Theoretical Commutative Algebra in Coq: Nakayama's Lemma

1 Introduction

A number of structures from abstract algebra have been formalized using the proof assistant Coq [7]. That said, there currently appears to be a computational emphasis with regard to higher-level algebra, with more formalization of discrete objects and algorithmic processes rather than the verification of more abstract theorems. For instance, the commutative-algebraic notion of a Gröbner basis, a special type of generating set for an ideal in a ring of polynomials, and which has cryptographic applications, has gathered significant attention (e.g., see [8]).

The mathematical field of commutative algebra not only has further applications to cryptography, but also in other areas of science and technology such as data science, and the study of the human genome. The field stems from the study of polynomial equations, and now centers around commutative rings, i.e., those whose multiplication satisfies the commutative law, along with ideals of these rings, and modules over them. Commutative algebra has meaningful connections with other areas of theoretical mathematics, including number theory and algebraic geometry.

We construct a formal proof of *Nakayama's lemma* [6, 2], a fundamental theorem in the field of commutative algebra, in Coq. Sometimes also called the Krull-Azumaya theorem[1] [5], Nakayama's lemma is typically introduced in a first graduate course in commutative algebra [1, 4, 3], and is applied widely in the field.

In order to prove Nakayama's lemma, we formalize certain algebraic structures that are essential to higher-level algebra, including *local rings* and *modules over commutative rings*. The notion of a module over a ring is an extension of the linear-algebraic notion of a vector space over a field, ubiquitous in mathematics and its applications. Nakayama's lemma exhibits one way that a finitely generated module over a commutative ring behaves like a vector space over a field.

Add link to github for code?

2 Mathematical basis and motivation

Nakayama's Lemma. *Let R be a commutative local ring with maximal ideal $\mathfrak{m}$, and suppose that M is a finitely generated R-module. If $M = \mathfrak{m}M$, then $M = 0$.*

3 Formalization

We started by defining a semigroup class, which declares a binary operation to be associative. From here, we build up through monoids, which introduce identities, to groups, which introduce inverses. Note the double equals in these definitions is notation for an arbitrary equivalence relation over the group's carrier set which acts as equality.

[1] Hideyuki Matsumura explains in his text *Commutative Algebra* [4]: *This simple but important lemma is due to T. Nakayama, G. Azumaya and W. Krull. Priority is obscure, and although it is usually called the Lemma of Nakayama, late Prof. Nakayama did not like the name.*

```
Infix "==" := equiv (at level 60, no associativity).
Class Semigroup := {
  semigroup_assoc:
    forall (a b c: Carrier),
      a <o> b <o> c == a <o> (b <o> c);
}.
Class Monoid := {
  monoid_semigroup :> Semigroup equiv op;
  monoid_ident_l:
    forall (a: Carrier), ident <o> a == a;
  monoid_ident_r:
    forall (a: Carrier), a <o> ident == a;
}.
Class Group := {
  group_monoid :> Monoid equiv op ident;
  group_inv_l:
    forall (a: Carrier), inv a <o> a == ident;
  group_inv_r:
    forall (a: Carrier), a <o> inv a == ident;
}.
```

The lines like `monoid_semigroup :> Semigroup equiv op;` simply coerce the monoid typeclass into a semigroup.

While we found later on that we did not need quotients, it is worth remarking that we got quotients working rather nicely in Coq using typeclasses. An algebraic quotient are an equivalence relation on the algebraic structure which preserve that structure. For example, quotient groups are formed by taking a subgroup and making every element of that subgroup equivalent to the identity. With `P` being the predicate for the subgroup, there are two ways to make an equivalence relation from this description.

```
Definition left_congru (a b: Carrier) :=
  P (inv a <o> b).
Definition right_congru (a b: Carrier) :=
  P (a <o> inv b).
```

When these two relations coincide, then we can prove that the equivalence relation(s) actually preserve the group structure. Subgroups which have this property are called *normal subgroups*.

```
Let normal_subgroup_congru_coincide :=
  forall (a b: Carrier),
    left_congru op inv P a b <->
    right_congru op inv P a b.

Theorem quotient_normal_subgroup_group:
  normal_subgroup_congru_coincide ->
  Group (left_congru op inv P) op ident inv.
```

It is because of quotients that we used equivalence relations to define the components of group structure. If we were to use the regular Leibniz equality, this would make it very difficult to say that a quotient group is another group. But by having the definition of a group depend upon an arbitrary equivalence relation, we enable our theory to state that a quotient group is simply a group with under a different equivalence. We didn't loose much as we could still rely upon Coq's setoid rewrite tactics. Setoids are types equipped with an equivalence relation.

