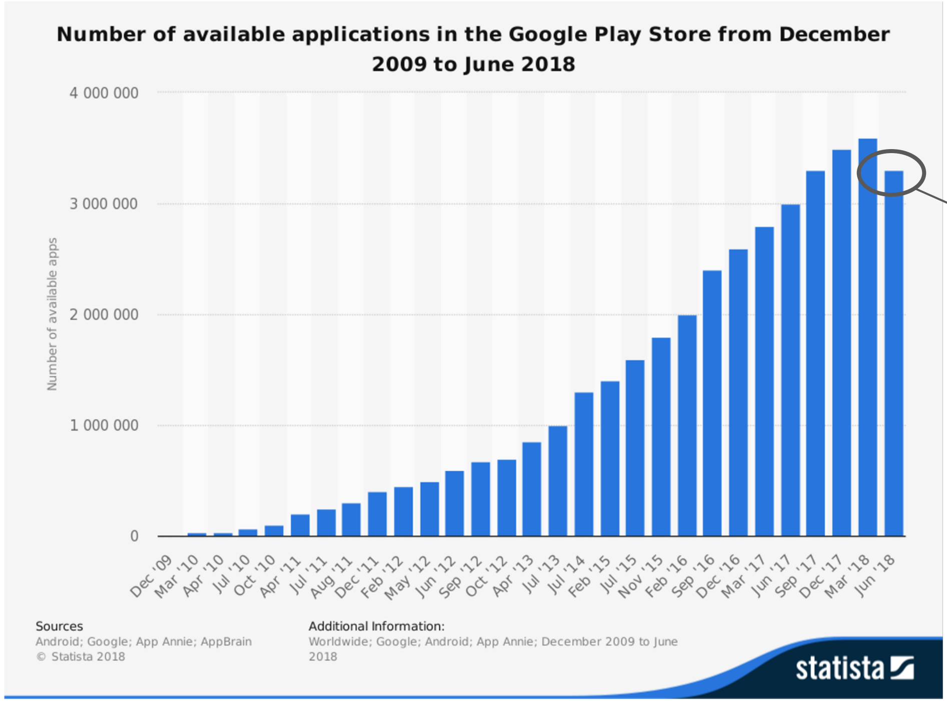
An Empirical Study of Android Test Generation Tools In Industrial Cases

Wenyu Wang [UI] Dengfeng Li [UI] Wei Yang [UT] Yurui Cao [UI]
Zhenwen Zhang [TC] Yuetang Deng [TC] Tao Xie [UI]

[UI] University of Illinois at Urbana-Champaign, USA
[UT] University of Texas at Dallas, USA
[TC] Tencent Inc., China

Automated Android Testing: Still Necessary?



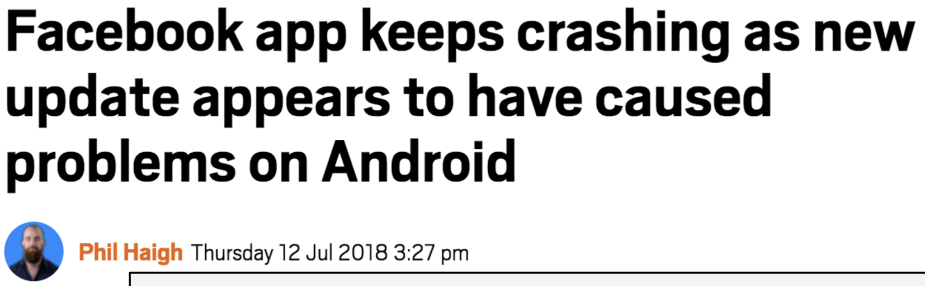
Number of available applications in the Google Play Store from December 2009 to June 2018

Sources
Android; Google; App Annie; AppBrain
© Statista 2018

Additional Information:
Worldwide; Google; Android; App Annie; December 2009 to June 2018

3.3m Android apps

Automated Android Testing: Still Necessary?



