Handling Cryptomorphism in Proof Assistants and Automated Concept Formation

Floris van Doorn, Michael R. Douglas and David McAllester*

May 11, 2021

Abstract

A mathematical concept can be defined in different ways. For example, a group can be defined as a set together with a group operation, an inverse operation and identity element such that the group axioms hold, or a group can be defined as a set together with a group operation such that the inverse operation and identity element exist. The term cryptomorphism was coined by Birkoff and popularized by Rota as a name for this phenomenon. Cryptomorphism is important in proof assistants and in systems for exploratory mathematics where we want to introduce new concepts while avoiding redundancy. We develop formal definitions of concept and cryptomorphism within a dependent type theory and describe tactics for establishing cryptomorphisms between different concept definitions.

Mathematical structures are at the core of mathematics. Structures are exemplified by number systems (natural numbers, integers, rationals, reals, *etc.*), by algebraic systems (semigroups, groups, modules, *etc.*), and by topological and geometric structures (topological spaces, manifolds, *etc.*). A common approach to group these structures in classes, used for example by Bourbaki, is the signature-axiom class (SAC). A signature-axiom class can be defined by a type expression of dependent type theory whose instances are the structures in that class. These structures consist of one or more carrier sets (sorts) together with data over the sorts — typically constants, functions, and predicates. The class expression (the type expression defining the class) also specifies axioms that the data must satisfy.

Here we are interested in defining and automatically recognizing equivalence between concepts — equivalence between type expressions that define signature-axiom classes. An example of such an equivalence is the two equivalent definitions of a group as a four-tuple of a set, a group operation, an inverse operation and an identity element or the equivalent definition of a group as a pair of a set and a group operation such that the inverse operation and identity element exist. Equivalent concept definitions are called cryptomorphic, a term coined

*All three authors contributed equally to this work.

by Birkoff and popularized by Rota. In modern treatments cryptomorphism is typically defined in terms of category theory. However, we do not wish to require a human user or automated mathematical exploration system to specify a category for each concept. Rather each concept, by virtue of being a type in a dependent type theory, is naturally associated with groupoid structure. In a model of type theory in which all lambda expressions denote functors over this inherent groupoid structure we can define cryptomorphism to simply mean that there exists a pair of lambda expressions establishing a bijection between the two classes. In an appropriate type theory it is possible to show that every class expression is cryptomorphic to a signature-axiom expression where the sorts, data, and axioms are explicitly segregated [1].

A library of concepts (signature-axiom class expressions) naturally forms a concept graph whose nodes are the concepts and whose edges include containment between concepts, for example an Abelian group is a subclass of groups, and functors mapping one concept into another. We wish to avoid redundancy in this concept graph — we want cryptomorphic concepts to be represented by a single node. We consider various types of edges in the concept graph, depending on the relation between two concepts.

Establishing a cryptomorphism between two concepts involves finding a pair of functors establishing a bijection between them. This is a special case of the problem of finding nontrivial functors between concepts. Finding useful functors is fundamental to automated mathematical exploration and we also consider this more general problem. Another important feature of automated mathematical exploration is to recognize that two structures are *not* cryptomorphic. We can do this by showing that the automorphism groups the structures are different.

We also consider the problem of defining an objective function for automated mathematical exploration. For this we consider a richer graph including theorems, proofs and tactics. We consider an objective defined in terms of coverage and size. Coverage is defined in terms of the fraction of automatically generated questions that have "obvious" answers where obviousness is defined by an automated tactic. Size is simply the size of the library used in computing obvious answers. Shorter proofs using more general concepts are considered superior. Automated library expansion can be done by improving coverage at the cost of size and then reducing size for a given coverage. We will propose Reinforcement learning methods for training a value function that predicts the utility (eventual out degree in the library graph) of new conjectures, concepts and tactics.

We have implemented some of these proposals in the Lean ITP system. This includes tactics for finding functors, and a concept graph extraction tool. The latter is derived from a tool developed by Patrick Massot, adapting it to the problem of extracting concepts as we have defined them.

References

[1] David McAllester, *Dependent type theory as related to the Bourbaki notions of structure and isomorphism*, 2021.