Polynomial rings with operators in Coq

William Simmons
(Hobart and William Smith Colleges)

Formal Methods in Mathematics/Lean Together 2020
Carnegie Mellon University
Jan. 10, 2020

Rings with additional operators

- Given a ring R, consider a collection $Op \subseteq R \to R$ of additive homomorphisms on R.
- If the elements of Op satisfy the usual product rule from calculus (for all $D \in Op$ and $a, b \in R$ we have $D(a \cdot b) = D(a) \cdot b + a \cdot D(b)$), we say (R, Op) is a *differential ring* equipped with *derivations* $D \in Op$.

Rings with additional operators

- Given a ring R, consider a collection $Op \subseteq R \to R$ of additive homomorphisms on R.
- If the elements of Op satisfy the usual product rule from calculus (for all $D \in Op$ and $a, b \in R$ we have $D(a \cdot b) = D(a) \cdot b + a \cdot D(b)$), we say (R, Op) is a *differential ring* equipped with *derivations* $D \in Op$.
- If the elements of Op are ring homomorphisms from R to itself, we call (R, Op) a *difference ring* with *morphisms* $D \in Op$.
- These are only two of many possibilities. We can study differential algebra, difference algebra, etc., being sure to respect the distinguished operations (e.g., if (R, D_R) and (S, D_S) are differential rings and $\varphi : R \to S$ is a ring homomorphism, then φ is a differential ring homomorphism if $\varphi(D_R(a)) = D_S(\varphi(a))$).

Polynomials over rings with additional operators

Given a ring $(R, \{D\})$ and an algebraic indeterminate X, we can consider the polynomial ring $R\{X\} := R[X, D(X), D^2(X), \ldots]$. Often we can extend D to $\overline{D}$ on $R\{X\}$ so that $(R\{X\}, \{\overline{D}\})$ is itself a ring of the same kind as $(R, \{D\})$ (e.g., differential/difference/...).

Polynomials over rings with additional operators

Given a ring $(R, \{D\})$ and an algebraic indeterminate X, we can consider the polynomial ring $R\{X\} := R[X, D(X), D^2(X), \ldots]$. Often we can extend D to $\overline{D}$ on $R\{X\}$ so that $(R\{X\}, \{\overline{D}\})$ is itself a ring of the same kind as $(R, \{D\})$ (e.g., differential/difference/...).

The same holds for more than one variable and for collections of multiple operators.

With a notion of such polynomials, we can consider ideals closed under the operators, define varieties, schemes, etc., and generally try to imitate the algebra and geometry of ordinary rings and fields.

What are they good for?

Ex: Elimination can facilitate numerical solutions:

$$\dot{x} = .7y + \sin(2.5z)$$
$$\dot{y} = 1.4x + \cos(2.5z)$$
$$1 = x^2 + y^2$$

are not all differential polynomial equations and there is no equation $\dot{z} = \ldots$, which is needed to run numerical algorithms like Euler's method.

What are they good for?

Ex: Elimination can facilitate numerical solutions:

$$\dot{x} = .7y + \sin(2.5z)$$
$$\dot{y} = 1.4x + \cos(2.5z)$$
$$1 = x^2 + y^2$$

are not all differential polynomial equations and there is no equation $\dot{z} = \ldots$, which is needed to run numerical algorithms like Euler's method. However, by reformulating as the equivalent

$$\dot{x} = .7y + s \qquad \dot{s} = 2.5\dot{z}c$$
$$\dot{y} = 1.4x + c \qquad \dot{c} = -2.5\dot{z}s$$
$$1 = x^2 + y^2 \qquad 1 = s^2 + c^2,$$

and performing differential elimination, we obtain a suitable system. (*Boulier, Differential Elimination and Biological Modelling, '07, pp. 9-11.*)

What are they good for?

Ex: Used to define objects like *differentially closed fields* that show up, e.g., in Hrushovski's model-theoretic proof of the geometric Mordell-Lang conjecture.

What are they good for?

Ex: Used to define objects like *differentially closed fields* that show up, e.g., in Hrushovski's model-theoretic proof of the geometric Mordell-Lang conjecture.

