb{N} \wedge (\text{Mod}[i, 5] = \text{Mod}[j, 5])) \wedge j \in \mathbb{N}\right\}\right]$,

under the assumption:

(Definition (symmetry)) $\forall_{Rel,S} \left(\text{symmetric}_S[Rel] :\Leftrightarrow \forall_{x,y} (x \in S \wedge y \in S \Rightarrow (\langle x, y\rangle \in Rel \Rightarrow \langle y, x\rangle \in Rel))\right)$.

Formula (Proposition (R symmetric on ℕ)), using (Definition (symmetry)), is implied by:

(1) $\underset{x,y}{\forall}\left(x \in \mathbb{N} \wedge y \in \mathbb{N} \Rightarrow \left(\langle x, y\rangle \in \left\{\langle i, j\rangle \mid \underset{i,j}{} i \in \mathbb{N} \wedge j \in \mathbb{N} \wedge (\text{Mod}[i, 5] = \text{Mod}[j, 5])\right\} \Rightarrow .\right.\right.$

$\left.\left.\langle y, x\rangle \in \left\{\langle i, j\rangle \mid \underset{i,j}{} i \in \mathbb{N} \wedge j \in \mathbb{N} \wedge (\text{Mod}[i, 5] = \text{Mod}[j, 5])\right\}\right)\right)$

We assume

(2) $x_0 \in \mathbb{N} \wedge y_0 \in \mathbb{N}$,

and show

(3) $\langle x_0, y_0\rangle \in \left\{\langle i, j\rangle \mid \underset{i,j}{} i \in \mathbb{N} \wedge j \in \mathbb{N} \wedge (\text{Mod}[i, 5] = \text{Mod}[j, 5])\right\} \Rightarrow .$

$\langle y_0, x_0\rangle \in \left\{\langle i, j\rangle \mid \underset{i,j}{} i \in \mathbb{N} \wedge j \in \mathbb{N} \wedge (\text{Mod}[i, 5] = \text{Mod}[j, 5])\right\}$

We prove (3) by the deduction rule.

We assume

(4) $\langle x_0, y_0\rangle \in \left\{\langle i, j\rangle \mid \underset{i,j}{} i \in \mathbb{N} \wedge j \in \mathbb{N} \wedge (\text{Mod}[i, 5] = \text{Mod}[j, 5])\right\}$

and show

(5) $\langle y_0, x_0\rangle \in \left\{\langle i, j\rangle \mid \underset{i,j}{} i \in \mathbb{N} \wedge j \in \mathbb{N} \wedge (\text{Mod}[i, 5] = \text{Mod}[j, 5])\right\}.$

From what we already know follows:

From (4) we know by definition of $\left\{T_x \mid \underset{x}{} P\right\}$ that we can choose an appropriate value such that

(6) $i1_0 \in \mathbb{N} \wedge i2_0 \in \mathbb{N} \wedge (\text{Mod}[i1_0, 5] = \text{Mod}[i2_0, 5])$,

(7) $\langle x_0, y_0\rangle = \langle i1_0, i2_0\rangle$.

Using available computation rules we can simplify the knowledge base:

Formula (7) simplifies to

(8) $(x_0 = i1_0) \wedge (y_0 = i2_0)$,

In order to prove (5) we have to show

(9) $\underset{i,j}{\exists}((i \in \mathbb{N} \wedge j \in \mathbb{N} \wedge (\text{Mod}[i, 5] = \text{Mod}[j, 5])) \wedge (\langle y_0, x_0\rangle = \langle i, j\rangle))$.

Since $i := y_0$ and $j := x_0$ solves the equational part of (9) it suffices to show

(10) $y_0 \in \mathbb{N} \wedge x_0 \in \mathbb{N} \wedge (\text{Mod}[y_0, 5] = \text{Mod}[x_0, 5])$.

We prove the individual conjunctive parts of (10):

Proof of (10.1) $y_0 \in \mathbb{N}$:

Formula (10.1) is true because it is identical to (2.2).

Proof of (10.2) $x_0 \in \mathbb{N}$:

Formula (10.2) is true because it is identical to (2.1).

