Conceptual Mutation Testing for Student Programming Misconceptions

Siddhartha Prasad
Ben Greenman
Tim Nelson
Shriram Krishnamurthi
Understanding CS Problems
Understanding CS Problems

Q. median
Understanding CS Problems

Q. median

feedback?
Understanding CS Problems

Q. median

feedback?
Understanding CS Problems

Q. median
Understanding CS Problems

Q. median
How to communicate?
How to communicate?

Test Cases
Q. median
Q. median

median [1] is 1
median [1, 2, 3] is 3
median [3, 3, 3] is 3
Q. median

median [1] is 1
median [1, 2, 3] is 3
median [3, 3, 3] is 3

What's wrong with these tests?
```python
include my-gdrive("median-code.arr")
# DO NOT CHANGE ANYTHING ABOVE THIS LINE

check:
median([list: 1]) is 1  
median([list: 1, 2, 3]) is 3  
median([list: 3, 3, 3, 3]) is 3

da
```
```plaintext
1 include my-gdrive("median-code.arr")
2 # DO NOT CHANGE ANYTHING ABOVE THIS LINE
3
4 check:
5 median([list: 1]) is 1
6 median([list: 1, 2, 3]) is 3
7 median([list: 3, 3, 3, 3]) is 3
8
9 end
```
```python
include my-gdrive("median-code.arr")

# DO NOT CHANGE ANYTHING ABOVE THIS LINE

class check:
    median([list: 1]) is 1
    median([list: 1, 2, 3]) is 2
    median([list: 3, 3, 3, 3]) is 3

    # Shows that Median is not Mean
    median([list: 1, 1, 3]) is 1

    # Shows that Median is not Mode
    median([list: 1, 1, 3, 4, 4]) is 3

end
```
include my-gdrive("median-code.arr")
#
# DO NOT CHANGE ANYTHING ABOVE THIS LINE

check:
median([list: 1]) is 1
median([list: 1, 2, 3]) is 2
median([list: 3, 3, 3, 3]) is 3

# Shows that Median is not Mean
median([list: 1, 1, 3]) is 1

# Shows that Median is not Mode
median([list: 1, 1, 3, 4, 4]) is 3

These tests are valid and consistent with the assignment handout. They caught 2 of 4 sample buggy programs. Add more test cases to improve this test suite's thoroughness.
What's **wrong** with these tests?
Q. median
Q. median

Tests must distinguish:

- mean
- median vs. mode
- middle ...

Valid & Thorough
How to check thoroughness?

Buggy solutions

*(mutation testing)*
RQ. How to design buggies?
RQ. How to design buggies?

Need to *discover* misconceptions
Prior Work:
Expert-Driven
Today, a recipe:
Buggies from Data
1. Design problem
1. Design problem

Running example: Doc Diff

docdiff ['a'] ['A'] is 1

docdiff ['one', 'two'] ['one'] is 1/2

docdiff ['hello'] ['world'] is 0
2. Collect invalid tests
2. Collect invalid tests
2. Collect invalid tests

Doc Diff ==> 1,500 invalids in ~1 week
3. Cluster tests by feature vector
3. Cluster tests by feature vector
Feature vectors $\Leftarrow\Rightarrow$ problem characteristics
Feature vectors $\leqslant$ problem characteristics

- Case-insensitive
- Words may repeat
- Diff may be a fraction
... [14 in total]
5. Sort clusters
5. Sort clusters
5. Sort clusters

<table>
<thead>
<tr>
<th></th>
<th>[typos]</th>
<th>diff is always zero</th>
<th>case sensitive</th>
<th>normalize by wrong mag.</th>
<th>normalize by wrong vector</th>
<th>[unknown]</th>
</tr>
</thead>
<tbody>
<tr>
<td>800</td>
<td><img src="#" alt="Red squares" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>68</td>
<td><img src="#" alt="Red squares" /> + <img src="#" alt="Green squares" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>66</td>
<td><img src="#" alt="Red squares" /> + <img src="#" alt="Green squares" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>52</td>
<td><img src="#" alt="Red squares" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>49</td>
<td><img src="#" alt="Red squares" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>48</td>
<td><img src="#" alt="Red squares" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
5. Sort clusters

6. Make buggies

- 800: [typos]
- 68: diff is always zero
- 66: case sensitive
- 52: normalize by wrong mag.
- 49: normalize by wrong vector
- 48: [unknown]
Too many buggies ==> overwhelming!
6. Make buggies

* Focus on 1-2
* Favor narrow characteristics
* Maximize subproblem coverage
6. Make buggies

* Focus on 1-2
* Favor narrow characteristics
* Maximize subproblem coverage
6. Make buggies

* Focus on 1-2

* Favor narrow characteristics

<table>
<thead>
<tr>
<th></th>
<th>typos</th>
<th>diff is always zero</th>
<th>case sensitive</th>
<th>case sensitive is more narrow</th>
<th>normalize by wrong mag.</th>
<th>normalize by wrong vector</th>
<th>[unknown]</th>
</tr>
</thead>
<tbody>
<tr>
<td>800</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>68</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>66</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>52</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>49</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>48</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Evaluation

