

# CSE P 504

Advanced topics in Software Systems

Fall 2022

## Delta Debugging

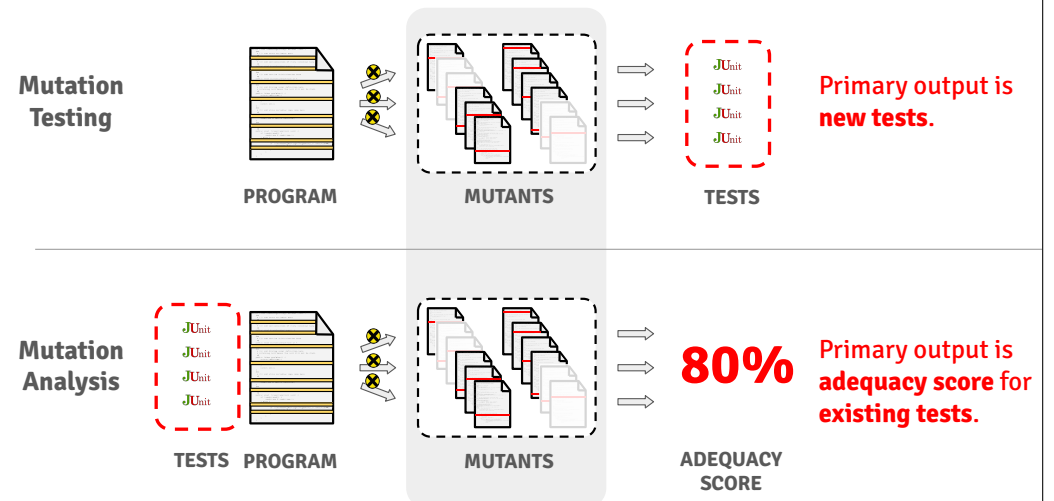
October 31, 2022

## Today

- Coverage-based testing: open discussion
  - Any open questions about coverage-based testing?
- Mutation-based testing: open discussion
  - What have you observed and learned?
  - Challenges: mutant comprehension vs. test creation?
  - Any open questions about mutation-based testing?
- Delta Debugging
  - Motivating examples
  - Live demo
  - A little quiz
  - Discussion

## Coverage- and Mutation-based testing: discussion

## Recap: Mutation Testing vs. Mutation Analysis



How expensive is mutation testing?

Is the mutation score meaningful?

## Delta Debugging

## This is a crashing test case

```
<td align=left valign=top>
<SELECT NAME="op sys" MULTIPLE SIZE=7>
<OPTION VALUE="All">All
<OPTION VALUE="Windows 3.1">Windows 3.1
<OPTION VALUE="Windows 95">Windows 95
<OPTION VALUE="Windows 98">Windows 98
<OPTION VALUE="Windows ME">Windows ME
<OPTION VALUE="Windows 2000">Windows 2000
<OPTION VALUE="Windows NT">Windows NT
<OPTION VALUE="Mac System 7">Mac System 7
<OPTION VALUE="Mac System 7.5">Mac System 7.5
<OPTION VALUE="Mac System 7.6.1">Mac System 7.6.1
<OPTION VALUE="Mac System 8.0">Mac System 8.0
<OPTION VALUE="Mac System 8.5">Mac System 8.5
<OPTION VALUE="Mac System 8.6">Mac System 8.6
<OPTION VALUE="Mac System 9.x">Mac System 9.x
<OPTION VALUE="MacOS X">MacOS X
<OPTION VALUE="Linux">Linux
<OPTION VALUE="BSDI">BSDI
<OPTION VALUE="FreeBSD">FreeBSD
<OPTION VALUE="NetBSD">NetBSD
<OPTION VALUE="OpenBSD">OpenBSD
<OPTION VALUE="AIX">AIX
<OPTION VALUE="BeOS">BeOS
<OPTION VALUE="HP-UX">HP-UX
<OPTION VALUE="IRIX">IRIX
<OPTION VALUE="Neutrino">Neutrino
<OPTION VALUE="OpenVMS">OpenVMS
<OPTION VALUE="OS/2">OS/2
<OPTION VALUE="OSF/1">OSF/1
<OPTION VALUE="Solaris">Solaris
<OPTION VALUE="SunOS">SunOS
<OPTION VALUE="other">other</SELECT></td>
<td align=left valign=top>
<SELECT NAME="priority" MULTIPLE SIZE=7>
<OPTION VALUE="--">--<OPTION VALUE="P1">P1<OPTION VALUE="P2">P2<OPTION
VALUE="P3">P3<OPTION VALUE="P4">P4<OPTION
VALUE="P5">P5</SELECT>
</td>
<td align=left valign=top>
<SELECT NAME="bug severity" MULTIPLE SIZE=7>
<OPTION VALUE="blocker">blocker<OPTION VALUE="critical">critical<OPTION
VALUE="major">major<OPTION
VALUE="normal">normal<OPTION VALUE="minor">minor<OPTION
VALUE="trivial">trivial<OPTION VALUE="enhancement">enhancement</SELECT>
</tr>
</table>
```

