Property Testing with derived idempotents

Brian Zeligson

April 2020

Table of Contents

Introduction to Property Testing

Isomorphisms: Ideal Property Testing Candidates

Idempotents: Making Property Testing Practical

Application: Deriving Idempotents for Property Testing

Live Coding: Property Testing FizzBuzz

Table of Contents

Introduction to Property Testing

Isomorphisms: Ideal Property Testing Candidates

Idempotents: Making Property Testing Practical

Application: Deriving Idempotents for Property Testing

Live Coding: Property Testing FizzBuzz

What is Property Testing?

$$\exists \Longrightarrow \forall$$

What is Property Testing?

$$\exists \implies \forall$$

Unit tests on cartoon steroids.

What is Property Testing?

$\exists \implies \forall$

Unit tests on cartoon steroids.

Take from the source:

QuickCheck
Hypothesis
JSVerify

Property Testing - A closer look at the examples

You don't know what your inputs are.

Property Testing - A closer look at the examples

You don't know what your inputs are.

Your properties are meant to hold over a broad set of inputs, they must be general.

Property Testing - A closer look at the examples

You don't know what your inputs are.

Your properties are meant to hold over a broad set of inputs, they must be general.

How do you make meaningful assertions without re-implementing the code under test?

Property Testing - A closer look at the examples

You don't know what your inputs are.

Your properties are meant to hold over a broad set of inputs, they must be general.

How do you make meaningful assertions without re-implementing the code under test?

Revisit:

QuickCheck
Hypothesis
JSVerify

Table of Contents

Introduction to Property Testing

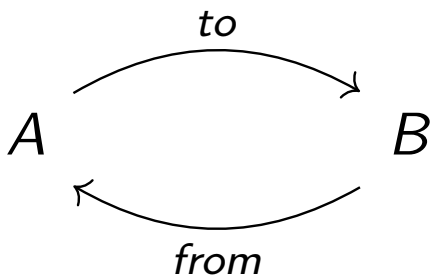
Isomorphisms: Ideal Property Testing Candidates

Idempotents: Making Property Testing Practical

Application: Deriving Idempotents for Property Testing

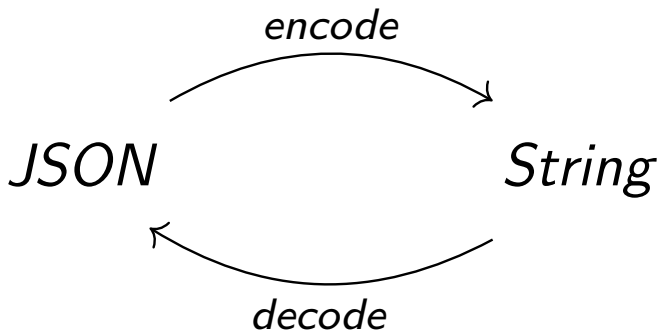
Live Coding: Property Testing FizzBuzz

Isomorphism Defined



$$\text{from}(\text{to}(\text{A})) = \mathit{from} \circ \mathit{to} = 1_A$$
$$\text{to}(\text{from}(\text{B})) = \mathit{to} \circ \mathit{from} = 1_B$$

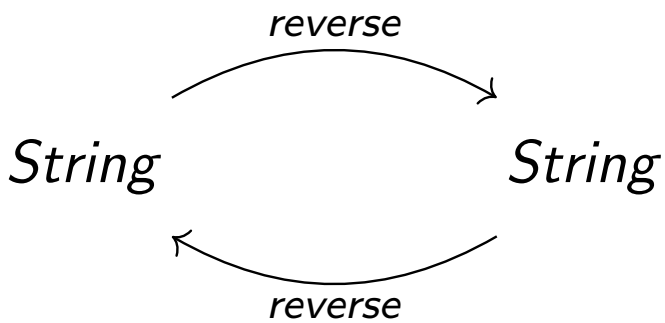
Isomorphism Example: encode $\iff$ decode



$$\text{decode}(\text{encode}(\text{JSON})) = \mathit{decode} \circ \mathit{encode} = 1_{\mathit{JSON}}$$

$$\text{encode}(\text{decode}(\text{String})) = \mathit{encode} \circ \mathit{decode} = 1_{\mathit{String}}$$