Proof of (10.3) $\text{Mod}[y_0, 5] = \text{Mod}[x_0, 5]$:

Formula (10.3), using (8.1), is implied by:

$$\text{Mod}[y_0, 5] = \text{Mod}[i1_0, 5],$$

which, using (6.3), is implied by:

$$\text{Mod}[y_0, 5] = \text{Mod}[i2_0, 5],$$

which, using (8.2), is implied by:

(11) $\text{Mod}[i2_0, 5] = \text{Mod}[i2_0, 5]$.

It can be easily verified that formula (11) always holds.

□

5.6 Computing ⋃ Solving ⊆ Proving

5.6.1 Representation of Equivalence Class

Proposition["class representation",
$\text{class}_{\mathbb{N},\{\langle i,j\rangle \mid i,j\in\mathbb{N}\ (\text{Mod}[i,5]=\text{Mod}[j,5])\}}[4] \supseteq \{4 + 5\,k \mid k\in\mathbb{N}\}$]

We will need two auxiliary Lemmata:

Lemma["Natural Numbers", any[i, j, k ∈ ℕ],
i + k j ∈ ℕ]

Lemma["Mod Properties", any[i, j, k],
Mod[i + j k, j] = i]

SetGlobals[SearchDepth → 50]

```
Prove[Proposition["class representation"],
  using → ⟨Lemma["Natural Numbers"], Lemma["Mod Properties"], Definition["Equivalence class"]⟩,
  built-in → ⟨Built-in["Numbers"], Built-in["Number Domains"], Built-in["Tuples"]⟩,
  by → SetTheoryPCSProver,
  ProverOptions → {SimplifyFormula → False, DisableProver → {PND}}] //
 Timing
```

```
{29.63 Second, - ProofObject -}
```

Prove:

(Proposition (class representation)) $\text{class}_{\mathbb{N},\{\langle i,j\rangle \mid (i\in\mathbb{N}\wedge(\text{Mod}[i,5]=\text{Mod}[j,5]))\wedge j\in\mathbb{N}\}}[4] \supseteq \{4+5*k \mid k\in\mathbb{N}\}$,

under the assumptions:

(Lemma (Natural Numbers)) $\forall_{i,j,k}\,(i\in\mathbb{N}\wedge j\in\mathbb{N}\wedge k\in\mathbb{N}\Rightarrow i+k*j\in\mathbb{N})$,

(Lemma (Mod Properties)) $\forall_{i,j,k}\,(\text{Mod}[i+j*k,\ j]=i)$,

(Definition (Equivalence class)) $\forall_{A,R,x}\left(\text{class}_{A,R}[x] := \{a \mid a\in A\wedge\langle a,\ x\rangle\in R\}\right)$.

Using available computation rules we can simplify the knowledge base:

Formula (Lemma (Natural Numbers)) simplifies to

(1) $\forall_{i,j,k}\left(\in_{\mathbb{N}}[i]\bigwedge\in_{\mathbb{N}}[j]\bigwedge\in_{\mathbb{N}}[k]\Rightarrow\in_{\mathbb{N}}[i+j*k]\right)$,

For proving (Proposition (class representation)) we choose

(2) $k1_0\in\{4+5*k \mid k\in\mathbb{N}\}$,

and show:

(3) $k1_0\in\text{class}_{\mathbb{N},\{\langle i,j\rangle \mid (i\in\mathbb{N}\wedge(\text{Mod}[i,5]=\text{Mod}[j,5]))\wedge j\in\mathbb{N}\}}[4]$.

From what we already know follows:

From (2) we know by definition of $\{T_x \mid P\}$ that we can choose an appropriate value such that

(4) $k2_0\in\mathbb{N}$,

(5) $k1_0=4+5*k2_0$.

