Beyond pdb

Delta Debugging in Python

Dr. Kristian Rother

krother@academis.eu

www.academis.eu

ACADEMIS

Python is not easy to debug

```
data = ["There are two kinds of people",
        "First, those who finish",
        "what they start"


SyntaxError: unexpected EOF while parsing
```

Python is not easy to debug

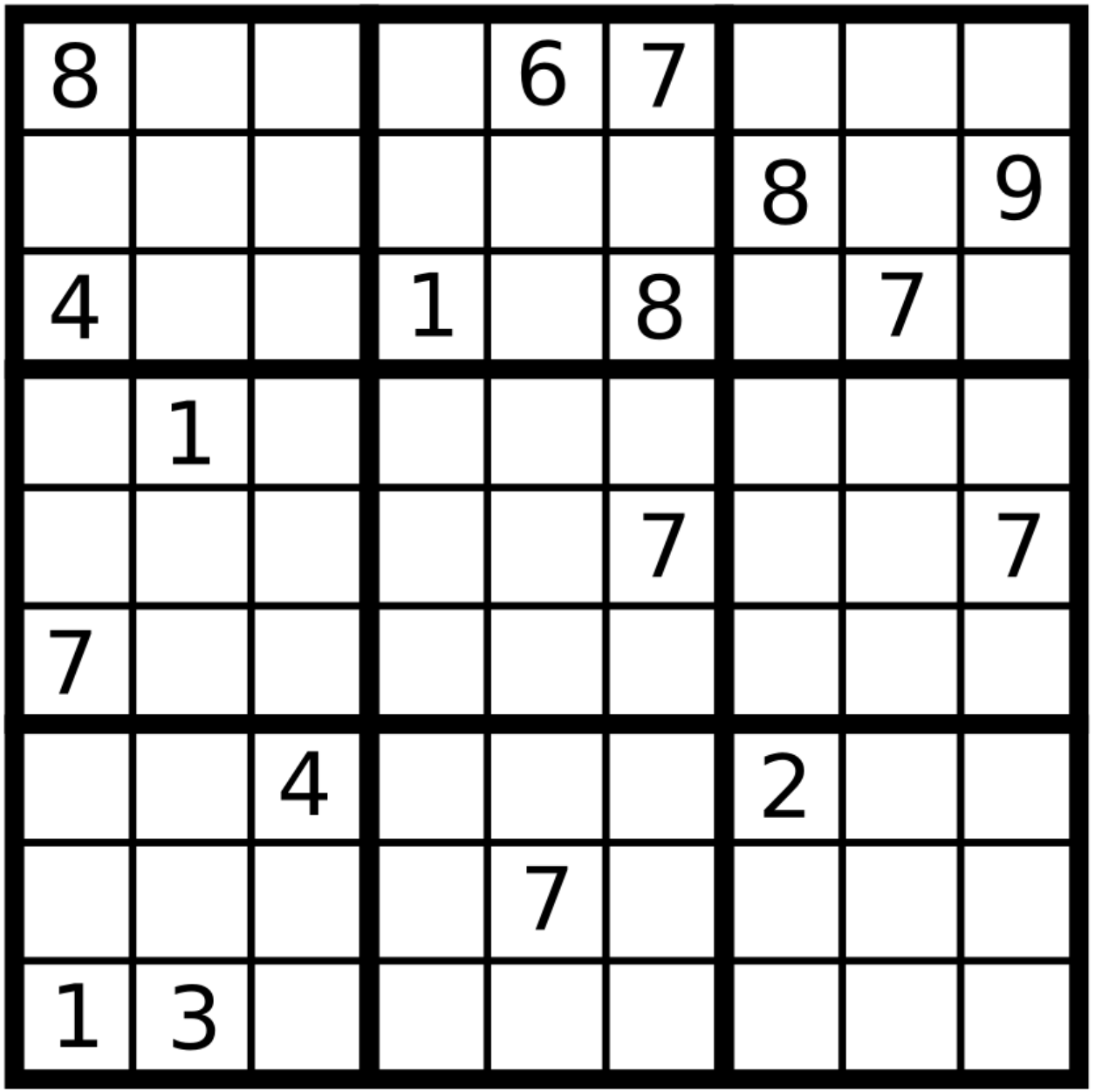
```
data = "There are 10 kinds of people",
       "those who can read binary",
       "and those who don't"]


IndentationError: unexpected indent
```

