

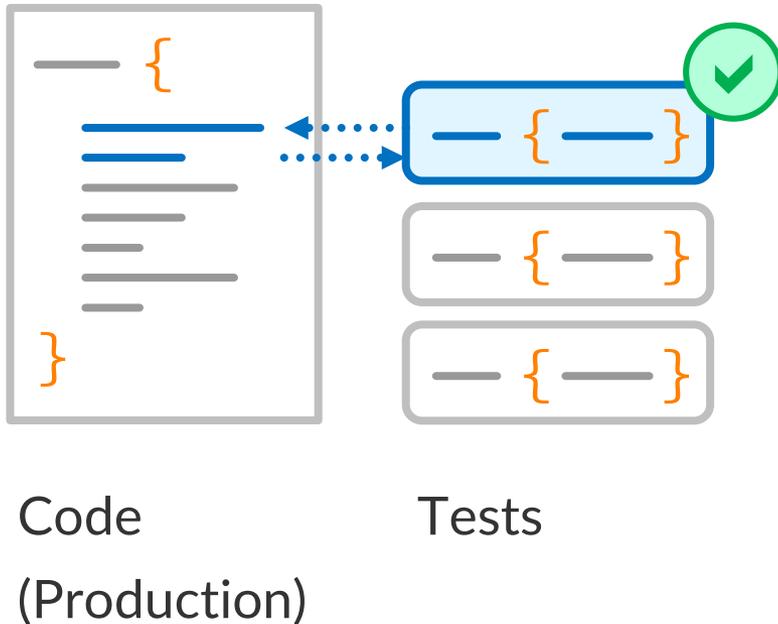
Faster Feedback through **Concept-based Test Prioritization**

Toni Mattis, Robert Hirschfeld

Research School for Service-Oriented Systems Engineering
Hasso Plattner Institute, University of Potsdam, Germany

