Hilbert's Nullstellensatz

Alexander Maletzky*

May 26, 2024

Abstract

This entry formalizes Hilbert's Nullstellensatz, an important theorem in algebraic geometry that can be viewed as the generalization of the Fundamental Theorem of Algebra to multivariate polynomials: If a set of (multivariate) polynomials over an algebraically closed field has no common zero, then the ideal it generates is the entire polynomial ring. The formalization proves several equivalent versions of this celebrated theorem: the weak Nullstellensatz, the strong Nullstellensatz (connecting algebraic varieties and radical ideals), and the field-theoretic Nullstellensatz. The formalization follows Chapter 4.1. of *Ideals, Varieties, and Algorithms* by Cox, Little and O'Shea.

Contents

*Funded by the Austrian Science Fund (FWF): grant no. P 29498-N31

1 Algebraically Closed Fields

theory *Algebraically-Closed-Fields*
imports *HOL−Computational-Algebra.Fundamental-Theorem-Algebra*
begin

lemma *prod-eq-zeroE*:
assumes *prod f I* = (*0*::*'a*::{*semiring-no-zero-divisors*,*comm-monoid-mult*,*zero-neq-one*})
obtains *i* **where** *finite I* **and** *i* ∈ *I* **and** *f i* = *0*
⟨*proof*⟩

lemma *degree-prod-eq*:
assumes *finite I* **and** ⋀*i*. *i* ∈ *I* ⟹ *f i* ≠ *0*
shows *Polynomial.degree* (*prod f I* :: -::*semiring-no-zero-divisors poly*) = (∑ *i*∈*I*. *Polynomial.degree* (*f i*))
⟨*proof*⟩

class *alg-closed-field* =
assumes *alg-closed-field-axiom*: ⋀*p*::*'a*::*field poly*. *0* < *Polynomial.degree p* ⟹ ∃ *z*. *poly p z* = *0*
begin

lemma *rootE*:
assumes *0* < *Polynomial.degree p*
obtains *z* **where** *poly p z* = (*0*::*'a*)
⟨*proof*⟩

lemma *infinite-UNIV*: *infinite* (*UNIV*::*'a set*)
⟨*proof*⟩

lemma *linear-factorsE*:
fixes *p* :: *'a poly*
obtains *c A m* **where** *finite A* **and** *p* = *Polynomial.smult c* (∏ *a*∈*A*. [:− *a*, *1*:] ^ *m a*)
and ⋀*a*. *m a* = *0* ⟷ *a* ∉ *A* **and** *c* = *0* ⟷ *p* = *0* **and** ⋀*z*. *poly p z* = *0* ⟷ (*c* = *0* ∨ *z* ∈ *A*)
⟨*proof*⟩

end

instance *complex* :: *alg-closed-field*
⟨*proof*⟩

end

2 Properties of the Lexicographic Order on Power-Products

theory *Lex-Order-PP*
  imports *Polynomials.Power-Products*
begin

We prove some useful properties of the purely lexicographic order relation on power-products.

lemma *lex-pm-keys-leE*:
  assumes *lex-pm s t* **and** $x \in$ *keys* (s::$'x$::*linorder* $\Rightarrow_0$ $'a$::*add-linorder-min*)
  obtains y **where** $y \in$ *keys t* **and** $y \leq x$
  ⟨*proof*⟩

lemma *lex-pm-except-max*:
  assumes *lex-pm s t* **and** *keys s* ∪ *keys t* ⊆ {..x}
  shows *lex-pm* (*except s* {x}) (*except t* {x})
⟨*proof*⟩

lemma *lex-pm-strict-plus-left*:
  assumes *lex-pm-strict s t* **and** $\bigwedge x\ y.\ x \in$ *keys t* $\Longrightarrow$ $y \in$ *keys u* $\Longrightarrow$ $x < y$
  shows *lex-pm-strict* (u + s) (t::- $\Rightarrow_0$ $'a$::*add-linorder-min*)
⟨*proof*⟩

end

3 Polynomial Mappings and Univariate Polynomials

theory *Univariate-PM*
  imports *HOL−Computational-Algebra.Polynomial Polynomials.MPoly-PM*
begin

3.1 Morphisms *pm-of-poly* and *poly-of-pm*

Many things in this section are copied from theory *Polynomials.MPoly-Type-Univariate*.

lemma *pm-of-poly-aux*:
  {t. (*poly.coeff p* (*lookup t x*) *when t* ∈ .[{x}]) ≠ *0*} =
      Poly-Mapping.single x ‘ {d. *poly.coeff p d* ≠ *0*} (**is** *?M* = -)
⟨*proof*⟩

lift-definition *pm-of-poly* :: $'x \Rightarrow$ $'a$ *poly* $\Rightarrow$ ($'x \Rightarrow_0$ *nat*) $\Rightarrow_0$ $'a$::*comm-monoid-add*
  is $\lambda x\ p\ t.$ (*poly.coeff p* (*lookup t x*)) *when t* ∈ .[{x}]
⟨*proof*⟩