- Crashed Mozilla
- How would you debug the problem?

## This is a crashing test case

```
<td align=left valign=top>
<SELECT NAME="op sys" MULTIPLE SIZE=7>
<OPTION VALUE="All">All
<OPTION VALUE="Windows 3.1">Windows 3.1
<OPTION VALUE="Windows 95">Windows 95
<OPTION VALUE="Windows 98">Windows 98
<OPTION VALUE="Windows ME">Windows ME
<OPTION VALUE="Windows 2000">Windows 2000
<OPTION VALUE="Windows NT">Windows NT
<OPTION VALUE="Mac System 7">Mac System 7
<OPTION VALUE="Mac System 7.5">Mac System 7.5
<OPTION VALUE="Mac System 7.6.1">Mac System 7.6.1
<OPTION VALUE="Mac System 8.0">Mac System 8.0
<OPTION VALUE="Mac System 8.5">Mac System 8.5
<OPTION VALUE="Mac System 8.6">Mac System 8.6
<OPTION VALUE="Mac System 9.x">Mac System 9.x
<OPTION VALUE="MacOS X">MacOS X
<OPTION VALUE="Linux">Linux
<OPTION VALUE="BSDI">BSDI
<OPTION VALUE="FreeBSD">FreeBSD
<OPTION VALUE="NetBSD">NetBSD
<OPTION VALUE="OpenBSD">OpenBSD
<OPTION VALUE="AIX">AIX
<OPTION VALUE="BeOS">BeOS
<OPTION VALUE="HP-UX">HP-UX
<OPTION VALUE="IRIX">IRIX
<OPTION VALUE="Neutrino">Neutrino
<OPTION VALUE="OpenVMS">OpenVMS
<OPTION VALUE="OS/2">OS/2
<OPTION VALUE="OSF/1">OSF/1
<OPTION VALUE="Solaris">Solaris
<OPTION VALUE="SunOS">SunOS
<OPTION VALUE="other">other</SELECT></td>
<td align=left valign=top>
<SELECT NAME="priority" MULTIPLE SIZE=7>
<OPTION VALUE="--">--<OPTION VALUE="P1">P1<OPTION VALUE="P2">P2<OPTION
VALUE="P3">P3<OPTION VALUE="P4">P4<OPTION
VALUE="P5">P5</SELECT>
</td>
<td align=left valign=top>
<SELECT NAME="bug severity" MULTIPLE SIZE=7>
<OPTION VALUE="blocker">blocker<OPTION VALUE="critical">critical<OPTION
VALUE="major">major<OPTION
VALUE="normal">normal<OPTION VALUE="minor">minor<OPTION
VALUE="trivial">trivial<OPTION VALUE="enhancement">enhancement</SELECT>
</tr>
</table>
```

- Crashed Mozilla
- How would you debug the problem?
- A minimal test case is: <SELECT>
- Can we automate the process of minimizing test cases?
- What's the naive approach for an optimal solution?