HPI Nanjing Workshop 3 – 4 Sept. 2019, Nanjing, China

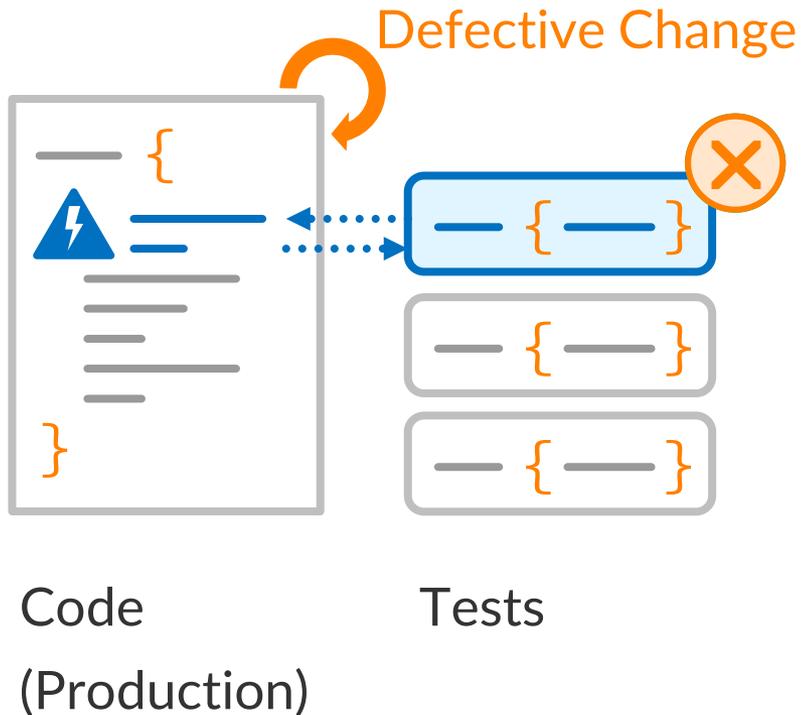
Testing in Software Engineering



Unit Test

1. Runs a small part of the code with example input
2. Checks if output is as expected

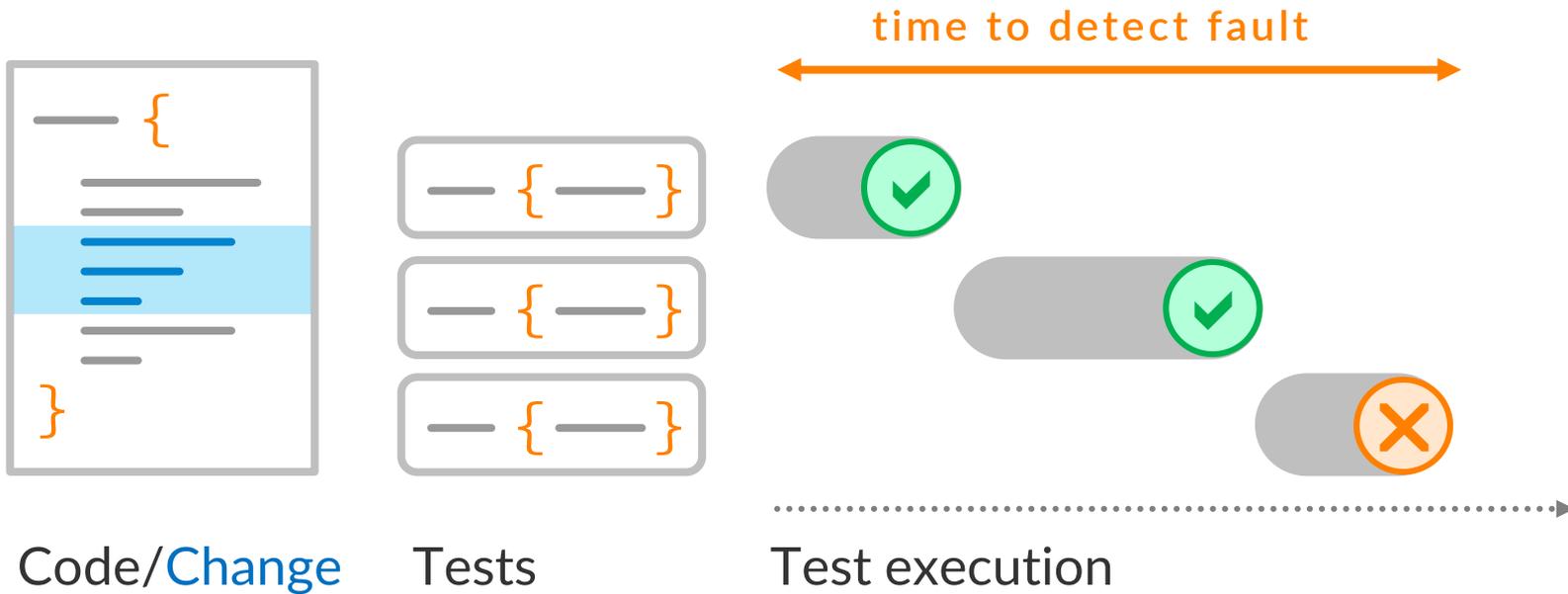
Testing in Software Engineering



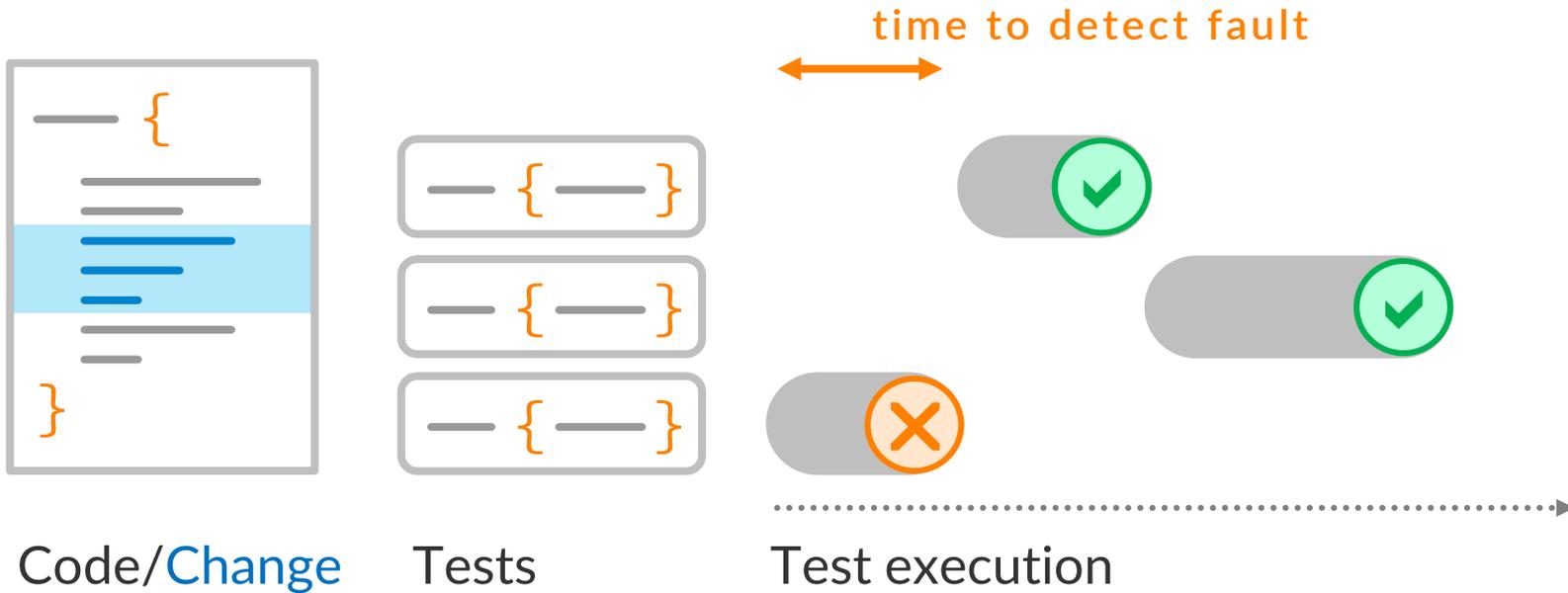
Unit Test

1. Runs a small part of the code with example input
2. Checks if output is as expected

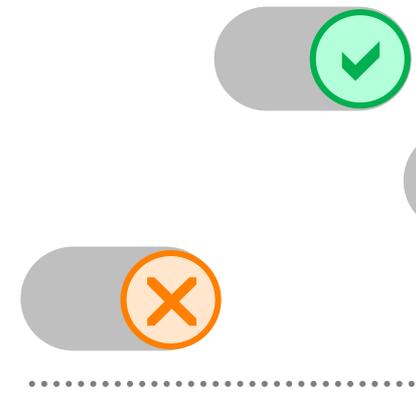
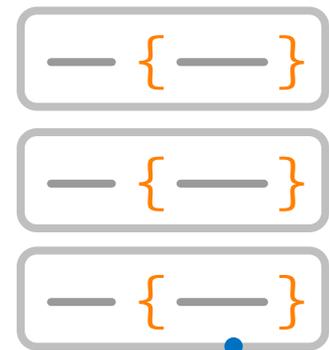
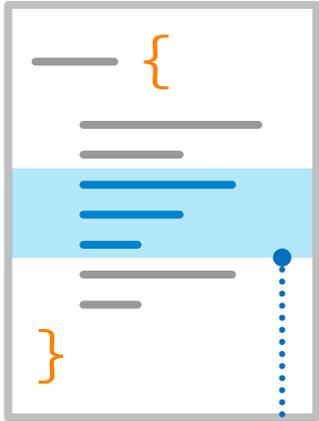
Goal: Immediate Feedback



Goal: Immediate Feedback



Lexical Test Prioritization



```
- return ...  
+ if session.user.is_admin:  
+     return ...
```

```
assert get(...) status == 403  
with login(admin_user):  
    assert get(...) == ...
```



