数理解析研究所講究録
1514巻 2006年 148-154

Presentation Hensel's Lemma in Isabelle/HOL

小林 英恒

Hidetsune Kobayashi

日本大学 理工学部 数学科

Department of Mathematics, Faculty of Science and Technology, Nihon University [*]

鈴木 秀男

Hideo Suzuki

能力開発総合大学校 東京校 情報技術科

Department of Information Technology, Tokyo Institute, Polytechnic University [†]

Abstract

Representation of Hensel's lemma in Isabelle/HOL is reported. The lemma is concerned with a factorization of a polynomial with coefficients in a valuation ring. A trial to make an automated inference system is also discussed.

1 Introduction

We see formula manipulation can solve problems of differentiation and integration. Moreover some formula manipulation language gives graphs that visualize mathematical properties. But one of the authors has long been unsatisfied to these formula manipulation languages, because they cannot treat theory itself.

Isabelle/HOL is one of the computer languages with that we can express theory itself. In this report we show that Isabelle/HOL has enough capability to express abstract mathematics by giving representation of Hensel's lemma as an example. Hensel's lemma treats polynomials over a discrete valuation ring, and it shows that a factorization over residue field can be lifted up to a factorization of the original polynomial ring over the valuation ring.

In the last section we report that a database and Isabelle/HOL are combined and we can store proofs in the database. We are trying to make an automated reasoning system that gives some steps of a proof automatically. It is interesting to give a question "is it possible to solve a problem from exercises, by choosing automatically proper lemmas or proof methods from the database", provided there is no answer to the problem in the database. To the goal, we have very long way, but we are thinking that it is not impossible to make a system giving automatically some steps of a part of proof.

[*] hikoba@math.cst.nihon-u.ac.jp
[†] hsuzuki@tokyo-pc.ac.jp

2 Valuation and valuation ring

We define a valuation ring by using a valuation. In this section we give definitions expressed in Isabelle/HOL and explain the meaning.

2.1 Valuation and Valuation ring

We give a definition of a valuation expressed in Isabelle/HOL:

```
valuation::"['b FieldType, 'b ⇒ ant] ⇒ bool"
  "valuation K v == v ∈ extensional (carrier K) ∧ v ∈ carrier K → Z∞
    ∧ v (0K) = ∞ ∧ (∀x∈((carrier K) - 0K).  v x ≠ ∞) ∧ (∀x∈(carrier K).
    ∀y∈(carrier K). v (x ·K y) = (v x) + (v y)) ∧ (∀x∈(carrier K).
    0 ≤ (v x) ⟶ 0 ≤ (v (1K +K x))) ∧ (∃x.  x ∈ carrier K ∧ (v x) ≠ ∞
    ∧ (v x) ≠ 0)"
```

The first two lines gives a type of a valuation. 'b FieldType is a type of algebraic object defined as:

```
record 'a FieldType = "'a RingType" + iOp ::  'a ⇒ 'a
```

This means that 'a FieldType is a set of components 'a RingType and one more component iOp :: 'a ⇒ 'a, where 'a RingType is a set of components having components "carrier" as a base set with element of type 'a, and two binary operations bOp1::'a ⇒ 'a ⇒ 'a and bOp2::'a ⇒ 'a ⇒ 'a and iOp1::'a ⇒ 'a and two units with respect to bOp1 and bOp2. One of the binary operations is called addition and the other called multiplication. The iOp above is the inverse operator of multiplication. We can illustrate this record as

```
(| carrier, bOp1, bOp2, iOp1, unit1, unit2, iOp |)
```

We give a definition of a field as

```
constdefs
  field ::  "('a, 'm) FieldType_scheme ⇒ bool"
    "field K == ring K ∧ (iOp K) ∈ (carrier K - 0K) → (carrier K - 0K)
    ∧ (∀x∈(carrier K - 0K).  (iOp K x) ·K x = 1K)"
```

This means a field is a FieldType satisfying above conditions. This is an expression of "field" in Isabelle/HOL.