Moving onwards, we then defined structures for rings, which have two binary operations: a commutative plus $+$ and a non-commutative times $\cdot$, and structures for commutative rings, where

multiplication is commutative and has an identity. Here we define *ideals* which are normal subgroups under addition and are absorbing with regards to multiplication, i.e. ra is in the ideal whenever a is in the ideal and r is any element of the ring. Again, we defined quotient rings but found them unnecessary in the end. Note that should an ideal I contain a unit, an element x with a multiplicative inverse x^{-1}, then I would be the entire ring by the absorbing property: given any element a of the ring, then ax^{-1} is also in the ring, and $ax^{-1}x = a1 = a \in I$. Maximal ideals were defined next. These are ideals that are proper subsets of the ring while having no larger ideal except for the ring itself. Below is the definition in Coq, which uses `P` as the predicate for the ideal.

```
Definition maximal_ideal :=
  exists (r: Carrier), (not (P r) /\
    forall (Q: Carrier -> Prop)
        (Q_proper: Proper (equiv ==> iff) Q)
        (Q_ideal: Ideal add zero minus mul Q),
      (forall (r: Carrier), P r -> Q r) ->
      (forall (r: Carrier), Q r) \/
        (forall (r: Carrier), Q r -> P r)).
```

Almost by definition, maximal ideals can contain no units, otherwise they would fail to be a proper subset of the ring. We could then define a local ring, which is a ring with a single maximal ideal.

```
Definition local_ring :=
  exists (P: Carrier -> Prop)
      (P_proper: Proper (equiv ==> iff) P)
      (P_ideal: Ideal add zero minus mul P),
    maximal_ideal P /\
    (forall (Q: Carrier -> Prop)
        (Q_proper: Proper (equiv ==> iff) Q)
        (Q_ideal: Ideal add zero minus mul Q),
      maximal_ideal Q -> forall (r: Carrier), P r <-> Q r).
```

Here we had to include an axiom that in commutative ring, any non-unit x is contained in some maximal ideal. This was made into an axiom as the standard mathematical argument is a proof with potentially infinitely many steps. The standard argument goes as follows.

> Set $I_1 \coloneqq (x)$ to be the principal ideal for x, i.e. the ideal generated by the single element x. If I_1 is not maximal, then there exists a strictly larger ideal I_2, i.e. $x \in I_1 \subsetneq I_2$. If I_2 is not maximal, then there exists another strictly larger ideal I_3. Continue these arguments infinitely many times if necessary in order to get an infinite chain of ideals containing x.
>
> $$x \in I_1 \subsetneq I_2 \subsetneq I_3 \subsetneq \cdots \subsetneq R$$
>
> It is a simple matter to show that $\bigcup_{k=1}^{\infty} I_k$ is also an ideal containing in x. So by Zorn's lemma, there exists a maximal ideal in R which contains x.

Zorn's lemma is equivalent to the axiom of choice. So we had to add at least one axiom here in order to proceed. By adding the axiom that we did, not only will we encapsulate an infinite argument, but we will also avoid the need for the axiom of choice.

Also, we used classical logic to prove that $1 - x$ is a unit where x is a non-unit in any local ring. The proof used the rule $\neg\neg P \to P$ in order to do a proof by contradiction.

The next structure defined were modules over rings which generalize vector spaces over fields. We needed to capture the notion of linear combinations of coefficients and module vectors. This was done by dependently typed vectors, i.e. lists parameterized by their length. Because there is an overload of the term "vector", we will use that term to refer to module vectors, and use the term "list" to mean length parameterized lists. As we don't use the simpler kind of lists, this avoids any

name collisions. A finitely generated module is like the vector space $\mathbf{R}^n$ in that there are finitely many vectors which can generate all other vectors, for $\mathbf{R}^n$ one such collection of generators are $\mathbf{e}_1 = \begin{pmatrix}1 & 0 & 0 & \cdots & 0\end{pmatrix}^T$, $\mathbf{e}_2 = \begin{pmatrix}0 & 1 & 0 & \cdots & 0\end{pmatrix}^T$, $\ldots$, $\mathbf{e}_n = \begin{pmatrix}0 & 0 & 0 & \cdots & 1\end{pmatrix}^T$. In our code, `M` is the type of module elements, `R` is the type of ring elements which act as coefficients, and `t A n` is a list whose elements are of type `A` and whose length is `n`.

```
Definition finitely_generated {n: nat}(basis: t M n) :=
  forall (vector: M),
    exists (coeffs: t R n),
      vector =M= linear_combin coeffs basis.
```

Next, we defined the product of an ideal I and a module M, denoted as IM, which is defined to be all linear combinations of vectors from M and coefficients from I. This can be easily shown to be a submodule of M. We represented this in Coq as a predicate over M.

```
Context (P: R -> Prop).
Context {P_proper: Proper (Requiv ==> iff) P}.
Context {P_ideal: Ideal Radd Rzero Rminus Rmul P}.

Definition ideal_module (x: M): Prop :=
  exists (n: nat)(coeffs: t R n)(vectors: t M n),
    Forall P coeffs /\
    x =M= linear_combin Madd Mzero action coeffs vectors.
```

The `Forall P coeffs` ensures that every element of the coefficient list `coeffs` satisfies the predicate `P`. From here, we can move to stating Nakayama's lemma.