definition *poly-of-pm* :: $'x \Rightarrow$ (($'x \Rightarrow_0$ *nat*) $\Rightarrow_0$ $'a$) $\Rightarrow$ $'a$::*comm-monoid-add poly*
  where *poly-of-pm x p* = *Abs-poly* ($\lambda d.$ *lookup p* (*Poly-Mapping.single x d*))

lemma *lookup-pm-of-poly-single* [*simp*]:
lookup (*pm-of-poly x p*) (*Poly-Mapping.single x d*) = *poly.coeff p d*
⟨*proof*⟩

lemma *keys-pm-of-poly*: *keys* (*pm-of-poly x p*) = *Poly-Mapping.single x* ‘ {*d*. *poly.coeff p d* ≠ *0*}
⟨*proof*⟩

lemma *coeff-poly-of-pm* [*simp*]: *poly.coeff* (*poly-of-pm x p*) *k* = *lookup p* (*Poly-Mapping.single x k*)
⟨*proof*⟩

lemma *pm-of-poly-of-pm*:
assumes *p* ∈ *P*[{*x*}]
shows *pm-of-poly x* (*poly-of-pm x p*) = *p*
⟨*proof*⟩

lemma *poly-of-pm-of-poly* [*simp*]: *poly-of-pm x* (*pm-of-poly x p*) = *p*
⟨*proof*⟩

lemma *pm-of-poly-in-Polys*: *pm-of-poly x p* ∈ *P*[{*x*}]
⟨*proof*⟩

lemma *pm-of-poly-zero* [*simp*]: *pm-of-poly x 0* = *0*
⟨*proof*⟩

lemma *pm-of-poly-eq-zero-iff* [*iff*]: *pm-of-poly x p* = *0* ⟷ *p* = *0*
⟨*proof*⟩

lemma *pm-of-poly-monom*: *pm-of-poly x* (*Polynomial.monom c d*) = *monomial c* (*Poly-Mapping.single x d*)
⟨*proof*⟩

lemma *pm-of-poly-plus*: *pm-of-poly x* (*p* + *q*) = *pm-of-poly x p* + *pm-of-poly x q*
⟨*proof*⟩

lemma *pm-of-poly-uminus* [*simp*]: *pm-of-poly x* (− *p*) = − *pm-of-poly x p*
⟨*proof*⟩

lemma *pm-of-poly-minus*: *pm-of-poly x* (*p* − *q*) = *pm-of-poly x p* − *pm-of-poly x q*
⟨*proof*⟩

lemma *pm-of-poly-one* [*simp*]: *pm-of-poly x 1* = *1*
⟨*proof*⟩

lemma *pm-of-poly-pCons*:
pm-of-poly x (*pCons c p*) =

monomial c 0 + punit.monom-mult (1::-::monoid-mult) (Poly-Mapping.single x 1) (pm-of-poly x p)
(**is** *?l = ?r*)
⟨*proof*⟩

lemma *pm-of-poly-smult* [*simp*]: *pm-of-poly x (Polynomial.smult c p) = c · pm-of-poly x p*
⟨*proof*⟩

lemma *pm-of-poly-times*: *pm-of-poly x (p * q) = pm-of-poly x p * pm-of-poly x (q::-::ring-1 poly)*
⟨*proof*⟩

lemma *pm-of-poly-sum*: *pm-of-poly x (sum f I) = (∑ i∈I. pm-of-poly x (f i))*
⟨*proof*⟩

lemma *pm-of-poly-prod*: *pm-of-poly x (prod f I) = (∏ i∈I. pm-of-poly x (f i :: -::ring-1 poly))*
⟨*proof*⟩

lemma *pm-of-poly-power* [*simp*]: *pm-of-poly x (p ^ m) = pm-of-poly x (p::-::ring-1 poly) ^ m*
⟨*proof*⟩

lemma *poly-of-pm-zero* [*simp*]: *poly-of-pm x 0 = 0*
⟨*proof*⟩

lemma *poly-of-pm-eq-zero-iff*: *poly-of-pm x p = 0 ⟷ keys p ∩ .[{x}] = {}*
⟨*proof*⟩

lemma *poly-of-pm-monomial*:
poly-of-pm x (monomial c t) = (Polynomial.monom c (lookup t x) when t ∈ .[{x}])
⟨*proof*⟩

lemma *poly-of-pm-plus*: *poly-of-pm x (p + q) = poly-of-pm x p + poly-of-pm x q*
⟨*proof*⟩

lemma *poly-of-pm-uminus* [*simp*]: *poly-of-pm x (- p) = - poly-of-pm x p*
⟨*proof*⟩

lemma *poly-of-pm-minus*: *poly-of-pm x (p - q) = poly-of-pm x p - poly-of-pm x q*
⟨*proof*⟩

lemma *poly-of-pm-one* [*simp*]: *poly-of-pm x 1 = 1*
⟨*proof*⟩

lemma *poly-of-pm-times*:

*poly-of-pm x (p * q) = poly-of-pm x p * poly-of-pm x (q::- ⇒₀ 'a::comm-semiring-1)*
⟨*proof*⟩

lemma *poly-of-pm-sum*: *poly-of-pm x (sum f I) = (∑ i∈I. poly-of-pm x (f i))*
⟨*proof*⟩

lemma *poly-of-pm-prod*: *poly-of-pm x (prod f I) = (∏ i∈I. poly-of-pm x (f i))*
⟨*proof*⟩

lemma *poly-of-pm-power* [*simp*]: *poly-of-pm x (p ^ m) = poly-of-pm x p ^ m*
⟨*proof*⟩