Isomorphism Example: reverse $\Longleftrightarrow$ reverse



$$\text{reverse}(\text{reverse}(\text{String})) = \mathit{reverse} \circ \mathit{reverse} = 1_{\mathit{String}}$$

$$\text{reverse}(\text{reverse}(\text{String})) = \mathit{reverse} \circ \mathit{reverse} = 1_{\mathit{String}}$$

Table of Contents

Introduction to Property Testing

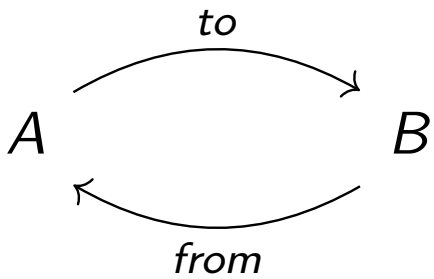
Isomorphisms: Ideal Property Testing Candidates

Idempotents: Making Property Testing Practical

Application: Deriving Idempotents for Property Testing

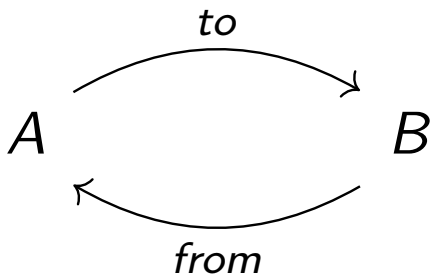
Live Coding: Property Testing FizzBuzz

Idempotent Defined



$$\text{to}(\text{from}(\text{B})) = to \circ from = 1_B$$

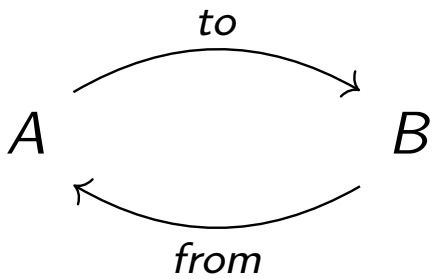
Idempotent Defined



$$\text{to}(\text{from}(\text{B})) = \mathit{to} \circ \mathit{from} = 1_B$$

$$\text{from}(\text{to}(\text{from}(\text{to}(\text{A})))) =$$

Idempotent Defined

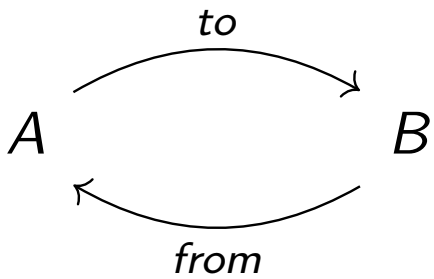


$$\text{to}(\text{from}(\text{B})) = \mathit{to} \circ \mathit{from} = 1_B$$

$$\text{from}(\text{to}(\text{from}(\text{to}(\text{A})))) =$$

$$\mathit{from} \circ \mathit{to} \circ \mathit{from} \circ \mathit{to} =$$

Idempotent Defined



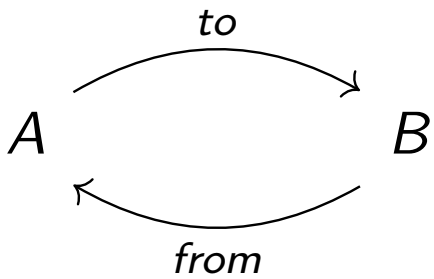
$$\text{to}(\text{from}(\text{B})) = to \circ from = 1_B$$

$$\text{from}(\text{to}(\text{from}(\text{to}(\text{A})))) =$$

$$from \circ to \circ from \circ to =$$

$$from \circ (to \circ from) \circ to =$$

Idempotent Defined



$$\text{to}(\text{from}(\text{B})) = to \circ from = 1_B$$

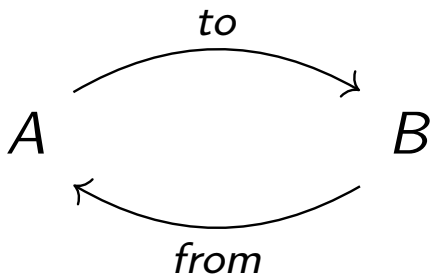
$$\text{from}(\text{to}(\text{from}(\text{to}(\text{A})))) =$$