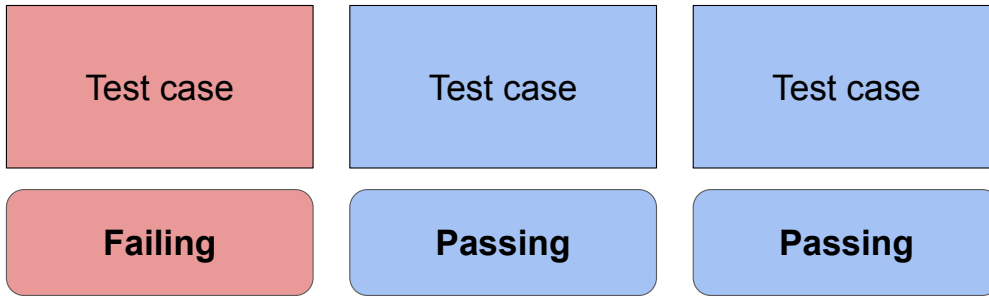
## Minimizing test cases

Test case

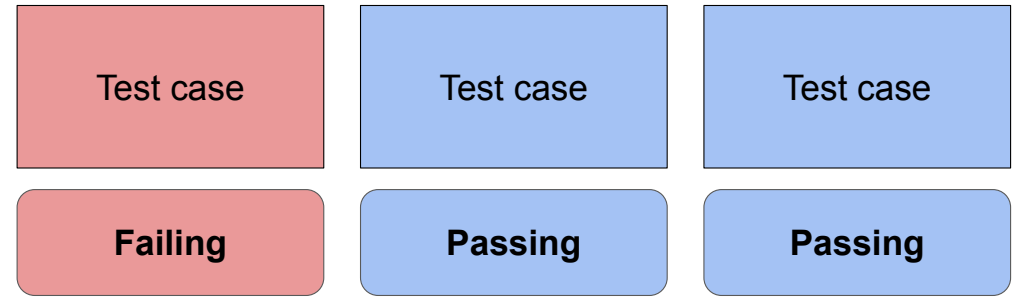
Test case

Test case

## Minimizing test cases

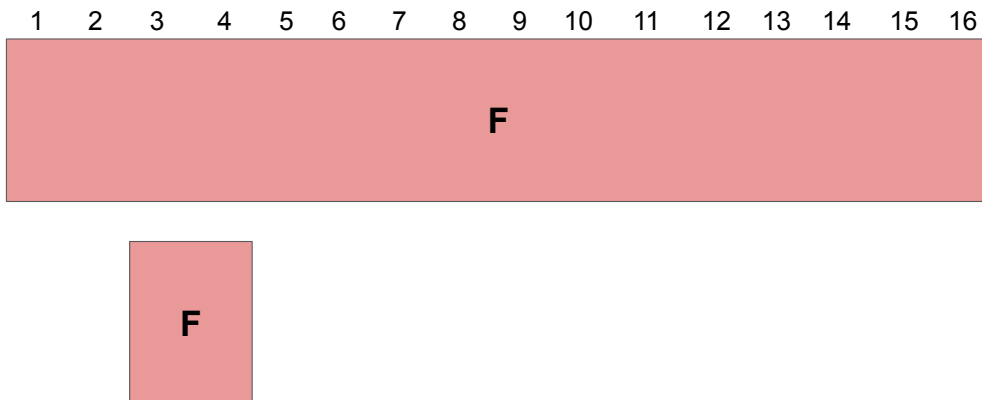
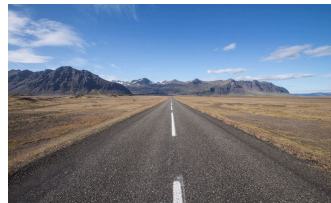


## Minimizing test cases



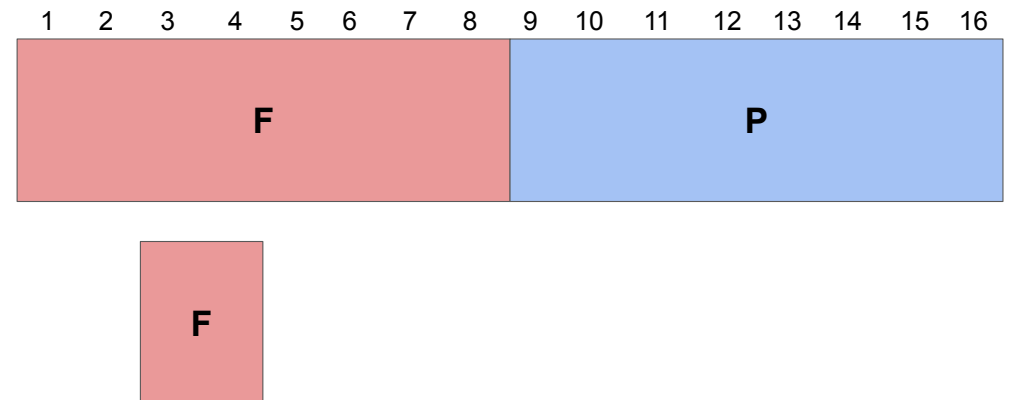
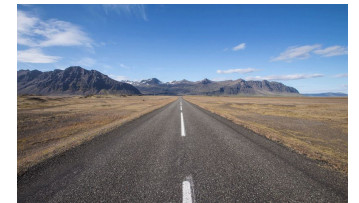
**Goal: Minimize the failing test case**