3.2 Evaluating Polynomials

lemma *poly-eq-poly-eval*: *poly (poly-of-pm x p) a = poly-eval (λy. a when y = x) p*
⟨*proof*⟩

corollary *poly-eq-poly-eval'*:
assumes *p ∈ P[{x}]*
shows *poly (poly-of-pm x p) a = poly-eval (λ-. a) p*
⟨*proof*⟩

lemma *poly-eval-eq-poly*: *poly-eval a (pm-of-poly x p) = poly p (a x)*
⟨*proof*⟩

3.3 Morphisms *flat-pm-of-poly* and *poly-of-focus*

definition *flat-pm-of-poly* :: *'x ⇒ (('x ⇒₀ nat) ⇒₀ 'a) poly ⇒ (('x ⇒₀ nat) ⇒₀ 'a::semiring-1)*
where *flat-pm-of-poly x = flatten ∘ pm-of-poly x*

definition *poly-of-focus* :: *'x ⇒ (('x ⇒₀ nat) ⇒₀ 'a) ⇒ (('x ⇒₀ nat) ⇒₀ 'a::comm-monoid-add) poly*
where *poly-of-focus x = poly-of-pm x ∘ focus {x}*

lemma *flat-pm-of-poly-in-Polys*:
assumes *range (poly.coeff p) ⊆ P[Y]*
shows *flat-pm-of-poly x p ∈ P[insert x Y]*
⟨*proof*⟩

corollary *indets-flat-pm-of-poly-subset*:
indets (flat-pm-of-poly x p) ⊆ insert x (⋃ (indets ‘ range (poly.coeff p)))
⟨*proof*⟩

lemma
shows *flat-pm-of-poly-zero* [*simp*]: *flat-pm-of-poly x 0 = 0*
and *flat-pm-of-poly-monom*: *flat-pm-of-poly x (Polynomial.monom c d) =*
punit.monom-mult 1 (Poly-Mapping.single x d) c
and *flat-pm-of-poly-plus*: *flat-pm-of-poly x (p + q) =*

flat-pm-of-poly x p + flat-pm-of-poly x q
and *flat-pm-of-poly-one* [*simp*]: *flat-pm-of-poly x 1 = 1*
and *flat-pm-of-poly-sum*: *flat-pm-of-poly x* (*sum f I*) = ($\sum$ *i*∈*I*. *flat-pm-of-poly x* (*f i*))
⟨*proof*⟩

lemma
shows *flat-pm-of-poly-uminus* [*simp*]: *flat-pm-of-poly x* (− *p*) = − *flat-pm-of-poly x p*
and *flat-pm-of-poly-minus*: *flat-pm-of-poly x* (*p* − *q*) =
flat-pm-of-poly x p − *flat-pm-of-poly x* (*q*::-::*ring poly*)
⟨*proof*⟩

lemma *flat-pm-of-poly-pCons*:
flat-pm-of-poly x (*pCons c p*) =
c + *punit.monom-mult 1* (*Poly-Mapping.single x 1*) (*flat-pm-of-poly x* (*p*::-::*comm-semiring-1 poly*))
⟨*proof*⟩

lemma *flat-pm-of-poly-smult* [*simp*]:
flat-pm-of-poly x (*Polynomial.smult c p*) = *c* ∗ *flat-pm-of-poly x* (*p*::-::*comm-semiring-1 poly*)
⟨*proof*⟩

lemma
shows *flat-pm-of-poly-times*: *flat-pm-of-poly x* (*p* ∗ *q*) = *flat-pm-of-poly x p* ∗ *flat-pm-of-poly x q*
and *flat-pm-of-poly-prod*: *flat-pm-of-poly x* (*prod f I*) =
($\prod$ *i*∈*I*. *flat-pm-of-poly x* (*f i* :: -::*comm-ring-1 poly*))
and *flat-pm-of-poly-power*: *flat-pm-of-poly x* (*p* ^ *m*) = *flat-pm-of-poly x* (*p*::-::*comm-ring-1 poly*) ^ *m*
⟨*proof*⟩

lemma *coeff-poly-of-focus-subset-Polys*:
assumes *p* ∈ *P*[*X*]
shows *range* (*poly.coeff* (*poly-of-focus x p*)) ⊆ *P*[*X* − {*x*}]
⟨*proof*⟩

lemma
shows *poly-of-focus-zero* [*simp*]: *poly-of-focus x 0 = 0*
and *poly-of-focus-uminus* [*simp*]: *poly-of-focus x* (− *p*) = − *poly-of-focus x p*
and *poly-of-focus-plus*: *poly-of-focus x* (*p* + *q*) = *poly-of-focus x p* + *poly-of-focus x q*
and *poly-of-focus-minus*: *poly-of-focus x* (*p* − *q*) = *poly-of-focus x p* − *poly-of-focus x q*
and *poly-of-focus-one* [*simp*]: *poly-of-focus x 1 = 1*
and *poly-of-focus-sum*: *poly-of-focus x* (*sum f I*) = ($\sum$ *i*∈*I*. *poly-of-focus x* (*f i*))
⟨*proof*⟩