A valuation v is a map of carrier K to Z∪∞ satisfying conditions listed above:

1. v is a map form the carrier K to Z_∞
2. $v(0_K) = \infty$
3. v (x) is not equal to ∞_ if $x \in$ carrier K - 0_K
4. $v\ (x \cdot_K y) = v\ (x) + v\ (y)$ forall x, y in carrier K
5. if $0 \leq v\ (x)$ then $0 \leq v\ (1_K + x)$
6. $\exists x \in K$ such that $v\ (x) \neq \infty$ and $v\ (x) \neq 0$

A valuation ring Vr K v is a ring whose carrier is a set of elements having non-negative value:

```
Vr::"['r FieldType, 'r ⇒ ant] ⇒ 'r RingType"
  "Vr K v == Sr K (x.  x ∈ carrier K ∧ 0 ≤ (v x))"
```

The set of all positive elements of the carrier (Vr K V) is the only one maximal ideal of Vr K v and the maximal ideal is principal. Moreover any nonzero ideal of (Vr K v) is a power of the maximal ideal:

```
lemma ideal_pow_vp:"[| field K; valuation K v; ideal (Vr K v) I;
                       I ≠ carrier (Vr K v); I ≠ 0_(Vr K v) |]
                       ⟹ I = (vp K v)^◇(Vr K v) (na (n_val K v (Ig K v I)))"
```

where, vp K v is the maximal ideal of the valuation ring. Since the maximal ideal is principal, any ideal of "Vr K v" is principal. There for the valuation ring is an integral domain, Noetherian. Hensel's lemma treats a polynomial in the ring (Vr K v)[X].

3 Convergence with respect to the (vp K v)-adic topology

A limit of a sequence of elements of the carrier K is defined as

```
limit ::"['b FieldType, 'b ⇒ ant, nat ⇒ 'b, 'b] ⇒ bool"
                              ("(4lim_ _ _ _)" [90,90,90,91]90)
"lim_K v f b == ∀N. ∃M. (∀n.  M < n ⟶
                              ((f n) +_K (-_K b)) ∈ (vp K v)^(Vr K v) (an N))"
```

Here K is the field on which the valuation v is defined. f is a function from a set of natural numbers to the carrier K. We can take this f as a series of the elements of the carrier K. b is an element of the carrier K, which is a limit of the series of elements f. We can prove that the limit of a series is unique, provided that the series converges. A Cauchy sequence is defined as

```
Cauchy_seq::"['b FieldType, 'b ⇒ ant, nat ⇒ 'b] ⇒ bool"
                              ("(3Cauchy_ _ _)" [90,90,91]90)
  "Cauchy_K v f == (∀n.  (f n) ∈ carrier K) ∧ (∀N. ∃M. (∀n m.  M < n
    ∧ M < m ⟶ ((f n) +_K (-_K (f m))) ∈ (vp K v)^(Vr K v) (an N)))"
```

It is easy to see that this definition is similar to the definition in complex analysis. We say K is complete with respect to the valuation v if any Cauchy sequence has a limit in K:

```
v_complete::"['b ⇒ ant, 'b FieldType] ⇒ bool"
                              ("(2Complete_ _)" [90,91]90)
  "Complete_v K == ∀f.  (Cauchy_K v f) ⟶ (∃b.  b ∈ (carrier K) ∧ lim_K v f b)"
```

We define a Cauchy sequence of polynomials from (Vr K v)[X] as

```
pol_Cauchy_seq::"[('b, 'm) RingType_scheme, 'b, 'b FieldType, 'b
     ⇒ ant, nat ⇒ 'b] ⇒ bool" ("(5PCauchy_ _ _ _ _)" [90,90,90,90,91]90)
  "PCauchy_R X K v F == (∀n.  (F n) ∈ carrier R) ∧ (∃d.  (∀n.
    deg R (Vr K v) X (F n) ≤ (an d))) ∧ (∀N. ∃M. (∀n m.
    M < n ∧ M < m ⟶ P_mod R (Vr K v) X ((vp K v)^(Vr K v) (an N))
    (F n +_R -_R (F m))))"
```

Here, R is the polynomial ring (Vr K v)[X] and F is a function of natural numbers having value in R, that is a series of polynomials. deg R (Vr K v) X (F n) is the degree of the polynomial F n and (an d) is the natural number d taken as a number in Z_∞ by type converter an.