## The happy path: binary search

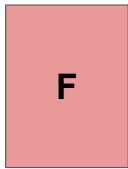


**Failing test with 16 lines.  
The minimal test has 2 lines.**

## The happy path: binary search



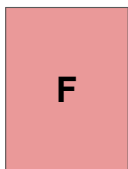
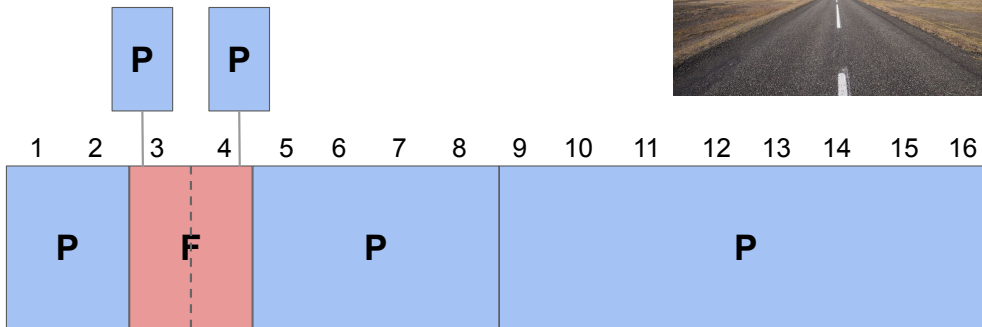
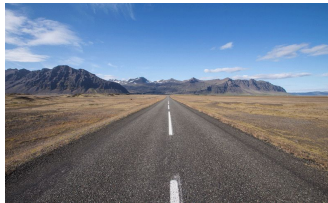
The happy path: binary search



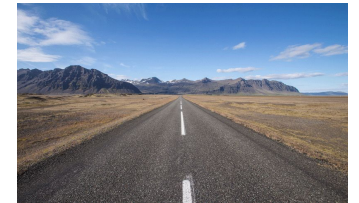
The happy path: binary search



The happy path: binary search



The happy path: binary search

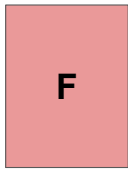


**Successfully minimized the failing test to 2 lines**

The not so happy path...



1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

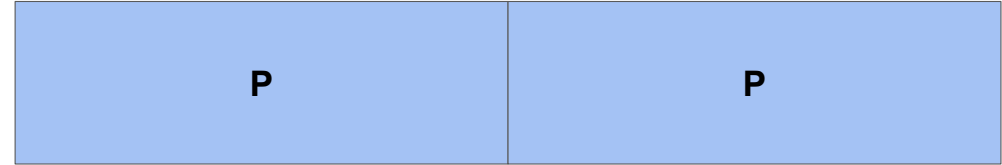


Suppose the failure pattern is more complex.

The not so happy path...



1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16



Binary search does not give optimal results.

## Delta debugging: binary search + X

## The DD algorithm

### Minimizing Delta Debugging Algorithm

Let  $test$  and  $c_x$  be given such that  $test(\emptyset) = \checkmark \wedge test(c_x) = \times$  hold.  
 The goal is to find  $c'_x = dmin(c_x)$  such that  $c'_x \subseteq c_x$ ,  $test(c'_x) = \times$ , and  $c'_x$  is 1-minimal.  
 The minimizing Delta Debugging algorithm  $dmin(c)$  is

