Debugging

Sergey Mechtaev

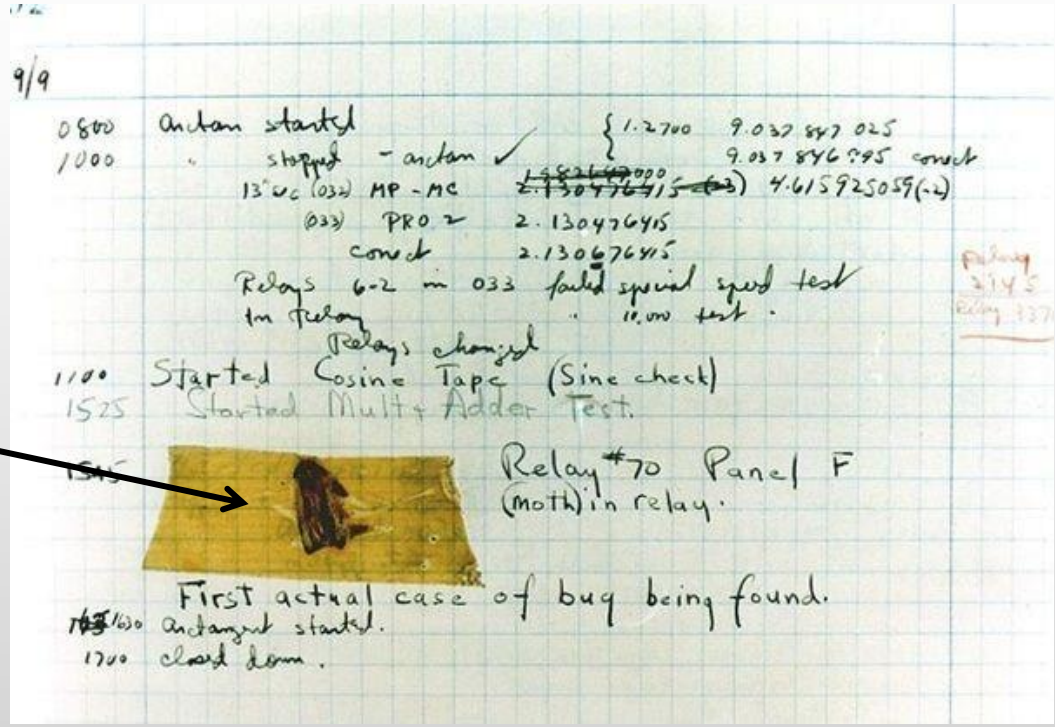
mechtaev@pku.edu.cn

Peking University

Debugging

Definition. The process of identifying and removing errors from computer hardware or software.

"The first bug"
(September 9, 1947)



Cost of Debugging

On average, developers spend **50%** of their programming time finding and fixing bugs.

This inefficiency is estimated to cost the global economy **$312 billion** annually.

Cambridge University, 2013

Debugging techniques

- Defensive programming
- Printf debugging
- Interactive debuggers
- System debuggers
- Delta debugging
- Automated program repair

Print Statements

"The most effective debugging tool is still careful thought, coupled with judiciously placed print statements.“ – Brian Kernighan

Famous programmers who prefer print statements:

- Linus Torvalds (Linux)
- Robert C. Martin (Agile Manifesto)
- Brian W. Kernighan and Rob Pike (UNIX)
- Guido van Rossum (Python)

Java Debugger – jdb

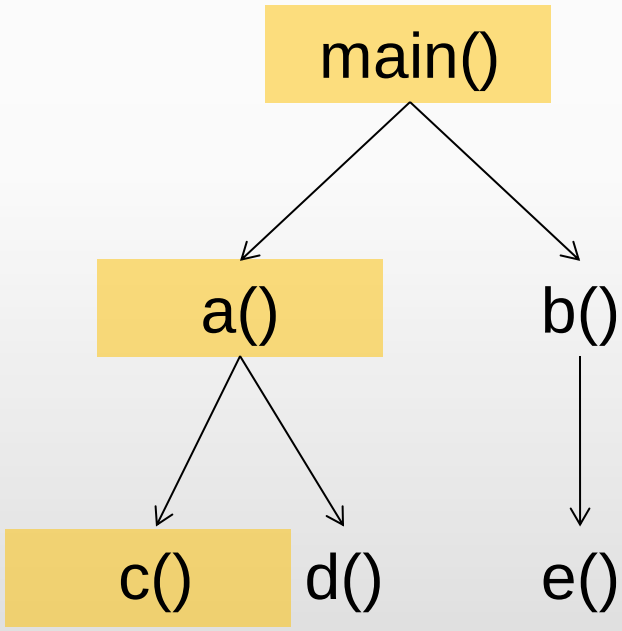
- Breakpoints – location or condition under which we want the execution to be suspended
  - Line Breakpoint
  - Method Breakpoint
  - Expression Breakpoint
  - Exception Breakpoint
- Allows to inspect:
  - Values of variables
  - Call stack
  - Threads

Call Stack

Methods

```
void main() {
    a();
    b();
}
void a() {
    c();
    d();
}
void b() {
    e();
}
```

Calls



System Debuggers – procfs

The **proc filesystem (procfs)** is a special filesystem in Unix-like operating systems that presents information about processes.

Useful files/directories:

- **/proc/PID/cmdline**, the command that originally started the process.
- **/proc/PID/cwd**, a symlink to the current working directory of the process.
- **/proc/PID/environ** contains the names and values of the environment variables that affect the process.
- **/proc/PID/fd**, a directory containing a symbolic link for each open file descriptor.

System Debugger – strace

Utility for Linux that monitors system calls, signal deliveries, and changes of process state.