lemma *poly-of-focus-eq-zero-iff* [*iff*]: *poly-of-focus x p = 0* $\longleftrightarrow$ *p = 0*
⟨*proof*⟩

lemma *poly-of-focus-monomial*:
poly-of-focus x (*monomial c t*) = *Polynomial.monom* (*monomial c* (*except t* {*x*})) (*lookup t x*)
⟨*proof*⟩

lemma
shows *poly-of-focus-times*: *poly-of-focus x* (*p* ∗ *q*) = *poly-of-focus x p* ∗ *poly-of-focus x q*
and *poly-of-focus-prod*: *poly-of-focus x* (*prod f I*) =
($\prod$ *i*∈*I*. *poly-of-focus x* (*f i* :: - ⇒$_0$ -::*comm-semiring-1*))
and *poly-of-focus-power*: *poly-of-focus x* (*p* ^ *m*) = *poly-of-focus x* (*p*::- ⇒$_0$ -::*comm-semiring-1*) ^ *m*
⟨*proof*⟩

lemma *flat-pm-of-poly-of-focus* [*simp*]: *flat-pm-of-poly x* (*poly-of-focus x p*) = *p*
⟨*proof*⟩

lemma *poly-of-focus-flat-pm-of-poly*:
assumes *range* (*poly.coeff p*) ⊆ *P*[− {*x*}]
shows *poly-of-focus x* (*flat-pm-of-poly x p*) = *p*
⟨*proof*⟩

lemma *flat-pm-of-poly-eq-zeroD*:
assumes *flat-pm-of-poly x p = 0* **and** *range* (*poly.coeff p*) ⊆ *P*[− {*x*}]
shows *p = 0*
⟨*proof*⟩

lemma *poly-poly-of-focus*: *poly* (*poly-of-focus x p*) *a* = *poly-eval* (λ-. *a*) (*focus* {*x*} *p*)
⟨*proof*⟩

corollary *poly-poly-of-focus-monomial*:
poly (*poly-of-focus x p*) (*monomial 1* (*Poly-Mapping.single x 1*)) = (*p*::- ⇒$_0$ -::*comm-semiring-1*)
⟨*proof*⟩

end

4 Hilbert's Nullstellensatz

theory *Nullstellensatz*
imports *Algebraically-Closed-Fields*
HOL−Computational-Algebra.Fraction-Field
Lex-Order-PP
Univariate-PM

Groebner-Bases.Groebner-PM
begin

We prove the geometric version of Hilbert's Nullstellensatz, i.e. the precise correspondence between algebraic varieties and radical ideals. The field-theoretic version of the Nullstellensatz is proved in theory *Nullstellensatz-Field*.

4.1 Preliminaries

lemma *finite-linorder-induct* [*consumes 1*, *case-names empty insert*]:
assumes *finite* (*A*::'*a*::*linorder set*) **and** *P* {}
and ⋀*a A*. *finite A* ⟹ *A* ⊆ {..<*a*} ⟹ *P A* ⟹ *P* (*insert a A*)
shows *P A*
⟨*proof*⟩

lemma *Fract-same*: *Fract a a* = (*1 when a* ≠ *0*)
⟨*proof*⟩

lemma *Fract-eq-zero-iff*: *Fract a b* = *0* ⟷ *a* = *0* ∨ *b* = *0*
⟨*proof*⟩

lemma *poly-plus-rightE*:
obtains *c* **where** *poly p* (*x* + *y*) = *poly p x* + *c* ∗ *y*
⟨*proof*⟩

lemma *poly-minus-rightE*:
obtains *c* **where** *poly p* (*x* − *y*) = *poly p x* − *c* ∗ (*y*::-::*comm-ring*)
⟨*proof*⟩

lemma *map-poly-plus*:
assumes *f* *0* = *0* **and** ⋀*a b*. *f* (*a* + *b*) = *f a* + *f b*
shows *map-poly f* (*p* + *q*) = *map-poly f p* + *map-poly f q*
⟨*proof*⟩

lemma *map-poly-minus*:
assumes *f* *0* = *0* **and** ⋀*a b*. *f* (*a* − *b*) = *f a* − *f b*
shows *map-poly f* (*p* − *q*) = *map-poly f p* − *map-poly f q*
⟨*proof*⟩

lemma *map-poly-sum*:
assumes *f* *0* = *0* **and** ⋀*a b*. *f* (*a* + *b*) = *f a* + *f b*
shows *map-poly f* (*sum g A*) = (∑ *a*∈*A*. *map-poly f* (*g a*))
⟨*proof*⟩

lemma *map-poly-times*:
assumes *f* *0* = *0* **and** ⋀*a b*. *f* (*a* + *b*) = *f a* + *f b* **and** ⋀*a b*. *f* (*a* ∗ *b*) = *f a* ∗ *f b*
shows *map-poly f* (*p* ∗ *q*) = *map-poly f p* ∗ *map-poly f q*