Using available computation rules we can simplify the knowledge base:

Formula (4) simplifies to

(6) $\underset{\mathbb{N}}{\in}[k2_0]$,

Formula (3), using (5), is implied by:

$4+5*k2_0 \in \text{class}_{\mathbb{N},\{\langle i,j\rangle \mid i\in\mathbb{N}\wedge j\in\mathbb{N}\wedge(\text{Mod}[i,5]=\text{Mod}[j,5])\}}[4]$,

which, using (Definition (Equivalence class)), is implied by:

(7) $4+5*k2_0 \in \left\{a \underset{a}{\mid} a\in\mathbb{N}\bigwedge\langle a,4\rangle\in\left\{\langle i,j\rangle \underset{i,j}{\mid} i\in\mathbb{N}\wedge j\in\mathbb{N}\wedge(\text{Mod}[i,5]=\text{Mod}[j,5])\right\}\right\}$.

In order to prove (7) we have to show:

(8) $4+5*k2_0\in\mathbb{N}\bigwedge\langle 4+5*k2_0,4\rangle\in\left\{\langle i,j\rangle \underset{i,j}{\mid} i\in\mathbb{N}\wedge j\in\mathbb{N}\wedge(\text{Mod}[i,5]=\text{Mod}[j,5])\right\}$.

Using available computation rules we simplify (8):

(9) $\underset{\mathbb{N}}{\in}[4+5*k2_0]\bigwedge\langle 4+5*k2_0,4\rangle\in\left\{\langle i,j\rangle \underset{i,j}{\mid} i\in\mathbb{N}\wedge j\in\mathbb{N}\wedge(\text{Mod}[i,5]=\text{Mod}[j,5])\right\}$.

We prove the individual conjunctive parts of (9):

Proof of (9.1) $\underset{\mathbb{N}}{\in}[4+5*k2_0]$:

Formula (9.1), using (1), is implied by:

(10) $\underset{\mathbb{N}}{\in}[4]\bigwedge\underset{\mathbb{N}}{\in}[5]\bigwedge\underset{\mathbb{N}}{\in}[k2_0]$.

Using available computation rules we simplify (10):

(11) $\underset{\mathbb{N}}{\in}[k2_0]$.

Formula (11) is true because it is identical to (6).

Proof of (9.2) $\langle 4+5*k2_0,4\rangle\in\left\{\langle i,j\rangle \underset{i,j}{\mid} i\in\mathbb{N}\wedge j\in\mathbb{N}\wedge(\text{Mod}[i,5]=\text{Mod}[j,5])\right\}$:

In order to prove (9.2) we have to show

(12) $\underset{i,j}{\exists}((i\in\mathbb{N}\wedge j\in\mathbb{N}\wedge(\text{Mod}[i,5]=\text{Mod}[j,5]))\wedge(\langle 4+5*k2_0,4\rangle=\langle i,j\rangle))$.

Using available computation rules we simplify (12):

(13) $\underset{i,j}{\exists}\left(\underset{\mathbb{N}}{\in}[i]\bigwedge\underset{\mathbb{N}}{\in}[j]\bigwedge(\text{Mod}[i,5]=\text{Mod}[j,5])\bigwedge(4+5*k2_0=i)\bigwedge(4=j)\right)$.

Since $i:=4+5*k2_0$ and $j:=4$ solves the equational part of (13) it suffices to show

(14) $(\text{Mod}[4+5*k2_0,5]=4)\bigwedge\underset{\mathbb{N}}{\in}[4+5*k2_0]\bigwedge\underset{\mathbb{N}}{\in}[4]$.

Using available computation rules we simplify (14):

(15) $(\text{Mod}[4 + 5 * k2_0, 5] = 4) \bigwedge \underset{\mathbb{N}}{\in} [4 + 5 * k2_0]$.

We prove the individual conjunctive parts of (15):

Proof of (15.1) $\text{Mod}[4 + 5 * k2_0, 5] = 4$:

Formula (15.1) is proved because it is an instance of (Lemma (Mod Properties)).

Proof of (15.2) $\underset{\mathbb{N}}{\in} [4 + 5 * k2_0]$:

Formula (15.2), using (1), is implied by:

(16) $\underset{\mathbb{N}}{\in} [4] \bigwedge \underset{\mathbb{N}}{\in} [5] \bigwedge \underset{\mathbb{N}}{\in} [k2_0]$.