Ex: They're fun! Many similarities to the ordinary case while harboring many open questions (e.g., complexity and undecidability)

Formalizing polynomials with operators

- Not currently represented in proof assistant libraries (to my knowledge)
- You can't simply name the variables (infinitely many)
- How to put differential, difference, and other polynomials in the same framework?
- There are many implementations of multivariate polynomials (e.g., Bernard, et al., '16, '17; `multinomials` library by Strub, et al.; Sternagel, et al., '17; `mv_polynomial.lean` by Hölzl, et al.), but not clear–to me, anyway–how to build polynomials with operators on top of these

Desiderata and decisions

- Represent these objects in an elementary fashion with a minimum of prerequisites
- Achieve as much generality as possible
- Decided to work in Coq
- Take advantage of existing libraries but don't require users to work within a particular framework (e.g., ringType in Mathematical Components)
- Go for a "deep embedding" of the theory rather than modeling polynomials as lists of coefficients or functions with finite support.

The basic type

```
Inductive op_poly (X Y Z : Type) : Type :=
  |Zeroop
  |Oneop
  |Coeff (_: X)
  |Var (_:Y)
  |Op (D:Z)(p: op_poly X Y Z)
  |Add (_ _:op_poly X Y Z)
  |Subtract (_ _: op_poly X Y Z)
  |Mult(_ _:op_poly X Y Z).
```

Axioms via an inductive prop

```
Inductive op_poly_eq (X Y Z : Type): (op_poly X Y Z) ->
(op_poly X Y Z) -> Prop:=
  |Add_Assoc p1 p2 p3: op_poly_eq (Add (Add p1 p2) p3)
  (Add p1 (Add p2 p3))
  |Add_Subtract_Assoc p1 p2 p3 : op_poly_eq (Subtract
  (Add p1 p2) p3)(Add p1 (Subtract p2 p3))
  |Add_Comm p1 p2: op_poly_eq (Add p1 p2)(Add p2 p1)
  |Mult_Assoc p1 p2 p3: op_poly_eq (Mult (Mult p1 p2) p3)
  (Mult p1 (Mult p2 p3))
  |DistribL p1 p2 p3: op_poly_eq (Mult p1 (Add p2 p3))
  (Add (Mult p1 p2) (Mult p1 p3))
  |DistribR p1 p2 p3: op_poly_eq(Mult(Add p1 p2) p3)
  (Add(Mult p1 p3) (Mult p2 p3))
  |Distrib_SubtractL p1 p2 p3: op_poly_eq (Mult p1
  (Subtract p2 p3)) (Subtract (Mult p1 p2) (Mult p1 p3))
  |Distrib_SubtractR p1 p2 p3: op_poly_eq (Mult
  (Subtract p1 p2) p3)(Subtract(Mult p1 p3) (Mult p2 p3))
```

Axioms via an inductive prop, cont.

```
  |Op_Add_Hom (D:Z) (p1 p2: op_poly X Y Z):  op_poly_eq
  (Op D (Add p1 p2))(Add (Op D p1)(Op D p2))
  |Op_Subtract_Hom (D:Z) (p1 p2: op_poly X Y Z):  op_poly_eq
  (Op D (Subtract p1 p2))(Subtract (Op D p1)(Op D p2))
  |Id_ZeroopR p: op_poly_eq (Add p (@Zeroop X Y Z))(p)
  |Id_ZeroopL p: op_poly_eq (Add (@Zeroop X Y Z) p)(p)
  |Id_OneopR p: op_poly_eq (Mult p (@Oneop X Y Z))(p)
  |Id_OneopL p: op_poly_eq (Mult (@Oneop X Y Z) p)(p)
  |Id_Subtract_equals p q: op_poly_eq (Subtract
  (Add p q) q)(p)
(*to make op_poly_eq an equiv reln*)
  |Refl_oppolyeq p: op_poly_eq p p
  |...
(*to make op_poly_eq respect application of operators, etc.*)
  |Id_Add p1 p2 p3 p4 (H1: op_poly_eq p1 p2)
  (H2: op_poly_eq p3 p4): op_poly_eq(Add p1 p3)(Add p2 p4)
  |...
```

Using setoids to enable rewriting mod the axioms

```
(**op_poly_eq_rel: op_poly_eq is an equiv reln*)
Add Parametric Relation (X Y Z : Type):
(op_poly X Y Z)(@op_poly_eq X Y Z)
  reflexivity proved by (@Refl_oppolyeq X Y Z)
  symmetry proved by (@Symm_oppolyeq X Y Z)
  transitivity proved by (@Trans_oppolyeq X Y Z)
  as op_poly_eq_rel.

(**opaddmorph: addition respects op_poly_eq*)
Add Parametric Morphism (X Y Z : Type): (@Add X Y Z)
with signature (@op_poly_eq X Y Z) ==> (@op_poly_eq X Y Z)
==>(@op_poly_eq X Y Z)
    as opaddmorph.
Proof. intros x y xyoppolyeq x0 y0 x0y0oppolyeq.
apply Id_Add; assumption. Qed.
```