Theorem 3.1 (Nakayama's lemma)**.** *Let M be a finitely generated module over a local ring R with maximal ideal $\mathfrak{m}$. If $\mathfrak{m}M = M$, then $M = 0$, which is to say that M is the zero module which consists of exactly one element, that being the zero vector.*

We needed a lemma before moving on to prove the main theorem which states that for a finitely generated module M with a basis $b_1, b_2, \ldots, b_n$, any element $x \in IM$ where I is an ideal can be written as a linear combination of the basis vectors with coefficients coming from I. The proof of this lemma is straightforward and follows the standard, informal mathematical argument which inductively goes through the vectors needed to generate x and shows that each vector can be rewritten in terms of the basis vectors times elements of the ideal I as follows.

$$\begin{aligned} x &= \sum_{k=1}^{m} u_k a_k \\ &= \sum_{k=1}^{m} u_k (r_{k1} b_1 + \cdots + r_{kn} b_n) \\ &= \sum_{j=1}^{n} (u_1 r_{1j} + \cdots + u_m r_{mj}) b_j \end{aligned}$$

Since each u_k comes from the ideal I, the absorbing property of ideals guarantees that $u_k r_{kj}$ is also contained in I. As ideals are also closed under addition, it follows that $u_1 r_{1j} + \cdots + u_m r_{mj}$ are all elements of I. Thus, x can be written as a linear combination of the basis vectors with the coefficients coming from I.

For the proof of Nakayama's lemma, it too follows the standard, informal mathematical argument. Do induction on the number of vectors needed to generate the module M. The base case where M is generated by 0 vectors is by definition true since an empty linear combination is conventionally taken to be the zero vector. The inductive case where M is generated by the vectors $b_1, \ldots, b_m, b_{m+1}$. By

assumption $M = \mathfrak{m}M$ with $b_1 \in M$. Then $b_1 \in \mathfrak{m}M$ meaning that by our lemma, there exists a linear combination for b_1, say

$$b_1 = u_1 b_1 + \cdots + u_{m+1} b_{m+1}$$

for some $u_1, \ldots, u_{m+1} \in \mathfrak{m}$. Collect the b_1 terms on the left-hand side of the equation to get that

$$(1 - u_1) b_1 = u_2 b_2 + \cdots + u_{m+1} b_{m+1}.$$

As mentioned when defining maximal ideals, $\mathfrak{m}$ can only contain non-units. Namely, u_1 must be a non-unit. Then we had already proven that $1 - u_1$ is a unit as we are in a local ring. Let v_1 be the multiplicative inverse of $1 - u_1$, and multiply it on both sides of the equation.

$$\begin{aligned} v_1(1 - u_1) b_1 &= v_1 u_2 b_2 + \cdots + v_1 u_{m+1} b_{m+1} \\ 1 b_1 &= \\ b_1 &= v_1 u_2 b_2 + \cdots + v_1 u_{m+1} b_{m+1} \end{aligned}$$

We have thus found that one of the basis vectors is not needed to generate this module. Using the induction hypothesis, we have that $M = 0$. Showing that the induction hypothesis holds in Coq takes more work than in an informal proof, but this extra work is just a lot of bookkeeping.

References

[1] M. F. Atiyah and I. G. Macdonald. *Introduction to commutative algebra*. Addison-Wesley Series in Mathematics. Westview Press, Boulder, CO, economy edition, 2016. For the 1969 original see [MR0242802]. 1

[2] Gorô Azumaya. On maximally central algebras. *Nagoya Math. J.*, 2:119–150, 1951. 1

[3] David Eisenbud. *Commutative algebra*, volume 150 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1995. With a view toward algebraic geometry. 1

[4] Hideyuki Matsumura. *Commutative algebra*, volume 56 of *Mathematics Lecture Note Series*. Benjamin/Cummings Publishing Co., Inc., Reading, Mass., second edition, 1980. 1

[5] Masayoshi Nagata. *Local rings*. Interscience Tracts in Pure and Applied Mathematics, No. 13. Interscience Publishers (a division of John Wiley & Sons, Inc.), New York-London, 1962. 1

[6] Tadasi Nakayama. A remark on finitely generated modules. *Nagoya Math. J.*, 3:139–140, 1951. 1

[7] The Coq Development Team. The Coq Proof Assistant, October 2019. 1

[8] Laurent Théery. A machine-checked implementation of buchberger's algorithm. *Journal of Automated Reasoning*, 20:107–137, 2001. 1