Orca

Differential Bug Localization in Large-Scale Services

Ranjita Bhagwan, Rahul Kumar, Chandra Maddila, Adithya Philip
Microsoft Research India
Devops Pipeline for Orion

**Code**
- 1000s of devs, 100s of commits/day

**Build**
- Dozens of builds/week

**Deploy**
- “Ring”-based deployment

**Monitor**
- Fine-grained monitoring
This is not traditional debugging | Can take hours!
State-of-the-art

• Look for related bugs [Lo et al., Srinivasa et al.]
  • Services are very dynamic and continuously changing. Similarity is limited!

• Instrument code and collect data [Liblit et al.]
  • Cannot disrupt current procedures

• Use extensive static/dynamic analysis [Lal et al.]
  • Too resource-intensive, the odd bug WILL slip through!
Post-Deployment Bugs: Observation

Textual similarity between symptom and code changes

Client-side

Operation not supported for type MailId

Support for datatype MailId added on server-side but not on a particular client

Server-side

Public MailSession GetMailSession(Object o, ClientType c, bool returnMailIds)
Textual similarity occurs between symptom and CLUSTER of commits made around the same time.
Post-Deployment Bugs: Observation

Buggy commit may not be part of symptomatic build

Build 1 → Build 2 → Build 3 → Build 4

Commit C to Build 3 caused the problem
Orca: Search Engine for Localizing Bugs

1. Differential Code Analysis of the Abstract Syntax Tree

2. Build Provenance Graph

3. Commit Risk Estimation
   Term relevance

Documents

Queries

On-call Engineer
Customer complaints
Alerts
Log error messages
Exceptions

RING 1
RING 2
RING 3
1. Differential Code Analysis

**Before**

Namespace Storage {
    ...
    class Connection {
        ...
        public Transaction BeginTx() {
            return Transaction.New(this);
        }
    }
}

**After**

Namespace Storage {
    ...
    class Connection {
        ...
        public Transaction BeginTx() {
            if (this.filter != null) {
                Transaction.MailFilter.Apply();
                this.source.PerfCounters.UpdateMailFilterCt();
            }
            return Transaction.New(this);
        }
    }
}

Namespace, Class, Method Changed: Storage, Connection, BeginTx
Custom tokenizer extracts important Keywords: Storage, Mail, Tx, MailFilter,
2. Build Provenance Graph

RING 1
1.0 \rightarrow 1.1 \rightarrow 1.2 \rightarrow 1.3 \rightarrow 1.4

RING 2
2.0 \rightarrow 2.1 \rightarrow 2.2

RING 3
3.1 \rightarrow 3.2

TIME

Security fix
Bug fix
Bug fix
Security fix
Alert!
3. Commit Risk Estimation

- Commits to some codepaths are more likely to cause bugs
- A developer with < 10 total commits is twice as likely to make a buggy commit
- Buggy commits had twice as many reviewer comments as safe commits
Presenting Orca Results
Orca Evaluation

Manually curated a list of $Q=48$ code-related incidents

Mean Reciprocal Rank (MRR) = \[
\frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i}
\]

<table>
<thead>
<tr>
<th>Metric</th>
<th>Results for top-10 ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Diff on AST</td>
</tr>
<tr>
<td>Recall</td>
<td>33</td>
</tr>
<tr>
<td>MRR</td>
<td>0.44</td>
</tr>
</tbody>
</table>
Orca Evaluation

Orca reduced the average number of commits that an On-Call Engineer has to examine by a factor of 3

“Awesome! And using this tool today we found [a] regression very quickly 😊”
Project Sankie

www.microsoft.com/en-us/research/project/sankie

CHLR:
Commit Risk Prediction

REX:
Related Entities

Reviewer Suggestions

Test/Build
FastLane:
Data-driven Test Minimization

Deploy
Distributor:
Attributing Deployment inefficiency

Monitor
ORCA:
Bug Localization
Decaf:
Explaining Latency Anomalies
Summary

- Orca is a commit-level bug localization tool

- Orca uses differential code analysis, build provenance graph, commit risk estimation

- On-call Engineers have found Orca useful: it finds the buggy commit is 77% of cases.
Thank you. Questions?
DevOps Pipeline for Orion

1000s of devs, 100s of commits/day

Testing severely restricted

Dozens of deployments/week

Fine-grained monitoring

Code

Microsoft

B

Find and revert the buggy commit!

Monitor

On-call Engineer

This is not debugging

Can take hours!

Anomalies tell us It is a code issue