$$dmin(c_x) = dmin_2(c_x, 2) \quad \text{where}$$

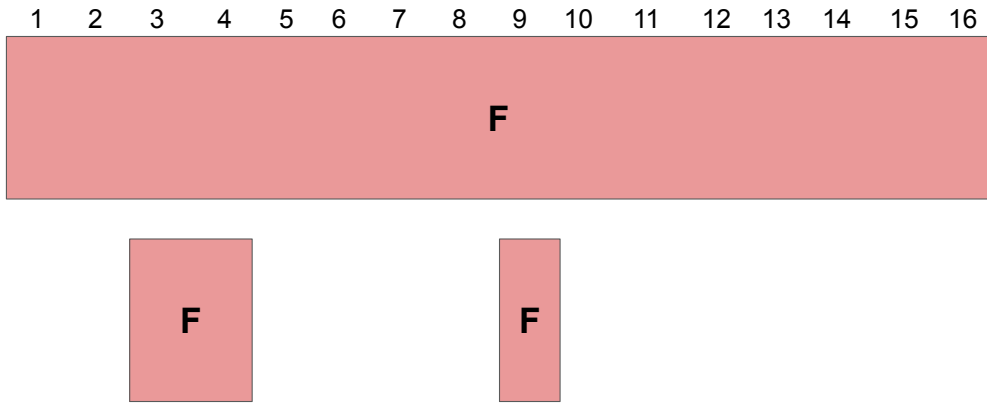
$$dmin_2(c'_x, n) = \begin{cases} dmin_2(\Delta_i, 2) & \text{if } \exists i \in \{1, \dots, n\} \cdot test(\Delta_i) = \times \text{ ("reduce to subset")} \\ dmin_2(\nabla_i, \max(n-1, 2)) & \text{else if } \exists i \in \{1, \dots, n\} \cdot test(\nabla_i) = \times \text{ ("reduce to complement")} \\ dmin_2(c'_x, \min(|c'_x|, 2n)) & \text{else if } n < |c'_x| \text{ ("increase granularity")} \\ c'_x & \text{otherwise ("done").} \end{cases}$$

where  $\nabla_i = c'_x - \Delta_i$ ,  $c'_x = \Delta_1 \cup \Delta_2 \cup \dots \cup \Delta_n$ , all  $\Delta_i$  are pairwise disjoint, and  $\forall \Delta_i \cdot |\Delta_i| \approx |c'_x|/n$  holds.  
 The recursion invariant (and thus precondition) for  $dmin_2$  is  $test(c'_x) = \times \wedge n \leq |c'_x|$ .

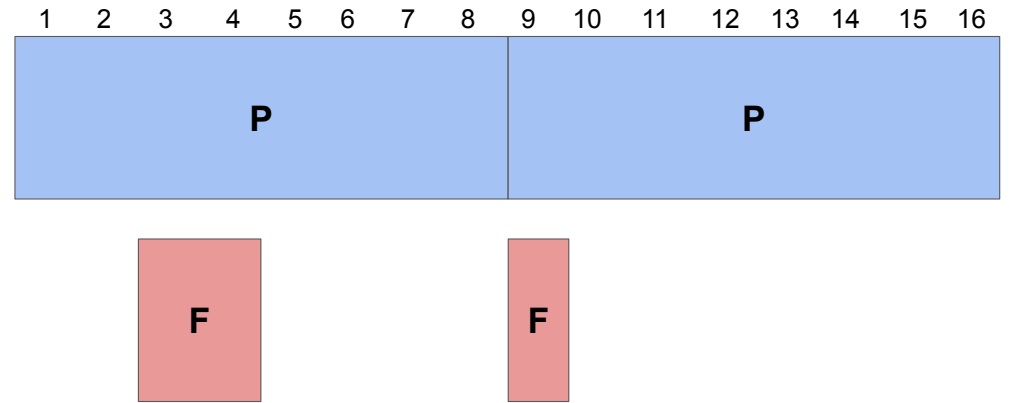
### Four basic steps:

