Anchored Metadata

Austin Bingham

@austin_bingham

The Problem

Associating Metadata with Code

Do not mutate

No linting

Disable tests

```
let context = Context::from_path(
    anchor.file_path(),
    anchor.context().offset(),
    anchor.context().topic().len() as u64,
    anchor.context().width())?;

let new_anchor = Anchor::new(
    anchor.file_path(),
    context,
    anchor.metadata().clone(),
    anchor.encoding().clone(),
)?;

let mut diff_strings: Vec<String> =
Vec::new();

let mut changed = false;

for diff in diff::lines(
    anchor.context().full_text().as_str(),
    new_anchor.context().full_text().as_str()
```

Cosmic Ray: Mutation Testing for Python

github.com/sixty-north/cosmic-ray

What is mutation testing?

Code under test + test suite

Introduce single change to code under test

Run test suite

Ideally, all changes will result in test failures

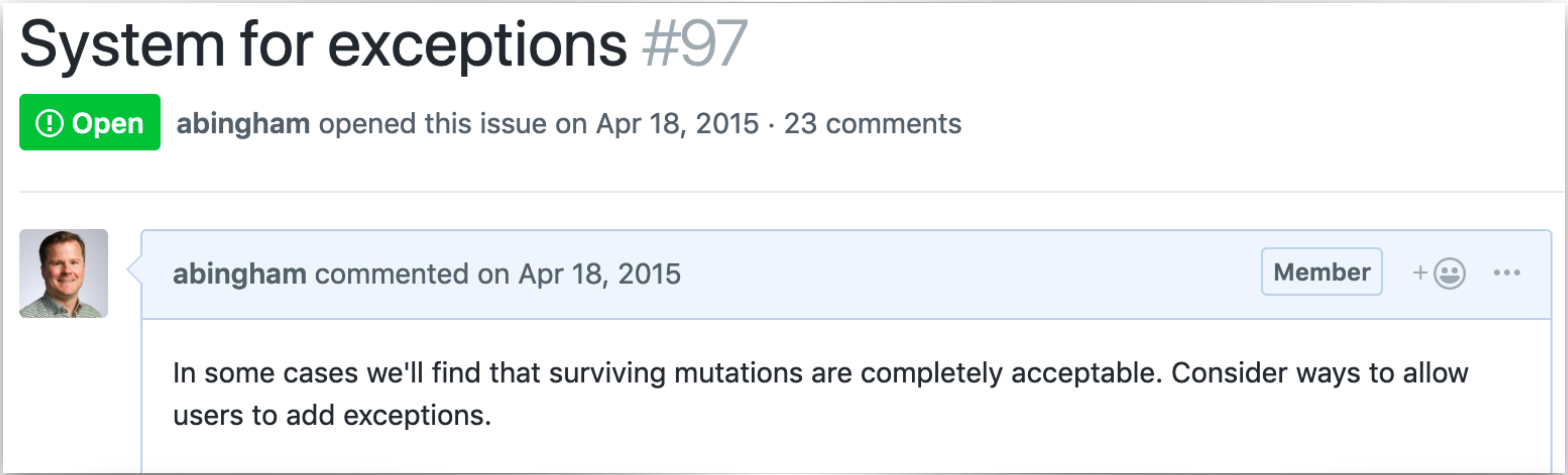
Equivalent Mutants

```
if __name__ == '__main__':
    # Code in here never
    # runs in tests
    run()
```

Equivalent Mutants

```
def consume(iterator, n):
    """Advance the iterator n-steps ahead.
       If n is none, consume entirely."""

    # Use functions that consume iterators at C speed.
    if n is None:
        # feed the entire iterator into a zero-length deque
        collections.deque(iterator, maxlen=0)
    else:
        # advance to the empty slice starting at position n
        next(islice(iterator, n, n), None)
```

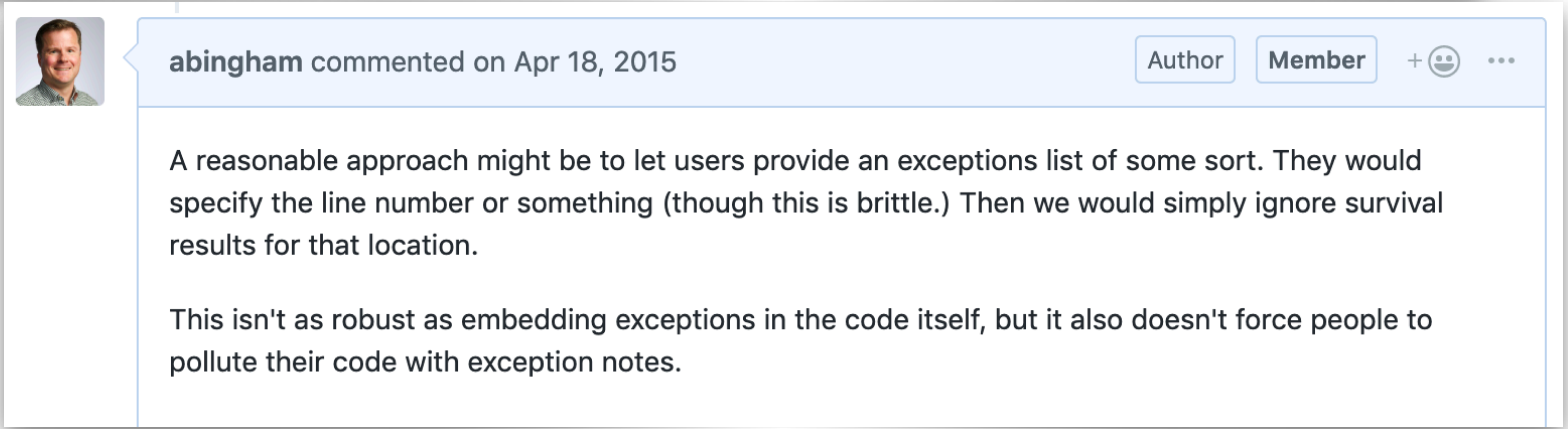


System for exceptions #97

Open abingham opened this issue on Apr 18, 2015 · 23 comments

abingham commented on Apr 18, 2015 Member

In some cases we'll find that surviving mutations are completely acceptable. Consider ways to allow users to add exceptions.



abingham commented on Apr 18, 2015 Author Member

A reasonable approach might be to let users provide an exceptions list of some sort. They would specify the line number or something (though this is brittle.) Then we would simply ignore survival results for that location.

This isn't as robust as embedding exceptions in the code itself, but it also doesn't force people to pollute their code with exception notes.

```python
from sphinx.util.osutil import (  # noqa
    SEP, os_path, relative_uri, ensuredir,
    walk, mtimes_of_files, movefile,
    copyfile, copytimes, make_filename,
    ustrftime)
from sphinx.util.nodes import (   # noqa
    nested_parse_with_titles, split_explicit_title,
    explicit_title_re,
    caption_ref_re)
from sphinx.util.matching import patfilter  # noqa
```

What’s wrong with inline metadata?

- Language-specific
- Collisions
- Clutters code
- Not robust against refactoring