$$from \circ to \circ from \circ to =$$

$$from \circ (to \circ from) \circ to =$$

$$from \circ 1_B \circ to =$$

Idempotent Defined



$$\text{to}(\text{from}(\text{B})) = to \circ from = 1_B$$

$$\text{from}(\text{to}(\text{from}(\text{to}(\text{A})))) =$$

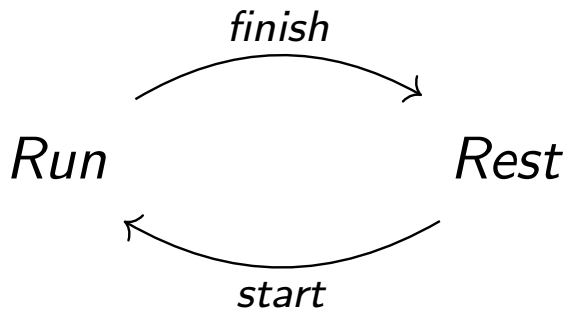
$$from \circ to \circ from \circ to =$$

$$from \circ (to \circ from) \circ to =$$

$$from \circ 1_B \circ to =$$

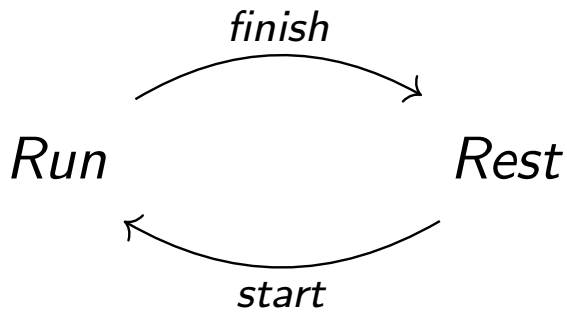
$$from \circ to$$

Idempotent Example: Run $\hookrightarrow$ Rest



$$\text{start}(\text{finish}(\text{Run})) = \mathit{start} \circ \mathit{finish} = 1_{\mathit{Run}}$$

Idempotent Example: Run $\hookrightarrow$ Rest



$$\text{start}(\text{finish}(\text{Run})) = \mathit{start} \circ \mathit{finish} = 1_{\mathit{Run}}$$

$$\text{finish}(\text{start}(\text{finish}(\text{start}(\text{Rest})))) =$$

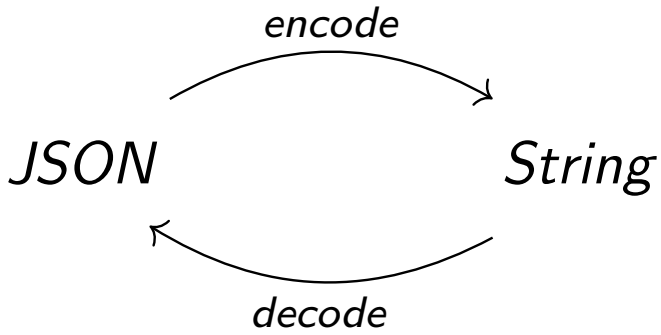
$$\mathit{finish} \circ \mathit{start} \circ \mathit{finish} \circ \mathit{start} =$$

$$\mathit{finish} \circ (\mathit{start} \circ \mathit{finish}) \circ \mathit{start} =$$

$$\mathit{finish} \circ (1_{\mathit{Run}}) \circ \mathit{start} =$$

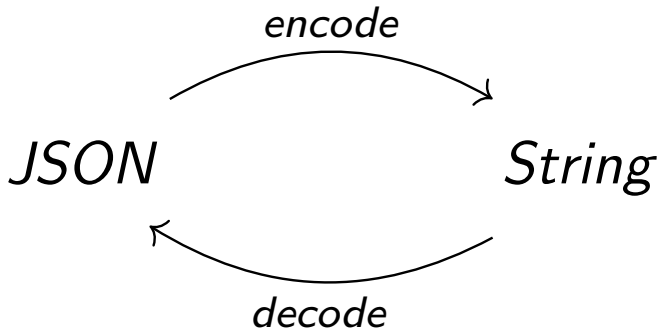
$$\mathit{finish} \circ \mathit{start}$$