1. Test each subset
2. Test each complement
3. Increase granularity
4. Reduce

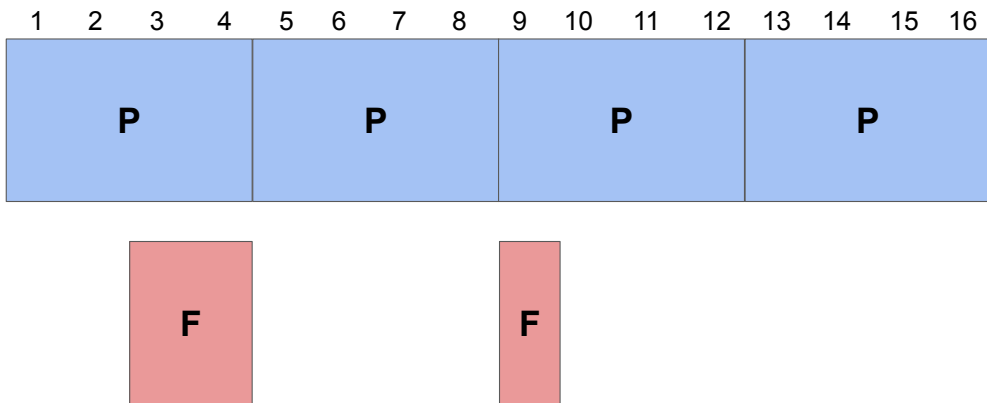
### Delta Debugging: mostly binary search



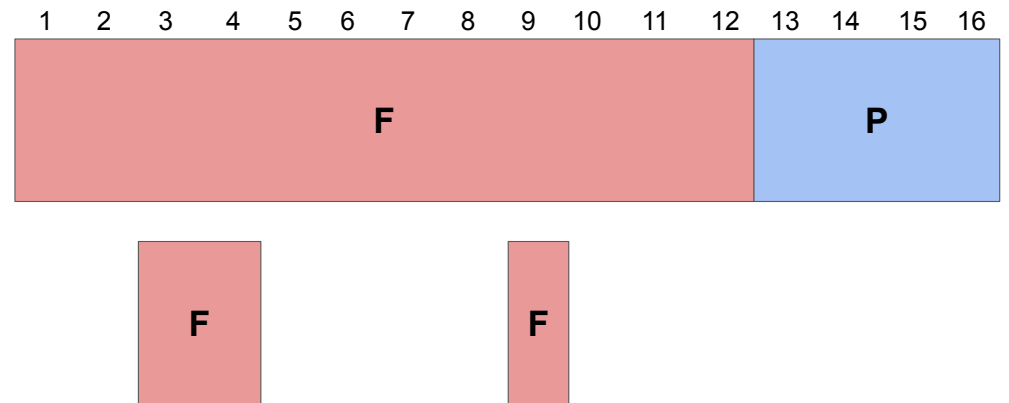
### Delta Debugging: mostly binary search