The Solution(?)
Externalized Metadata

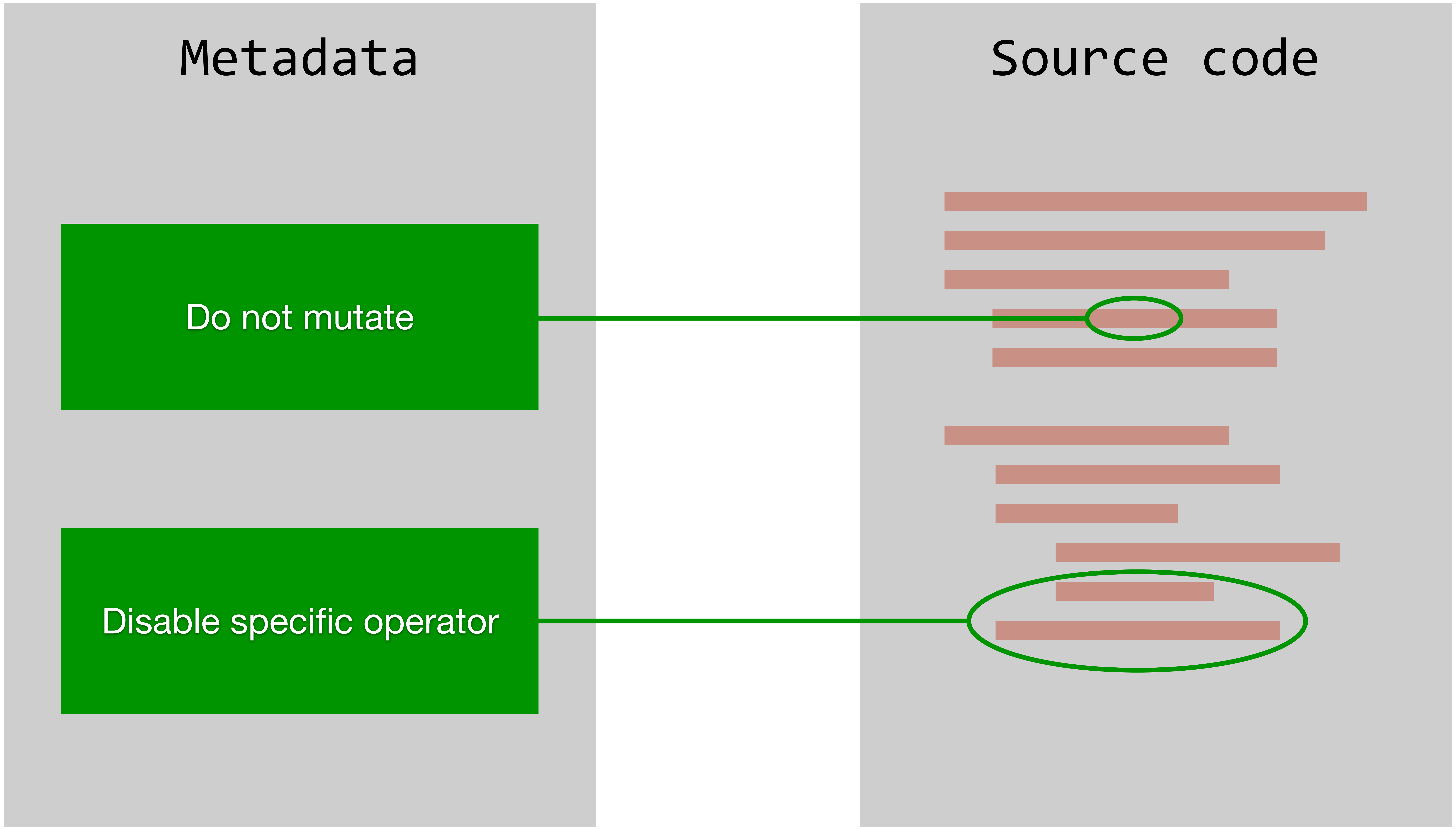
Rob Smallshire

David MacIver

Rob Smallshire

David MacIver

Beer

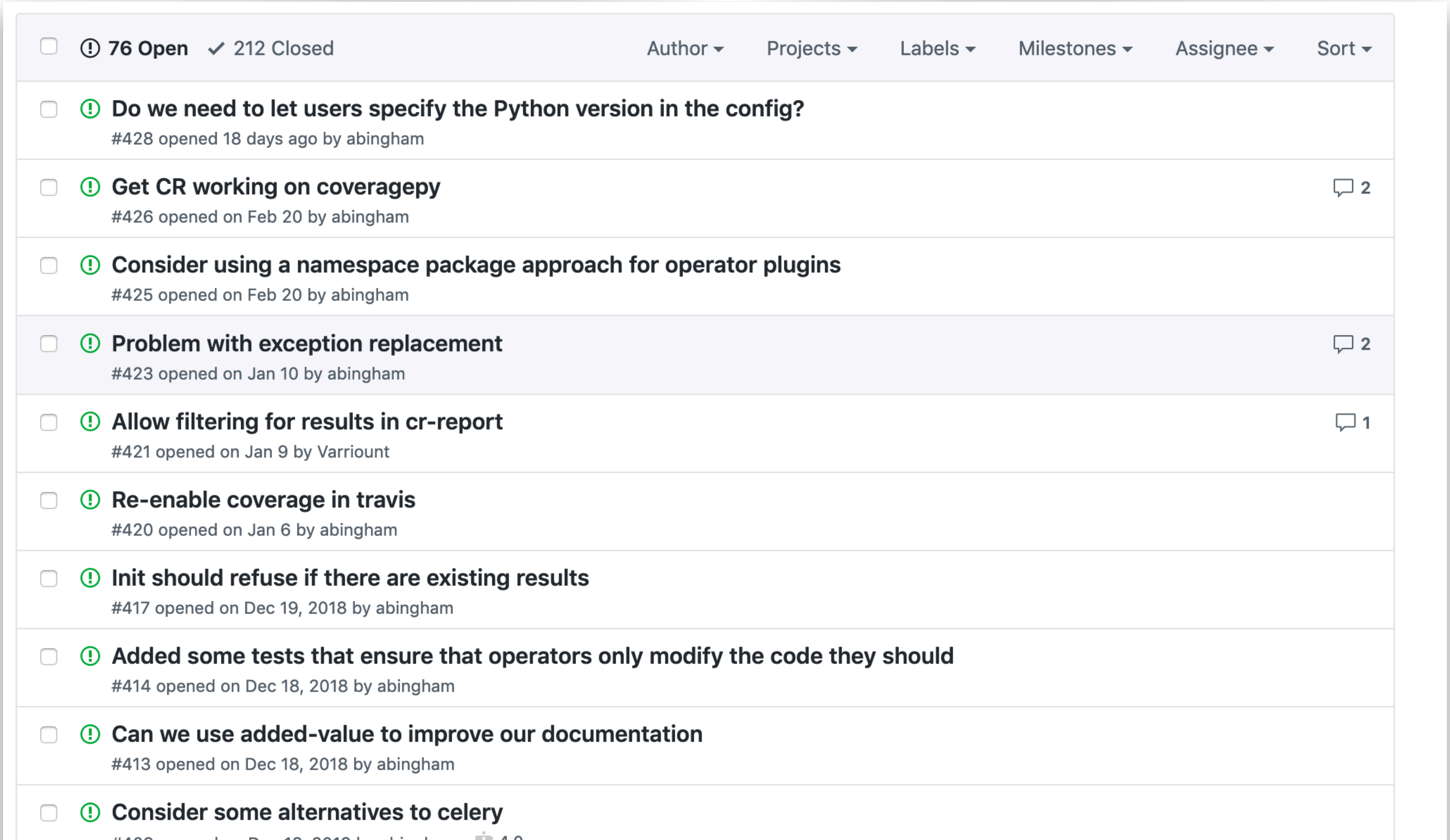

Metadata
Source code
Do not mutate
Disable specific operator

What happens when the code changes?

Image by tasokkassaris, creativecommons.org/licenses/by-sa/3.0/

The Challenge

Keep Metadata Aligned with Changing Source Code



Do we need to let users specify the Python version in the config?
#428 opened 18 days ago by abingham

Get CR working on coveragepy 2
#426 opened on Feb 20 by abingham

Consider using a namespace package approach for operator plugins
#425 opened on Feb 20 by abingham

Problem with exception replacement 2
#423 opened on Jan 10 by abingham

Allow filtering for results in cr-report 1
#421 opened on Jan 9 by Varriount

Re-enable coverage in travis
#420 opened on Jan 6 by abingham

Init should refuse if there are existing results
#417 opened on Dec 19, 2018 by abingham

Added some tests that ensure that operators only modify the code they should
#414 opened on Dec 18, 2018 by abingham

Can we use added-value to improve our documentation
#413 opened on Dec 18, 2018 by abingham

Consider some alternatives to celery

Metadata Anchoring for Source Code: Robust Location Descriptor Definition, Building and Interpreting

Conference Paper · August 2013

DOI: 10.1007/978-3-642-40173-2_30

CITATIONS
3

READS
91

2 authors:

