

Property-based-testing

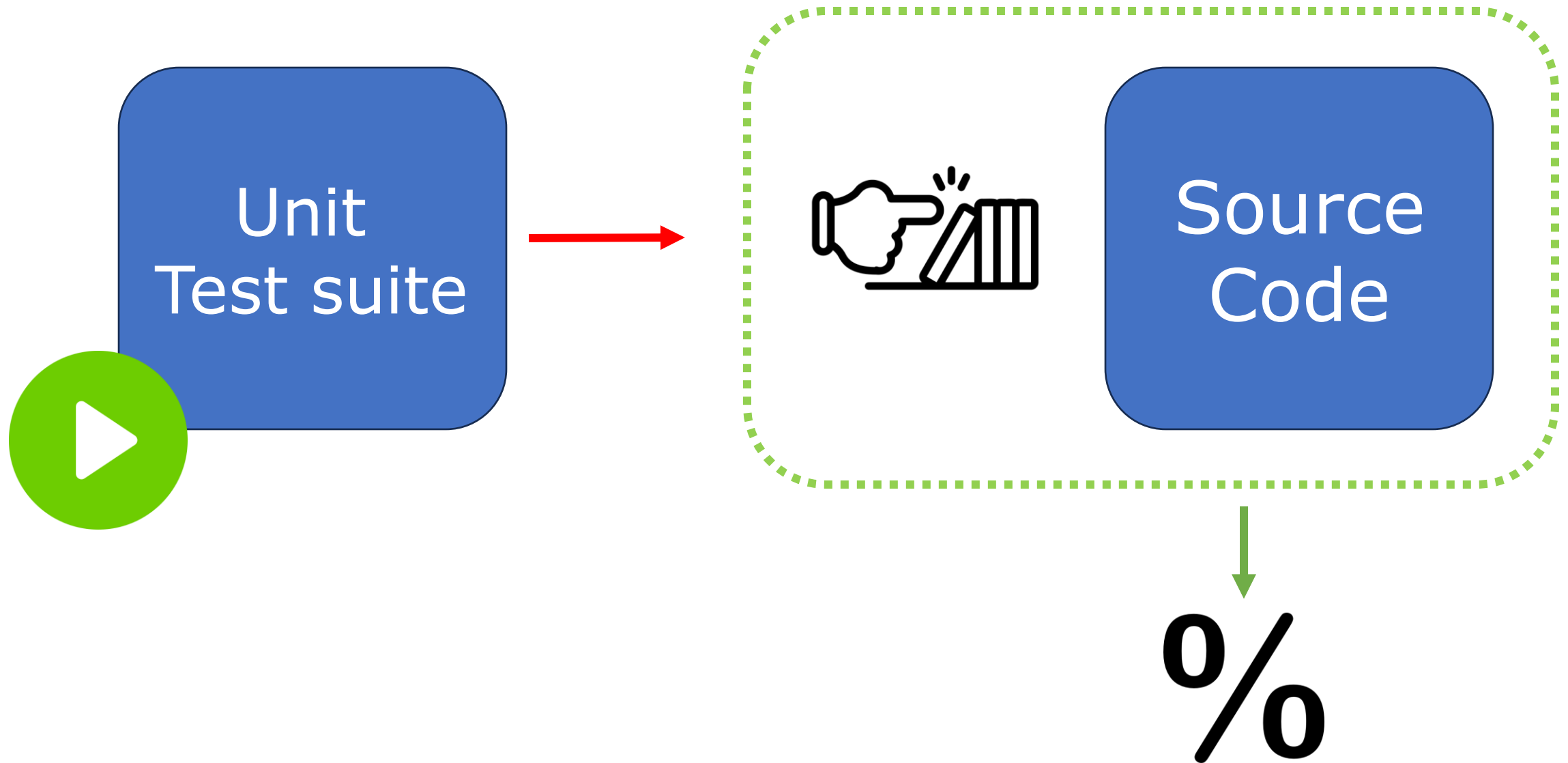
- *Unveiling the truth on test reduction* -

When have you done enough testing?...

