Formalizing the Ring of Witt Vectors

Johan Commelin
Robert Y. Lewis

Certified Programs and Proofs
Jan 17, 2021

Overview

We have:

- defined the type of Witt vectors $\mathbb{W}R$ over a base ring R
- defined the ring structure on $\mathbb{W}R$
- proved that $\mathbb{W}(\mathbb{Z}/p\mathbb{Z}) \simeq \mathbb{Z}_p$

in the Lean proof assistant.

In this talk we will:

- explain why this is an achievement.
- see some techniques that made this formalization possible.

We will not:

- cover much of the mathematics of Witt vectors.
- see many details of the formalization.

Mathematical maturity

Witt vectors are a “mathematically mature” topic.

- Don’t try to unfold all the details.
- Follow high level strategies.

Proof assistants aren’t good at this! How to formalize?

- Isolate the mathematics underlying these strategies.
- Write meta code that simulates the reasoning patterns.

Defining Witt vectors

Fix a ring R and prime p. A p-typical Witt vector over R is a sequence of elements of R:

$$x = (\dots x_2, x_1, x_0) \quad x_i \in R$$

We can add, subtract, and multiply Witt vectors:

$$x + y = (\dots S_2(x_2, x_1, x_0, y_2, y_1, y_0), S_1(x_1, x_0, y_1, y_0), S_0(x_0, y_0))$$

where $\{S_i\}$ is a family of polynomials depending on p.

The details here are intricate.

Witt vector API

To make use of Witt vectors you need some operations:

Teichmüller $\tau : R \to \mathbb{W}R$	$r \mapsto (\ldots, 0, 0, r)$	multiplicative, zero preserving
Verschiebung $V : \mathbb{W}R \to \mathbb{W}R$	$(\ldots, x_2, x_1, x_0) \mapsto (\ldots, x_1, x_0, 0)$	additive
scalar multiplication $[n] : \mathbb{W}R \to \mathbb{W}R$	$x \mapsto n \cdot x$	additive
Frobenius $F : \mathbb{W}R \to \mathbb{W}R$	lift $r \mapsto r^p$ to $\mathbb{W}R$	ring hom
ghost map $W : \mathbb{W}R \to (\mathbb{N} \to R)$	apply *n*th Witt polynomial	ring hom, not injective

Challenge! How to prove properties of these operations without digging too deep into the definition of ring operations?

Strategies for proving Witt vector operation identities

Strategy 1

1. First prove the identity for rings R in which p is invertible.
2. Then prove the identity for polynomial rings over the integers.
3. Finally, use the natural surjective ring homomorphism $\mathbb{Z}[(X_r)_{r\in R}] \to R$ to deduce the identity for arbitrary rings R.

Strategy 2

1. Ignore the fact that the ghost map is not injective in general.
2. Apply the ghost map to both sides of the identity, and prove that the resulting claim is true in $R^{\mathbb{N}}$.

Strategy 2: high risk, high reward

Hazewinkel writes:

> *There are pitfalls in calculating with ghost components as is done here. Such a calculation gives a valid proof of an identity or something else only if it is a universal calculation; that is, makes no use of any properties beyond those that follow from the axioms for a unital commutative ring only.*

Mathematical maturity: if you don't know what you're doing, following this strategy is dangerous!

Polynomial functions

Definition

Let $f_R \colon \mathbb{W}R \to \mathbb{W}R$ be a family of functions where R ranges over all commutative rings. f_R is a polynomial function if there is a family of polynomials $\varphi_n \in \mathbb{Z}[X_0, X_1, \ldots]$ such that for every commutative ring R and each $n \in \mathbb{N}$ and $x = (\ldots x_1, x_0) \in \mathbb{W}R$,

$$f_R(x)_n = \varphi_n(x_0, x_1, \ldots).$$

Theorem (extensionality)

Let $f_R, g_R : \mathbb{W}R \to \mathbb{W}R$ be polynomial functions. If for all $x \in \mathbb{W}\mathbb{Z}$ and $n \in \mathbb{N}$ we have

$$w_n(f_{\mathbb{Z}}(x)) = w_n(g_{\mathbb{Z}}(x)),$$

then $f_R = g_R$ for every ring R.

Strategy 2, refined

Strategy 2

- Show that both sides of the identity are polynomial functions.
- Use extensionality to reduce this to a computation on ghost components.

Why is this good?

- Polynomial functions are well behaved under composition.
- Calculations on ghost components are mostly mechanical.

These identity proofs become almost completely automatic.

Identity proofs, automated

```
/-- The "projection formula" for Frobenius and Verschiebung. -/
lemma verschiebung_mul_frobenius (x y : 𝕎 R) :
    verschiebung (x * frobenius y) = verschiebung x * y :=
by { ghost_calc x y, rintro ⟨⟩; ghost_simp [mul_assoc] }
```

Identity proofs, automated

```
/-- The "projection formula" for Frobenius and Verschiebung. -/
lemma verschiebung_mul_frobenius (x y : 𝕎 R) :
    verschiebung (x * frobenius y) = verschiebung x * y :=
by { ghost_calc x y, rintro ⟨⟩; ghost_simp [mul_assoc] }
    ↑
```

```
p : ℕ
_inst_1 : fact (nat.prime p)
R : Type u_1
_inst_2 : comm_ring R
x y : witt_vector p R
⊢ ⇑verschiebung (x * ⇑frobenius y) = ⇑verschiebung x * y
```

Identity proofs, automated

```
/-- The "projection formula" for Frobenius and Verschiebung. -/
lemma verschiebung_mul_frobenius (x y : 𝕎 R) :
    verschiebung (x * frobenius y) = verschiebung x * y :=
by { ghost_calc x y, rintro ⟨⟩; ghost_simp [mul_assoc] }
                       ↑
```

```
p : ℕ
_inst_1 : fact (nat.prime p)
R : Type u_1
R._inst : comm_ring R
x y : witt_vector p R
⊢ ∀ (n : ℕ), ⇑(ghost_component n) ⇑(verschiebung (x * ⇑frobenius y)) =
    ⇑(ghost_component n) ⇑(verschiebung x * y)
```

It works!

The ring of Witt vectors over $\mathbb{Z}/p\mathbb{Z}$ is isomorphic to the ring of *p*-adic integers:

```
def equiv : 𝕎 (zmod p) ≃+* ℤ_[p] := ...
```

Concluding thoughts

- We can formalize mathematically mature topics with the right idioms.
- $\sim$ 3500 LOC specifically on Witt vectors corresponds to 7 dense pages of Hazewinkel.
- A little metaprogramming goes a long way.