pol_limit is defined as

```
pol_limit::"[('b, 'm) RingType_scheme, 'b, 'b FieldType, 'b ⇒ ant,
      nat ⇒ 'b, 'b] ⇒ bool" ("(6Plimit_ _ _ _ _ _)" [90,90,90,90,90,91]90)
   "Plimit_R X K v F p == (∀n.  (F n) ∈ carrier R) ∧ (∀N. ∃M. (∀m.  M < m
     ⟶ P_mod R (Vr K v) X ((vp K v)^((Vr K v) (an N))) ((F m) +_R -_R p)))"
```

Now we can state Hensel's lemma as

Lemma (Hensel)

K : a field

v : a valuation of K, and K is complete with respect to v

O : the valuation ring

M : the maximal ideal of O

f ∈ O[x] a polynomial with coefficients in O

g' and h' are relatively prime polynomials in (O/M)[X] such that

f' = g' h' in (O/M)[X], where f' is the image of f

⟹

∃g, h elements of O[x] such that f = g h and deg g ≤ g', and g' is the image of g and h' is the image of h.

This lemma is expressed in Isabelle/HOL as:

```
theorem Hensel:"[|field K; valuation K v; Complete_v K; ring R;
  polyn_ring R (Vr K v) X; ring S; polyn_ring S (ringF ((Vr K v)
  /_r (vp K v))) Y; f ∈ carrier R; f ≠ 0_R; g' ∈ carrier S;
  h' ∈ carrier S; 0 < deg S (ringF ((Vr K v) /_r (vp K v))) Y g';
  0 < deg S (ringF ((Vr K v) /_r (vp K v))) Y h'; ((ext_rH R
  (Vr K v) X S (ringF ((Vr K v) /_r (vp K v))) Y (pj (Vr K v)
  (vp K v))) f) = g' ·_S h'; rel_prime_pols S (ringF ((Vr K v)
  /_r (vp K v))) Y g' h'|] ⟹
  ∃g h.  g ∈ carrier R ∧ h ∈ carrier R
  ∧ deg R (Vr K v) X g ≤ deg S (ringF ((Vr K v) /_r (vp K v))) Y g'
  ∧ f = g ·_R h"
```

In the above expression, polyn_ring R (Vr K v) X means polynomial ring R that is equal to (Vr K v)[X]. ringF ((Vr K v) $/_r$ (vp K v)) is the residue class field. (ext_rH R (Vr K v) X S (ringF ((Vr K v) $/_r$ (vp K v))) Y (pj (Vr K v) (vp K v))) is a ring homomorphism from (Vr K v)[X] to (ringF ((Vr K v) $/_r$ (vp K v)))[Y] which is a natural extension of the ring homomorphism (pj (Vr K v) (vp K v)): (Vr K v) → (Vr K v) $/_r$ (vp K v).

4 A polynomial ring over a valuation ring

A polynomial is expressed in several ways, for example $1 + X = 1 + X + 0X^2 = 1 + X + 0X^2 + 0X^3 = ...$ This fact makes a definition of polynomial ring a little complicated. At first, we define

```
pol_coeff::"[('a, 'more) RingType_scheme, nat, nat ⇒ 'a] ⇒ bool"
  "pol_coeff S n f == f ∈ Nset n → carrier S"

pol_coeffs::"('a, 'more) RingType_scheme ⇒ (nat ⇒ 'a) set"
  "pol_coeffs S == ∪X. ∃n.  X = Nset n → carrier S"

coeff_len::"[('a, 'more) RingType_scheme, nat ⇒ 'a] ⇒ nat"
  "coeff_len S f == SOME n.  f ∈ (Nset n → carrier S)"

coeff_max::"[('a, 'b) RingType_scheme, nat, nat ⇒ 'a] ⇒ nat"
  "coeff_max S n f == n_max j.  j ≤ n ∧ f j ≠ 0_S"

polyn_expr::"[('a, 'more) RingType_scheme, 'a, nat, nat ⇒ 'a] ⇒ 'a"
  "polyn_expr R X n f == eSum R (λj.  (f j) ·_R (X^R j)) n"

algfree_cond::"[('a, 'm) RingType_scheme, ('a, 'm1) RingType_scheme, 'a]
                ⇒ bool"
  "algfree_cond R S X == ∀n f.  pol_coeff S n f ∧ eSum R (λj.  (f j) ·_R (X^R j))
   n = 0_R ⟶ (∀j∈Nset n.  f j = 0_R)"

polyn_ring::"[('a, 'm) RingType_scheme, ('a, 'm1) RingType_scheme, 'a]
                ⇒ bool"
  "polyn_ring R S X == algfree_cond R S X ∧ ring R ∧ ¬ zeroring R
   ∧ Subring R S ∧ X ∈ carrier R ∧ (∀g∈carrier R. ∃f.  f ∈ pol_coeffs S
   ∧ g = eSum R (λj.  (f j) ·_R (X^R j)) (coeff_len S f))"
```