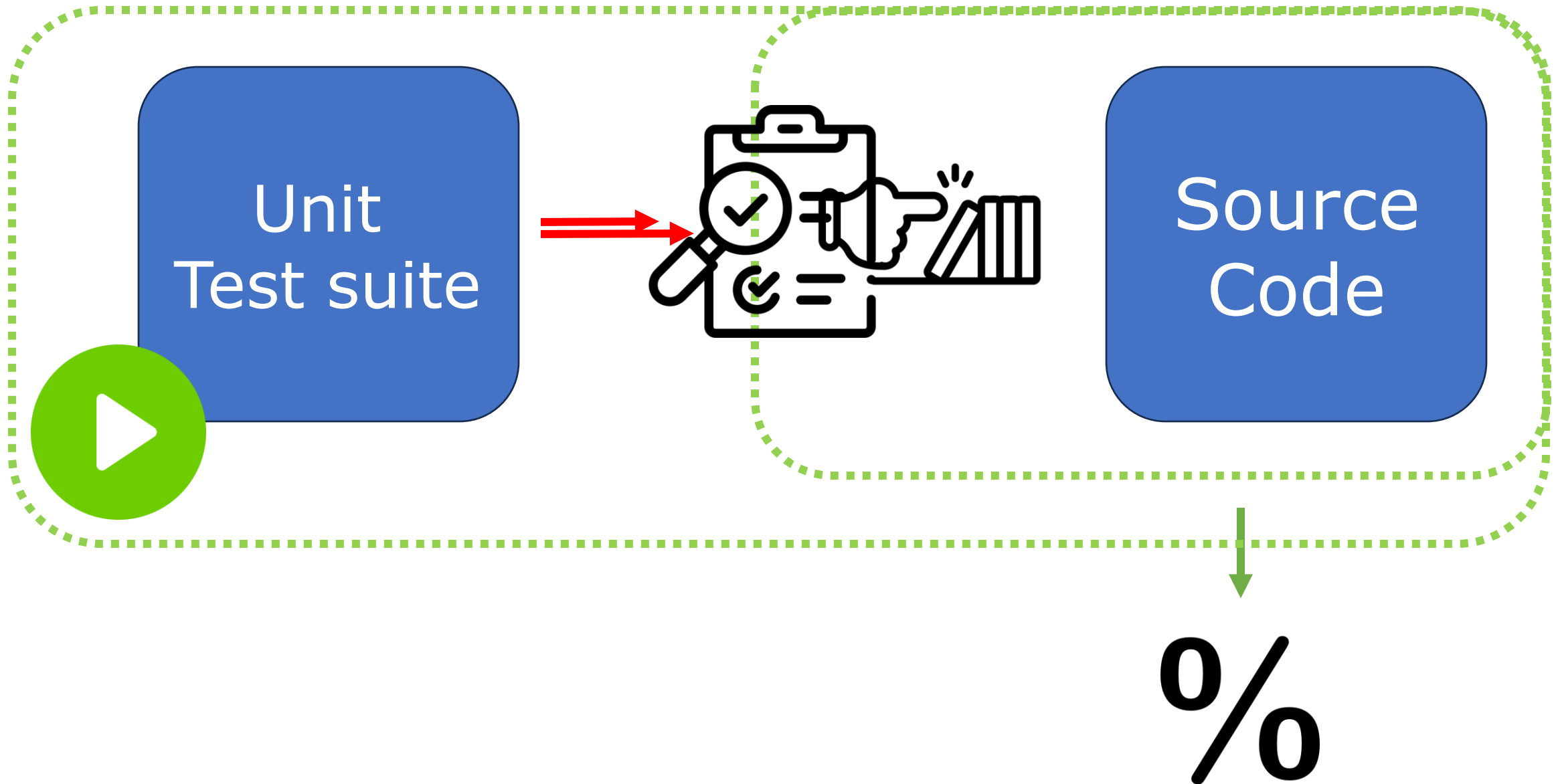
[illegible]

Code Coverage \neq Test Coverage

Code coverage



Test coverage




How to get a high(er) coverage?