Karol Rástočný
Slovak University of Technology in Bratislava

16 PUBLICATIONS **43** CITATIONS

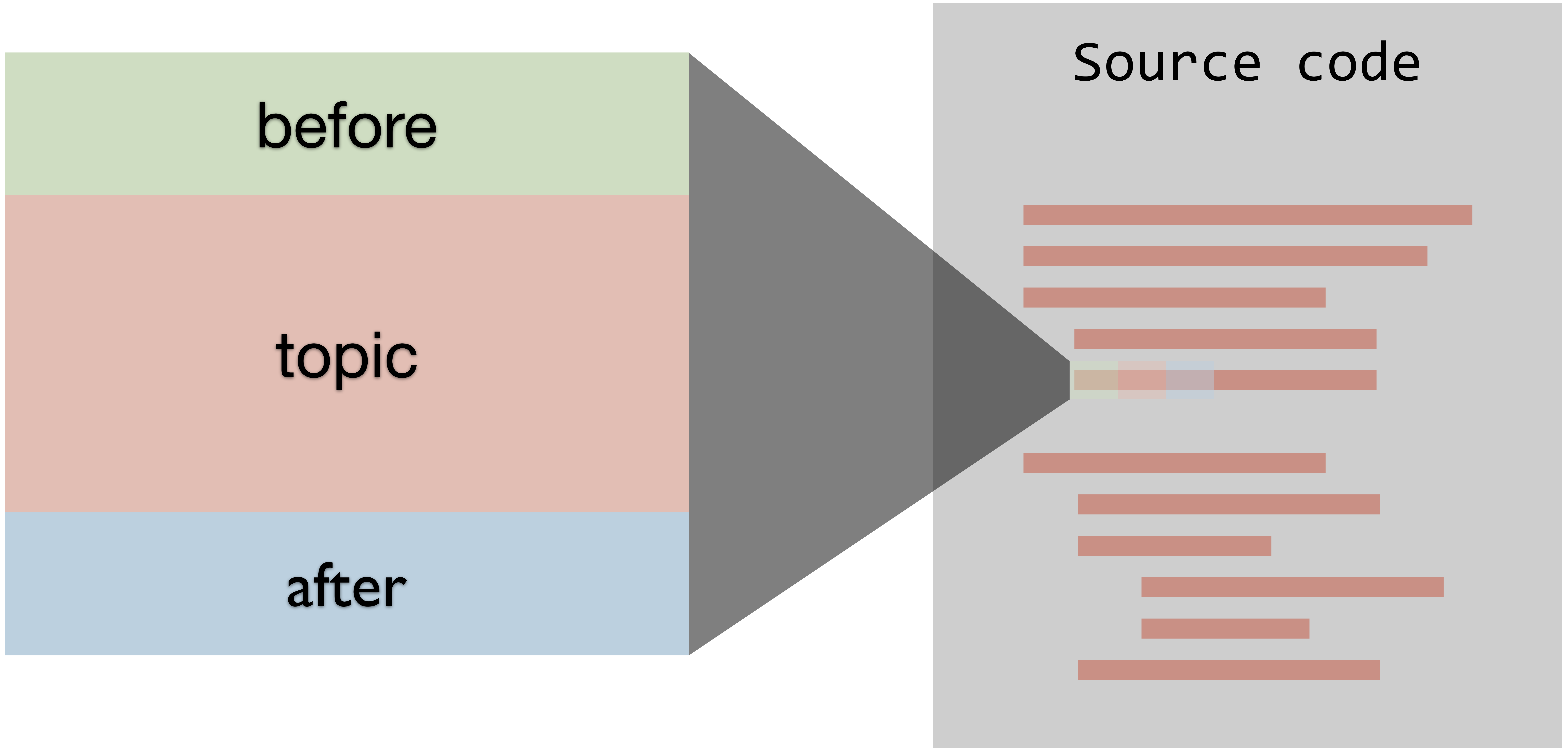
SEE PROFILE

Maria Bielikova
Slovak University of Technology in Bratislava

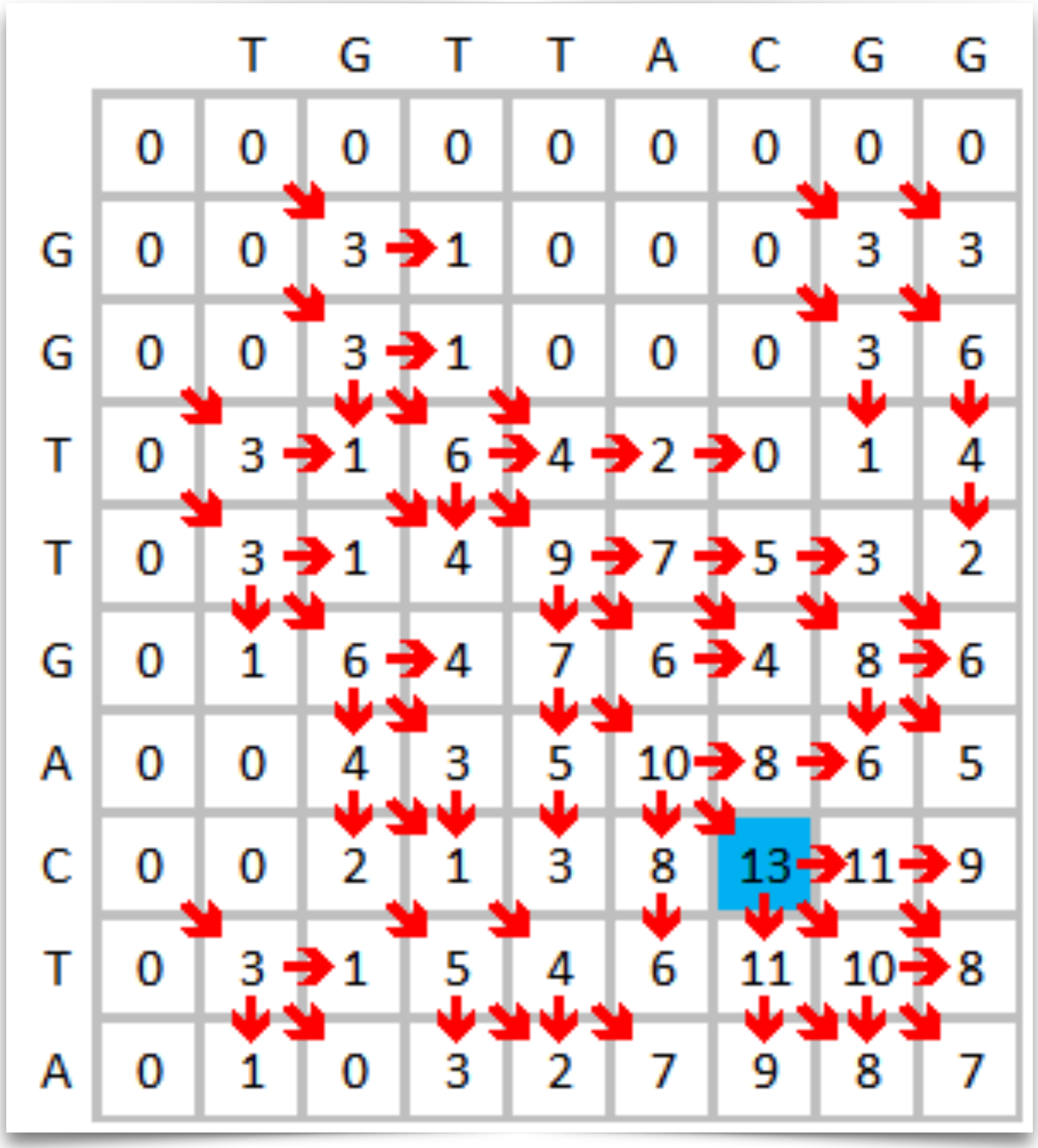
309 PUBLICATIONS **1,850** CITATIONS

SEE PROFILE

Context



Smith-Waterman Alignment Algorithm

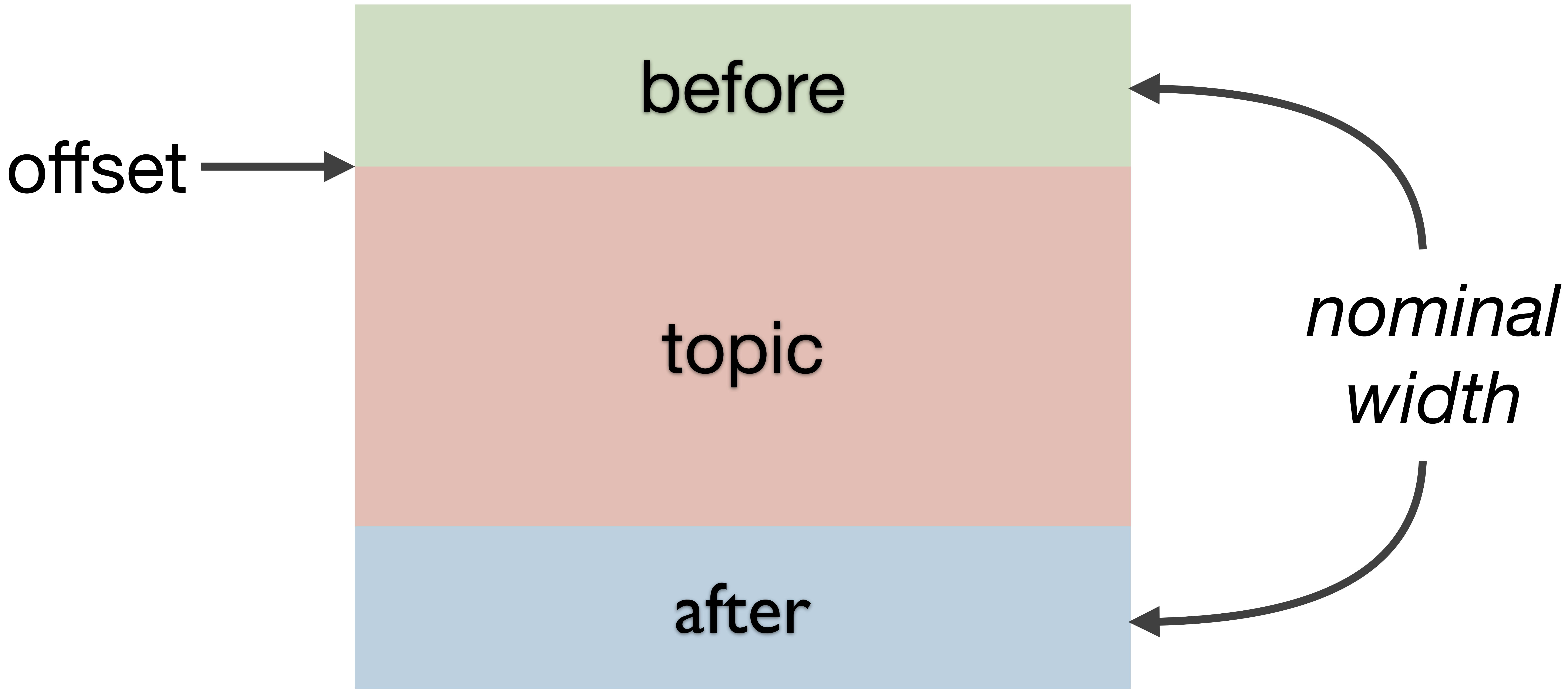


		T	G	T	T	A	C	G	G
	0	0	0	0	0	0	0	0	0
G	0	0	3	1	0	0	0	3	3
G	0	0	3	1	0	0	0	3	6
T	0	3	1	6	4	2	0	1	4
T	0	3	1	4	9	7	5	3	2
G	0	1	6	4	7	6	4	8	6
A	0	0	4	3	5	10	8	6	5
C	0	0	2	1	3	8	13	11	9
T	0	3	1	5	4	6	11	10	8
A	0	1	0	3	2	7	9	8	7

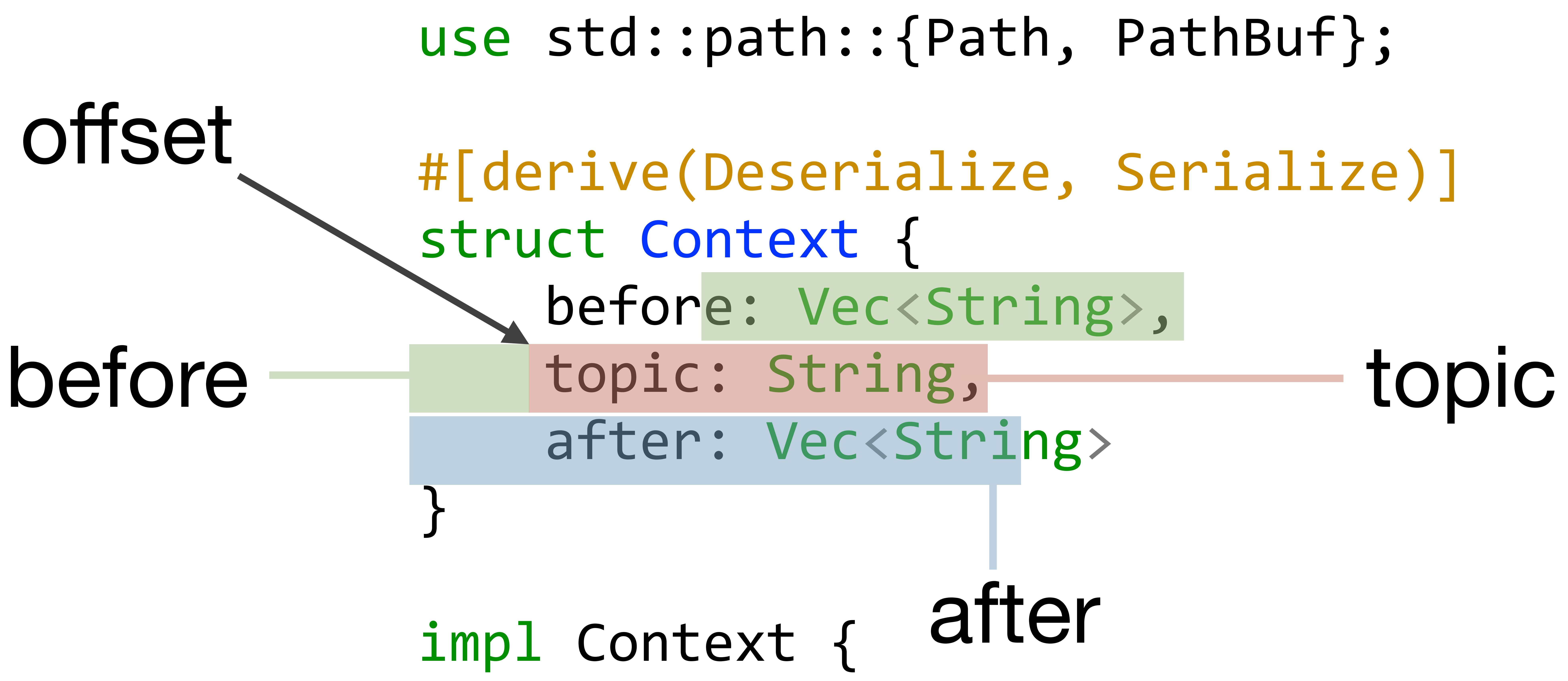
By Yz cs5160, creativecommons.org/licenses/by-sa/4.0/deed.en