⟨*proof*⟩

lemma *poly-Fract*:
assumes *set* (*Polynomial.coeffs p*) ⊆ *range* (λ*x*. *Fract x 1*)
obtains *q m* **where** *poly p* (*Fract a b*) = *Fract q* (*b* ^ *m*)
⟨*proof*⟩

lemma (**in** *ordered-term*) *lt-sum-le-Max*: *lt* (*sum f A*) ≼$_t$ *ord-term-lin.Max* {*lt* (*f a*) | *a*. *a* ∈ *A*}
⟨*proof*⟩

4.2 Ideals and Varieties

definition *variety-of* :: (('*x* ⇒$_0$ *nat*) ⇒$_0$ '*a*) *set* ⇒ ('*x* ⇒ '*a*::*comm-semiring-1*) *set*
where *variety-of F* = {*a*. ∀*f*∈*F*. *poly-eval a f* = *0*}

definition *ideal-of* :: ('*x* ⇒ '*a*::*comm-semiring-1*) *set* ⇒ (('*x* ⇒$_0$ *nat*) ⇒$_0$ '*a*) *set*
where *ideal-of A* = {*f*. ∀ *a*∈*A*. *poly-eval a f* = *0*}

abbreviation $\mathcal{V}$ ≡ *variety-of*
abbreviation $\mathcal{I}$ ≡ *ideal-of*

lemma *variety-ofI*: (⋀*f*. *f* ∈ *F* ⟹ *poly-eval a f* = *0*) ⟹ *a* ∈ $\mathcal{V}$ *F*
⟨*proof*⟩

lemma *variety-ofI-alt*: *poly-eval a* ' *F* ⊆ {*0*} ⟹ *a* ∈ $\mathcal{V}$ *F*
⟨*proof*⟩

lemma *variety-ofD*: *a* ∈ $\mathcal{V}$ *F* ⟹ *f* ∈ *F* ⟹ *poly-eval a f* = *0*
⟨*proof*⟩

lemma *variety-of-empty* [*simp*]: $\mathcal{V}$ {} = *UNIV*
⟨*proof*⟩

lemma *variety-of-UNIV* [*simp*]: $\mathcal{V}$ *UNIV* = {}
⟨*proof*⟩

lemma *variety-of-antimono*: *F* ⊆ *G* ⟹ $\mathcal{V}$ *G* ⊆ $\mathcal{V}$ *F*
⟨*proof*⟩

lemma *variety-of-ideal* [*simp*]: $\mathcal{V}$ (*ideal F*) = $\mathcal{V}$ *F*
⟨*proof*⟩

lemma *ideal-ofI*: (⋀*a*. *a* ∈ *A* ⟹ *poly-eval a f* = *0*) ⟹ *f* ∈ $\mathcal{I}$ *A*
⟨*proof*⟩

lemma *ideal-ofD*: *f* ∈ $\mathcal{I}$ *A* ⟹ *a* ∈ *A* ⟹ *poly-eval a f* = *0*
⟨*proof*⟩

lemma *ideal-of-empty* [*simp*]: $\mathcal{I}\ \{\} = \mathit{UNIV}$
⟨*proof*⟩

lemma *ideal-of-antimono*: $A \subseteq B \Longrightarrow \mathcal{I}\ B \subseteq \mathcal{I}\ A$
⟨*proof*⟩

lemma *ideal-ideal-of* [*simp*]: *ideal* $(\mathcal{I}\ A) = \mathcal{I}\ A$
⟨*proof*⟩

lemma *ideal-of-UN*: $\mathcal{I}\ (\bigcup\ (A\ `\ J)) = (\bigcap j{\in}J.\ \mathcal{I}\ (A\ j))$
⟨*proof*⟩

corollary *ideal-of-Un*: $\mathcal{I}\ (A \cup B) = \mathcal{I}\ A \cap \mathcal{I}\ B$
⟨*proof*⟩

lemma *variety-of-ideal-of-variety* [*simp*]: $\mathcal{V}\ (\mathcal{I}\ (\mathcal{V}\ F)) = \mathcal{V}\ F$ (**is** - = *?V*)
⟨*proof*⟩

lemma *ideal-of-inj-on*: *inj-on* $\mathcal{I}$ (*range* ($\mathcal{V}$::(('*x* ⇒$_0$ *nat*) ⇒$_0$ '*a*::*comm-semiring-1*) *set* ⇒ -))
⟨*proof*⟩

lemma *ideal-of-variety-of-ideal* [*simp*]: $\mathcal{I}\ (\mathcal{V}\ (\mathcal{I}\ A)) = \mathcal{I}\ A$ (**is** - = *?I*)
⟨*proof*⟩

lemma *variety-of-inj-on*: *inj-on* $\mathcal{V}$ (*range* ($\mathcal{I}$::('*x* ⇒ '*a*::*comm-semiring-1*) *set* ⇒ -))
⟨*proof*⟩

lemma *image-map-indets-ideal-of*:
assumes *inj f*
shows *map-indets f* ` $\mathcal{I}\ A = \mathcal{I}\ ((\lambda a.\ a \circ f)\ -`\ (A{::}('x \Rightarrow 'a{::}\mathit{comm\text{-}semiring\text{-}1})\ \mathit{set})) \cap P[\mathit{range}\ f]$
⟨*proof*⟩

lemma *variety-of-map-indets*: $\mathcal{V}$ (*map-indets f* ` *F*) = $(\lambda a.\ a \circ f)\ -`\ \mathcal{V}\ F$
⟨*proof*⟩