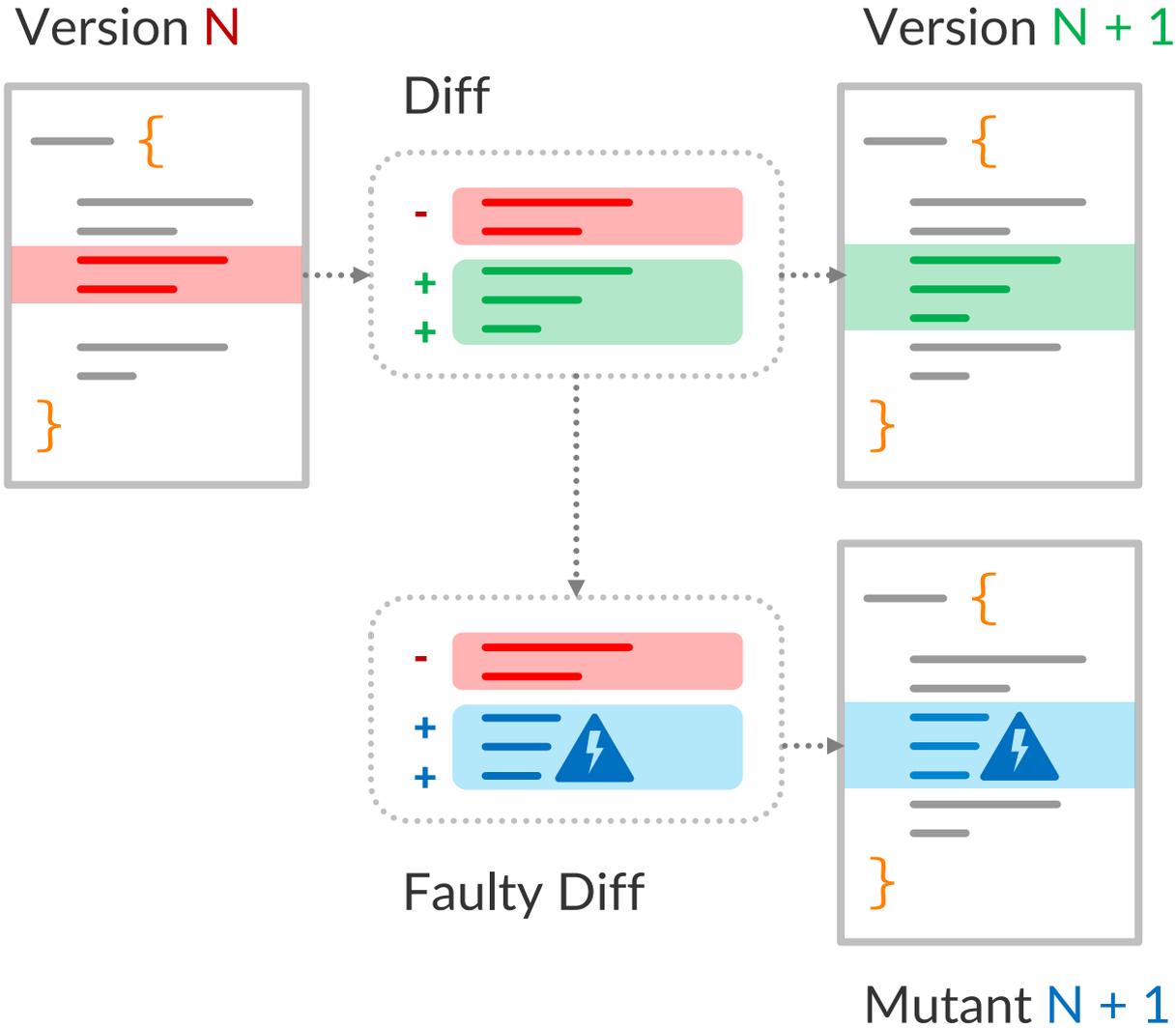
Hypothesis

Test cases that **share vocabulary** with the most recent **change** are more likely to fail

Approach

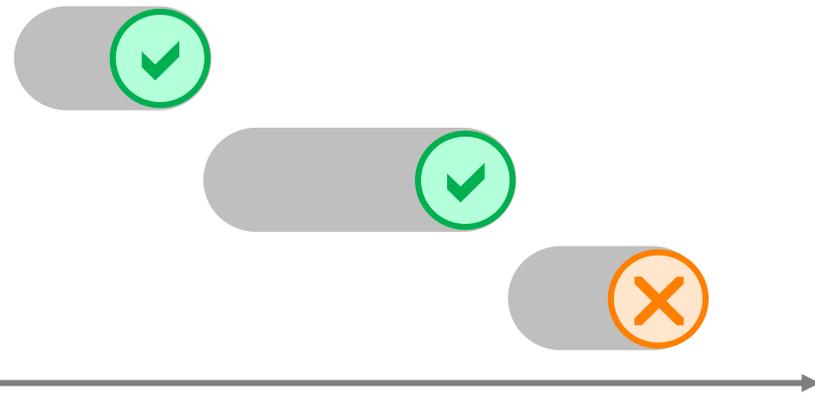
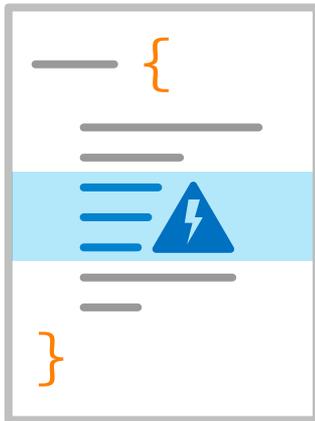
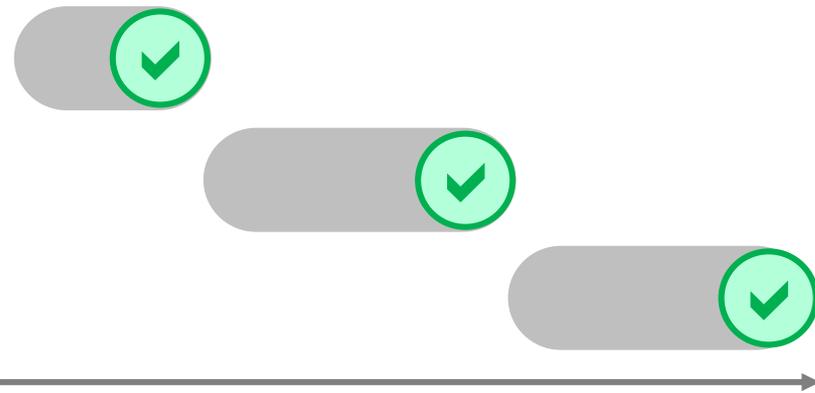
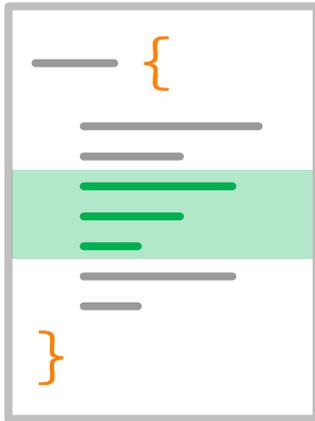
1. Seed faulty changes
2. Run tests
3. Re-order tests based on lexical similarity
4. Check how much earlier failures occur