Idempotent Example: JSON $\hookrightarrow$ String



$$\text{decode}(\text{encode}(\text{JSON})) = \mathit{decode} \circ \mathit{encode} = 1_{\mathit{JSON}}$$

Idempotent Example: JSON $\hookrightarrow$ String



$$\text{decode}(\text{encode}(\text{JSON})) = \mathit{decode} \circ \mathit{encode} = 1_{\mathit{JSON}}$$

$$\text{encode}(\text{decode}(\text{encode}(\text{decode}(\text{String})))) =$$

$$\mathit{encode} \circ \mathit{decode} \circ \mathit{encode} \circ \mathit{decode} =$$

$$\mathit{encode} \circ (\mathit{decode} \circ \mathit{encode}) \circ \mathit{decode} =$$

$$\mathit{encode} \circ (1_{\mathit{JSON}}) \circ \mathit{decode} =$$

$$\mathit{encode} \circ \mathit{decode}$$

Table of Contents

Introduction to Property Testing

Isomorphisms: Ideal Property Testing Candidates

Idempotents: Making Property Testing Practical

Application: Deriving Idempotents for Property Testing

Live Coding: Property Testing FizzBuzz

Making Properties Easy

We know that properties are easy and effective when we have an isomorphism.

```
from hypothesis import given
from hypothesis.strategies import text


@given(text())
def test_decode_inverts_encode(s):
    assert decode(encode(s)) == s
```

Making Properties Easy

We know that properties are easy and effective when we have an isomorphism.

```
from hypothesis import given
from hypothesis.strategies import text


@given(text())
def test_decode_inverts_encode(s):
    assert decode(encode(s)) == s
```

What about when we don't have an isomorphism?

Making Properties Easy

We know that properties are easy and effective when we have an isomorphism.

```
from hypothesis import given
from hypothesis.strategies import text


@given(text())
def test_decode_inverts_encode(s):
    assert decode(encode(s)) == s
```

What about when we don't have an isomorphism?

Can we *find* an isomorphism?

Finding an isomorphism

FizzBuzz does not belong to an isomorphism.

```
from typing import List
def fizzbuzz(nums: List[int]) -> List[str]:
  res: List[str] = []
  for num in nums:
    s = ""
    if num % 3:
      s += "Fizz"
    if num % 5:
      s += "Buzz"
    if s == "":
      s = str(num)
    res.append(s)
  return res
```

Finding an isomorphism

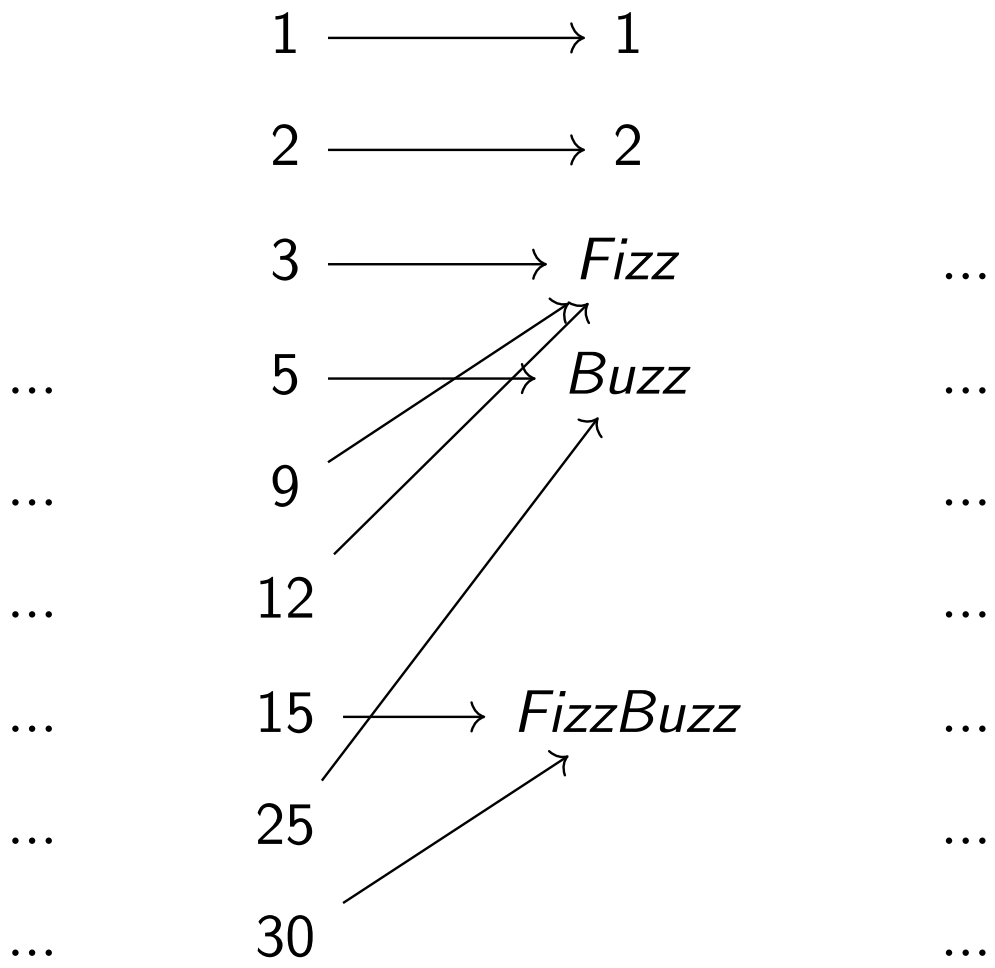
FizzBuzz does not belong to an isomorphism.

```
from typing import List
def fizzbuzz(nums: List[int]) -> List[str]:
  res: List[str] = []
  for num in nums:
    s = ""
    if num % 3:
      s += "Fizz"
    if num % 5:
      s += "Buzz"
    if s == "":
      s = str(num)
    res.append(s)
  return res
```