Some notation

```
Notation "p1 + p2":= (Add p1 p2): op_poly_scope.
Notation "p1 * p2":= (Mult p1 p2): op_poly_scope.
Notation "p1 - p2":= (Subtract p1 p2): op_poly_scope.
Notation "a $ p":= (Mult (@Coeff _ _ _ a) p): op_poly_scope.
Notation "D @ p":= (Op D p) (at level 40): op_poly_scope.
Notation "0":= (@Zeroop _ _ _): op_poly_scope.
Notation "1":= (@Oneop _ _ _): op_poly_scope.
Notation "p1 =' p2":= (op_poly_eq p1 p2)(at level 70).
Notation "p ^ n":= (powerop n p): op_poly_scope.
Notation "n $$ p":= (repeat_addop n p): op_poly_scope.
Notation " D [[ n ]] @ p ":= (repeat_op n D p):op_poly_scope.
Notation "( - p )":= (@neg_op _ _ _ p) : op_poly_scope.
```

Additional assumptions as props

```
Definition derivs (X Y Z : Type) (D: Z): Prop:= forall
(p q : op_poly X Y Z),D@(p*q)=' p*(D@q)+(D@p)*q.

Definition comm_op (X Y Z : Type): Prop:=
forall (p q: op_poly X Y Z), p*q =' q*p.

Lemma derivs_power (p : op_poly X Y Z)(H:@derivs X Y Z D)
(Comm_H:comm_op X Y Z): forall (n:nat), D@(p^n) ='
(n$$(p^(n-1)))*(D@p).
```

Example: a differential ideal generated by x

```
Definition left_ideal_algop (X Y Z : Type)
(I : op_poly X Y Z -> Prop) : Prop :=
(I 0) /\ forall (p q : op_poly X Y Z), ((I p /\ I q) ->
I (p + q)) /\ forall (p q : op_poly X Y Z),
(I p -> I (q * p)).

Definition left_ideal_op (X Y Z : Type)
(I : op_poly X Y Z -> Prop) : Prop :=
(left_ideal_algop I) /\ forall (p : op_poly X Y Z)(D : Z),
(I p -> I (D@p)).
```

Example: a differential ideal generated by x, cont.

```
Section ideal.
Inductive Z : Type :=
  | D.
Variables (X Y: Type)(y1 : Y) (derivH : @derivs X Y Z D)
(commH : @comm_op X Y Z).
Let v1:= @Var X Y Z y1.
Definition I1 (t : op_poly X Y Z) : Prop :=
exists (u1 : op_poly X Y Z), t =' u1 * v1.

Lemma I1_is_left_algideal: left_ideal_algop I1.

Fixpoint nth_level_left_I1 (n : nat)(t : op_poly X Y Z) :
Prop := if n is n.+1 then exists (q u : op_poly X Y Z),
(nth_level_left_I1 n q) /\t =' q + u * (D[[n.+1]]@v1)
else (I1 t).
```

Example: a differential ideal generated by x, cont.

```
Lemma nth_level_left_algideal (n : nat): left_ideal_algop
(nth_level_left_I1 n).

Lemma nth_level_left_cumulative_I1 (i j:nat)(t:op_poly
X Y Z)(H1 : i < j)(H2 : nth_level_left_I1 i t) :
nth_level_left_I1 j t.

Lemma Deriv_up_one_level_I1 (n : nat) : forall
(t : op_poly X Y Z),  nth_level_left_I1 n t ->
nth_level_left_I1 n.+1 (D@t).

Definition I1_op (t : op_poly X Y Z) : Prop :=
 exists n, nth_level_left_I1 n t.

Lemma I1_is_left_ideal_op: left_ideal_op I1_op.

End ideal.
```

Current status, ongoing work, and aspirations

- Rings of polynomials with operators can be described simply in Coq in a way that is very general and allows for fairly smooth proving.
- I am working on defining degrees and rankings in order to define elimination algorithms.
- I want to confirm that the set-up is reasonably compatible with multiple implementations of rings.
- Formalize exciting theorems!

Thank you!