Fault Seeding



Fault Seeding

Version $N + 1$



Mutant $N + 1$

Sampling Run

Fault Seeding

Negate condition

```
(if session.user.is_admin:)
  ..
```

```
if (not) session.user.is_admin:
  ..
```

Swap operator

```
average = total(/)count
```

```
average = total(*)count
```

Change number

```
response.status = (404)
```

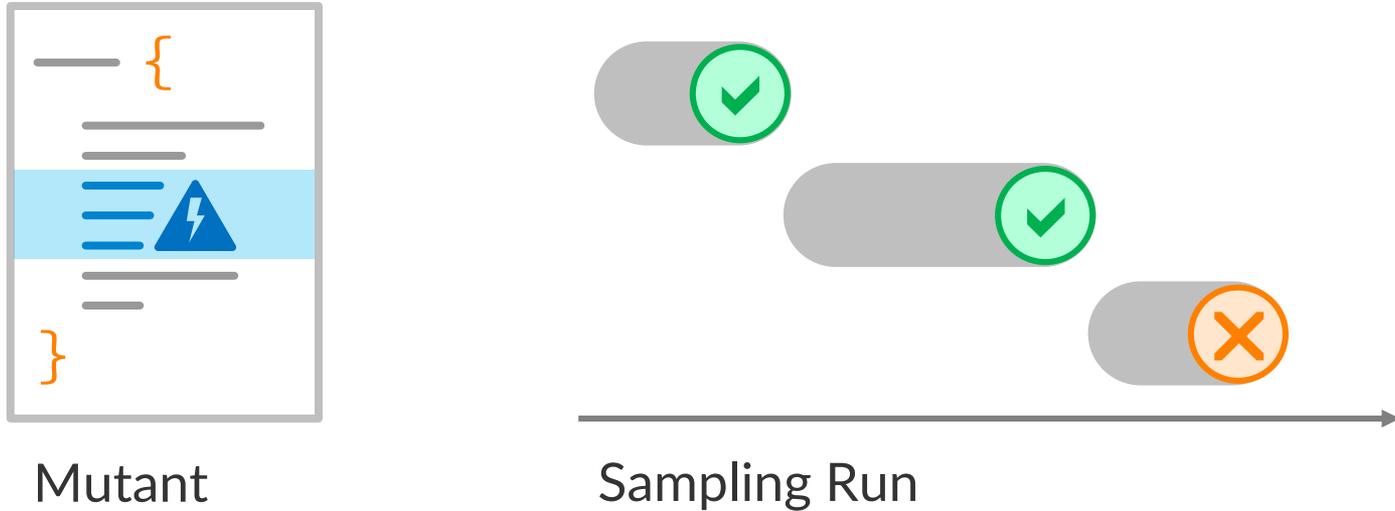
```
response.status = (405)
```

Drop call

```
(user_profile.save())
return redirect(...)
```

```
()
return redirect(...)
```

Feature Extraction



```
if not session.user.is_admin:
    return ...
```

{ admin, user, session ... }

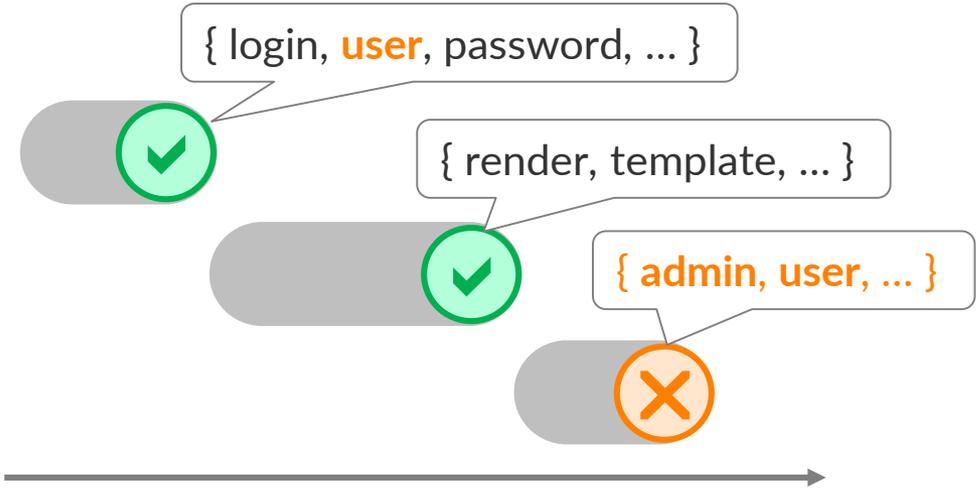
```
assert get(...).status == 403
with login(admin_user):
    assert get(...) == ...
```