What is the closest isomorphism we can find?

It helps to take a different perspective.

FizzBuzz as a Set Function



We cannot have an isomorphism because inputs *collapse* onto outputs.

This prevents construction of an inverse.

FizzBuzz as a Set Function, Partitioned Domain

$$\{1\} \longrightarrow 1$$

$$\{2\} \longrightarrow 2$$

$$\{3, 6, 9, 12, ...\} \longrightarrow \textit{Fizz}$$

$$\{4\} \longrightarrow 4$$

$$\{5, 10, 20, 25, ...\} \longrightarrow \textit{Buzz}$$

$$\{7\} \longrightarrow 7 \qquad ...$$

$$... \qquad \{15, 30, ...\} \longrightarrow \textit{FizzBuzz}$$

$$\{16\} \longrightarrow 16$$

We have an isomorphism, can we fix the input type?

FizzBuzz as a Set Function, Partitioned Domain

$$\{1\} \longrightarrow 1$$

$$\{2\} \longrightarrow 2$$

$$\{3, 6, 9, 12, ...\} \longrightarrow \textit{Fizz}$$

$$\{4\} \longrightarrow 4$$

$$\{5, 10, 20, 25, ...\} \longrightarrow \textit{Buzz}$$

$$\{7\} \longrightarrow 7 \qquad ...$$

$$... \qquad \{15, 30, ...\} \longrightarrow \textit{FizzBuzz}$$

$$\{16\} \longrightarrow 16$$

We have an isomorphism, can we fix the input type?

With an idempotent.

FizzBuzz^{-1} as a Set Function, Idempotent

We just pick one value from each input set.

$$
\begin{array}{lll}
 & 1 \longleftarrow 1 & \\
 & 2 \longleftarrow 2 & \\
 & 3 \longleftarrow \textit{Fizz} & \\
 & 4 \longleftarrow 4 & \\
 & 5 \longleftarrow \textit{Buzz} & \\
 & 7 \longleftarrow 7 & \ldots \\
\ldots & 15 \longleftarrow \textit{FizzBuzz} & \\
 & 16 \longleftarrow 16 &
\end{array}
$$

This can be pre-composed with FizzBuzz to create an identity on the output set.
This means we have an idempotent on the input set.

Table of Contents

Introduction to Property Testing

Isomorphisms: Ideal Property Testing Candidates

Idempotents: Making Property Testing Practical

Application: Deriving Idempotents for Property Testing

Live Coding: Property Testing FizzBuzz