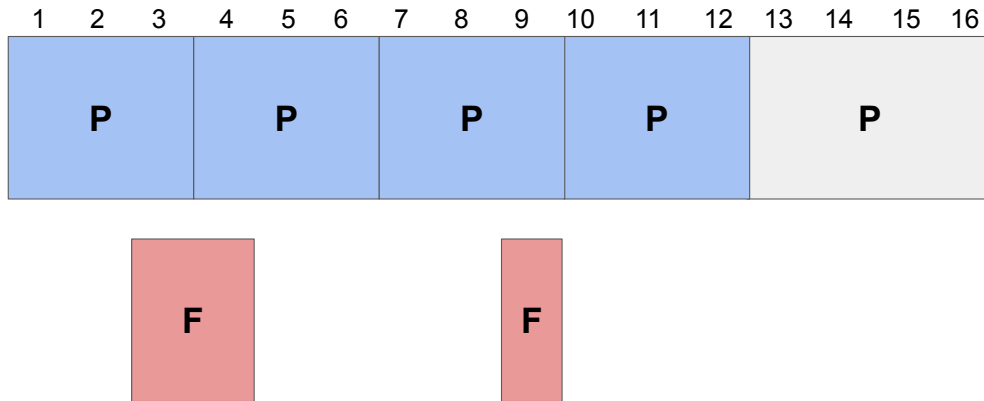
### Delta Debugging: granularity



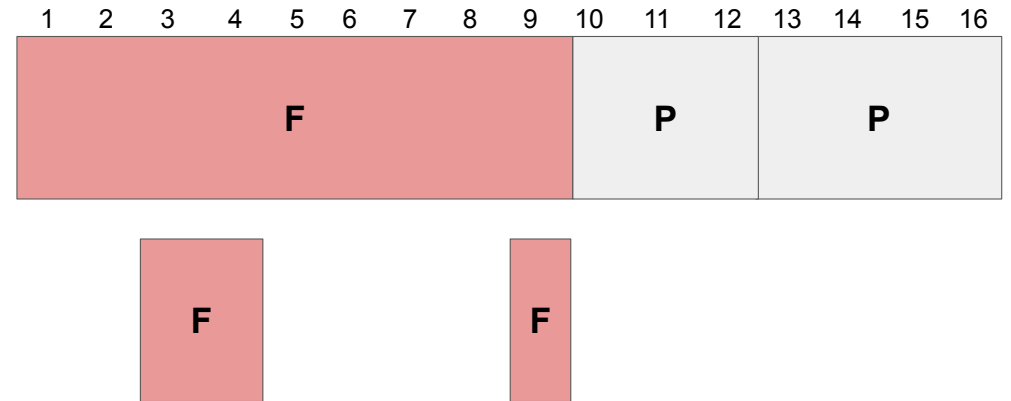
### Delta Debugging: complements



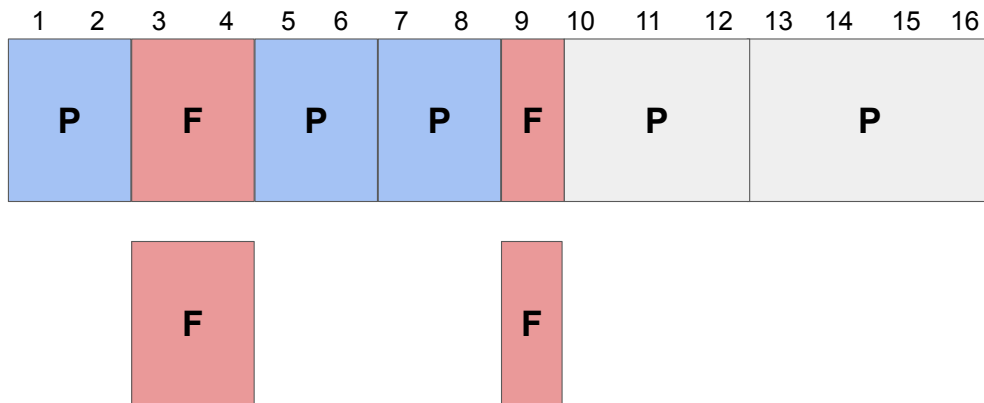
## Delta Debugging: reduce



## Delta Debugging: reduce



## Delta Debugging: 1-minimality



**Failure inputs must be deterministic and monotone.**

## Delta debugging: live examples



## A little quiz

### Program and initial test case

- Program  $P$  takes as input a **String of  $a_s$  and  $b_s$** .
- $P$  **crashes** whenever the input String contains an **even number of  $a_s$  AND an odd number of  $b_s$** .
- Assume **character-level** granularity.
- **Initial crashing test case** is: **babab**.

### Determine the following test cases (using DD)

1. Smallest
2. Local minimum but not smallest
3. 1-minimal of size 3



## A little quiz

### Program and initial test case

- Program  $P$  takes as input a **String of  $a_s$  and  $b_s$** .
- $P$  **crashes** whenever the input String contains an **even number of  $a_s$  AND an odd number of  $b_s$** .
- Assume **character-level** granularity.
- **Initial crashing test case** is: **babab**.

### Determine the following test cases (using DD)

1. Smallest **b**
2. Local minimum but not smallest **NONE**
3. 1-minimal of size 3 **aab**



## A little quiz

### Program and initial test case

- Program  $P$  takes as input an **Array of integers  $a$** .
- $P$  **crashes** whenever  $a$  contains **42**.
- **Initial crashing test case** is: **2424**

### Complete the following table

Iteration	$n$	input	$\Delta_1, \dots, \Delta_n$ $\nabla_1, \dots, \nabla_n$
1		2424	



## A little quiz

### Program and initial test case

- Program  $P$  takes as input an **Array of integers  $a$** .
- $P$  **crashes** whenever  $a$  contains **42**.
- **Initial crashing test case** is: **2424**

### Complete the following table

Iteration	$n$	input	$\Delta_1, \dots, \Delta_n$ $\nabla_1, \dots, \nabla_n$
1	2	2424	24, (24)
2	4	2424	2, 4, (2), (4), 424, 224, 244, 242
3	3	424	(4), (2), (4), (24), 44, 42
4	2	42	(4),( 2)



## Delta debugging: summary

### **Discussion**

- Non-deterministic programs
- Input structure and granularity
- Monotonicity
- Complexity