{ admin, user, status, login, ... }

Prioritization

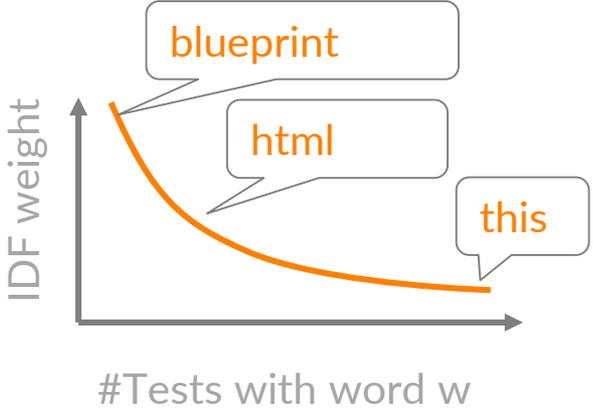


Mutant



Sampling Run

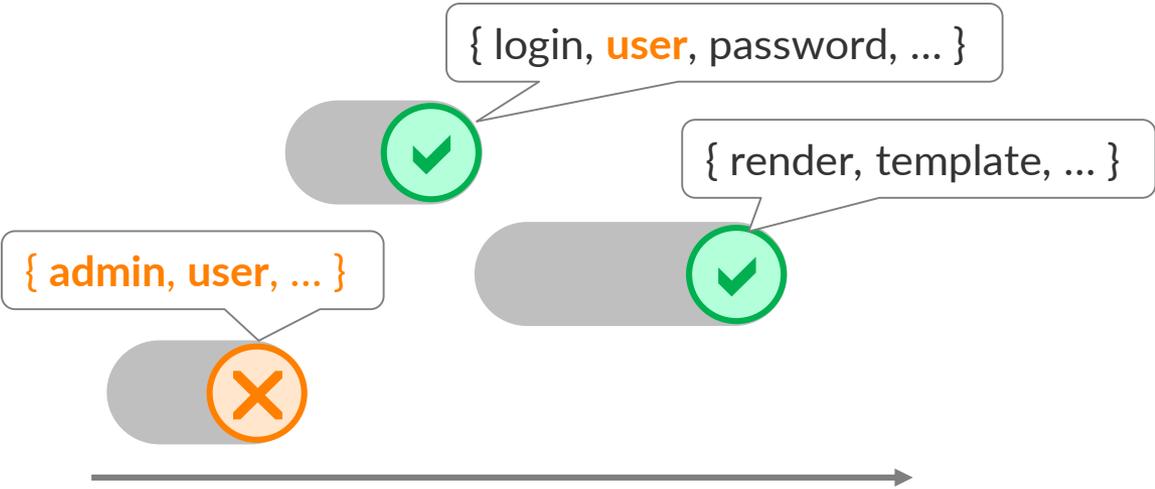
TF-IDF:



Prioritization



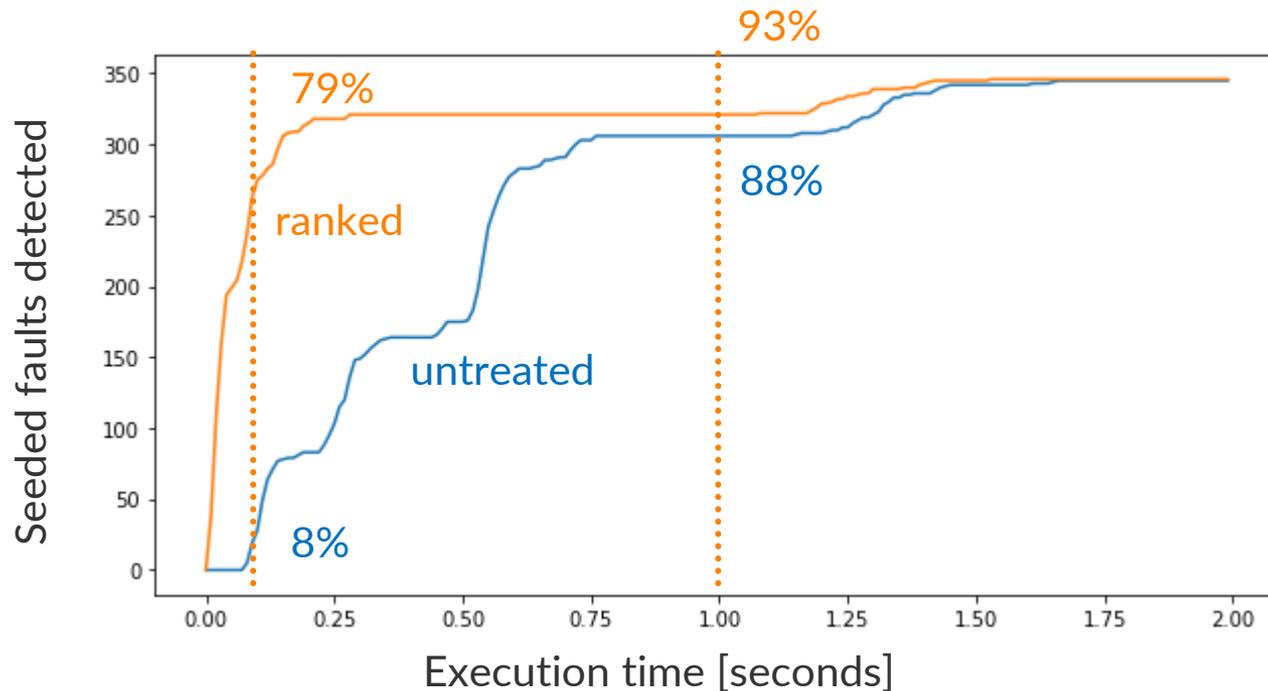
Mutant



Sorted sampling run results

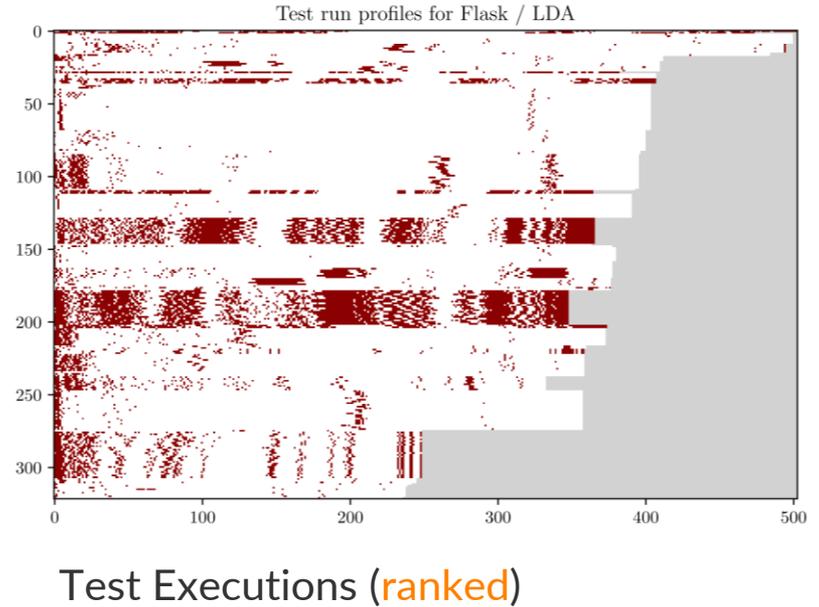
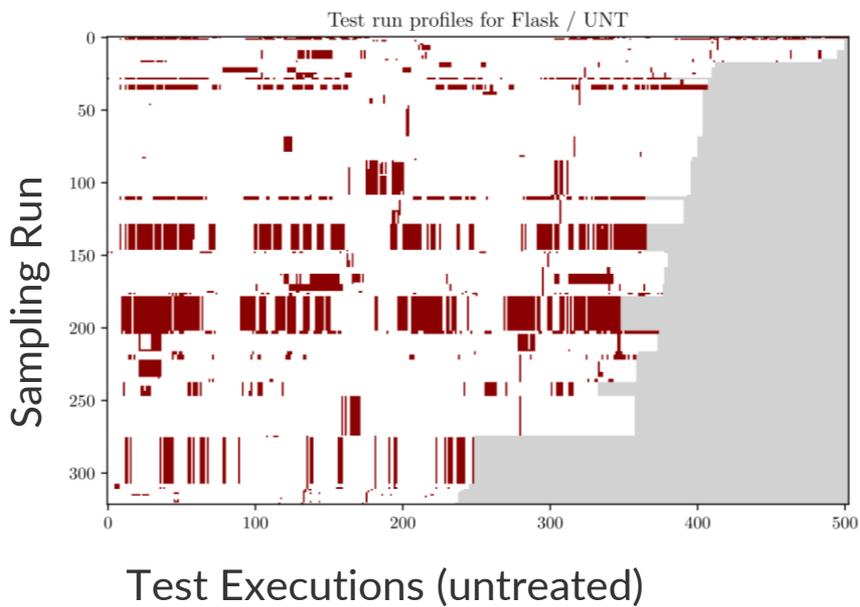
Case Study: Flask

