Mutation Tests

https://www.filmandtvnow.com/paramount-announce-teenage-mutant-ninja-turtles-sequel-2016/

Luzern, 21.03.2022

Mike Mugglin

Content

What is Mutation Testing ?

- Mutation Testing is a type of software testing in which certain statements of the source code are changed/mutated to check if the test cases are able to find errors in source code. The goal of Mutation Testing is ensuring the quality of test cases in terms of robustness that it should fail the mutated source code
- Mutant tests are there to ensure you will catch unintended results of future modifications
- It’s a way to evaluate the quality of your existing tests

How does the Mutant Testing work?

- A Mutation Test change something in the source code. This is called a «mutant»
- The applied change is called «mutator»
- The existing tests are then executed against the mutant
- If the test against the mutant fails => the mutant is killed
  If the test against the mutant passes => the mutant survived
- The percentage of the killed mutants is measured and shows the quality of the existing tests

Coverage vs mutation

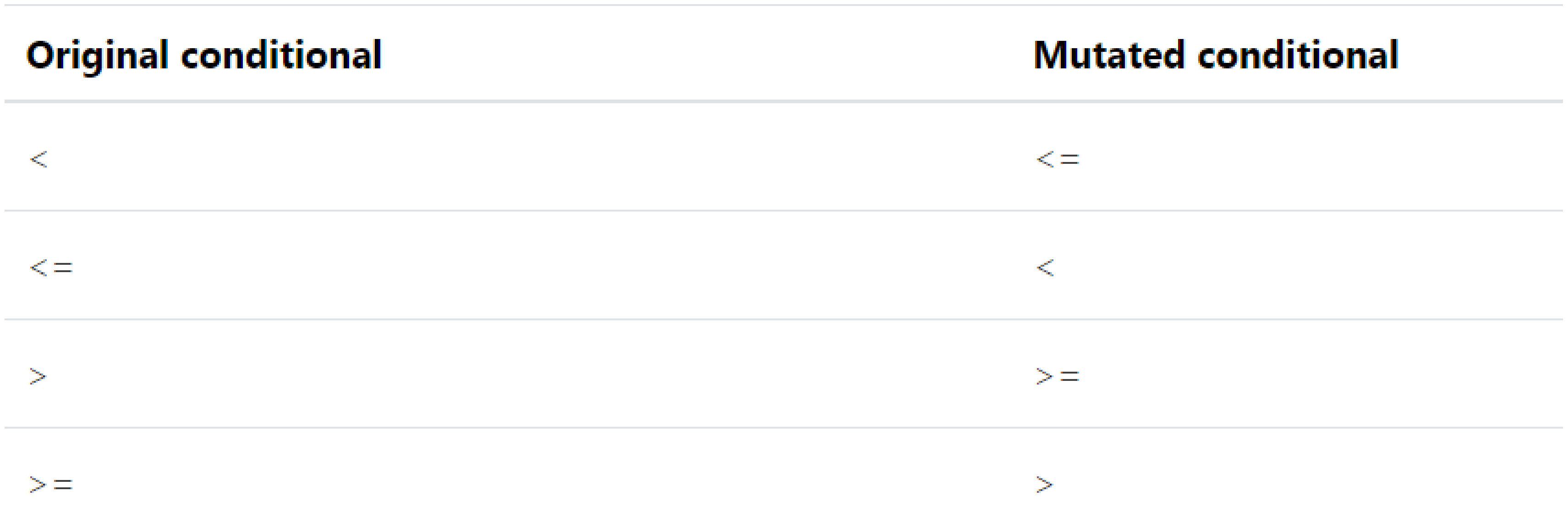
Coverage:
What percentage of the code that is executed by your tests. Usually this is not enough to determine test quality
=> *Let you easily find untested lines of code. Makes no statement about the quality of the test*

Mutation Test:
What percentage of the mutants are killed. It shows if your tests are able to detect subtle errors
=> *Gives you a pretty good idea of the quality of your tests*

Mutators

For every mutation, a mutation operator is applied to a piece of code. Mutation operator performs a slight modification on code in order to change its behavior.
Mutations are usually introduced to code one by one rather than multiple of them at the same time.

