A property-based testing library

Jason Chen CJ Quines Matthew Ho

April 5, 2022

In software engineering, testing code is considered to be a vital part of delivering quality software. However, writing unit tests is extremely time consuming, and it is easy for the programmer to miss certain cases for more complicated functions. Therefore, it is desirable to try to automate this processes either partially, or entirely.

We build a QuickCheck-like tool that allows users to formulate properties of programs and then automatically runs tests based on the properties. Properties that users want to test will be specified via a DSL embedded in Scheme. The properties will be represented as functions along with typing information of the parameters, where the types will enable the generation of test data. To generate the actual test data, we choose random testing, with a uniform distribution over the type by default. This is interesting because it allows "generator deriving": we automatically have a way to generate lists of integers if we have a way to generate integers.

We also allow users to specify their own distribution over types via generics to parallel real world data. To bias the generator towards poking around edge cases, we can use this feature to specify distributions testing extremes, or around what would be the boundaries if we did partition testing. These would be built-in, and the user would be able to specify an alternate distribution as well.

One potential extension to this idea is test shrinking: if a counterexample is found for a property, the library will automatically simplify the failure case to a minimal failing example via a built-in (and user-defined) rule system. This involves defining a generic shrinking function for each type, which is interesting in the case of recursive types.