- **open** – opens files
- **execve** – starts programs
- **write** – writes to files

Delta-debugging

Definition. A methodology to automate the debugging of programs using a scientific approach of hypothesis-trial-result loop.

Applications:

- Isolating failure-inducing program input (e.g. an HTML page that makes a Web browser fail)
- Isolating failure-inducing user interaction (e.g. the keystrokes that make a program crash)
- Isolating failure-inducing changes to the program code (e.g. after a failing regression test)

Delta-debugging

- Given
  - A set $C = \{c_1, c_2, \ldots, c_n\}$
  - A function $\text{Interesting}: C \rightarrow \{\text{Yes}, \text{No}\}$
  - $\text{Interesting}(C) = \text{Yes}$
  - Interesting is monotonic, unambiguous and consistent
- The delta debugging algorithm returns a one-minimal Interesting subset M of C:
  - $\text{Interesting}(M) = \text{Yes}$
  - $\forall\, m \in M, \text{Interesting}(M \setminus \{m\}) = \text{No}$

Naïve Solution

- Try all subsets of C to find the smallest one that is $\mathrm{Interesting}$
- Problem: there are $2^{|C|}$ subsets
- We want a polynomial-time solution

Assumptions

- Monotonic
  - $\text{Interesting}(X) \Rightarrow \text{Interesting}(X \cup \{c\})$
- Unambiguous
  - $\text{Interesting}(X) \wedge \text{Interesting}(Y) \Rightarrow \text{Interesting}(X \cap Y)$
- Consistent
  - $\text{Interesting}(X) = \text{Yes}$ or $\text{Interesting}(X) = \text{No}$

Interesting function

Valid examples

$$Interesting(C) \Leftrightarrow 3 \in C \wedge 8 \in C$$

where 3 and 8 are just some concrete elements

Monotonic, unambiguous, consistent

Invalid examples

- Interesting executes flacky test (that sometimes passes, sometimes fails) – not consistent
- $Interesting(C) \Leftrightarrow p \in C$ where p is any prime number – not unambiguous

Polynomial-Time Solution

$\mathrm{DD}(P, \{c_1, c_2, \ldots, c_n\}) =$

if $n = 1$ **then return** $\{c_1\}$

$P_1 = \{c_1, \ldots, c_{\frac{n}{2}}\}$

$P_2 = \{c_{\frac{n}{2}+1}, \ldots, c_n\}$

if $\mathrm{Interesting}(P \cup P_1)$ **then return** $\mathrm{DD}(P, P_1)$

if $\mathrm{Interesting}(P \cup P_2)$ **then return** $\mathrm{DD}(P, P_2)$

else return $\mathrm{DD}(P \cup P_1, P_2) \cup \mathrm{DD}(P \cup P_2, P_1)$

Example: {3,6} is smallest interesting subset of {1,…,8}

1 2 3 4 5 6 7 8	Interesting

Example: {3,6} is smallest interesting subset of {1,…,8}

1 2 3 4 5 6 7 8	Interesting
1 2 3 4	No

Example: {3,6} is smallest interesting subset of {1,…,8}

1 2 3 4 5 6 7 8	Interesting
1 2 3 4	No
. . . . 5 6 7 8	No

Example: {3,6} is smallest interesting subset of {1,…,8}

1 2 3 4 5 6 7 8	Interesting
1 2 3 4	No
. . . . 5 6 7 8	No
1 2 . . 5 6 7 8	No

Example: {3,6} is smallest interesting subset of {1,…,8}

1 2 3 4 5 6 7 8	Interesting
1 2 3 4	No
. . . . 5 6 7 8	No
1 2 . . 5 6 7 8	No
. . 3 4 5 6 7 8	Yes

Example: {3,6} is smallest interesting subset of {1,…,8}

1 2 3 4 5 6 7 8	Interesting
1 2 3 4	No
. . . . 5 6 7 8	No
1 2 . . 5 6 7 8	No
. . 3 4 5 6 7 8	Yes
. . 3 . 5 6 7 8	Yes

Example: {3,6} is smallest interesting subset of {1,…,8}

1 2 3 4 5 6 7 8	Interesting
1 2 3 4	No
. . . . 5 6 7 8	No
1 2 . . 5 6 7 8	No
. . 3 4 5 6 7 8	Yes
. . 3 . 5 6 7 8	Yes
1 2 3 4 5 6 . .	Yes

Example: {3,6} is smallest interesting subset of {1,…,8}

1 2 3 4 5 6 7 8	Interesting
1 2 3 4	No
. . . . 5 6 7 8	No
1 2 . . 5 6 7 8	No
. . 3 4 5 6 7 8	Yes
. . 3 . 5 6 7 8	Yes
1 2 3 4 5 6 . .	Yes
1 2 3 4 5 . . .	No

Example: {3,6} is smallest interesting subset of {1,…,8}

1 2 3 4 5 6 7 8	Interesting
1 2 3 4	No
. . . . 5 6 7 8	No
1 2 . . 5 6 7 8	No
. . 3 4 5 6 7 8	Yes
. . 3 . 5 6 7 8	Yes
1 2 3 4 5 6 . .	Yes
1 2 3 4 5 . . .	No
1 2 3 4 . 6 . .	Yes

Example: {3,6} is smallest interesting subset of {1,…,8}

1 2 3 4 5 6 7 8	Interesting
1 2 3 4	No
. . . . 5 6 7 8	No
1 2 . . 5 6 7 8	No
. . 3 4 5 6 7 8	Yes
. . 3 . 5 6 7 8	Yes
1 2 3 4 5 6 . .	Yes
1 2 3 4 5 . . .	No
1 2 3 4 . 6 . .	Yes