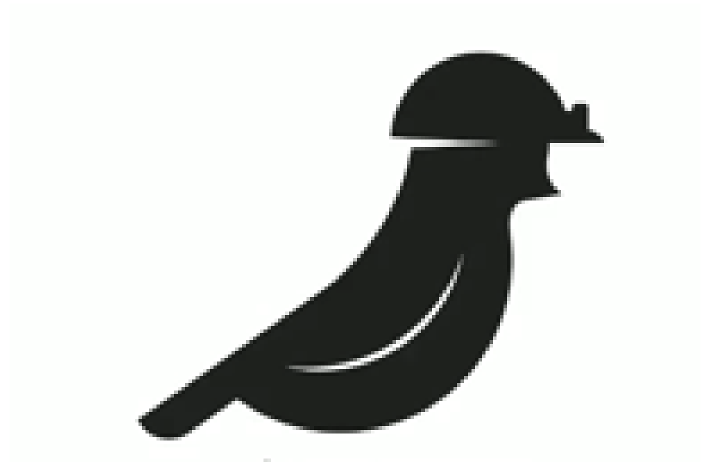
Example for a conditional mutator:



Original conditional	Mutated conditional
<	<=
<=	<
>	>=
>=	>

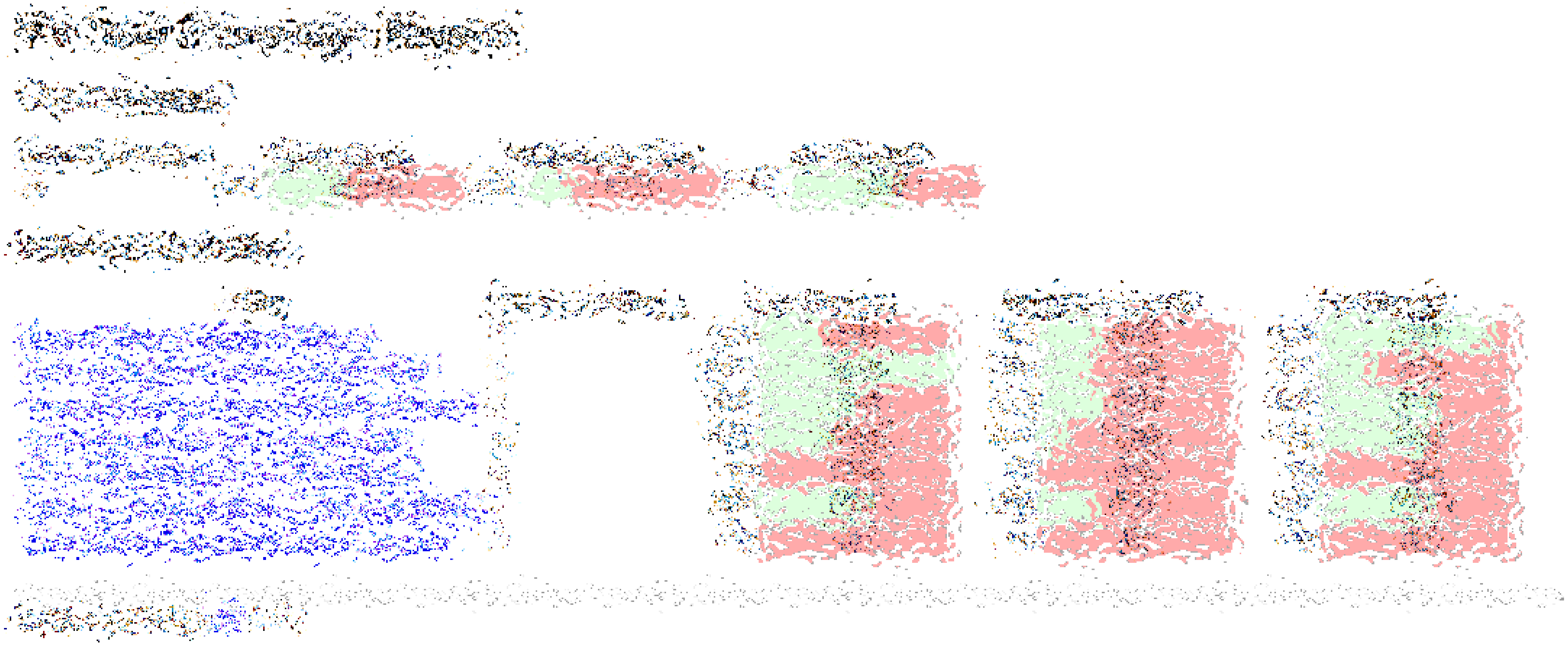
Pit test framework

- Pit test is a Mutation Test framework
- Supports Java & Maven
- Can be highly customized
- Open-source
- The PIT test framework runs your unit tests against automatically modified versions of your application code
- https://pitest.org/