Python web framework, 74 diffs, 413 seeded faults



Case Study: Flask

Python web framework, 74 commits, 413 seeded faults



Case Study: Flask

Limitation: **Pull Requests**

» Largest type of “change”

The screenshot shows a GitHub pull request for the Flask project. At the top, the repository path is 'pallets / flask'. Below this, there are navigation tabs: 'Code', 'Issues 36', 'Pull requests 18', and 'Insights'. The main content of the pull request is titled 'Merge pull request #2709 from davidism/docs-theme' with the description 'use theme provided by pallets-sphinx-themes'. Below the title, it shows the source branch 'master (#2709)' and the target branch '1.0.2 ... 1.0'. The pull request was committed by 'davidism' on 'Apr 18, 2018' and is marked as 'Verified'. At the bottom, it states 'Showing 25 changed files with 190 additions and 590 deletions.' The number '25' is highlighted with an orange box.

276 features

Case Study: Flask

Limitations:

- » Pull Requests
- » Distinguishing names (“NoAppException”) split into generic words (“no”, “app”, “exception”)

GitHub, Inc. (US) | <https://github.com/pallets/flask/commit/5fba092c22bee738ad2818d587d75ae17>

```

2 flask/cli.py
@@ -243,7 +243,7 @@ def locate_app(script_info, module_name, app_name, raise_if_not_
243 243         )
244 244         elif raise_if_not_found:
245 245             raise NoAppException(
246 -         'Could not import "{name}.".format(name=module_name)
246 +         'Could not import "{name}.".format(name=module_name)
247 247     )
248 248     else:
249 249         return
  
```

of 276...

'no', 'app', 'exception',

```

def test_locate_app_raises(test_apps, iname, aname):
    info = ScriptInfo()

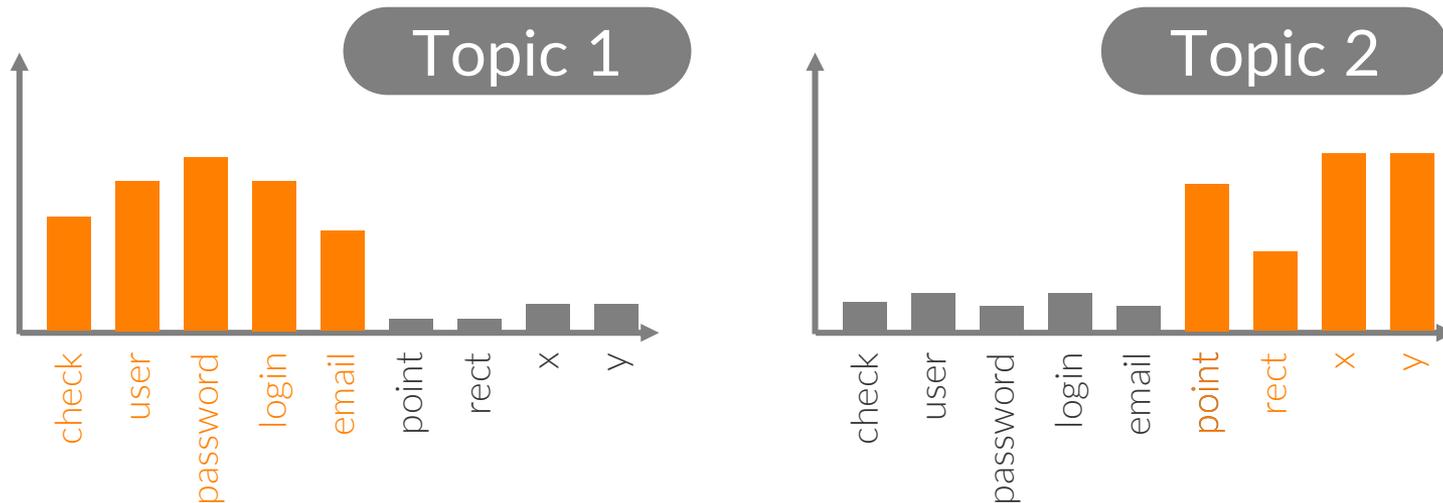
    with pytest.raises(NoAppException):
        locate_app(info, iname, aname)
  