Happy path



Different cases



Edge cases



Error Guessing



Happy path



Different cases



Edge cases



Decision table

SIGN UP

Email

Password

SIGN UP

Conditions	Case #1	Case #2	Case #3	Case #4
#1	True	False	False	True
#2	False	True	False	True

ACCOUNT LOGIN

Policy Number

User Name

Password

Login

Conditions	Case #1	Case #2	Case #3	Case #4	Case #5	Case #6	Case #7	Case #8
#1	True	True	True	True	False	False	False	False
#2	False	True	False	True	False	True	False	True
#3	False	False	True	True	False	True	True	False

$$X = 2^N$$

N	X
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	...

How many conditions can you define?



```
from SUT import add_numbers
```

```
def test_simple_sum():
```

```
    result = add_numbers([1,2])
```

```
    assert result == 3
```

SPECIFIC

`[1, 2] == 3`



```
from SUT import add_numbers

def test_simple_sum():
    result = add_numbers([1,2])
    assert result == 3
```

GENERIC --> DESCRIBING

“list of integers”

“integer”

“sum(*list of integers*)”

```
from hypothesis import given
import hypothesis.strategies as st
from SUT import add_numbers

@given(st.lists(st.integers))
def test_simple_sum(example):
    result = add_numbers(example)
    assert isinstance(result, int)
    assert added_nums == sum(example)
```

Build a dragon

1. Array of integers
2. Minimum of 2 items

1. Type (output)
2. Content input VS content output
3. All items are 'lesser then or eaqual to' the next item

```
from hypothesis import given
from collections import Counter
import hypothesis.strategies as st
from SUT import sort_algo

@given(st.lists(st.integers, min_size=2))
def test_simple_sum(l):
    s = sort_algo(l)

    print(f'list: {l}')

    assert isinstance(s, list)
    assert Counter(l) == Counter(s)
    assert all(x <= y for x, y in zip(s, s[1:]))
```