4.3 Radical Ideals

definition *radical* :: '*a*::*monoid-mult set* ⇒ '*a set* ($\sqrt{(-)}$ [*999*] *999*)
where *radical F* = $\{f.\ \exists\, m.\ f \mathbin{\hat{}} m \in F\}$

lemma *radicalI*: $f \mathbin{\hat{}} m \in F \Longrightarrow f \in \sqrt{F}$
⟨*proof*⟩

lemma *radicalE*:
assumes $f \in \sqrt{F}$

obtains m **where** $f \mathbin{\hat{}} m \in F$
⟨*proof*⟩

lemma *radical-empty* [*simp*]: $\sqrt{\{\}} = \{\}$
⟨*proof*⟩

lemma *radical-UNIV* [*simp*]: $\sqrt{UNIV} = UNIV$
⟨*proof*⟩

lemma *radical-ideal-eq-UNIV-iff*: $\sqrt{ideal\ F} = UNIV \longleftrightarrow ideal\ F = UNIV$
⟨*proof*⟩

lemma *zero-in-radical-ideal* [*simp*]: $0 \in \sqrt{ideal\ F}$
⟨*proof*⟩

lemma *radical-mono*: $F \subseteq G \Longrightarrow \sqrt{F} \subseteq \sqrt{G}$
⟨*proof*⟩

lemma *radical-superset*: $F \subseteq \sqrt{F}$
⟨*proof*⟩

lemma *radical-idem* [*simp*]: $\sqrt{\sqrt{F}} = \sqrt{F}$
⟨*proof*⟩

lemma *radical-Int-subset*: $\sqrt{(A \cap B)} \subseteq \sqrt{A} \cap \sqrt{B}$
⟨*proof*⟩

lemma *radical-ideal-Int*: $\sqrt{(ideal\ F \cap ideal\ G)} = \sqrt{ideal\ F} \cap \sqrt{ideal\ G}$
⟨*proof*⟩

lemma *ideal-radical-ideal* [*simp*]: $ideal\ (\sqrt{ideal\ F}) = \sqrt{ideal\ F}$ (**is** - = *?R*)
⟨*proof*⟩

lemma *radical-ideal-of* [*simp*]: $\sqrt{\mathcal{I}\ A} = \mathcal{I}\ (A::(- \Rightarrow -::semiring\text{-}1\text{-}no\text{-}zero\text{-}divisors)\ set)$
⟨*proof*⟩

lemma *variety-of-radical-ideal* [*simp*]: $\mathcal{V}\ (\sqrt{ideal\ F}) = \mathcal{V}\ (F::(- \Rightarrow_0 -::semiring\text{-}1\text{-}no\text{-}zero\text{-}divisors)\ set)$
⟨*proof*⟩

lemma *image-map-indets-radical*:
assumes *inj f*
shows *map-indets f ‘* $\sqrt{F} = \sqrt{(map\text{-}indets\ f\ `\ (F::(- \Rightarrow_0 'a::comm\text{-}ring\text{-}1)\ set))} \cap P[range\ f]$
⟨*proof*⟩

4.4 Geometric Version of the Nullstellensatz

lemma *weak-Nullstellensatz-aux-1*:
assumes $\bigwedge i.\ i \in I \Longrightarrow g\ i \in ideal\ B$
obtains c **where** $c \in ideal\ B$ **and** $(\prod i{\in}I.\ (f\ i + g\ i)\ \hat{}\ m\ i) = (\prod i{\in}I.\ f\ i\ \hat{}\ m\ i) + c$
⟨*proof*⟩

lemma *weak-Nullstellensatz-aux-2*:
assumes *finite X* **and** $F \subseteq P[insert\ x\ X]$ **and** $X \subseteq \{..<x::'x::\{countable,linorder\}\}$
and $1 \notin ideal\ F$ **and** $ideal\ F \cap P[\{x\}] \subseteq \{0\}$
obtains $a::'a::alg\text{-}closed\text{-}field$ **where** $1 \notin ideal\ (poly\text{-}eval\ (\lambda\text{-}.\ monomial\ a\ 0)\ `\ focus\ \{x\}\ `\ F)$
⟨*proof*⟩

lemma *weak-Nullstellensatz-aux-3*:
assumes $F \subseteq P[insert\ x\ X]$ **and** $x \notin X$ **and** $1 \notin ideal\ F$ **and** $\neg\ ideal\ F \cap P[\{x\}] \subseteq \{0\}$
obtains $a::'a::alg\text{-}closed\text{-}field$ **where** $1 \notin ideal\ (poly\text{-}eval\ (\lambda\text{-}.\ monomial\ a\ 0)\ `\ focus\ \{x\}\ `\ F)$
⟨*proof*⟩

theorem *weak-Nullstellensatz*:
assumes *finite X* **and** $F \subseteq P[X]$ **and** $\mathcal{V}\ F = (\{\}::('x::\{countable,linorder\} \Rightarrow 'a::alg\text{-}closed\text{-}field)\ set)$
shows $ideal\ F = UNIV$
⟨*proof*⟩