2020: 3 tasks

2021

2022
Evaluation

2020
- 3 tasks

2021

2022

Evaluation

2020
- 3 tasks

2021

2022

2020, 2021 ==> test
2022 ==> deploy
% explainable invalid tests

explainable = 1- or 2-

old buggies

old buggies

new buggies
% explainable invalid tests
explainable = 1- or 2-

New buggies explain more errors

old buggies
old buggies
new buggies
### High Effect Sizes for 2022

<table>
<thead>
<tr>
<th>Matchup</th>
<th>Problem</th>
<th>95% CI</th>
<th>p value</th>
</tr>
</thead>
<tbody>
<tr>
<td>2022 vs 2020</td>
<td>DocDiff</td>
<td>[-0.75, -0.57]</td>
<td>1.35E-29</td>
</tr>
<tr>
<td></td>
<td>Nile</td>
<td>[-0.55, -0.26]</td>
<td>9.07E-14</td>
</tr>
<tr>
<td></td>
<td>FileSys</td>
<td>[-0.35, -0.21]</td>
<td>2.35E-10</td>
</tr>
<tr>
<td>2022 vs 2021</td>
<td>DocDiff</td>
<td>[-0.70, -0.51]</td>
<td>6.87E-29</td>
</tr>
<tr>
<td></td>
<td>Nile</td>
<td>[-0.27, -0.07]</td>
<td>1.82E-3</td>
</tr>
<tr>
<td></td>
<td>FileSys</td>
<td>[-0.33, -0.19]</td>
<td>2.32E-9</td>
</tr>
<tr>
<td>2021 vs 2020</td>
<td>DocDiff</td>
<td>[-0.07, 0.08]</td>
<td>4.60E-1</td>
</tr>
<tr>
<td></td>
<td>Nile</td>
<td>[-0.39, -0.13]</td>
<td>1.15E-17</td>
</tr>
<tr>
<td></td>
<td>FileSys</td>
<td>[-0.06, 0.03]</td>
<td>2.52E-1</td>
</tr>
</tbody>
</table>
Weeks of Data > Years of Tuning

Promising approach for new problems
Recipe to uncover misconceptions
semi-automatic
Recipe to uncover misconceptions
semi-automatic

Data ==> better teaching
What's next? Hinting
What's next? Hinting

INCORRECT
CONSEQUENTLY, THOROUGHNESS
UNKNOWN

These tests do not match the behavior described by the assignment.

13 overlap([list: "a", "b"], [list: "b"]) is 0

The assignment says:
Overlap must be proportional to the dot product of two vectors.
Deep Goal:

**Rigorous methods for CS Ed research**
Let's talk!
1. design problem
2. identify characteristics
3. collect invalid tests
4. cluster by feature vector
5. analyze top clusters
6. select buggies
Future

Data collection is a bottleneck
~1 semester ramp-up

+70% typos! How to reduce?
D4 / Data Druid
Table 8  Our 2022 chaffs gave 1-m/2-m outcomes significantly more often than prior chaffs. The 2021 vs. 2020 results are similar except for Nile, which used D4 in 2021.

<table>
<thead>
<tr>
<th>Matchup</th>
<th>Assignment</th>
<th>p value</th>
<th>Z score</th>
<th>Effect Size [95% CI] (Cohen’s D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2022 vs 2020</td>
<td>DocDiff</td>
<td>1.35E-29</td>
<td>-11.24</td>
<td>0.66 [-0.75, -0.57]</td>
</tr>
<tr>
<td></td>
<td>Nile</td>
<td>9.07E-14</td>
<td>-7.36</td>
<td>-0.41 [-0.55, -0.26]</td>
</tr>
<tr>
<td></td>
<td>Filesystem</td>
<td>2.35E-10</td>
<td>-6.22</td>
<td>-0.28 [-0.35, -0.21]</td>
</tr>
<tr>
<td>2022 vs 2021</td>
<td>DocDiff</td>
<td>6.87E-29</td>
<td>-11.09</td>
<td>-0.61 [-0.70, -0.51]</td>
</tr>
<tr>
<td></td>
<td>Nile</td>
<td>1.82E-03</td>
<td>-2.91</td>
<td>-0.17 [-0.27, -0.07]</td>
</tr>
<tr>
<td></td>
<td>Filesystem</td>
<td>2.32E-09</td>
<td>-5.86</td>
<td>-0.26 [-0.33, -0.19]</td>
</tr>
<tr>
<td>2021 vs 2020</td>
<td>DocDiff</td>
<td>4.60E-01</td>
<td>0.1</td>
<td>0 [-0.07, 0.08]</td>
</tr>
<tr>
<td></td>
<td>Nile</td>
<td>1.15E-17</td>
<td>-8.48</td>
<td>-0.26 [-0.39, -0.13]</td>
</tr>
<tr>
<td></td>
<td>Filesystem</td>
<td>2.52E-01</td>
<td>-0.67</td>
<td>-0.02 [-0.06, 0.03]</td>
</tr>
</tbody>
</table>