Shrinking

```
===== test session starts =====
platform win32 -- Python 3.9.2, pytest-6.2.2, py-1.10.0, pluggy-0.13.1
rootdir: C:\Users\mwalid\OneDrive - TestCoders B.V\Documents\codechallenges\sudoku
plugins: hypothesis-6.12.0
collected 1 item

tests.py list: [0, 0]
list: [0, 0]
list: [-16, 1446194188, -5668, 30468, -21342, -13400268685896341822404318779962026388, 10]
list: [-7005, -92, 3656995437321075716]
list: [-1971873409119115395, 3656995437321075716]
list: [-281872320070016, -281872319939201, 16188]
list: [-281872320070016, -281872319939201, -385, -105, 3768, 841, -8299584243642436951, -103, -11269, 16316, -18, 2153, -18745, -27367, 31966, 119, 19224]
list: [-382719829154503565, 23044, 26937, 112, 1781]
list: [17357, -10616, -14567, 137509421272331155061951298633472455864, -23629]
list: [17357, -10616, -14567, 137509421272331155061951298633472455864, -23629]
list: [-382719829154503565, 23044, 26937, 112, 1781]
list: [-382719829154503565, 23044, 26937, 112, 1781]
list: [-382719829154503565, 23044, 129]
list: [4, 129]
list: [-382719829154503565, 23044]
list: [-382719829154503565, 129]
list: [23044, 129]
list: [0, 90]
list: [129, 23044]
list: [23044, 0]
list: [23044, 0]
list: [0, 0]
list: [0, 0]
list: [6660, 0]
list: [2564, 0]
list: [516, 0]
list: [4, 0]
list: [2, 0]
list: [1, 0]
list: [0, 0]
list: [1, 0]
list: [0, 0]
list: [1, 0]
list: [1, 0]
list: [1, -2]
list: [-2, 0]
list: [0, 1]
Falsifying example: test_sort_algo
    l=[1, 0],
)
list: [1, 0]
F

===== FAILURES =====
_____ test_sort_algo _____

    @given(st.lists(st.integers(), min_size=2))
    > def test_sort_algo():

tests.py:7:
-----
l = [1, 0]

    @given(st.lists(st.integers(), min_size=2))
    def test_sort_algo():
        s = sort_algo(l)
        print(f'list: {l}')
        assert isinstance(s, list)
        assert Counter(l) == Counter(s)
    > assert all(x == y for x, y in zip(s, s[1:]))
    E
    E   + where False = all(generator object test_sort_algo.<locals>.<genexpr> at 0x000001C913040200)

tests.py:12: AssertionError
===== Hypothesis =====
Falsifying example: test_sort_algo
    l=[1, 0],
)
list: [1, 0]
F

===== short test summary info =====
FAILED tests.py::test_sort_algo - assert False
===== 1 failed in 0.62s =====
```

```
tests.py list: [0, 0]
list: [0, 0]
list: [-16, 1446194188, -5668, 30468, -21342, -13400268685896341822404318779962026388, 10]
list: [-7005, -92, 3656995437321075716]
list: [-1971873409119115395, 3656995437321075716]
list: [-281872320070016, -281872319939201, 16188]
list: [-281872320070016, -281872319939201, -385, -105, 3768, 841, -8299584243642436951, -103, -11269, 16316, -18, 2153, -18745, -27367, 31966, 119, 19224]
list: [-382719829154503565, 23044, 26937, 112, 1781]
list: [17357, -10616, -14567, 137509421272331155061951298633472455864, -23629]
list: [17357, -10616, -14567, 137509421272331155061951298633472455864, -23629]
list: [-382719829154503565, 23044, 26937, 112, 1781]
list: [-382719829154503565, 23044, 26937, 112, 1781]
list: [-382719829154503565, 23044, 129]
list: [4, 129]
list: [-382719829154503565, 23044]
list: [-382719829154503565, 129]
list: [23044, 129]
list: [0, 90]
list: [129, 23044]
list: [23044, 0]
list: [23044, 0]
list: [0, 0]
list: [0, 0]
list: [6660, 0]
list: [2564, 0]
list: [516, 0]
list: [4, 0]
list: [2, 0]
list: [1, 0]
list: [0, 0]
list: [1, 0]
list: [0, 0]
list: [1, 0]
list: [1, 0]
list: [1, -2]
list: [-2, 0]
list: [0, 1]
Falsifying example: test_sort_algo
    l=[1, 0],
)
list: [1, 0]
F
```


Shrinking

$[-38271929154503565, 23044, 26937, 112, 1781]$



$[-38271929154503565, 23044, 129]$



$[23044, 129]$



$[23044, 0]$



!!! >>> $[1, 0]$ <<< !!!