Reasons your program is failing:

- The code is buggy
- The tests are buggy
- The data is buggy
- You have no idea which of the three applies

8	4	1	3	6	7	5	2	9
5	6	7	9	2	4	8	3	1
2	9	3	1	5	8	4	7	6
3	1	8	4	9	6	7	5	2
4	5	9	2		1	6	8	3
7	2	6	8	3	5	1	9	4
9	8	4	7	1	3	2	6	5
6	7	2	5	4	9	3	1	8
1	3	5	6	8	2	9	4	7



ACADEMIS

Sudoku Solver

- Formulated as linear equations
- `import pulp`
- 333+ constraints
- *using an LES is overkill*

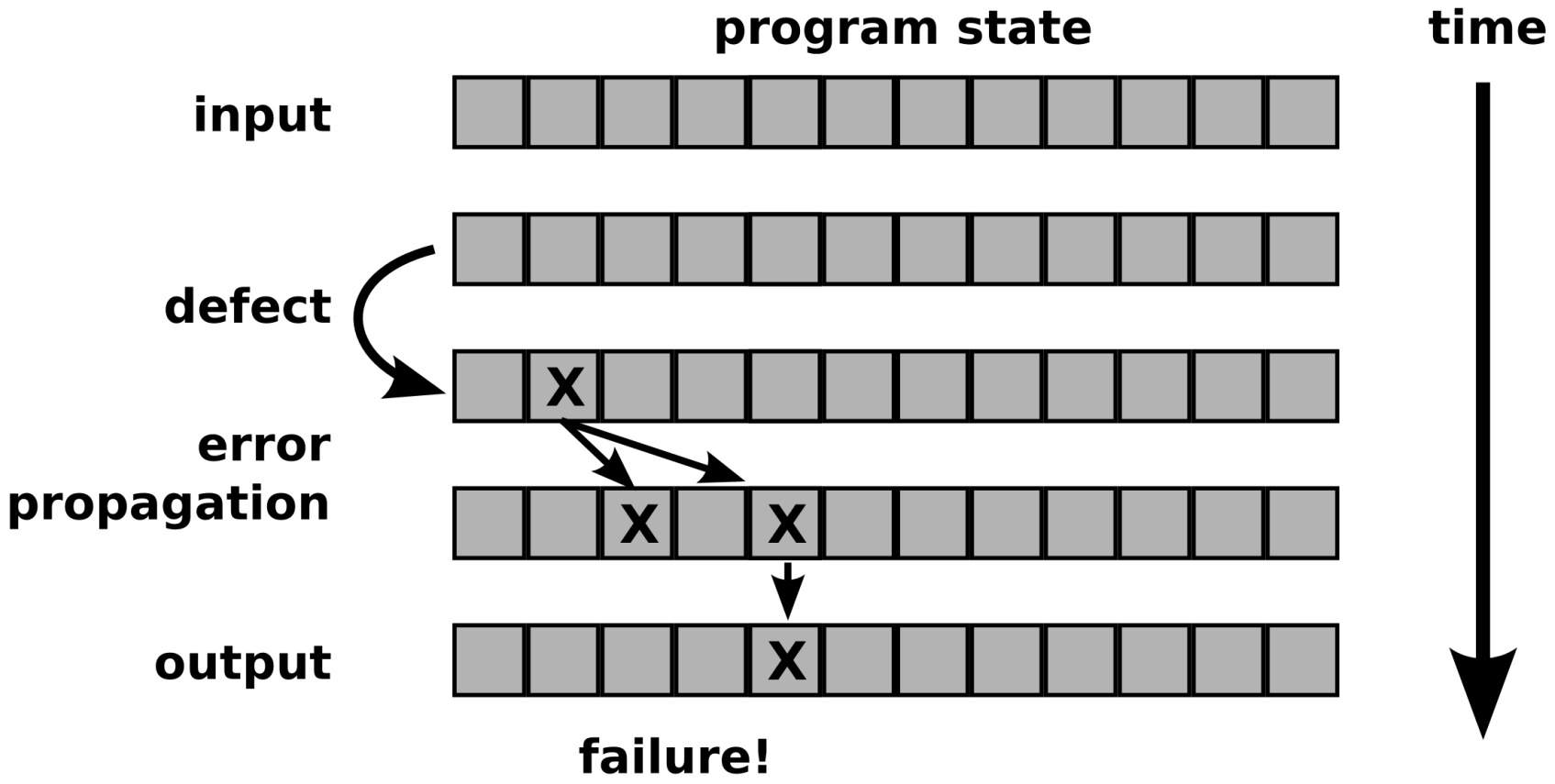
8	4	1	3	6	7	5	2	9
5	6	7	9	2	4	8	3	1
2	9	3	1	5	8	4	7	6
3	1	8	4	9	6	7	5	2
4	5	9	2	7	1	6	8	3
7	2	6	8	3	5	1	9	4
9	8	4	7	1	3	2	6	5
6	7	2	5	4	9	3	1	8
1	3	5	6	8	2	9	4	7