Data Structures

Context



Context

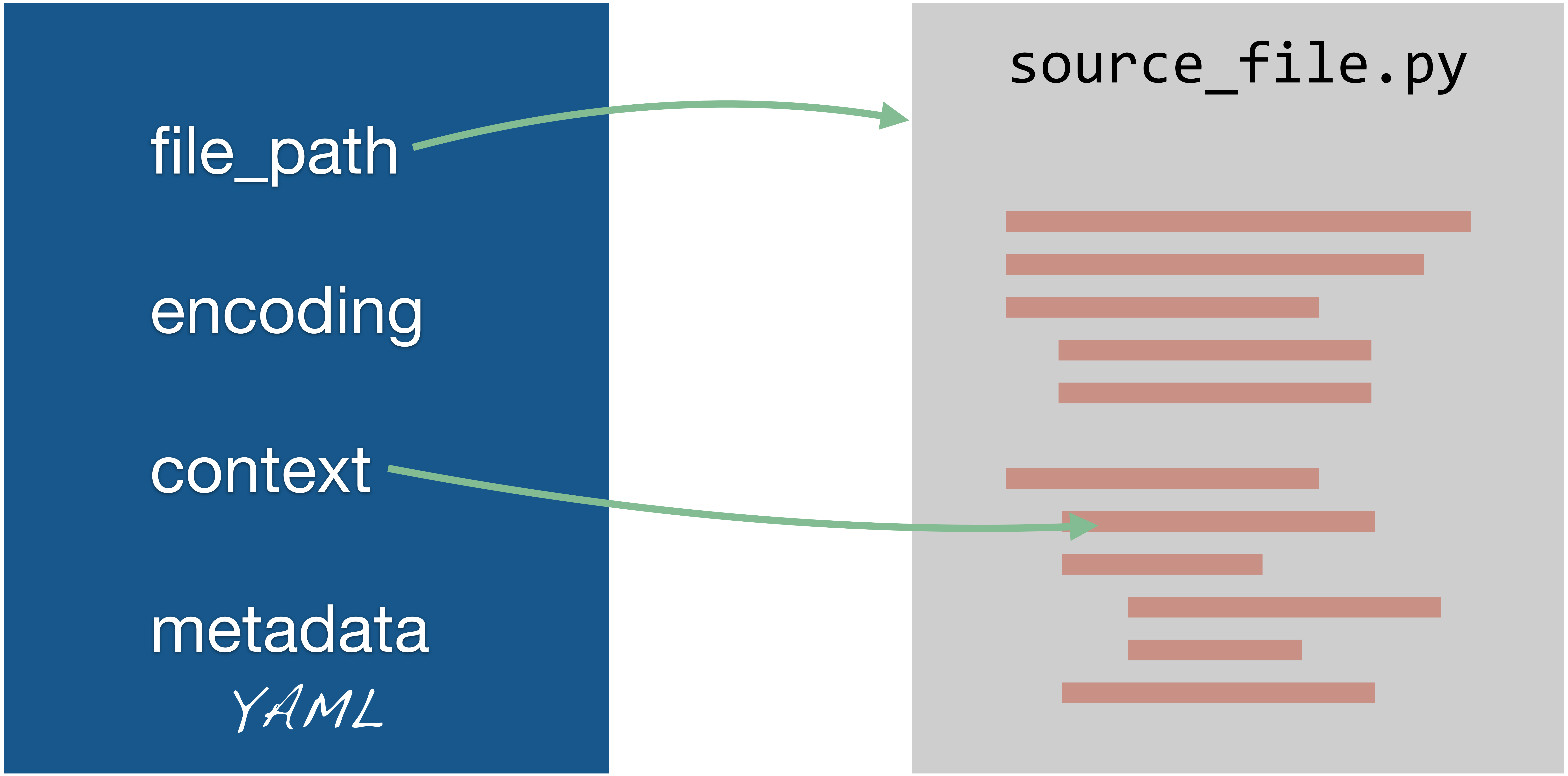


```rust
use std::path::{Path, PathBuf};

#[derive(Deserialize, Serialize)]
struct Context {
    before: Vec<String>,
    topic: String,
    after: Vec<String>
}

impl Context {
```

Anchor



Algorithms

Smith-Waterman

- Genomics: aligning nucleic acid sequences
- Finds all potentially optimal alignments
- Applies scoring and gap penalty functions
- Optimal alignments are found through backtracking

Basic Idea

For each pair of input elements, the score is the maximum of an afferent alignment score plus:

a) a scoring function if the alignment is contiguous
b) a gap penalty if there is a discontinuity

All maximal scores represent equally optimal alignments.

The alignments are the paths from maximum cell scores back through contributory alignments until a zero is reached.

GACCG

GCCA

Construct the score matrix

		G	A	C	C	G
G						
C						
C						
A						

Initialize edge to zeros

		G	A	C	C	G
	0	0	0	0	0	0
G	0					
C	0					
C	0					
A	0					

S_{ij} score is maximum of:

- $S_{i-1,j-1}$ + score_func(A[i], B[j])
- $S_{i,j-1}$ + gap_penalty()
- $S_{i-1,j}$ + gap_penalty()
- Zero

```
score_func(a, b):
  3 if a == b else -3

gap_penalty():
  -2
```

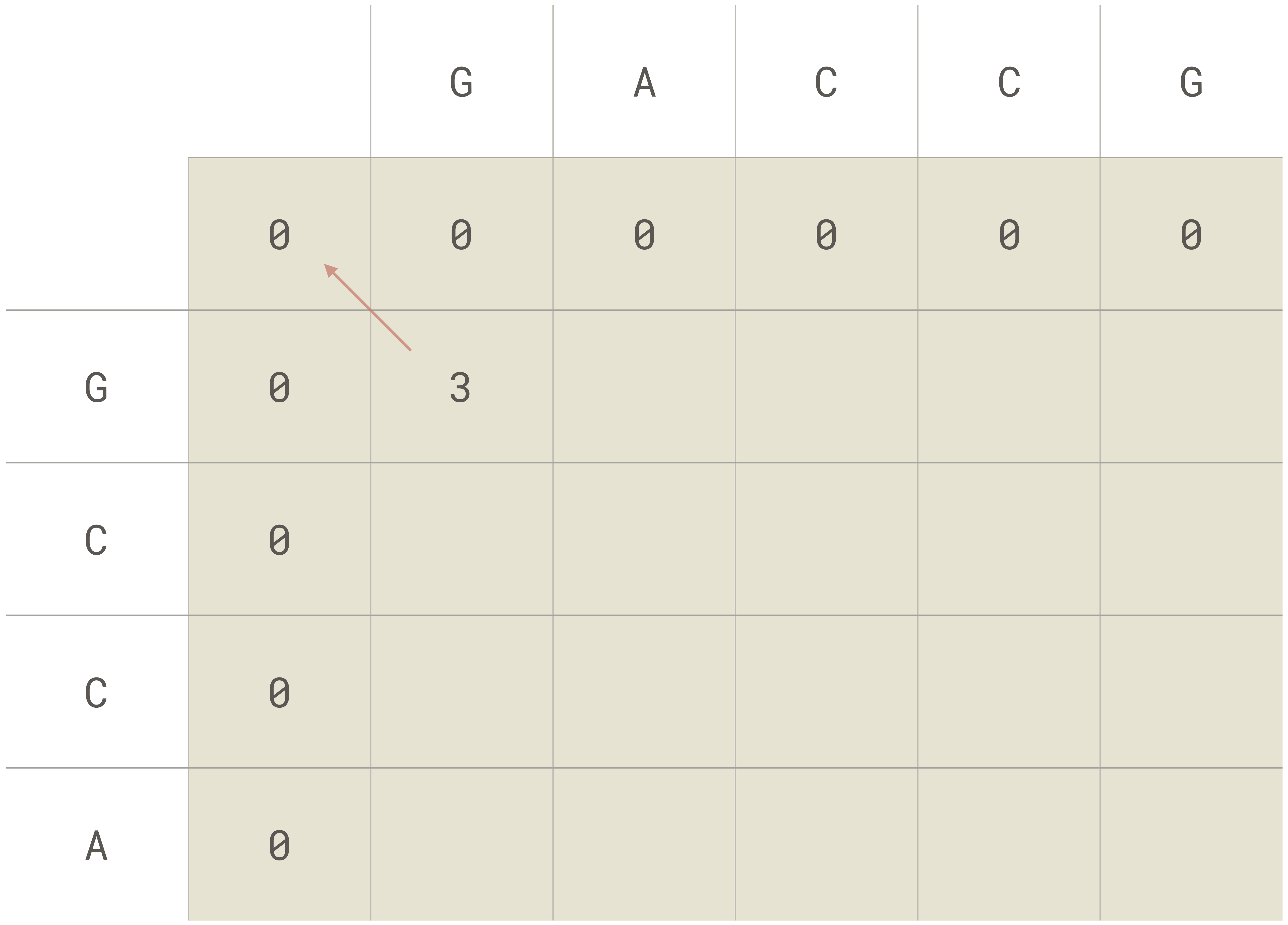
		G	A	C	C	G
	0	0	0	0	0	0
G	0					
C	0					
C	0					
A	0					