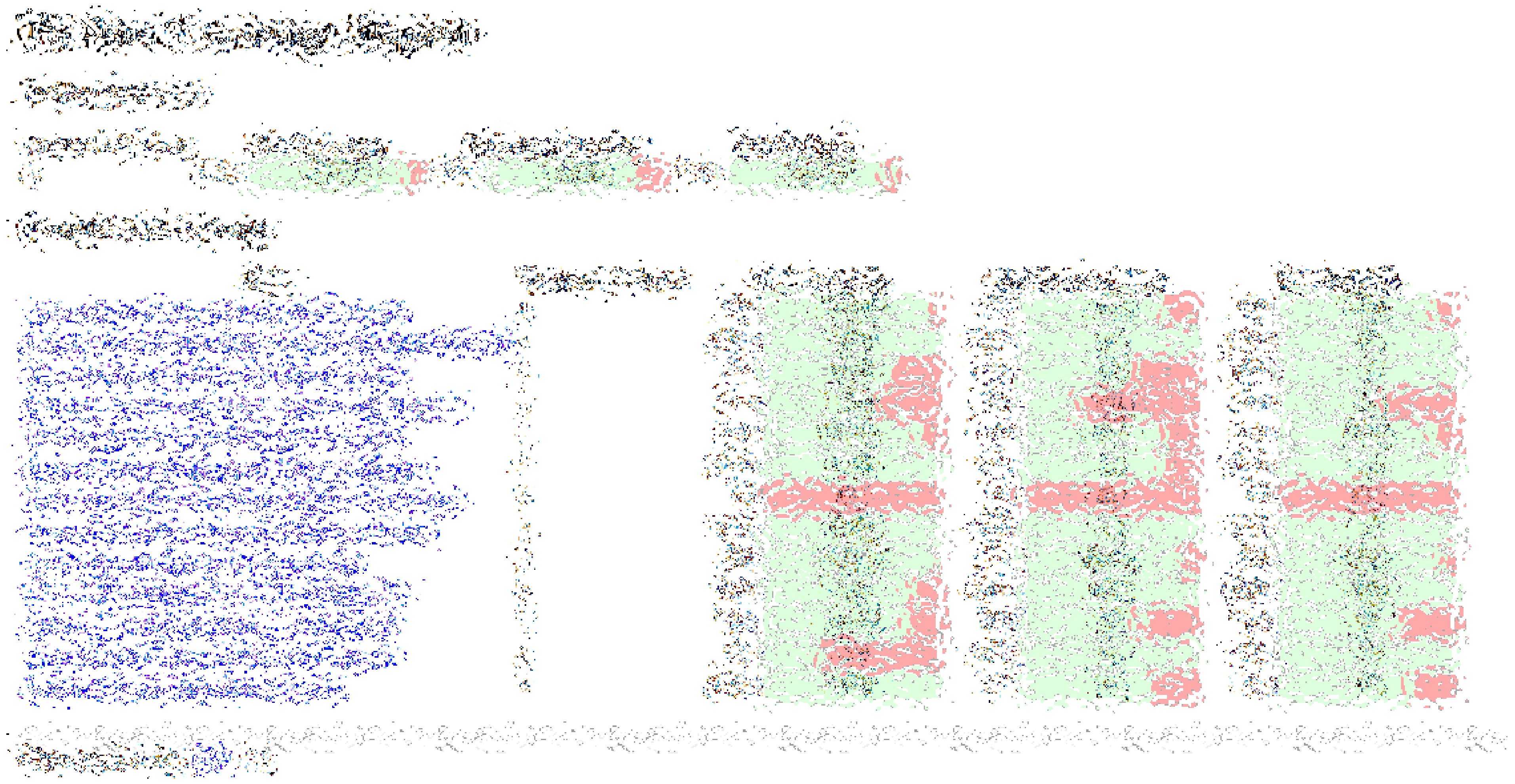
Pit test in the CSS codebase

OLDER CODE



Pit test in the CSS codebase

RECENTLY WRITTEN CODE



Pros vs Cons

Pros:

- Mutation Tests are very, very useful to measure the quality of your tests (Only code coverage ist not enough in my opinion)
- Mutation Tests can help discover possible test leaks

Cons:

- In a larger codebase the number of the generated mutants is extremely high and it takes a long time to generate these mutants and to perform all tests

Conclusion

- Code coverage is a deceptive metric to rely on. For me the Mutation Tests are a better metric
- The Mutation Test can be used to improve the quality of your tests
- But it doesn’t guarantee a bug-free system
- It needs extra effort for a large codebase
- There are also uninteresting failure in your tests which you have to identify first. This takes quite a lot of time !

- **Our newly written code in production has a very good test quality !!!**

Thank you

References:
https://medium.com/swlh/mutation-testing-the-most-comprehensive-way-to-test-your-software-674f645bfc21
https://en.wikipedia.org/wiki/Mutation_testing
https://www.guru99.com/mutation-testing.html#1
https://pitest.org/
https://pedrorijo.com/blog/intro-mutation/