ACADEMIS

Andreas Zeller

- Udacity course: **“Software Debugging”**
- Author of the book: **“Why Programs Fail”**
- Published Delta Debugging article in 1999

Program failures and defects



ACADEMIS

Beyond pdb

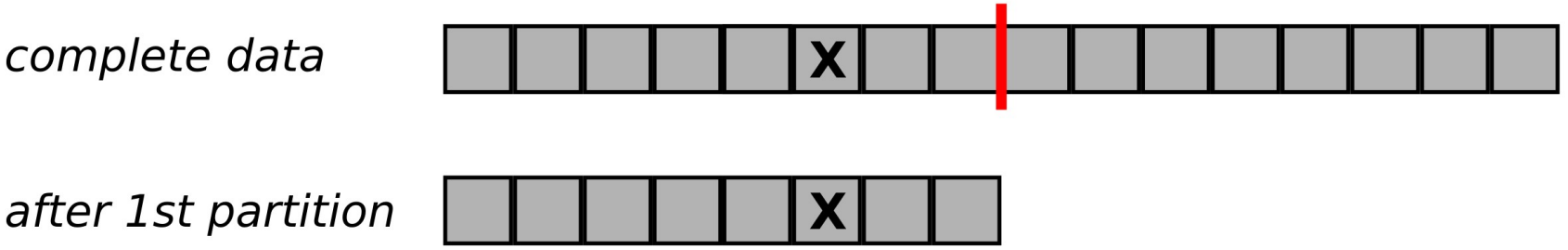
Partitioning data

ACADEMIS

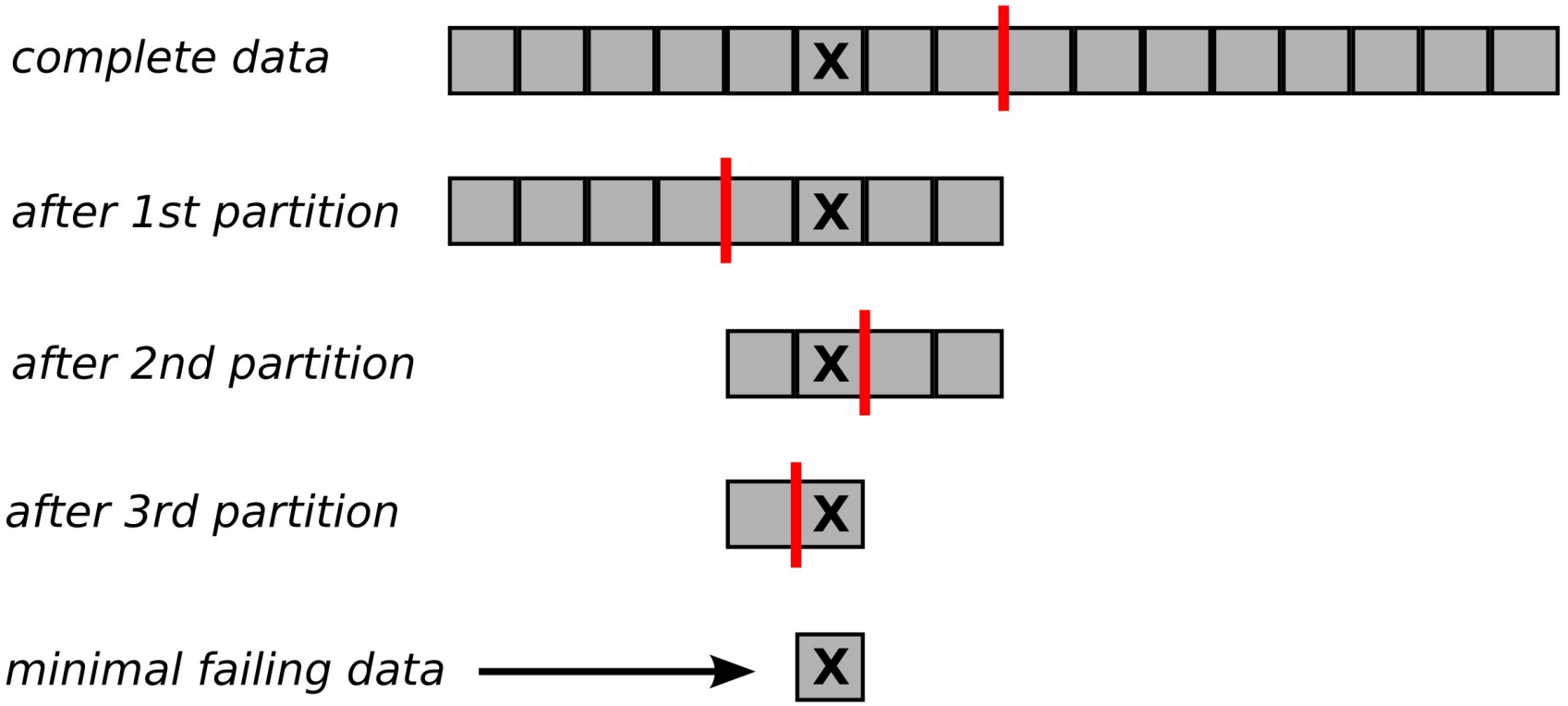
Partitioning data with binary search

complete data

Partitioning data with binary search