Using available computation rules we simplify (16):

(17) $\underset{\mathbb{N}}{\in} [k2_0]$.

Formula (17) is true because it is identical to (6).

□

5.6.2 Transitivity of Induced Relation

Proposition["induced by partition is transitive", any[P], with[is-partition[P]],
is-transitive[induced-relation[P]]]

Auxiliary:

Proposition["intersecting are equal", any[P], with[is-partition[P]],
$\forall_{X,Y \in P} (X \cap Y \neq \emptyset \Rightarrow (X = Y))$]

```
SetGlobals[SearchDepth → 80]
```

```
Prove[Proposition["induced by partition is transitive"],
    using → ⟨Definition["transitivity"], Definition["induced relation"], Proposition["intersecting are equal"]⟩,
    built-in → Built-in["Tuples"],
    by → SetTheoryPCSProver,
    ProverOptions → {GRWTarget → {"goal", "kb"}, ModusPonensKB → True,
        InferMembershipIntersection → True, AllowIntroduceQuantifiers → True,
        ApplyBuiltIns → {"kb"}, DisableProver → {PND}}] // Timing
```

```
{338.24 Second, - ProofObject -}
```

Prove:

(Proposition (induced by partition is transitive)) $\underset{P}{\forall}$ (is-partition[P] ⇒ is-transitive[induced-relation[P]]),

under the assumptions:

(Definition (transitivity)) $\underset{R}{\forall}\left(\text{is-transitive}[R] :\Leftrightarrow \underset{x,y,z}{\forall} (\langle x, y\rangle \in R \wedge \langle y, z\rangle \in R \Rightarrow \langle x, z\rangle \in R)\right)$,

(Definition (induced relation)) $\underset{S}{\forall}\left(\text{induced-relation}[S] :\Leftrightarrow \left\{\langle x, y\rangle \mid \underset{x,y}{\exists}\, \underset{M\in S}{\exists} (x \in M \wedge y \in M)\right\}\right)$,

(Proposition (intersecting are equal))

$\underset{P}{\forall}\left(\text{is-partition}[P] \Rightarrow \underset{X,Y}{\forall} (X \in P \wedge Y \in P \Rightarrow ((X \bigcap Y \neq \{\}) \Rightarrow (X = Y)))\right)$.

We assume

(1) is-partition[P_0],

and show

(2) is-transitive[induced-relation[P_0]].

Formula (2), using (Definition (transitivity)), is implied by:

(3) $\underset{x,y,z}{\forall} (\langle x, y\rangle \in \text{induced-relation}[P_0] \wedge \langle y, z\rangle \in \text{induced-relation}[P_0] \Rightarrow \langle x, z\rangle \in \text{induced-relation}[P_0])$.

We assume

(4) $\langle x_0, y_0\rangle \in \text{induced-relation}[P_0] \wedge \langle y_0, z_0\rangle \in \text{induced-relation}[P_0]$,

and show

(5) $\langle x_0, z_0\rangle \in \text{induced-relation}[P_0]$.

From what we already know follows:

From (4.1) we can infer

(6) induced-relation[P_0] ≠ {}.

Formula (5), using (Definition (induced relation)), is implied by:

(8) $\langle x_0, z_0\rangle \in \left\{\langle x, y\rangle \mid \underset{x,y}{\exists}\, \underset{M\in P_0}{\exists} (x \in M \wedge y \in M)\right\}$.

In order to prove (8) we have to show

(9) $\underset{x,y}{\exists}\left(\underset{M}{\exists} (M \in P_0 \wedge (x \in M \wedge y \in M)) \bigwedge (\langle x_0, z_0\rangle = \langle x, y\rangle)\right)$.

Since $x := x_0$ and $y := z_0$ solves the equational part of (9) it suffices to show

(10) $\underset{M}{\exists} (M \in P_0 \wedge (x_0 \in M \wedge z_0 \in M))$.

Formula (6), by (Definition (induced relation)), implies:

(11) $\left\{\langle x, y\rangle \mid \underset{x,y}{\exists}\ \underset{M \in P_0}{\exists}\ (x \in M \wedge y \in M)\right\} \neq \{\}$.