lemma *radical-idealI*:
assumes *finite X* **and** $F \subseteq P[X]$ **and** $f \in P[X]$ **and** $x \notin X$
and $\mathcal{V}\ (insert\ (1 - punit.monom\text{-}mult\ 1\ (Poly\text{-}Mapping.single\ x\ 1)\ f)\ F) = \{\}$
shows $(f::('x::\{countable,linorder\} \Rightarrow_0 nat) \Rightarrow_0 'a::alg\text{-}closed\text{-}field) \in \sqrt{}ideal\ F$
⟨*proof*⟩

corollary *radical-idealI-extend-indets*:
assumes *finite X* **and** $F \subseteq P[X]$
and $\mathcal{V}\ (insert\ (1 - punit.monom\text{-}mult\ 1\ (Poly\text{-}Mapping.single\ None\ 1)\ (extend\text{-}indets\ f))\ (extend\text{-}indets\ `\ F)) = \{\}$
shows $(f::(-::\{countable,linorder\} \Rightarrow_0 nat) \Rightarrow_0 -::alg\text{-}closed\text{-}field) \in \sqrt{}ideal\ F$
⟨*proof*⟩

theorem *Nullstellensatz*:
assumes *finite X* **and** $F \subseteq P[X]$
and $(f::(-::\{countable,linorder\} \Rightarrow_0 nat) \Rightarrow_0 -::alg\text{-}closed\text{-}field) \in \mathcal{I}\ (\mathcal{V}\ F)$
shows $f \in \sqrt{}ideal\ F$
⟨*proof*⟩

theorem *strong-Nullstellensatz*:
assumes *finite X* **and** $F \subseteq P[X]$

shows $\mathcal{I}\ (\mathcal{V}\ F) = \sqrt{}ideal\ (F::((\text{-}::\{countable,linorder\} \Rightarrow_0 nat) \Rightarrow_0 \text{-}::alg\text{-}closed\text{-}field)\ set)$
⟨*proof*⟩

The following lemma can be used for actually *deciding* whether a polynomial is contained in the radical of an ideal or not.

lemma *radical-ideal-iff*:
assumes *finite X* **and** $F \subseteq P[X]$ **and** $f \in P[X]$ **and** $x \notin X$
shows $(f::(\text{-}::\{countable,linorder\} \Rightarrow_0 nat) \Rightarrow_0 \text{-}::alg\text{-}closed\text{-}field) \in \sqrt{}ideal\ F \longleftrightarrow$
$1 \in ideal\ (insert\ (1 - punit.monom\text{-}mult\ 1\ (Poly\text{-}Mapping.single\ x\ 1)\ f)\ F)$
⟨*proof*⟩

end

5 Field-Theoretic Version of Hilbert's Nullstellensatz

theory *Nullstellensatz-Field*
imports *Nullstellensatz HOL−Types-To-Sets.Types-To-Sets*
begin

Building upon the geometric version of Hilbert's Nullstellensatz in *Nullstellensatz.Nullstellensatz*, we prove its field-theoretic version here. To that end we employ the 'types to sets' methodology.

5.1 Getting Rid of Sort Constraints in Geometric Version

We can use the 'types to sets' approach to get rid of the *countable* and *linorder* sort constraints on the type of indeterminates in the geometric version of the Nullstellensatz. Once the 'types to sets' methodology is integrated as a standard component into the main library of Isabelle, the theorems in *Nullstellensatz.Nullstellensatz* could be replaced by their counterparts in this section.

lemmas *radical-idealI-internalized* = *radical-idealI*[*unoverload-type* $'x$]

lemma *radical-idealI*:
assumes *finite X* **and** $F \subseteq P[X]$ **and** $f \in P[X]$ **and** $x \notin X$
and $\mathcal{V}\ (insert\ (1 - punit.monom\text{-}mult\ 1\ (Poly\text{-}Mapping.single\ x\ 1)\ f)\ F) = \{\}$
shows $(f::('x \Rightarrow_0 nat) \Rightarrow_0 'a::alg\text{-}closed\text{-}field) \in \sqrt{}ideal\ F$
⟨*proof*⟩

corollary *radical-idealI-extend-indets*:
assumes *finite X* **and** $F \subseteq P[X]$
and $\mathcal{V}\ (insert\ (1 - punit.monom\text{-}mult\ 1\ (Poly\text{-}Mapping.single\ None\ 1)\ (extend\text{-}indets\ f))$