pol_coeff S n f is a function from Nset n to the carrier S of a ring S. This is taken as n+1 elements of the carrier S: $a_0, a_1, ..., a_n$.

Later, we use these elements as coefficients of a polynomial $a_0 + a_1X + ... + a_nX^n$.

pol_coeffs S is the set of series of elements of S. coeff_len S f is a number n such that f : Nset n → carrier S. We do not claim coeff_len S f is uniquely determined. coeff_max S n f is the maximum number j such that f j $\neq 0_S$. This determines the degree of the polynomial having f as coefficients. polyn_expr R X n f is eSum R (λ j. (f j) $\cdot_R$ $(X^{\wedge R j})$) n. Here eSum stands for the symbol Σ, therefore polyn_expr R X n f = $\sum_{j=0}^{n}$ (f j) X^j

Algfree_cond gives a condition that $1, X, X^2, ..., X^n, ...$ is linearly independent. The definition of the polynomial ring gives ordinary polynomial ring.

In the last section, we described polyn_ring R C X means a polynomial ring C[X] which is written as R, hence R = C[X].

5 Representation of Hensel's construction

Hensel's lemma gives a method to give factors of f in O[X], from factors g' and h' in (O/M)[Y] of f' the image of f in (O/M)[Y]. To explain this procedure, we need some preliminary tools.

```
coeff_sol::"[('a, 'b) RingType_scheme, ('a, 'b1) RingType_scheme, 'a, 'a,
           nat ⇒ 'a] ⇒ bool"
  "coeff_sol R S X g f == f ∈ pol_coeffs S
    ∧ g = polyn_expr R X (coeff_len S f) f"
```

coeff_sol R S X g f is a set of coefficients f to the polynomial g in the polynomial ring R = S[X]: g = $\sum_{j=0}^{\mathrm{coeff\_len}\ f} (f\ j)\ X^j$

```
P_mod::"[('a, 'm) RingType_scheme, ('a, 'm1) RingType_scheme, 'a, 'a set, 'a]
           ⇒ bool"
  "P_mod R S X P p == p = 0_R ∨ (p ≠ 0_R ∧ (∀j∈Nset (coeff_len S (SOME f.
    coeff_sol R S X p f)).  ((SOME f.  coeff_sol R S X p f) j) ∈ P))"
```

P_mod R S X P p means if p = 0_R then true else ∀j∈Nset (coeff_len S (SOME f. coeff_sol R S X p f)). ((SOME f. coeff_sol R S X p f) j) ∈ P)) where Nset (coeff_len S (SOME f. coeff_sol R S X p f)) is the set of natural numbers from 0 through (coeff_len S (SOME f. coeff_sol R S X p f)). Roughly speaking, P_mod R S X P p means p = 0 mod P, where P is an ideal of S.

The following lemma explains P_mod:

```
lemma P_mod_mod:"[|ring R; ring S; polyn_ring R S X; ideal S I; p ∈ carrier R;
                 pol_coeff S n c; p = polyn_expr R X n c|] ⟹
                 (∀j∈Nset n.  c j ∈ I) = (P_mod R S X I p)"
```

If P is a principal ideal S ◇ t then P_mod is written as

```
lemma Pmod_0_principal:"[|ring R; integral_domain S; polyn_ring R S X;
                 t ∈ carrier S; g ∈ carrier R; P_mod R S X (S ◇ t) g|]
                 ⟹ ∃h∈ carrier R. g = t ·_R h"
```