S_{ij} score is maximum of:

- $S_{i-1,j-1}$ + score_func(A[i], B[j])
- $S_{i,j-1}$ + gap_penalty()
- $S_{i-1,j}$ + gap_penalty()
- Zero

```
score_func(a, b):
  3 if a == b else -3

gap_penalty():
  -2
```



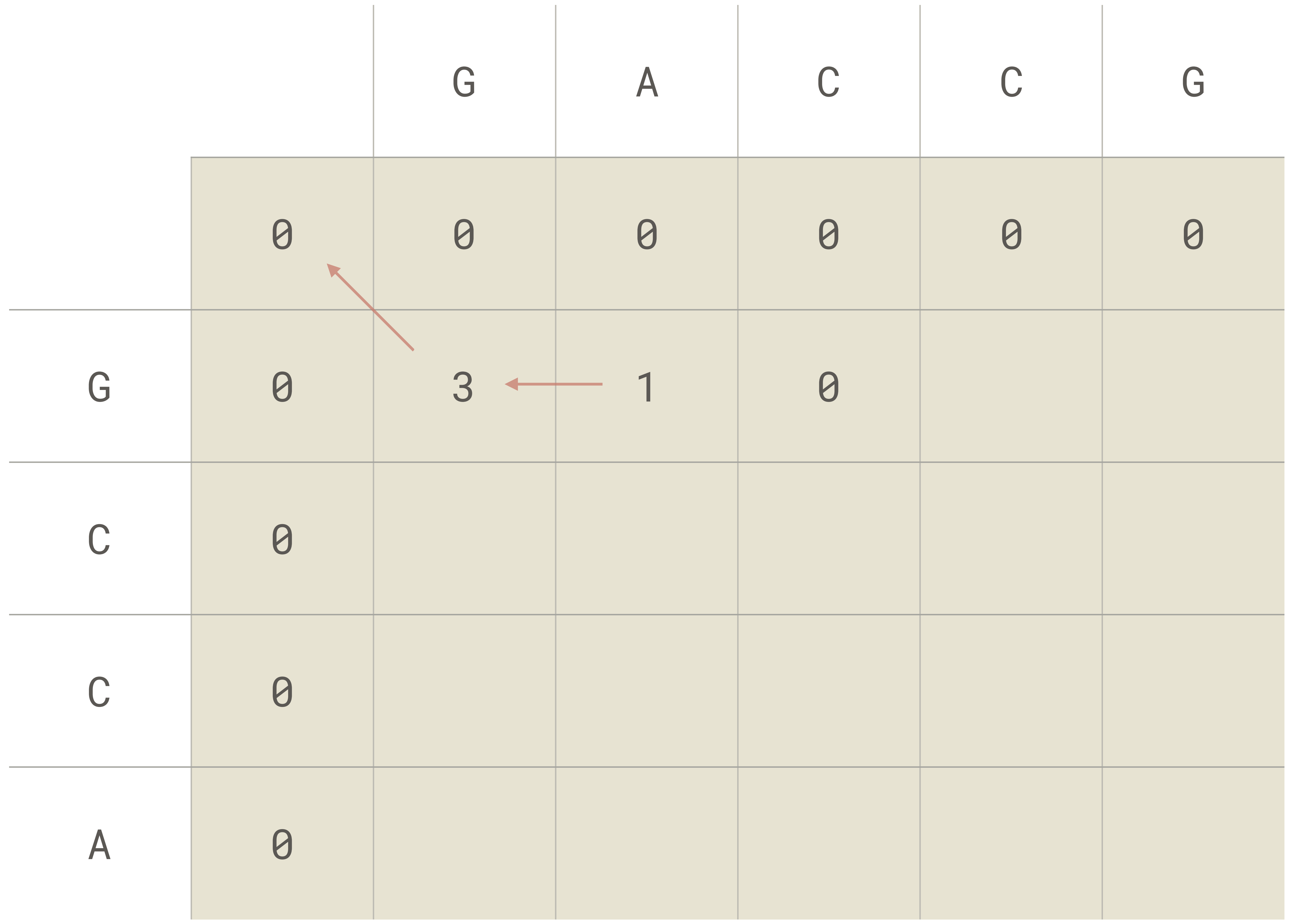
		G	A	C	C	G
	0	0	0	0	0	0
G	0	3				
C	0					
C	0					
A	0					

S_{ij} score is maximum of:

- $S_{i-1,j-1}$ + score_func(A[i], B[j])
- $S_{i,j-1}$ + gap_penalty()
- $S_{i-1,j}$ + gap_penalty()
- Zero

```
score_func(a, b):
  3 if a == b else -3

gap_penalty():
  -2
```



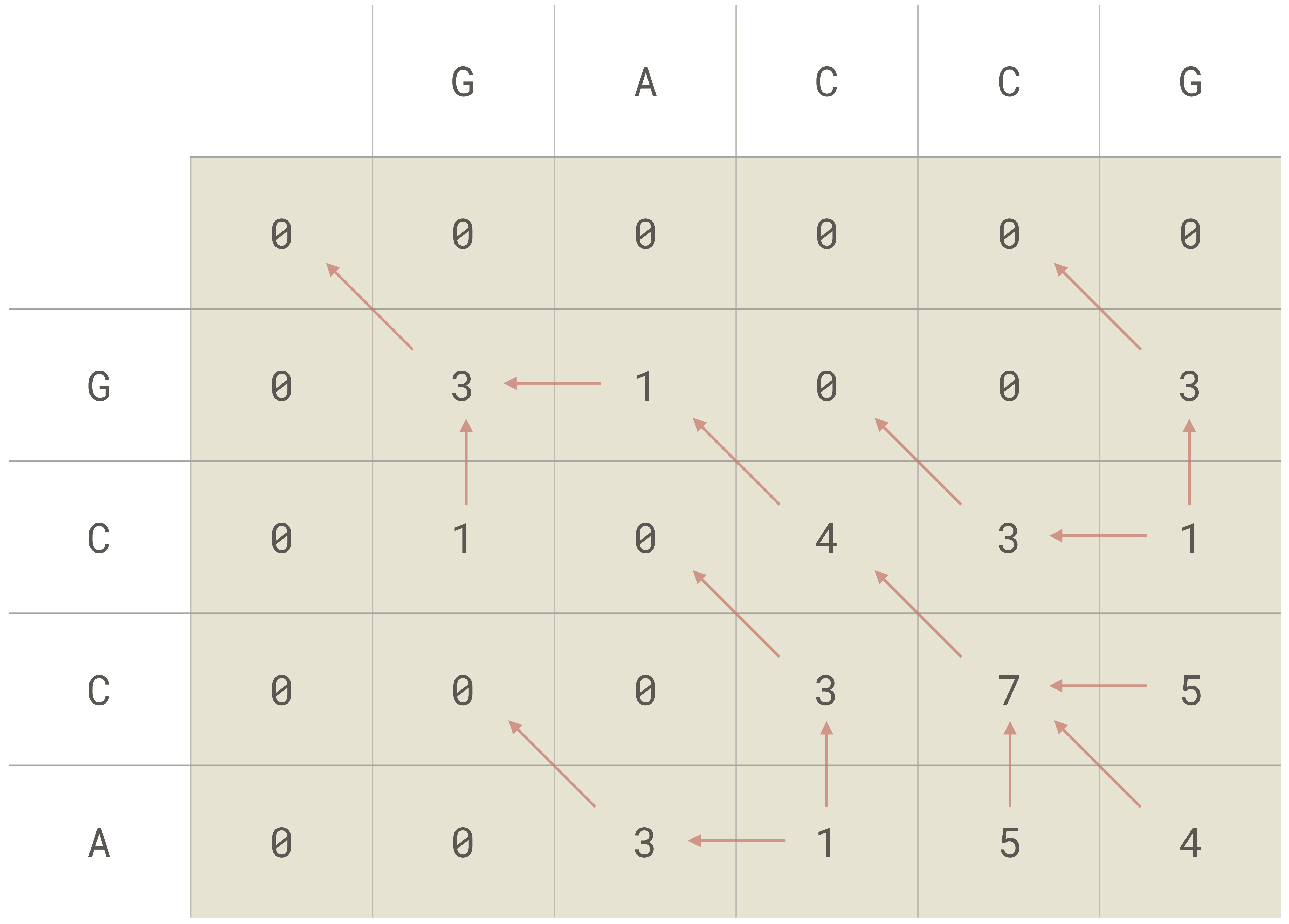
		G	A	C	C	G
	0	0	0	0	0	0
G	0	3	1	0		
C	0					
C	0					
A	0					

S_{ij} score is maximum of:

- $S_{i-1,j-1}$ + score_func(A[i], B[j])
- $S_{i,j-1}$ + gap_penalty()
- $S_{i-1,j}$ + gap_penalty()
- Zero