```

Exploiting Topicality

Changed identifiers:
 user, password, check

Test to prioritize:
 login, email, password

Topic/Concept Model

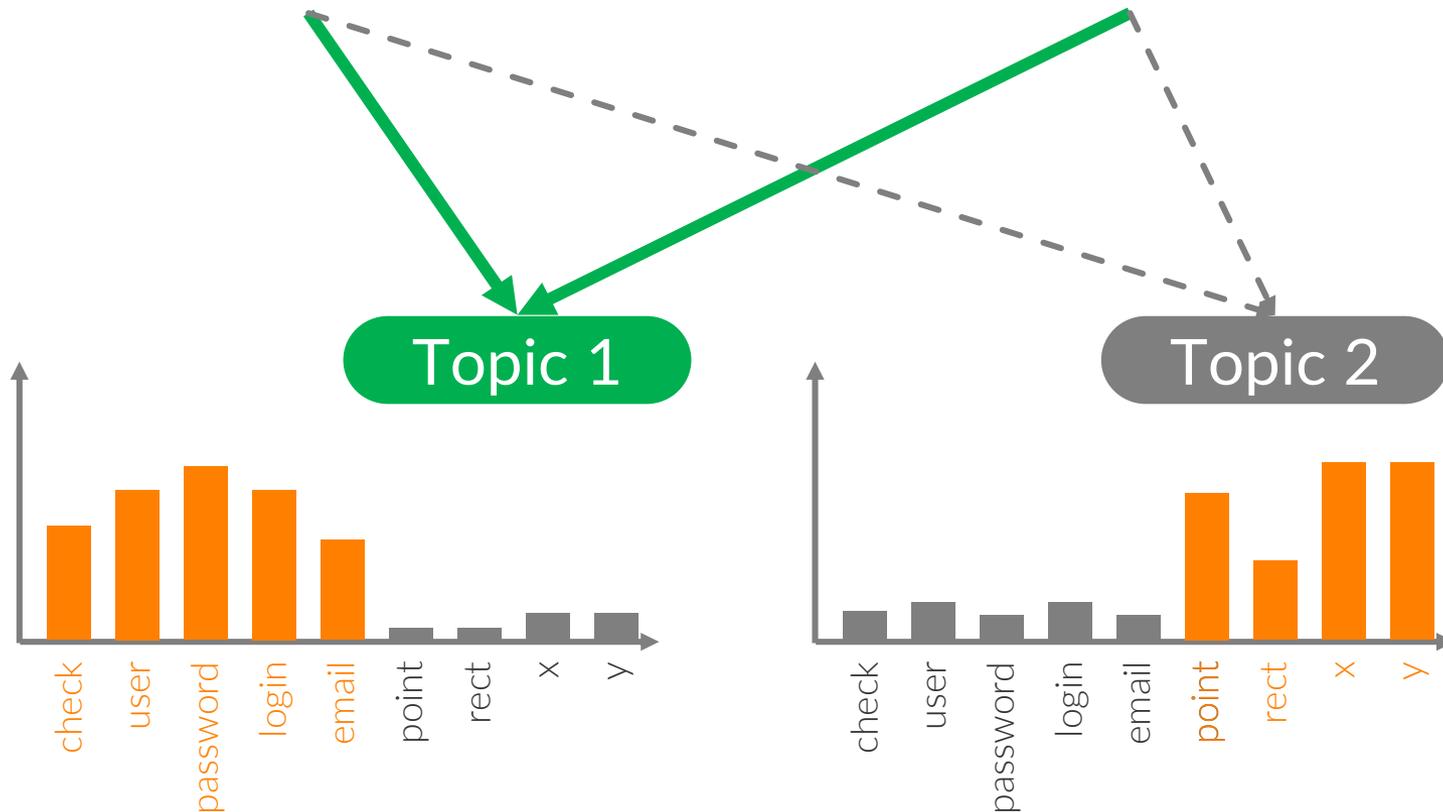


...

Exploiting Topicality

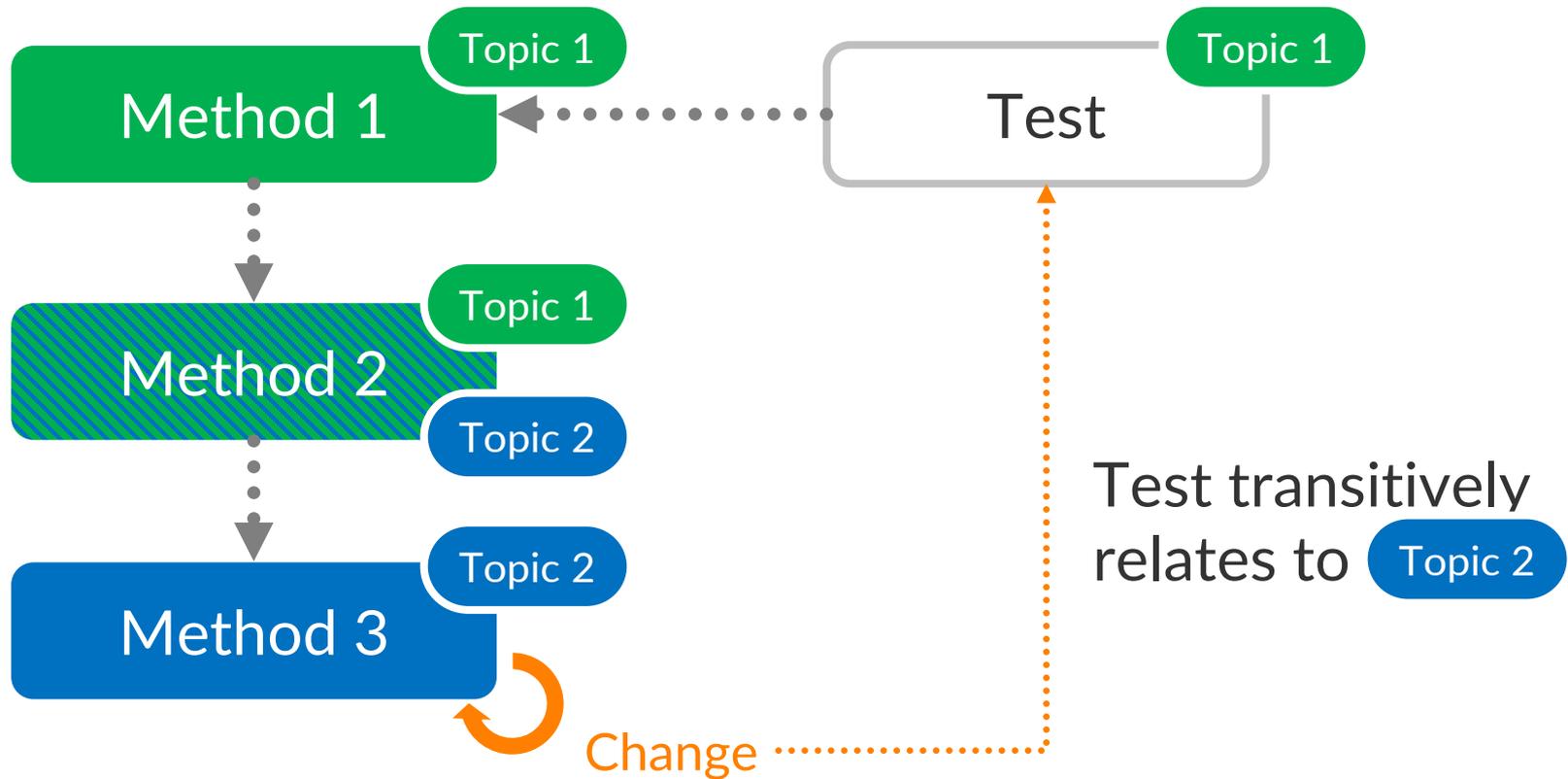
Changed identifiers:
 user, password, check