A key lemma for Hensel's lemma is:

```
lemma P_mod_diffxxx2:"[|ring R; integral_domain S; polyn_ring R S X;
  t ∈ carrier S; t ≠ 0_S; maximal_ideal S (S ◇ t); ring R';
  polyn_ring R' (ringF (S /_r (S ◇ t))) Y; f ∈ carrier R; g ∈ carrier R;
  h ∈ carrier R; deg R S X g ≤ deg R' (ringF (S /_r (S ◇ t)))
  Y (ext_rH R S X R' (ringF (S /_r (S ◇ t))) Y (pj S (S ◇ t)) g);
  deg R S X h + deg R' (ringF (S /_r (S ◇ t))) Y (ext_rH R S X R'
  (ringF (S /_r (S ◇ t))) Y (pj S (S ◇ t)) g) ≤ deg R S X f;
  0 < deg R' (ringF (S /_r (S ◇ t))) Y (ext_rH R S X R'
  (ringF (S /_r (S ◇ t))) Y (pj S (S ◇ t)) g);
  0 < deg R' (ringF (S /_r (S ◇ t))) Y (ext_rH R S X R'
  (ringF (S /_r (S ◇ t))) Y (pj S (S ◇ t)) h);
  rel_prime_pols R' (ringF (S /_r (S ◇ t))) Y (ext_rH R S X R'
  (ringF (S /_r (S ◇ t))) Y (pj S (S ◇ t)) g) (ext_rH R S X R'
  (ringF (S /_r (S ◇ t))) Y (pj S (S ◇ t)) h);
  P_mod R S X (S ◇ (t^-S m)) (f +_R -_R (g ·_R h)); 0 < m|] ⟹
```

∃g1 h1. g1 ∈carrier R ∧ h1 ∈ carrier R ∧ (deg R S X g1 ≤ deg R'
(ringF (S $/_r$ (S ◇ t))) Y (ext_rH R S X R' (ringF (S $/_r$ (S ◇ t)))
Y (pj S (S ◇ t)) g1)) ∧ P_mod R S X (S ◇ ($t^{\hat{}S\,m}$)) (g $+_R$ $-_R$ g1)
∧ (deg R S X h1 + deg R' (ringF (S $/_r$ (S ◇ t))) Y (ext_rH R S X R'
(ringF (S $/_r$ (S ◇ t))) Y (pj S (S ◇ t)) g1) ≤ deg R S X f)
∧ P_mod R S X (S ◇ ($t^{\hat{}S\,m}$)) (h $+_R$ $-_R$ h1) ∧ P_mod R S X (S ◇ ($t^{\hat{}S\,(Suc\,m)}$))
(f $+_R$ ($-_R$ (g1 $\cdot_R$ h1)))"

Using this lemma, we construct factors of the original polynomial inductively.

6 Proof database and automated inference

Isabelle is written in ML and the proof interface is ProofGeneral which is based on xemacs. Using ProofGeneral, we can execute proof of a lemma step by step. ProofGeneral and Isabelle is combined in a complicated way, hence it is not easy to take knowledge out from Isabelle and put it into the knowledge database. Conversely, taking proper lemmas from the database and putting them into Isabelle can be said automated inference if the chosen lemmas can proceed proof properly.

We have already made a system putting Isabelle proof procedure step by step into a database. Therefore, the database contains not only lemmas but whole proof steps of the lemmas.

Since proof of a lemma is based on the mathematical concepts already obtained, therefore we have to look for the database to find proper lemma. Of course, it is not guaranteed that we can obtain a proper lemma for a proof of new lemma A. In such case, there are two ways. One is to make a(some) lemma(s) for the proof of A, and put the lemma(s) and proof steps to the database and then try to prove A. Another way is very elementary such that to prove A from the definition and axioms.

But, in any case, the database is useful to check definitions, axioms and lemmas.

A big problem to make an automated inference system is, how we see a lemma is proper or not for a proof of new lemma. In general we have a strategy or key idea to solve a mathematical problem. But even we list up key ideas and store them in the database, we need a method to choose proper ideas.

参考文献

[1] L.C.Paulson : Isabelle - A Generic Theorem Prover, Springer Verlag (1994).

[2] Isabelle : http://www.cl.cam.ac.uk/Research/HVG/Isabelle

[3] Hideo Suzuki, Masakazu Funato : On the Implementation of a Proof Database, Journal of JSSAC, vol.10, No.2(2003).