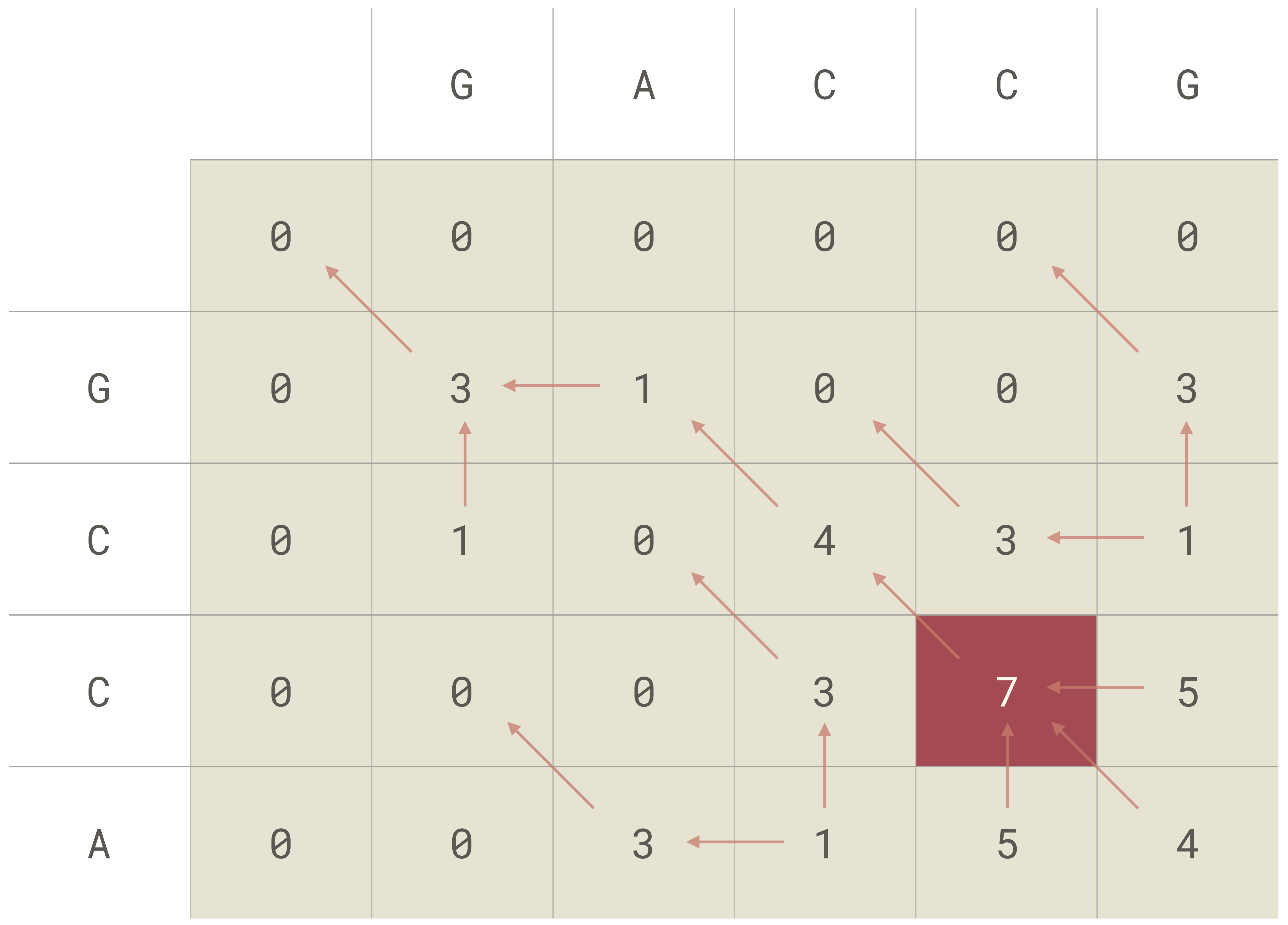
```
score_func(a, b):
  3 if a == b else -3

gap_penalty():
  -2
```



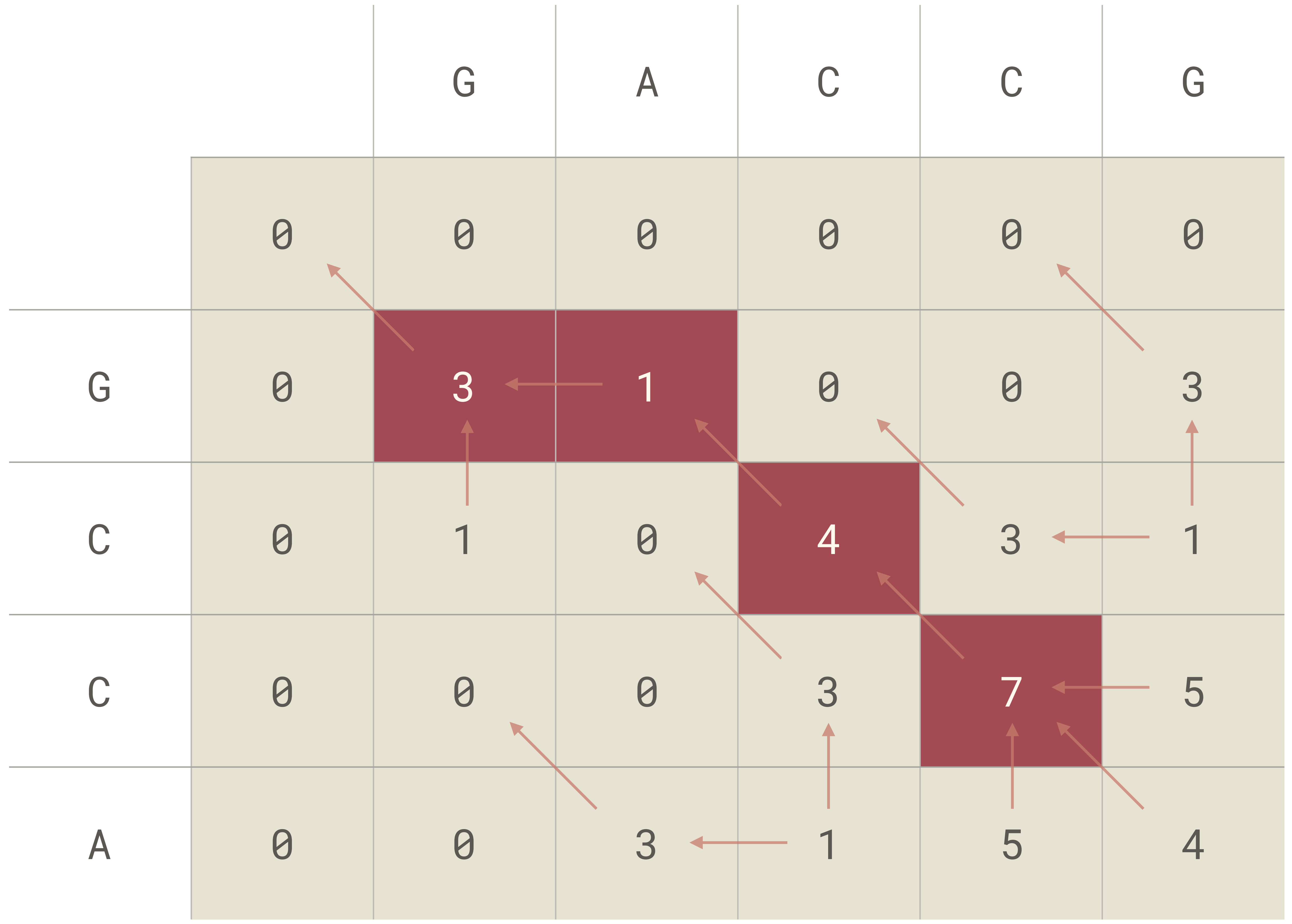
		G	A	C	C	G
	0	0	0	0	0	0
G	0	3	1	0	0	3
C	0	1	0	4	3	1
C	0	0	0	3	7	5
A	0	0	3	1	5	4

Find maximum score(s)