(*extend-indets* ' *F*)) = {}
shows (*f*::- ⇒$_0$ -::*alg-closed-field*) ∈ √*ideal F*
⟨*proof*⟩

theorem *Nullstellensatz*:
assumes *finite X* **and** *F* ⊆ *P*[*X*]
and (*f*::- ⇒$_0$ -::*alg-closed-field*) ∈ *I* (*V F*)
shows *f* ∈ √*ideal F*
⟨*proof*⟩

theorem *strong-Nullstellensatz*:
assumes *finite X* **and** *F* ⊆ *P*[*X*]
shows *I* (*V F*) = √*ideal* (*F*::(- ⇒$_0$ -::*alg-closed-field*) *set*)
⟨*proof*⟩

theorem *weak-Nullstellensatz*:
assumes *finite X* **and** *F* ⊆ *P*[*X*] **and** *V F* = ({}::(- ⇒ -::*alg-closed-field*) *set*)
shows *ideal F* = *UNIV*
⟨*proof*⟩

lemma *radical-ideal-iff*:
assumes *finite X* **and** *F* ⊆ *P*[*X*] **and** *f* ∈ *P*[*X*] **and** *x* ∉ *X*
shows (*f*::- ⇒$_0$ -::*alg-closed-field*) ∈ √*ideal F* ⟷
1 ∈ *ideal* (*insert* (1 − *punit.monom-mult* 1 (*Poly-Mapping.single x* 1)
f) *F*)
⟨*proof*⟩

5.2 Field-Theoretic Version of the Nullstellensatz

Due to the possibility of infinite indeterminate-types, we have to explicitly add the set of indeterminates under consideration to the definition of maximal ideals.

definition *generates-max-ideal* :: '*x set* ⇒ (('*x* ⇒$_0$ *nat*) ⇒$_0$ '*a*::*comm-ring-1*) *set* ⇒ *bool*
where *generates-max-ideal X F* ⟷ (*ideal F* ≠ *UNIV* ∧
(∀ *F'*. *F'* ⊆ *P*[*X*] ⟶ *ideal F* ⊂ *ideal F'* ⟶ *ideal F'* = *UNIV*))

lemma *generates-max-idealI*:
assumes *ideal F* ≠ *UNIV* **and** ⋀*F'*. *F'* ⊆ *P*[*X*] ⟹ *ideal F* ⊂ *ideal F'* ⟹ *ideal F'* = *UNIV*
shows *generates-max-ideal X F*
⟨*proof*⟩

lemma *generates-max-idealI-alt*:
assumes *ideal F* ≠ *UNIV* **and** ⋀*p*. *p* ∈ *P*[*X*] ⟹ *p* ∉ *ideal F* ⟹ 1 ∈ *ideal* (*insert p F*)
shows *generates-max-ideal X F*
⟨*proof*⟩

lemma *generates-max-idealD*:
assumes *generates-max-ideal X F*
shows *ideal F* $\neq$ *UNIV* **and** $F' \subseteq P[X] \Longrightarrow$ *ideal F* $\subset$ *ideal F'* $\Longrightarrow$ *ideal F'* = *UNIV*
⟨*proof*⟩

lemma *generates-max-ideal-cases*:
assumes *generates-max-ideal X F* **and** $F' \subseteq P[X]$ **and** *ideal F* $\subseteq$ *ideal F'*
obtains *ideal F* = *ideal F'* | *ideal F'* = *UNIV*
⟨*proof*⟩

lemma *max-ideal-UNIV-radical*:
assumes *generates-max-ideal UNIV F*
shows $\sqrt{}$*ideal F* = *ideal F*
⟨*proof*⟩

lemma *max-ideal-shape-aux*:
(λ*x*. *monomial 1* (*Poly-Mapping.single x 1*) − *monomial* (*a x*) *0*) ‘ *X* ⊆ *P*[*X*]
⟨*proof*⟩

lemma *max-ideal-shapeI*:
generates-max-ideal X ((λ*x*. *monomial* (*1*::*'a*::*field*) (*Poly-Mapping.single x 1*) − *monomial* (*a x*) *0*) ‘ *X*)
(**is** *generates-max-ideal X ?F*)
⟨*proof*⟩

We first prove the following lemma assuming that the type of indeterminates is finite, and then transfer the result to arbitrary types of indeterminates by using the ‘types to sets’ methodology. This approach facilitates the proof considerably.

lemma *max-ideal-shapeD-finite*:
assumes *generates-max-ideal UNIV* (*F*::((*'x*::*finite* ⇒$_0$ *nat*) ⇒$_0$ *'a*::*alg-closed-field*) *set*)
obtains *a* **where** *ideal F* = *ideal* (*range* (λ*x*. *monomial 1* (*Poly-Mapping.single x 1*) − *monomial* (*a x*) *0*))
⟨*proof*⟩

lemmas *max-ideal-shapeD-internalized* = *max-ideal-shapeD-finite*[*unoverload-type 'x*]

lemma *max-ideal-shapeD*:
assumes *finite X* **and** *F* ⊆ *P*[*X*]
and *generates-max-ideal X* (*F*::((*'x* ⇒$_0$ *nat*) ⇒$_0$ *'a*::*alg-closed-field*) *set*)
obtains *a* **where** *ideal F* = *ideal* ((λ*x*. *monomial 1* (*Poly-Mapping.single x 1*) − *monomial* (*a x*) *0*) ‘ *X*)
⟨*proof*⟩

theorem *Nullstellensatz-field*:

assumes *finite* X **and** $F \subseteq P[X]$ **and** *generates-max-ideal* X (F::(- $\Rightarrow_0$ -::*alg-closed-field*) *set*)
and $x \in X$
shows $\{0\} \subset$ *ideal* $F \cap P[\{x\}]$
⟨*proof*⟩

end

References

[1] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms.* Undergraduate Texts in Mathematics. Springer, 2007.