Vertex algebras in Lean: coming soon?

Scott Carnahan

Dept. of Mathematics, University of Tsukuba

2025年1月15日
Lean Together 2025

Where do we see vertex algebras?

Representations of affine Lie algebras

Affine Lie algebras are central extensions of loop algebras $\mathfrak{g}[t, t^{-1}]$ of simple Lie algebras. Smooth representations have natural actions of certain vertex algebras.

Monstrous Moonshine

Vertex algebras give natural representations of sporadic simple groups, like the monster, and connect them with modular functions. "If you're studying the monster and not using vertex algebras, you're basically working with a stone axe." – Borcherds (1998 Fields medal)

A mathematically rigorous part of 2d conformal field theory

In physics, conformal field theory comes up in the study of second order phase transitions, and string worldsheets.

A personal reason to formalize:

Theorem (Huang-Lepowsky-Zhang, around 2010)

If a vertex operator algebra satisfies (technical conditions) then the category of its modules has a (rather natural) braided tensor structure.

The proof is unpublished, and about 400 pages, split into 8 ArXiv preprints.
I have a paper with M. Miyamoto: "Regularity of fixed-point vertex operator subalgebras" (ArXiv: 1603.05645) that (a) uses their theorem in an essential way, and (b) has been rejected (in part) for depending on this un-refereed result.
People who use our result may have the same referee trouble!

A vertex algebra over a `CommRing R` is a triple $(V, \mathbf{1}, Y)$, where:

- V is an R-module,
- $\mathbf{1} \in V$ is a distinguished "unit" vector, and
- $Y : \mathbb{Z} \times V \times V \to V$ is a family of bilinear products, written $(n, u, v) \mapsto u_n v$,

satisfying the following axioms:

- (bounded) For any $u, v \in V$, $u_n v = 0$ for $n \gg 0$.
- (unit) $u_{-1}\mathbf{1} = u$ and $u_n \mathbf{1} = 0$ for $n \geq 0$
- (Jacobi identity) For any $u, v, w \in V$ and $p, q, r \in \mathbb{Z}$,

$$\sum_{i \geq 0} \binom{p}{i} (u_{r+i} v)_{p+q-i} w = \sum_{i \geq 0} (-1)^i \binom{r}{i} u_{p+r-i}(v_{q+i} w) - \sum_{i \geq 0} (-1)^{r+i} \binom{r}{i} v_{q+r-i}(u_{p+i} w).$$

What do we need to formalize this?

1. For $(V, \mathbf{1})$, use `AddCommGroupWithOne V` and `Module R V`.
2. For $Y : \mathbb{Z} \times V \times V \to V$, we can use $\mathbb{Z} \to V \to_l [R]\, V \to_l [R]\, V$ such that for any $u, v \in V$, $u_n v = 0$ for $n \gg 0$.
3. For sums like $\sum_{i \geq 0} (-1)^i \binom{r}{i} u_{p+r-i}(v_{q+i} w)$, we need binomial coefficients with integers on top.

Alternative: use generating functions (later)

1. Set $Y(u, x)v = \sum_{n \in \mathbb{Z}} u_n v x^{-n-1}$, so $Y : V \otimes V \to V((x))$ - use `LaurentSeries V`?
2. Jacobi identity seems to involve composites - maybe need `LaurentSeries (LaurentSeries V)`.

General binomial coefficients $\binom{x}{n}$: need a way to say that for any element x and natural number n, the product $x(x-1)\cdots(x-n+1)$ is uniquely divisible by $n!$.

First try (Aug. 2023)

```
class BinomialSemiring (R: Type _) extends Semiring R
where
  inj_smul_factorial : ∀ (n : ℕ) (r s :  R),
    n.factorial * r = n.factorial * s → r = s
  exist_binomial_coeffs : ∀ (r :  R) (n : ℕ),
    ∃ (x :  R), n.factorial * x =
      Polynomial.eval r (pochhammer R n)
```

Problems: Uses choice unnecessarily, casts from $\mathbb{N}$ instead of using `nsmul`, $\forall$ can be absorbed, etc.

On Zulip, Junyan Xu suggested making the quotient an explicit function. Other refinements from PR review.

Current form

```
class BinomialRing (R : Type*) [AddCommMonoid R]
    [Pow R ℕ] where
  nsmul_right_injective {n : ℕ} (h: n ≠ 0) :
    Injective (n • · : R → R)
  multichoose : R → ℕ → R
  factorial_nsmul_multichoose (r : R) (n : ℕ) :
    n.factorial • multichoose r n =
      (ascPochhammer ℕ n).smeval r
```

Given `AddCommGroupWithOne R`, we define `Ring.choose x n` as `multichoose (r - n + 1) n`.

First try: direct attack

```
structure VertexAlgebra [CommRing R]
    [AddCommGroupWithOne V] [Module R V] where
  Y : ℤ → V →ₗ[R] V →ₗ[R] V
  bound (u v : V) : ∃ n : ℕ, m > n → Y m u v = 0
  mul_neg_one_unit (u : V) : Y (-1) u (1 : V) = u
  mul_nat_unit (u : V) (n : ℕ) : Y n u (1 : V) = 0
  jacobi (u v w : V) (r s t : ℤ) :
    finsum (fun i ↦ (Ring.choose r i) •
      Y (r+s-i) (Y (t+i) u v) w) =
    finsum (fun i ↦ (negOnePow i) •
      (Ring.choose t i) • Y (r+t-i) u (Y (s+i) v w)) -
    finsum (fun i ↦ (negOnePow (t+i)) •
      (Ring.choose t i) • Y (s+t-i) v (Y (r+i) u w))
```

Problems

- Working with big equations is cumbersome.
- Need lots of explicit choice to work with `bound`.
- The literature uses lots of power series manipulations, which are hard to translate to explicit coefficient manipulations.