Trace view

```
===== test session starts =====
platform win32 -- Python 3.9.2, pytest-6.2.2, py-1.11.0, pluggy-0.13.1
rootdir: C:\Users\mwal\OneDrive - TestCoders B.V\Documents\codechallenges\sudoku
plugins: hypothesis-6.12.0
collected 1 item

tests.py list: [0, 0]
list: [0, 0]
list: [-10, 1446194100, -5600, 30460, -21342, -1340020005096341822404310779962026308, 10]
list: [-7005, -82, 3056995437321075716]
list: [-1971873409119115305, 3056995437321075716]
list: [-201872320070016, -201872319939202, 10100]
list: [-201872320070016, -201872319939202, -305, -105, 3700, 041, -8299504243642436951, -103, -11269, 16310,
-10, 2153, -10745, -27207, 32060, 119, 10224]
list: [-302720029154503565, 23044, 20037, 112, 1701]
list: [17357, -10010, -14567, 137509421272331155001951290033472455004, -23629]
list: [17357, -10010, -14567, 137509421272331155001951290033472455004, -23629]
list: [-302720029154503565, 23044, 20037, 112, 1701]
list: [-302720029154503565, 23044, 20037, 112, 1701]
list: [-302720029154503565, 23044, 129]
list: [4, 129]
list: [-302720029154503565, 23044]
list: [-302720029154503565, 129]
list: [23044, 129]
list: [0, 90]
list: [129, 23044]
list: [23044, 0]
list: [23044, 0]
list: [0, 0]
list: [0, 0]
list: [0000, 0]
list: [2564, 0]
list: [516, 0]
list: [4, 0]
list: [2, 0]
list: [1, 0]
list: [0, 0]
list: [1, 0]
list: [1, 0]
list: [0, 0]
list: [1, 0]
list: [1, 0]
list: [1, -2]
list: [-2, 0]
list: [0, 1]
Falsifying example: test_sort_algo(
  l=[1, 0],
)
list: [1, 0]
F

===== FAILURES =====
_____ test_sort_algo _____

    @given(st.lists(st.integers(), min_size=2))
    def test_sort_algo(l):
>
E
E + where False = all(<generator object test_sort_algo.<locals>.<genexpr> at 0x0000013040000000>)

tests.py:12: AssertionError

----- Hypothesis -----
Falsifying example: test_sort_algo(
  l=[1, 0],
)

===== short test summary info =====
FAILED tests.py::test_sort_algo - assert False

===== 1 failed in 0.62s =====
```

```
l = [1, 0]

@given(st.lists(st.integers(), min_size=2))
def test_sort_algo(l):
    s = sort_algo(l)
    print(f'list: {l}')
    assert isinstance(s, list)
    assert Counter(l) == Counter(s)
    assert all(x <= y for x, y in zip(s, s[1:]))
    assert False
    + where False = all(<generator object test_sort_algo.<locals>.<genexpr> at 0x0000013040000000>)

tests.py:12: AssertionError

----- Hypothesis -----
Falsifying example: test_sort_algo(
  l=[1, 0],
)

===== short test summary info =====
FAILED tests.py::test_sort_algo - assert False

===== 1 failed in 0.62s =====
```

Build **custom** strategies

```
class Dog:

    def __init__(self, name, breed):
        self.name = name
        self.breed = breed

    def __str__(self) -> str:
        return f'{self.name}({self.breed})'
```

```
st_dogs = st.builds(Dog,
                    name=st.text(),
                    breed=st.sampled_from(MY_BREEDS))

@given(st_dogs)
def test_animals(animal):
    print(animal)
```



```
@(Toller)
(Toller)
@(Toller)
[](Labrador)
ýU[] (Pointer)
[]&wÃ½ ½igã·s(Labrador)
[] (Pointer)
¿JLl¿gDê$»TSÔð»½[]I(Griffon)
[](Pointer)
·¹$[]ò· ør[] $[](Griffon)
¶l(Labrador)
¶(Toller)
¶→89[]#X ▼^lĩh(Pointer)
    [](Pointer)
        (Pointer)
        Z!!i(Labrador)
        Ø\!!i(Labrador)
    dZ!!i(Labrador)
[]Qr Cànã$øÃ½i@[]Ãñ(Pointer)
[]eĒã*(Labrador)
[]«Ý
    Æ(Toller)
[]«Ý
    Æ(Toller)
&/ (Pointer)
!!dWY[]ĩô, (Pointer)
»[] (Pointer)
£[]·ãVã$»[]°[]Ü[]bÔ»¹ ▼[](Labrador)
[](Toller)
wp7v(Toller)[]m=[]£Ý(Labrador)
>[](Labrador)
    +i(Griffon)
&Ôr♦IøI½+£ð(Labrador)
ÔÔ(Labrador)
Ô(Toller)
ÔÔ[]eõqpW ▼4[]ôçĒ(Toller)
[]![]@n[]Ã(Griffon)
```

Omg, that is sooo hypothesis!..

```
[](Pointer)
·¹$[]ò· ør[] $[](Griffon)
¶l(Labrador)
¶(Toller)
¶→89[]#X ▼^lĩh(Pointer)
    [](Pointer)
        (Pointer)
        Z!!i(Labrador)
        Ø\!!i(Labrador)
    dZ!!i(Labrador)
[]Qr Cànã$øÃ½i@[]Ãñ(Pointer)
[]eĒã*(Labrador)
[]«Ý
    Æ(Toller)
[]«Ý
    Æ(Toller)
&/ (Pointer)
!!dWY[]ĩô, (Pointer)
```

