A gentle introduction to property based testing

Alexander Færøy

BornHack 2017

August 28, 2017

Outline

- Ignore "the Erlang part".
- Unit testing vs. property based testing.
- Defining propety tests using Proper in Erlang.
- Questions

Concepts

- Generators.
- Shrinking strategies.

Base64 encoder and decoder

```
-type encode(binary()) -> binary().
encode(Data) ->
    ...

-type decode(binary()) -> binary().
decode(Data) ->
    ...

-type is_valid(binary()) -> boolean().
is_valid(Data) ->
    ...
```

Base64 encoder and decoder

Unit testing

```
test_b64_encoding() -> [
    ?assertEqual(<<"">>, encode(<<"">>)),
    ?assertEqual(<<"YWJj">>, encode(<<"abc">>))
].
```

Base64 encoder and decoder

Unit testing

```
test_b64_decoding() -> [
    ?assertEqual(<<"">>, decode(<<"">>)),
    ?assertEqual(<<"abc">>, decode(<<"YWJj">>))
].
```

Base64 encoder and decoder

Testing properties

```
prop_iso() ->
    ?FORALL(Data, binary(),
        begin
            EncodedData = encode(Data),
            Data =:= decode(EncodedData)
        end).
```

Base64 encoder and decoder

Testing properties

```
id(X) ->
    X.

prop_iso() ->
    ?FORALL(Data, binary(),
        begin
            EncodedData = id(Data),
            Data =:= id(EncodedData)
        end).
```

Base64 encoder and decoder

Testing properties

```
prop_encoding_output_length() ->
    ?FORALL(Data, binary(),
        begin
            EncodedData = encode(Data),
            DataSize = byte_size(Data),
            byte_size(EncodedData) =:=
                ((4 * DataSize div 3) + 3)
                    band (bnot 3)
        end).
```

Base64 encoder and decoder

Testing properties

```
-define(BASE64_ALPHABET,
    lists:seq($0, $9) ++
    lists:seq($a, $z) ++
    lists:seq($A, $Z) ++
    "+/").

is_valid(Data) when is_list(Data) ->
    valid_string(Data, ?BASE64_ALPHABET).

valid_string(Input, Alphabet) ->
    Set = ordsets:from_list(Alphabet),
    lists:all(fun (C) ->
        ordsets:is_element(C, Set)
    end, Input).
```

Base64 encoder and decoder

Testing properties

```
prop_encoding_alphabet() ->
    ?FORALL(Data, binary(),
        begin
            EncodedData = encode(Data),
            is_valid(EncodedData)
        end).
```

Base64 encoder and decoder

Testing properties

```
3> proper:quickcheck(b64:prop_encoding_alphabet()).
..!
Failed: After 3 test(s).
<<4>>

Shrinking .(1 time(s))
<<0>>
```

Base64 encoder and decoder

Testing properties

```
-define(BASE64_ALPHABET,
    lists:seq($0, $9) ++
    lists:seq($a, $z) ++
    lists:seq($A, $Z) ++
    "+/=").
```

Base64 encoder and decoder

Testing properties

```
3> proper:quickcheck(b64:prop_encoding_alphabet()).
.....................................................
OK: Passed 100 test(s).
```

Missing

- Using Niffy to test C code.
- Stateful symbolic execution.
- Concurrency testing.

Other implementations

Erlang	Proper
Python	Hypothesis
Haskell	QuickCheck
Rust	quickcheck

Questions?