		G	A	C	C	G
	0	0	0	0	0	0
G	0	3	1	0	0	3
C	0	1	0	4	3	1
C	0	0	0	3	7	5
A	0	0	3	1	5	4

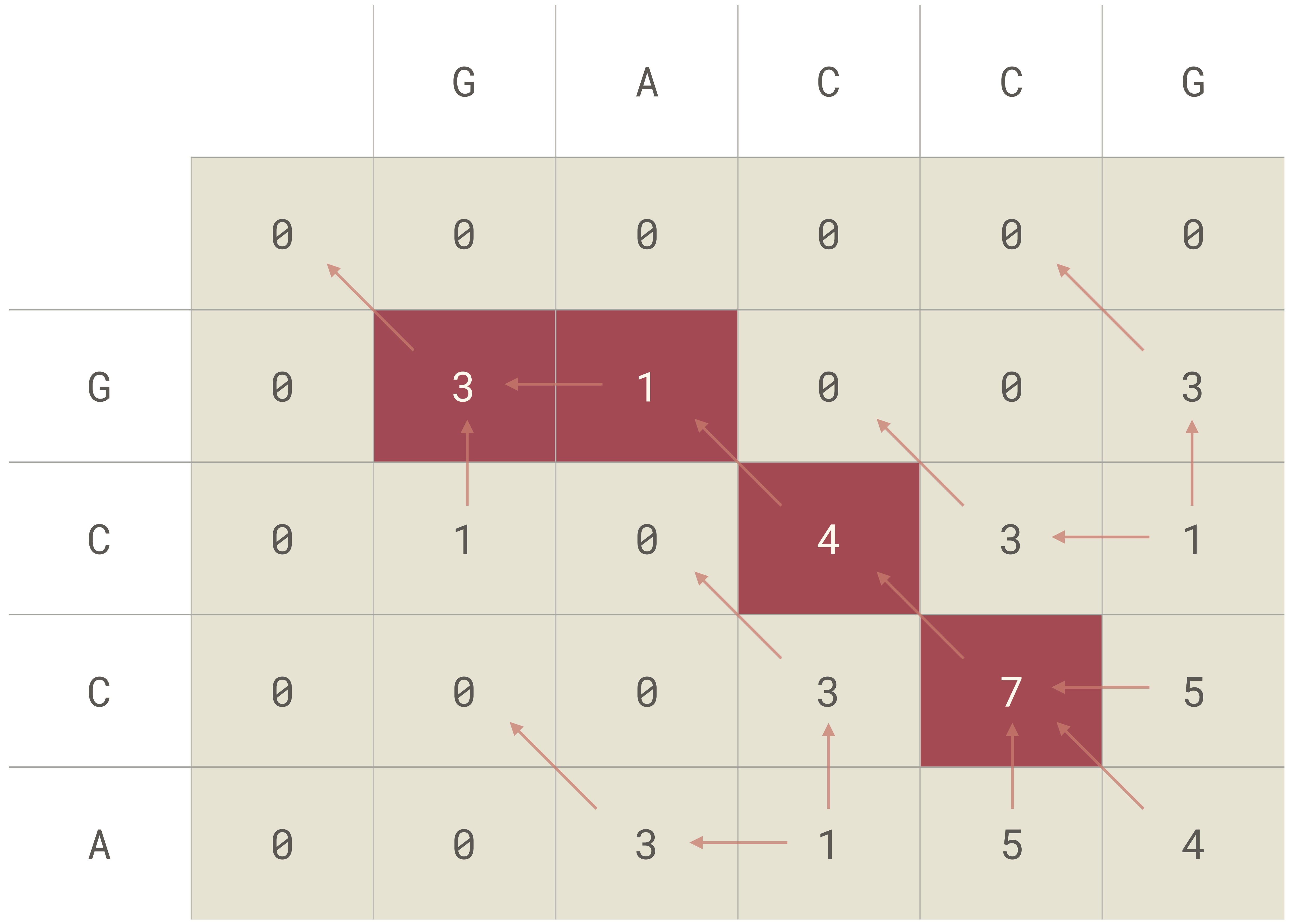
Backtrace to a zero



		G	A	C	C	G
	0	0	0	0	0	0
G	0	3	1	0	0	3
C	0	1	0	4	3	1
C	0	0	0	3	7	5
A	0	0	3	1	5	4

Alignment:

```
G A C C
| | | |
G – C C
```



		G	A	C	C	G
	0	0	0	0	0	0
G	0	3	1	0	0	3
C	0	1	0	4	3	1
C	0	0	0	3	7	5
A	0	0	3	1	5	4

Anchor Update

Align entire context with new source code

Find topic within alignment

Create new context from realigned topic

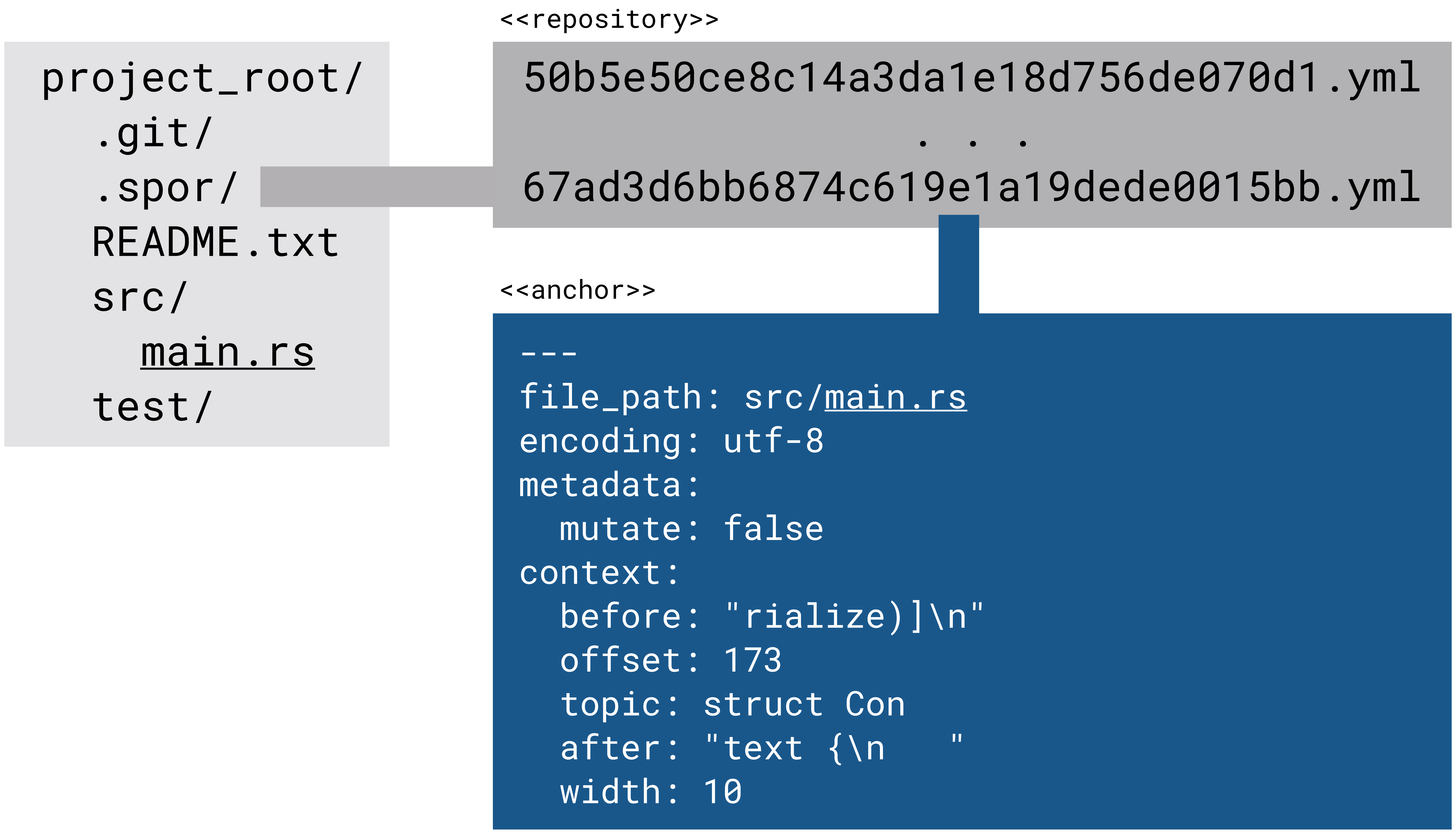
spor:
Anchored metadata

github.com/abingham/rust_spor

github.com/abingham/spor

Rust logo copyright Mozilla: https://creativecommons.org/licenses/by/4.0/

File structure



```
project_root/
  .git/
  .spor/
  README.txt
  src/
    main.rs
  test/
```

<<repository>>

```
50b5e50ce8c14a3da1e18d756de070d1.yml
              . . .
67ad3d6bb6874c619e1a19dede0015bb.yml
```

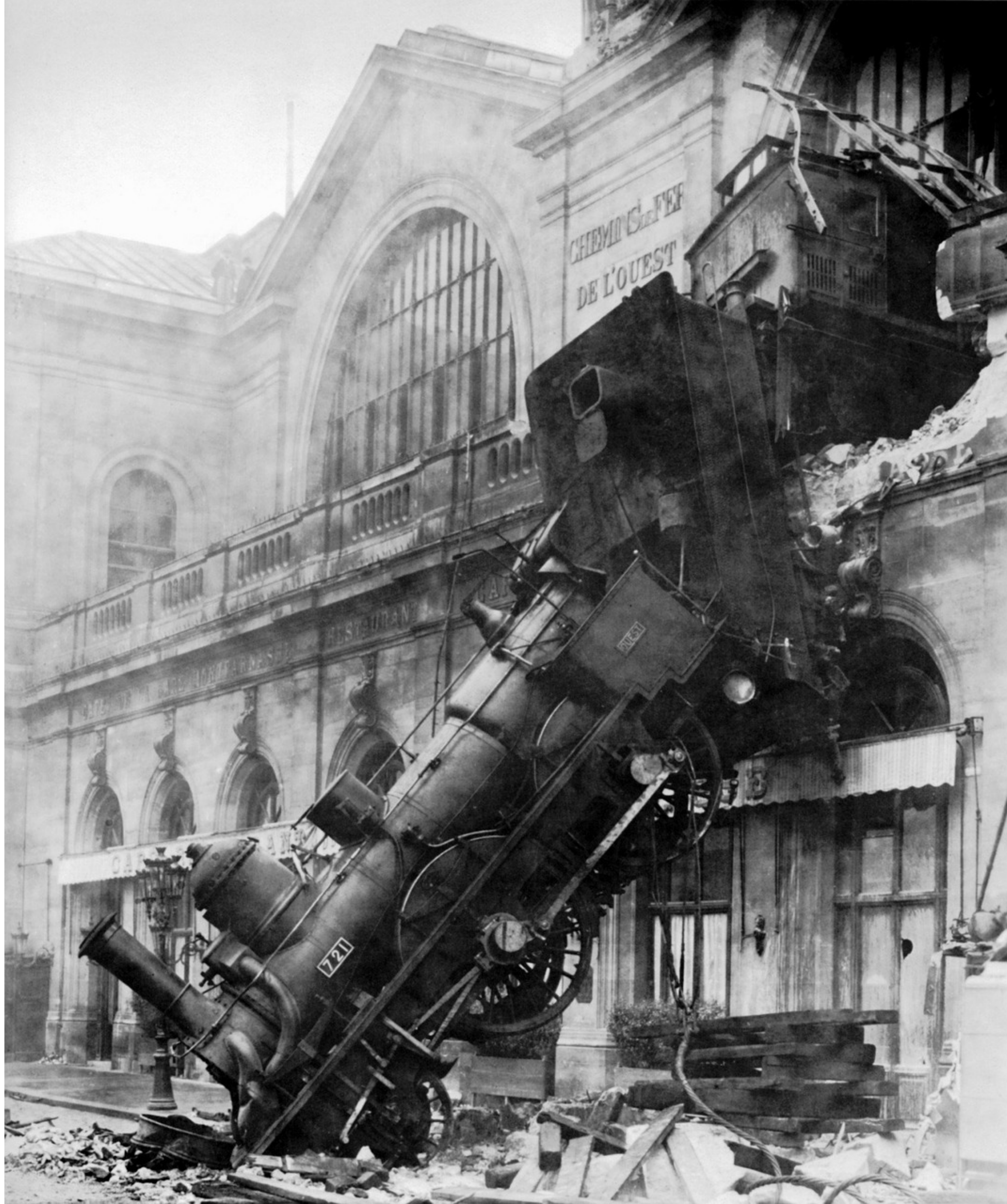
<<anchor>>

```
---
file_path: src/main.rs
encoding: utf-8
metadata:
  mutate: false
context:
  before: "rialize)]\n"
  offset: 173
  topic: struct Con
  after: "text {\n   "
  width: 10
```

Command-line Interface

```
$ spor -h
Usage:
  spor init
  spor add <source-file> <offset> <width> <context-width>
  spor list <source-file>
  spor details <id>
  spor diff <anchor-id>
  spor status
  spor update
```

Demo



Future Work

IDE Integration

.gitignore script.md anchor.rs ! 9bbc4665-01bf-4b69-90b9-7103964e763f.yml (deleted from disk)

You, a month ago | 1 author (You)

```
use std::cmp::max;       You, 7 months ago • Repository starting to come together.
use std::fs::File;
use std::io::{BufReader, Error, ErrorKind, Read, Result, Seek, SeekFrom};
use std::path::{Path, PathBuf};

#[derive(Debug, Deserialize, Serialize)]
pub struct Context {
    before: String,
    offset: u64,
    topic: String,
    after: String,
    width: u64,
}

impl Context {
    pub fn from_path(
        path: &Path,
        offset: u64,
        width: u64,
        context_width: u64
```

```
Metadata: {meta: data}
ID: 9bbc4665-01bf-4b69-90b9-7103964e763f
```

Edit...

PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL 1: zsh

Anchoring directories

```
project_name
├── README.rst
├── setup.py
└── src
    └── package_name
        ├── __init__.py
        ├── version.py
        └── subpackage
            ├── mod1.py
            └── mod2.py
```

Do not mutate

Source Control Integration

```
git mv foo.py bar.py
```

$\Big\downarrow$

Update anchors to foo.py

And much more!

- Third-party Smith-Waterman
- Alternative tokenization
- Explore scoring functions
- Function ensembles
- Storing anchor history
- Match-quality warnings
- Semantic anchors

Python to Rust

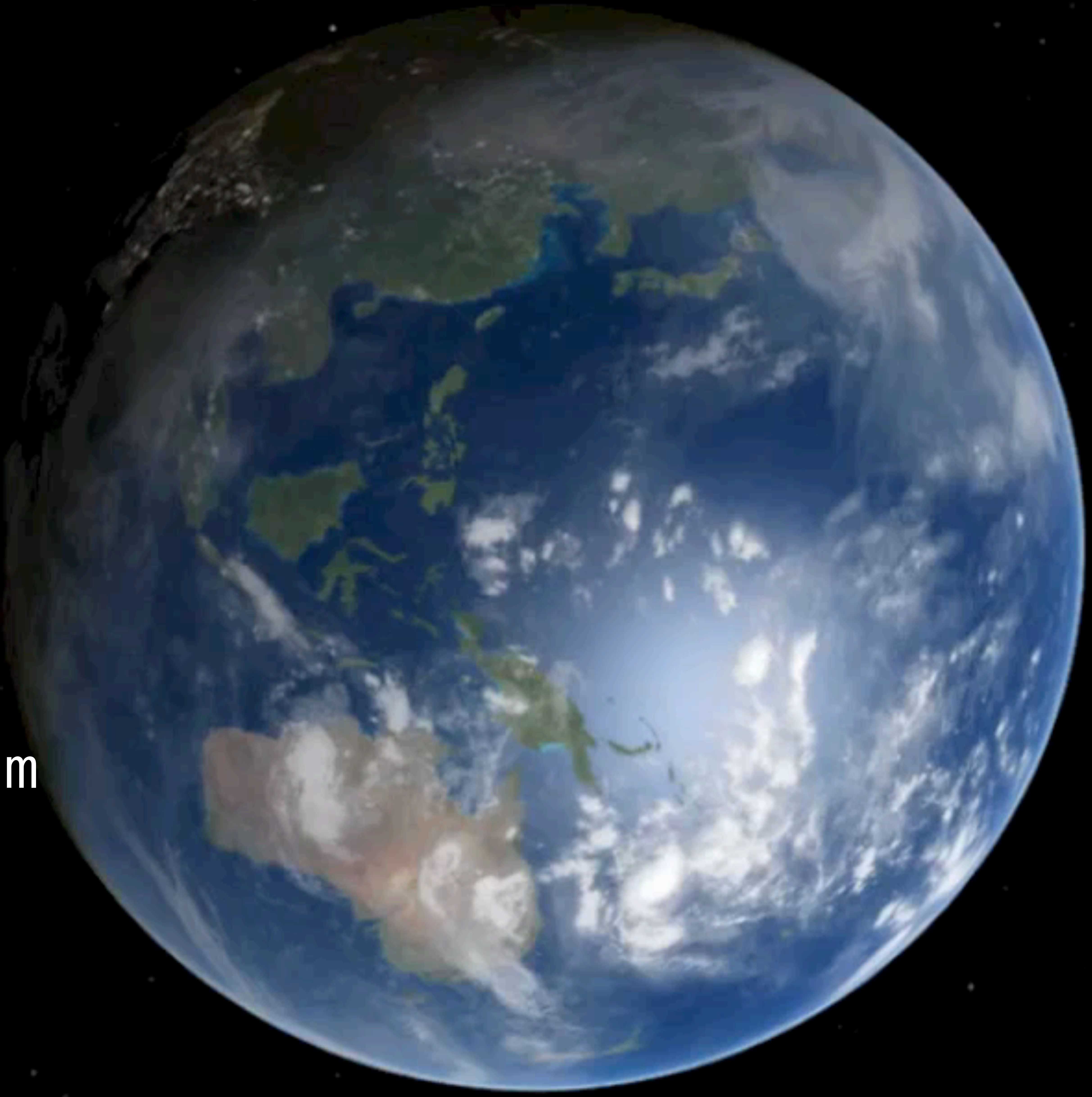
Speed

Curiosity!

Learning curve

Very positive!

- Nice tooling
- Fast development cycle
- Fast execution
- Maintainable
- Robust (feeling!)



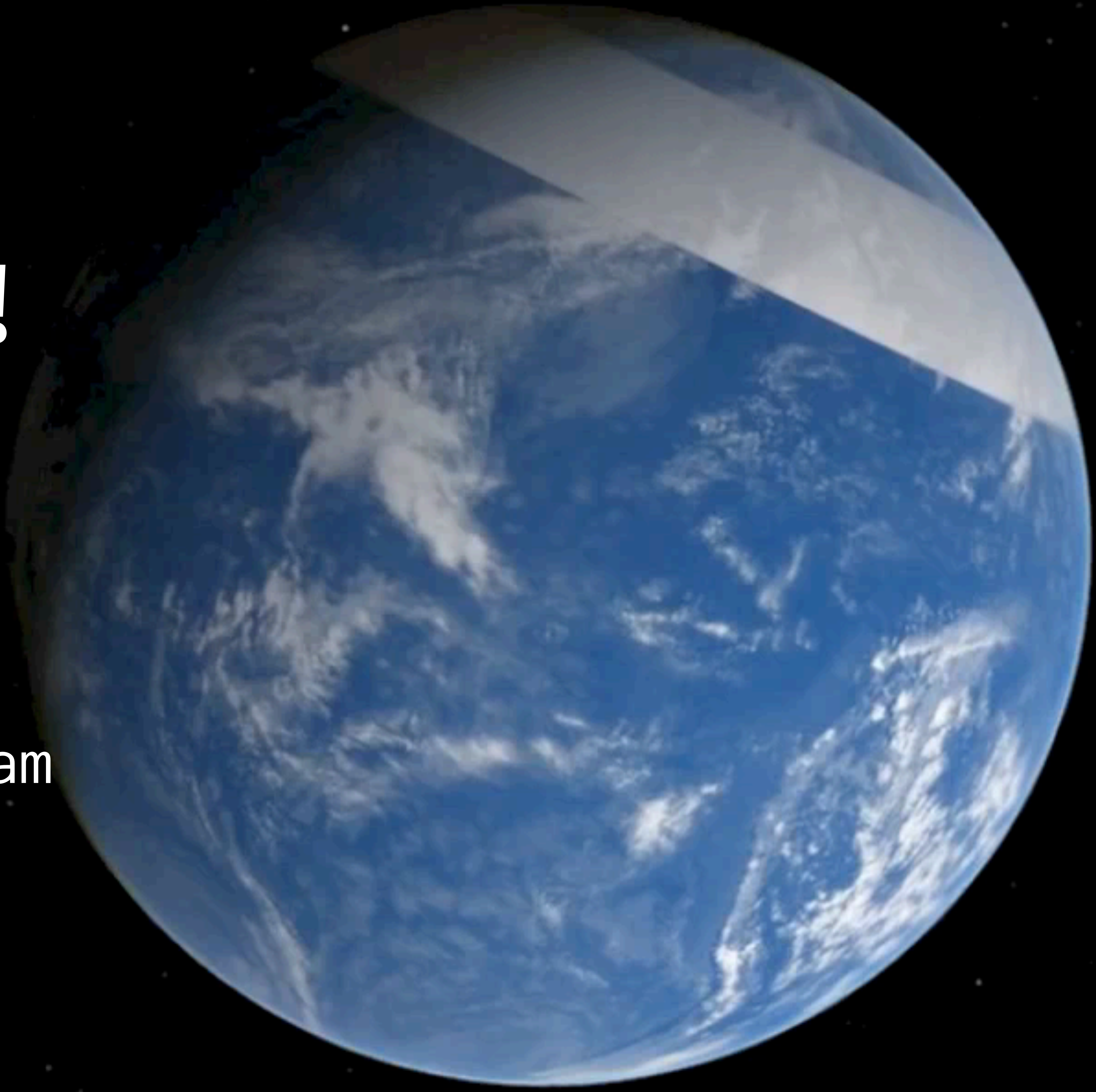
Thank you!

Austin Bingham

@austin_bingham

SixtyNORTH

@sixty_north



Thank you!

Austin Bingham

@austin_bingham

SixtyNORTH

@sixty_north