SCHEMES IN LEAN

KEVIN BUZZARD, CHRIS HUGHES, KENNY LAU, AND RAMON FERNÁNDEZ MIR

Abstract. We talk about the issues which arose when formalising Grothendieck's notion of a *scheme*, and the construction of affine schemes, in the Lean theorem prover.

1. Introduction

In my (KB) talk, I will talk about how and why we formalised a mathematical object called a *scheme* in the Lean theorem prover. I will provide context and background in the talk; in this extended abstract we stick mostly to the issues which arose in the formalisation process.

A scheme is a mathematical object whose definition and basic properties are usually taught at MSc or early PhD level in a typical mathematics department. The basic idea behind the definition is simple, and we start by describing it informally.

A *ring* is a mathematical structure where one can perform addition, subtraction and multiplication, and various natural axioms are satisfied. The integers $\mathbb{Z} = \{\dots, -2, -1, 0, 1, 2, \dots\}$ are an example of a ring. Note that all rings in this abstract are commutative, meaning that one of the natural axioms mentioned above is $a \times b = b \times a$.

Informally speaking, given a geometric object X like $\mathbb{R}^3$ or a circle, the set of real-valued functions defined on X is a ring; one can define addition and multiplication of functions "pointwise" – if f and g are functions $X \to \mathbb{R}$, one can set $(f + g)(x) = f(x) + g(x)$ and so on. If X has some extra structure (like a topology, or the structure of a manifold) then one can also consider spaces of continuous functions, or differentiable functions on X, and these are also rings. Analysis of these rings (perhaps done using algebraic methods) may be able to shed light on the geometric structure of X.

In the 1960s Grothendieck turned this situation on its head. Given as input a ring R, Grothendieck was able to define a geometric object $X := \mathrm{Spec}\,(R)$ and give a definition of "function" on X such that the ring of functions on X was, in a natural way, isomorphic to R. Such an object X is called an *affine scheme*, and a *scheme* is a geometric object which "locally looks like an affine scheme", that it, it is the union of open subsets each of which is isomorphic to an affine scheme. The devil, however, is in the details, and it is these details which the authors formalised.

The history of the project is as follows. The first author (KB) is a mathematician who wanted to learn about theorem provers, and was surprised to learn early on that even though these programs had been around for decades, what seemed to be being formalised in the main was theorems (sometimes with very intricate and long proofs) about elementary objects. He hence embarked on a project to formalise some elementary theorems about more complex mathematical objects, as a way of learning how to use a theorem prover. When the scale of the task became apparent, the second (CH) and third (KL) authors (both first year mathematics undergraduates at the time, who had been attending KB's "Xena project" formal verification club at Imperial College London) joined the project, writing ring theory libraries and proving theorems necessary for the definition of a scheme, and the proof that affine schemes were schemes. The resulting code base was in parts extremely amateurish, perhaps unsurprisingly, given that none of the authors had ever worked with formal proof verification systems before. The fourth author (RM) is a joint mathematics and computer science MSc student at Imperial College who is rewriting the code base from scratch, supervised by KB who had now learnt from the many mistakes he made in the original code base.

The first author was supported in part by EPSRC grant EP/L025485/1.

The original code is at https://github.com/kbuzzard/lean-stacks-project and the refactoring is at https://github.com/ramonfmir/lean-scheme.

2. Mathematical details.

The formal notion of "geometric object" referred to above which Grothendieck uses is a *locally-ringed space*. A locally-ringed space is a topological space equipped with a sheaf of rings such that all the stalks of the sheaf are local. Each of the technical terms in this definition can be taken apart further of course – for example a sheaf of rings is a presheaf of rings satisfying a further axiom, and a presheaf of rings is a contravariant functor from the category of open sets of the space to the category of rings.

To define a scheme in Lean one first has to define the notion of a locally ringed space. The main part of the work here, given the state of Lean's maths library when we embarked upon this project, was to define the following notions: a direct limit of rings (and the universal property of this limit), presheaves and sheaves, and the construction of a locally ringed space structure on an open subset of a locally ringed space. None of these constructions presented any particular difficulty.

More serious hurdles appeared when constructing schemes. A general scheme is a union of affine schemes, and the main work in the project was to construct the topological space $\mathrm{Spec}\,(R)$ associated to a ring R, and to show that it was a scheme. The main problem was in defining the presheaf of rings, and showing it was a sheaf, so we finish this abstract with a summary of the problems we faced.

There were two major hurdles to defining the presheaf of rings on $\mathrm{Spec}\,(R)$. The first involved making a robust API for localisation of rings. The second was the realisation that mathematicians are extremely good at identifying two isomorphic objects as "equal", especially in situations where the isomorphism is "canonical" a weasel word which apparently has no formal definition. In the middle of a mathematical proof, a ring might suddenly be replaced with a canonically isomorphic ring without warning. Our initial approach to this was rather ugly, however it got much cleaner after we learnt of Neil Strickland's predicate which classifies maps between rings which are isomorphic to localisations, and this predicate played a key role in the refactoring, greatly simplifying some arguments.

Showing the presheaf was a sheaf was a two step process. The construction we formalised was the argument in Johan de Jong's stacks project. The first step involves showing that the presheaf satisfies the sheaf axiom on basic open sets; this boiled down to a combinatorial lemma in ring theory. Originally it was proved for localisations; in the refactoring it will be proved using Strickland's predicate (this is the only part of the refactoring yet to be completed). A more formal argument then shows that the sheaf property propagates to the entire space. In formalising this argument we learnt a lot about equality in type theory, for example the notion that if two open subsets U and V of a space with a sheaf were equal, then the values the sheaf takes on U and V should not be regarded as equal (equality of types) but instead as isomorphic. This key realisation, one of many realisations we learnt at the Zulip Lean chat, turned several arguments from battles in dependent type theory to trivialities.

The first three authors would like to thank Mario Carneiro and the other regulars in the Lean Zulip chat room, for teaching them how to use the software.

Email address: k.buzzard@imperial.ac.uk

Department of Mathematics, Imperial College London

Email address: christopher.hughes17@imperial.ac.uk

Department of Mathematics, Imperial College London

Email address: kin.lau17@imperial.ac.uk

Department of Mathematics, Imperial College London

Email address: ramon.fernandez-i-mir15@imperial.ac.uk

Department of Mathematics, Imperial College London