We did not find any ground formula to match a part of (10).

Formula (4.1), by (Definition (induced relation)), implies:

(12) $\langle x_0, y_0\rangle \in \left\{\langle x, y\rangle \mid \underset{x,y}{\exists}\ \underset{M \in P_0}{\exists}\ (x \in M \wedge y \in M)\right\}$.

From what we already know follows:

From (12) we know by definition of $\left\{T_x \mid \underset{x}{P}\right\}$ that we can choose an appropriate value such that

(13) $\underset{M}{\exists}\ (M \in P_0 \wedge (x1_0 \in M \wedge x2_0 \in M))$,

(14) $\langle x_0, y_0\rangle = \langle x1_0, x2_0\rangle$.

Using available computation rules we can simplify the knowledge base:

Formula (13) simplifies to

(15) $\underset{M}{\exists}\ (M \in P_0 \wedge x1_0 \in M \wedge x2_0 \in M)$,

Formula (14) simplifies to

(16) $(x_0 = x1_0) \wedge (y_0 = x2_0)$,

By (15) we can take appropriate values such that:

(17) $M_0 \in P_0 \wedge x1_0 \in M_0 \wedge x2_0 \in M_0$.

From what we already know follows:

From (17.1) we can infer

(18) $P_0 \neq \{\}$.

From (17.2) we can infer

(19) $M_0 \neq \{\}$.

Because parts of the knowledge base match a part of (10), we try to find an instance of (10).

Now, let $M := M_0$. Thus, for proving (10) it is sufficient to prove:

(21) $M_0 \in P_0 \wedge (x_0 \in M_0 \wedge z_0 \in M_0)$.

We prove the individual conjunctive parts of (21):

Proof of (21.1) $M_0 \in P_0$:

Formula (21.1) is true because it is identical to (17.1).

Proof of (21.2) $x_0 \in M_0 \wedge z_0 \in M_0$:

We prove the individual conjunctive parts of (21.2):

Proof of (21.2.1) $x_0 \in M_0$:

Formula (21.2.1), using (16.1), is implied by:

(22) $x1_0 \in M_0$.

Formula (22) is true because it is identical to (17.2).

Proof of (21.2.2) $z_0 \in M_0$:

Formula (4.2), by (16.2), implies:

$\langle x2_0, z_0 \rangle \in \text{induced-relation}[P_0]$,

which, by (Definition (induced relation)), implies:

(23) $\langle x2_0, z_0 \rangle \in \left\{ \langle x, y \rangle \mid_{x,y} \exists_{M \in P_0} (x \in M \wedge y \in M) \right\}$.

From what we already know follows:

From (23) we know by definition of $\left\{ T_x \mid_x P \right\}$ that we can choose an appropriate value such that

(24) $\exists_M (M \in P_0 \wedge (x3_0 \in M \wedge x4_0 \in M))$,

(25) $\langle x2_0, z_0 \rangle = \langle x3_0, x4_0 \rangle$.

Using available computation rules we can simplify the knowledge base:

Formula (24) simplifies to

(26) $\exists_M (M \in P_0 \wedge x3_0 \in M \wedge x4_0 \in M)$,

Formula (25) simplifies to

(27) $(x2_0 = x3_0) \wedge (z_0 = x4_0)$,

By (26) we can take appropriate values such that:

(28) $M_1 \in P_0 \wedge x3_0 \in M_1 \wedge x4_0 \in M_1$.

From what we already know follows:

From (28.2) we can infer

(30) $M_1 \neq \{\}$.

Formula (21.2.2), using (27.2), is implied by:

(32) $x4_0 \in M_0$.

Formula (17.3), by (27.1), implies:

(33) $x3_0 \in M_0$.

From what we already know follows:

From (28.2) together with (33) we know

(35) $x3_0 \in M_1 \bigcap M_0$.

From what we already know follows:

From (35) we can infer

(36) $M_1 \bigcap M_0 \neq \{\}$.