Test Reduction

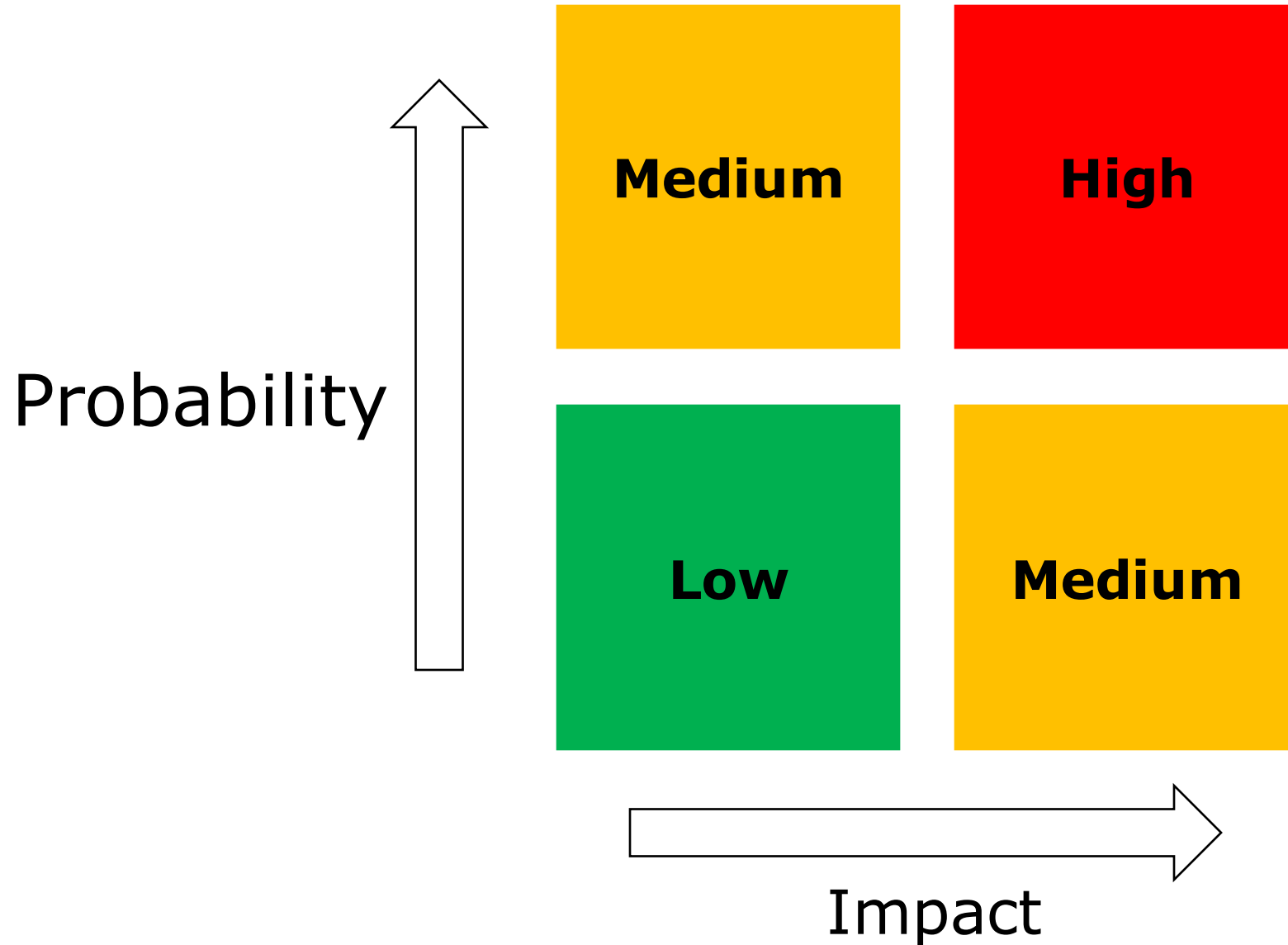


PBT Characteristics

- Increase of coverage
- Possibly decrease test code (not test cases)

**When have you done enough
testing?...**

RISK BASED TESTING



“Build a dragon”