Solution:

Develop formal power series API, rewrite axioms in terms of formal power series.

- $Y(u,x)v = \sum_{n\in\mathbb{Z}} u_n v x^{-n-1}$, so $Y(u,x)$ is a linear map $V \to V((x))$.
- Jacobi identity can be split into "locality" and "associativity" for maps $V \to V((x))$.

Example: Locality

Fact: For any u, v in a vertex algebra V, there is some $n \in \mathbb{N}$ such that $(x-y)^n Y(u,x)Y(v,y) = (x-y)^n Y(v,y)Y(u,x)$.
Proof: Take t big enough, so the left side of the Jacobi identity vanishes. The other sums give the $x^{-r-1}y^{-s-1}$ coefficients of $(x-y)^t Y(u,x)Y(v,y)$ and $(x-y)^t Y(v,y)Y(u,x)$.

Composites

We want to make sense of $(x-y)^n \bullet Y(u,x)Y(v,y)$ for $n \in \mathbb{Z}$.
$Y(u,x)Y(v,y)$ is an R-linear map $V \to V((x))((y))$.
The target admits scalar multiplication from $R((x))((y))$, so that is where we expand $(x-y)^n$.

Rather than iterating `LaurentSeries`, we consider the more general framework of Hahn series.

If Γ is a poset, a `HahnSeries` is a formal power series $\Gamma \to R$ with "partially well-ordered" support.
When Γ is an `OrderedAddCommMonoid` and R is a `Ring`, we get `Ring (HahnSeries Γ R)`.
`LaurentSeries R` is `HahnSeries ℤ R`.
`LaurentSeries (LaurentSeries R)` is `HahnSeries ℤ ×| ℤ R`.

Problem: diamond (pointed out to me by Eric Weiser)

Given a `Module R V` instance, it is dangerous to make a `Module (HahnSeries Γ R) (HahnSeries Γ V)` instance - get a diamond for `V = HahnSeries Γ R`.

Solution: type alias

We define `HahnModule` as a type alias for `HahnSeries`, and define an instance `Module (HahnSeries Γ R) (HahnModule Γ' R V)`.

First, we define heterogeneous vertex operators, as a general form of left multiplication:

```
abbrev HVertexOperator (Γ R V W : Type*)
    [PartialOrder Γ] [CommRing R] [AddCommGroup V]
    [Module R V] [AddCommGroup W] [Module R W] :=
  V →ₗ[R] (HahnModule Γ R W)
```

These have an action of `HahnSeries Γ R`. Special case:

```
abbrev VertexOperator (R V : Type*) [CommRing R]
    [AddCommGroup V] [Module R V] :=
  HVertexOperator ℤ R V V
```

We have a "normalized coefficient" function: `ncoeff n` takes the x^{-n-1} coefficient.

"Residue product" vertex operators

$$A(y)_n B(y) = \mathrm{Res}_{x=0}(x-y)^n A(x)B(y) - \mathrm{Res}_{x=0}(x-y)^n B(y)A(x).$$

Here, $n \in \mathbb{Z}$, and $\mathrm{Res}_{x=0}$ just takes the x^{-1}-coefficient.

```
structure VertexAlgebra (R V : Type*) [CommRing R]
    [AddCommGroupWithOne V] [Module R V] where
  Y : V →ₗ[R] VertexOperator R V
  unit_neg_one (u :  V) : (Y u (1 :V)).ncoeff (-1) = u
  unit_nat (u :  V) (n : ℕ ) :
    (Y u (1 :  V)).ncoeff n = 0
  local (u v :  V) : isLocal (Y u) (Y v)
  resProd_eq (u v :  V) (n : ℤ) :
    resProd (Y u) n (Y v) = (Y u v).ncoeff n
```

Necessities for basic theory

- Some infinite dimensional Lie algebras, their central extensions, and their "smooth" representations
- $(x-y)^n \bullet Y(u,x)Y(v,y)$ for $n \in \mathbb{Z}$.

Intermediate theory

- Twisted modules: $V \otimes M \to M((z^{1/N}))$
- Log-intertwining operators: $M_1 \otimes M_2 \to z^r M_3((z))[\log z]$
- "Composites" of log-intertwining operators.

More advanced theory

- Analytic correlation functions $\langle \phi, Y(u_1,x_1)\cdots Y(u_n,x_n)v\rangle$ for $x_1,\ldots,x_n$ on a Riemann surface.
- Some diff. eqs. (satisfied by the correlation functions)
- Semi-infinite cohomology

What I really want: Good API for rational functions $(x-y)^k(x-z)^m(y-z)^n, (k,m,n \in \mathbb{Z})$.

Application: Dong's Lemma for residue products

Given $A, B, C : V \to V((x))$, if A, B, and C are local, then A and $B_n C$ is local, for any $n \in \mathbb{Z}$.

Standard proof involves substitutions, like $(x-z) = (x-y) + (y-z)$. I'd like to write this using something more concise than `monomial 1 toLex(toLex(0,1),0) + monomial (-1) toLex(toLex(1,0),0)`.

Things I learned:

- Definitions are hard to get right.
- It is good to get an early start at defining things, even if they are likely to be bad at first.
- If you ask a question on Zulip, you may get an answer to a better question that you didn't think to ask.
- PR review is great for learning good style and new ideas.
- If a proof seems like a grind, maybe some API is missing.