Formula (36), by (Proposition (intersecting are equal)), implies:

(37) $\underset{P}{\forall}$ (is-partition[P] $\wedge$ $M_0 \in P \wedge M_1 \in P \Rightarrow (M_1 = M_0)$).

Formula (17.1), by (37), implies:

(71) is-partition[P_0] $\Rightarrow (M_1 \in P_0 \Rightarrow (M_1 = M_0))$.

From (1) and (71) we obtain by modus ponens

(72) $M_1 \in P_0 \Rightarrow (M_1 = M_0)$.

From (28.1) and (72) we obtain by modus ponens

(73) $M_1 = M_0$.

Formula (32) is true because of (28.3) and (73).

□

Comments on the Method (from the extended abstract)

The proof starts with a general "P" (prove) phase, which structures the proof according to the structure of the goal and the knowledge base. Natural deduction inference rules are used to eliminate universal quantifier(s) and to decompose compound formulae in the goal and in the knowledge base. As soon as one is left with atomic formulae (e.g. formula (2) and (5)) the "P" phase is terminated and a "C" (compute, simplify) phase is initiated, which uses particular formulae in the knowledge base in order to rewrite the goal or other formulae in the knowledge base. In this phase, definitions for function and predicate symbols are applied to goal and knowledge. After some cycles of "P" and "C" phases we arrive at formula (8), which then initiates a special "P" phase for handling formulae whose outermost symbol is a set operator. In this example, the goal is of the form $t \in \left\{T_{\underset{x}{x}} \mid c\right\}$, which, "by definition", introduces an existential quantifier. Note however, that the definition of this variant of the set constructor is applied as an inference rule in the special set theory "P" phase, and it need not be given as a definition in the knowledge base.

Formula (9) is an existential goal, which in many situations describes a proof problem that can be handled by *solving* the quantified formula (or parts of it) with respect to the quantified variables. This is done now in an "S" (solve) phase. It of course depends on the structure of the formula, which solving technique is appropriate, but all available powerful black–box solving techniques (Gauss method, Gröbner bases method, Collins' algorithm) are natural candidates to be applied in this step. In the example, one part of the formula is a tuple equation, which, after transforming it into a *system of individual equations*, can be solved by the standard

Mathematica "Solve" command. We then continue to proof the remaining part of formula (9). Formula (10) is still an existential goal, but this time, due to the structure of the quantified formula, no solve technique is available, and we therefore suspend the "S" phase.

The steps from formula (12) until (17) are again cycles of "P" and "C", where (13) and (14) are derived by using definitions built into the prover, and, by application of *Theorema* language semantics of logical connectives and tuples, simplified to (15) and (16), repectively. Having new knowledge in the knowledge base, we now re−enterthe "S" phase and we solve the existential goal (10) by basic predicate logic solving, i.e. we try to instanciate (10) guided by available formulae in the knowledge base. We now leave the "S" phase and continue in a "P"−"C"cycles. By a special inference technique for set theory, we infer formula (35): if we know $x \in A$ and also $x \in B$ then we can infer $x \in A \bigcap B$! Formula (35) is now the key to make use of the auxiliary property about partitions given in the knowledge base and the proof quickly succeeds.

6 Conclusion

Two main theorems in the area of equivalence relations, induced relations, and partitions

Theorem["induced by factor set of R is R itself", any[R], with[is-equivalence[R]],
induced-relation[factor-set[R]] = R]

Theorem["factor set of induced by P is P itself", any[P], with[is-partition[P]],
factor-set[induced-relation[P]] = P]

together with **all auxiliary propositions and lemmata** starting from the definitions have been proved **completely automatized** using the Set Theory PCS Prover. After this case study we think that structuring proving into alternating cycles of P−C−S

6.1

is a feasible approach for structuring automated theorem provers,

6.2

gives naturally structured proofs, **AND MOST IMPORTANTLY**

6.3

opens the door for incorporating powerful known solving and computing techniques from the field of Computer−Algebra(Gröbner bases, Collins' algorithm, Gauss method, Lazy Narrowing, etc.) into automated theorem provers.