Partitioning data with binary search



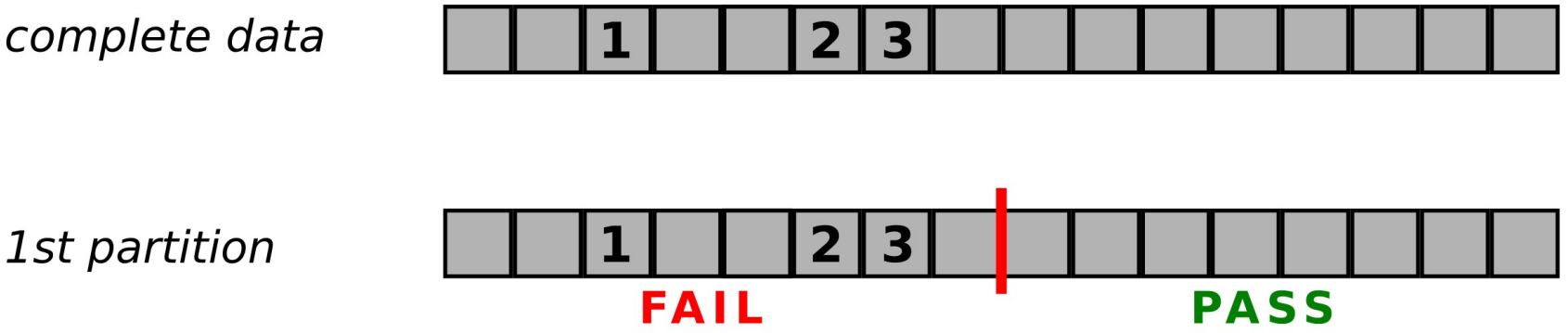
Beyond pdb

Partitioning data

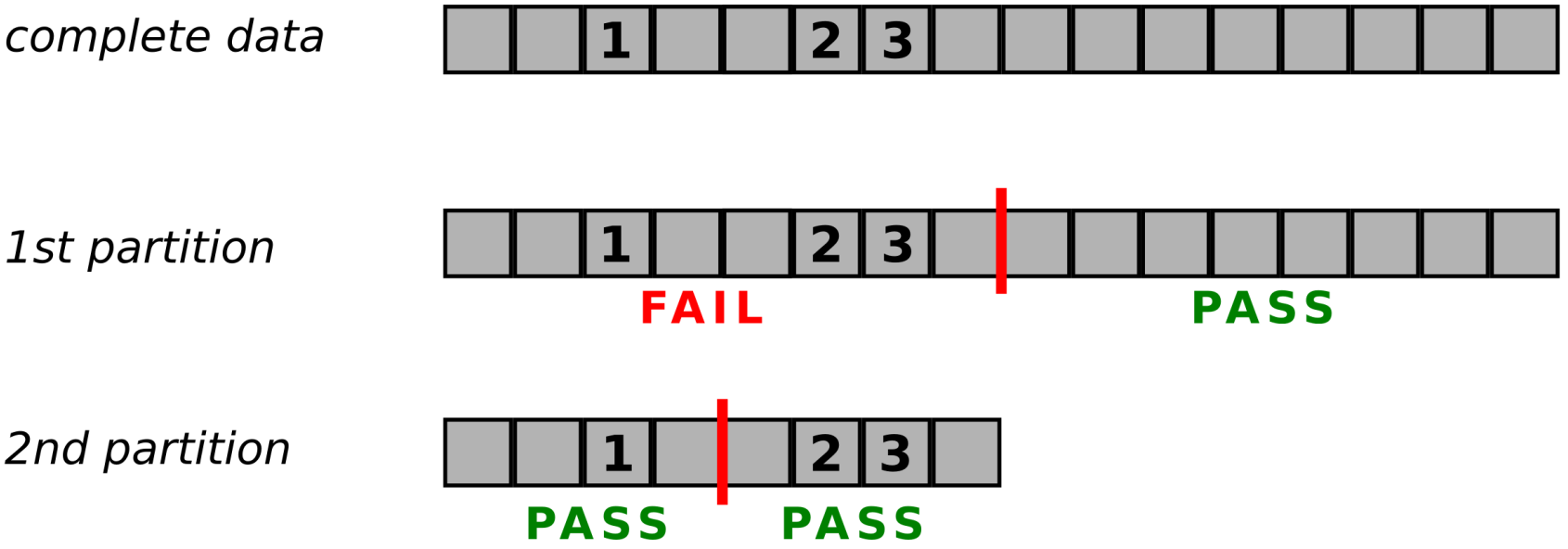
with increasing granularity

ACADEMIS

1, 2, 3 only fail together



1, 2, 3 only fail together



ACADEMIS

1, 2, 3 only fail together

after 1st partition

re-partitioning (granularity=3)

FAIL

PASS

PASS

ACADEMIS

1, 2, 3 only fail together

after 2nd partition

FAIL

3rd partition
(granularity=3)

PASS

PASS

PASS

ACADEMIS

1, 2, 3 only fail together

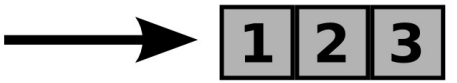
after 2nd partition

FAIL

3rd partition (granularity=4)

PASS

FAIL



PASS

PASS

ACADEMIS

Beyond pdb

Partitioning data

with increasing granularity

is called Delta Debugging.

Delta Debugging results in a minimal failing subset

Time Complexity

Best case:

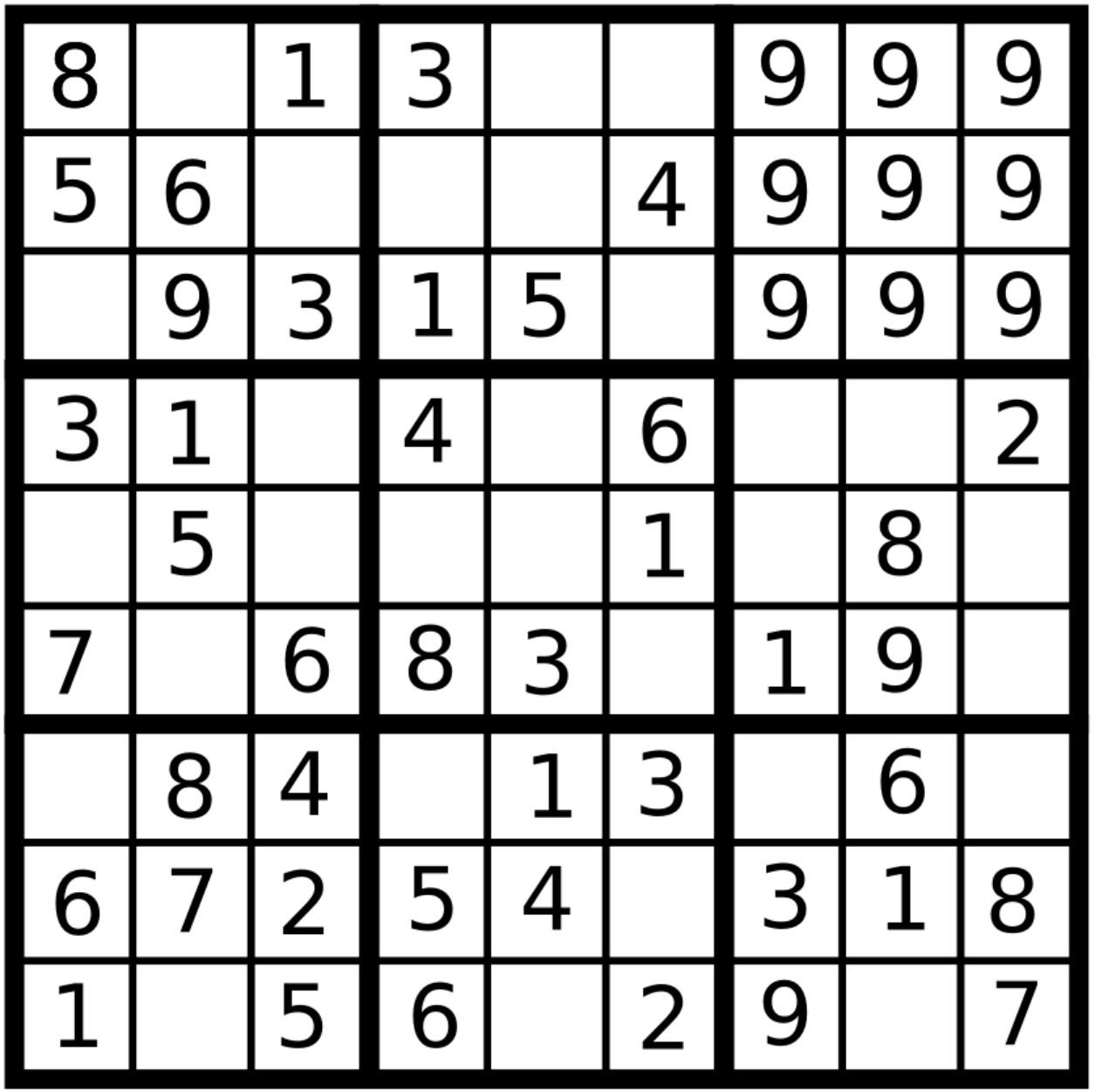
all fail

O(log n)

Worst case:

all pass

O(n)


8 1 3 9 9 9
5 6 4 9 9 9
9 3 1 5 9 9 9
3 1 4 6 2
5 8
7 6 8 3 1 9
8 4 1 3 6
6 7 2 5 4 3 1 8
1 5 6 2 9 7

Delta Debugging

Pros

- Does not require knowlede about the nature of the bug
- Works on any iterable type
- Works well with random tests
- Related approach used by `git bisect`

Cons

- You need a test first
- Partitions must not have side effects
- Few examples documented since 1999
- Does not give the **union** of failing subsets

ACADEMIS

Beyond pdb

Partitioning data

with increasing granularity

is called Delta Debugging.

Delta Debugging results in a minimal failing subset

for any partitionable type

(HTML, key strokes, git commits, Sudoku)

Source code and references

github.com/krother/delta_debug

Dr. Kristian Rother

krother@academis.eu

www.academis.eu

ACADEMIS

Debugging Techniques

A biased and incomplete enumeration

Debug	Test	Analyze	Fail early
print	unittest	pylint	data invariants
introspection	py.test	pyflakes	contracts
pdb	doctest	mypy	assertions
deduction	Jenkins	reviews	logging
git bisect	Selenium	Z-notation	loop invariants

ACADEMIS