Test to prioritize:
 login, email, password



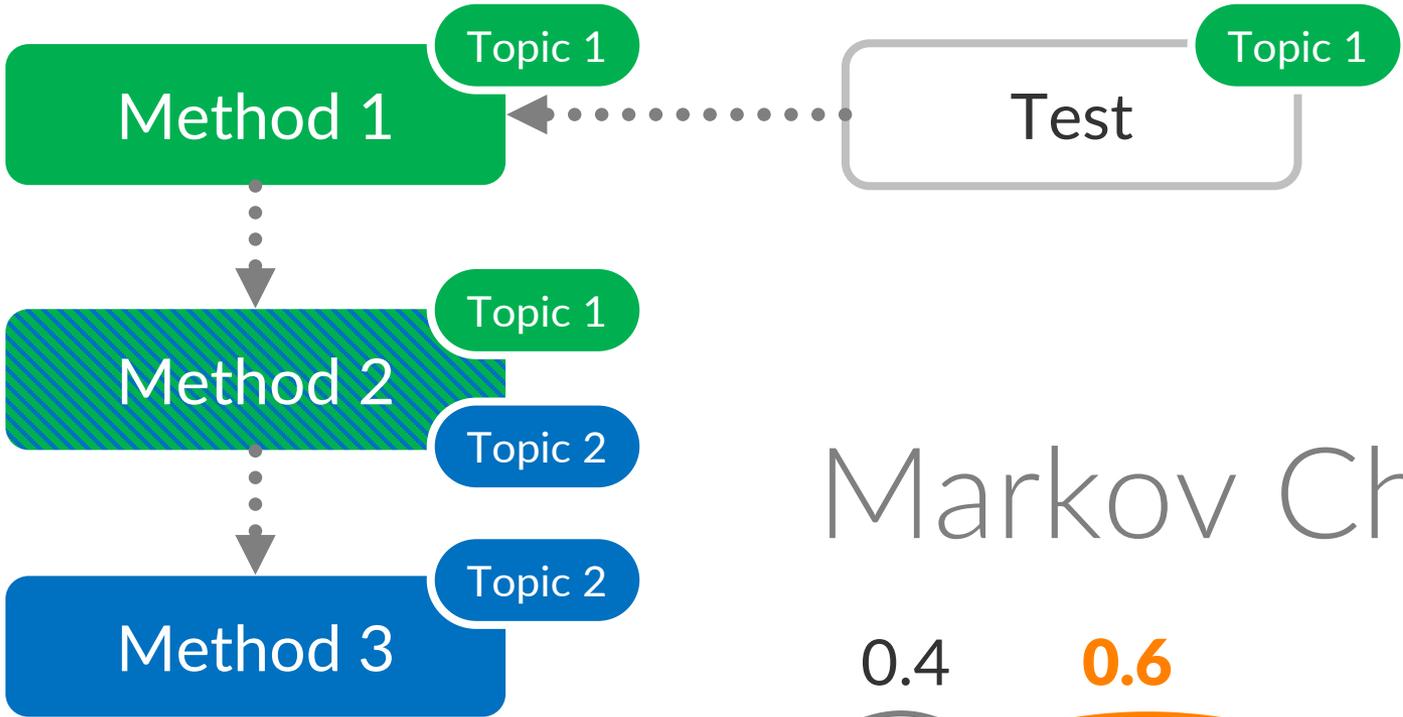
...

Approximating the Call Graph

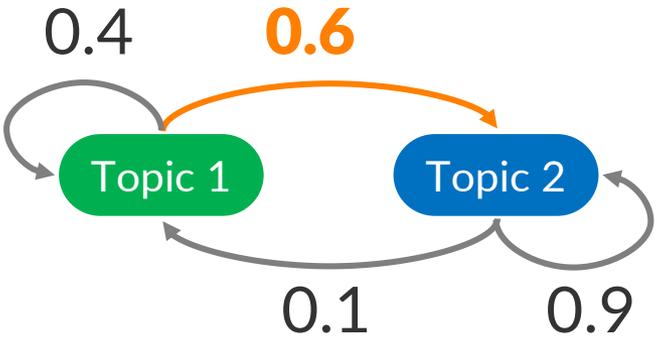


Test transitively relates to **Topic 2**

Approximating the Call Graph

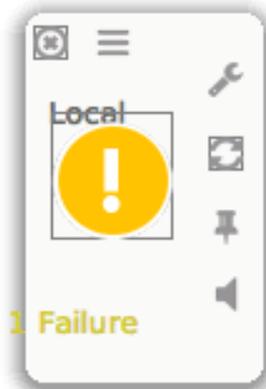


Markov Chain



Live Testing Tools

AutoTDD runs a selected **set of tests** whenever another selected **set of code locations** is changed

A screenshot of a GitHub repository page for 'AutoTDD'. The browser address bar shows 'https://github.com/hpi-swa-teaching/AutoTDD'. The page title is 'AutoTDD with Travis-CI Support'. Below the title, there is a 'build passing' status indicator. The main heading is 'Installation', followed by a list of instructions: '1. Make sure you have metacello-work installed.' and '2. Load the project with:'. At the bottom, there is a code snippet: 'Metacello new' and 'baseline: 'AutoTDD':'.

GitHub, Inc. (US) | https://github.com/hpi-swa-teaching/AutoTDD

AutoTDD with Travis-CI Support

build passing

Installation

1. Make sure you have [metacello-work](#) installed.
2. Load the project with:

```
Metacello new  
baseline: 'AutoTDD':
```

Conclusion

- » **Change-based fault seeding** is an effective method to generate many failures distributed like actual changes
- » **Lexical information** can be exploited to quickly guess which tests may fail
- » There is more potential in exploring **topicality** and **Markov properties** of the vocabulary