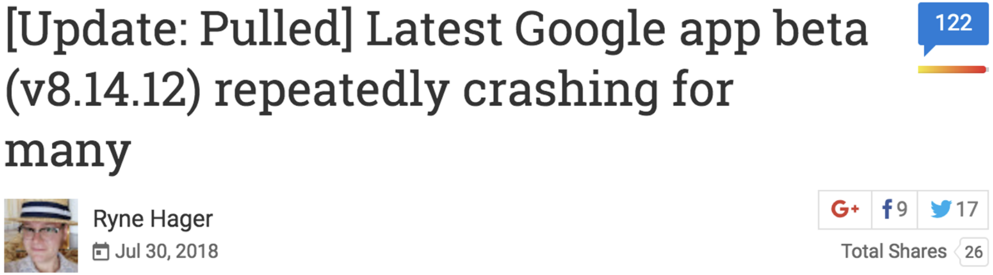
Facebook app keeps crashing as new update appears to have caused problems on Android

Phil Haigh Thursday 12 Jul 2018 3:27 pm

https://metro.co.uk/2018/07/12/facebook-app-keeps-crashing-new-update-appears-caused-problems-android-7708786/

10+ unique crashes on apps like *AccuWeather*, *Gmail*, *Yelp*, ...

https://www.androidpolice.com/2018/07/30/latest-google-app-beta-v8-14-12-repeatedly-crashing-many-android-p/

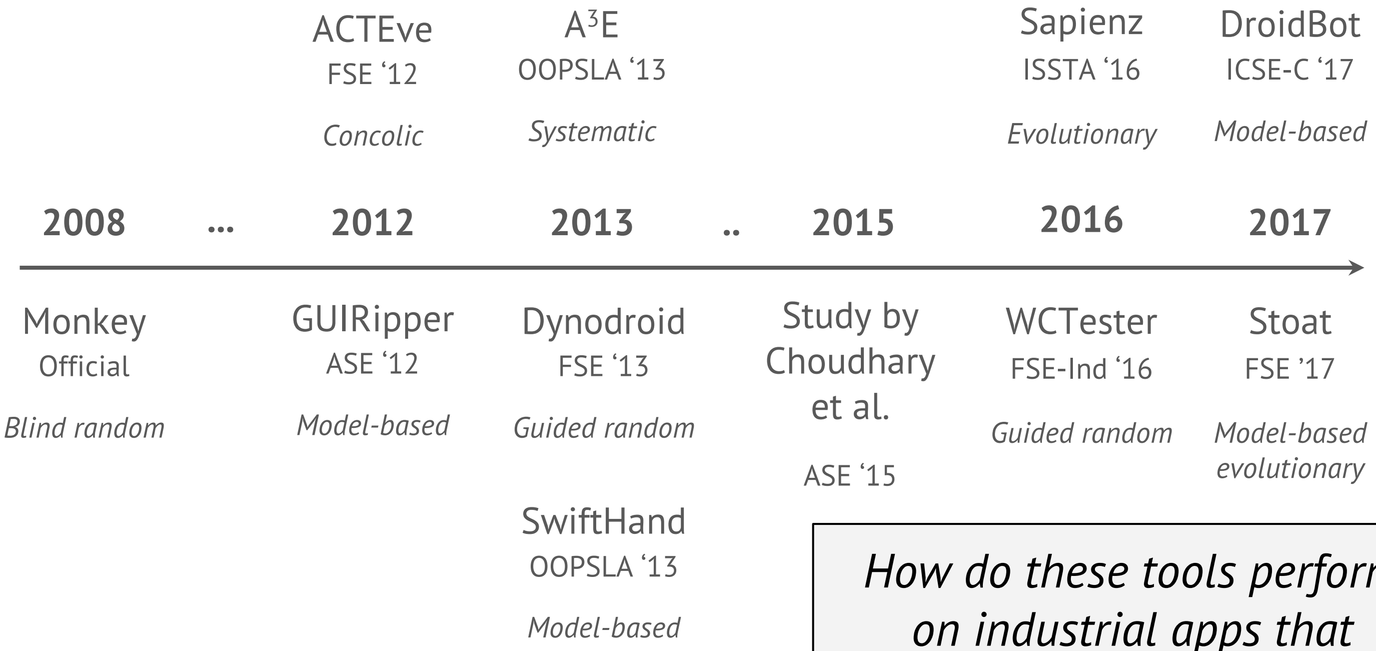


[Update: Pulled] Latest Google app beta (v8.14.12) repeatedly crashing for many

Ryne Hager

Jul 30, 2018

Android Test Generation Tools: A Retrospective



2008 ... 2012 2013 .. 2015 2016 2017

ACTEve
FSE '12
Concolic

A^3E
OOPSLA '13
Systematic

Sapienz
ISSTA '16
Evolutionary

DroidBot
ICSE-C '17
Model-based

Monkey
Official
Blind random

GUIRipper
ASE '12
Model-based

Dynodroid
FSE '13
Guided random

SwiftHand
OOPSLA '13
Model-based

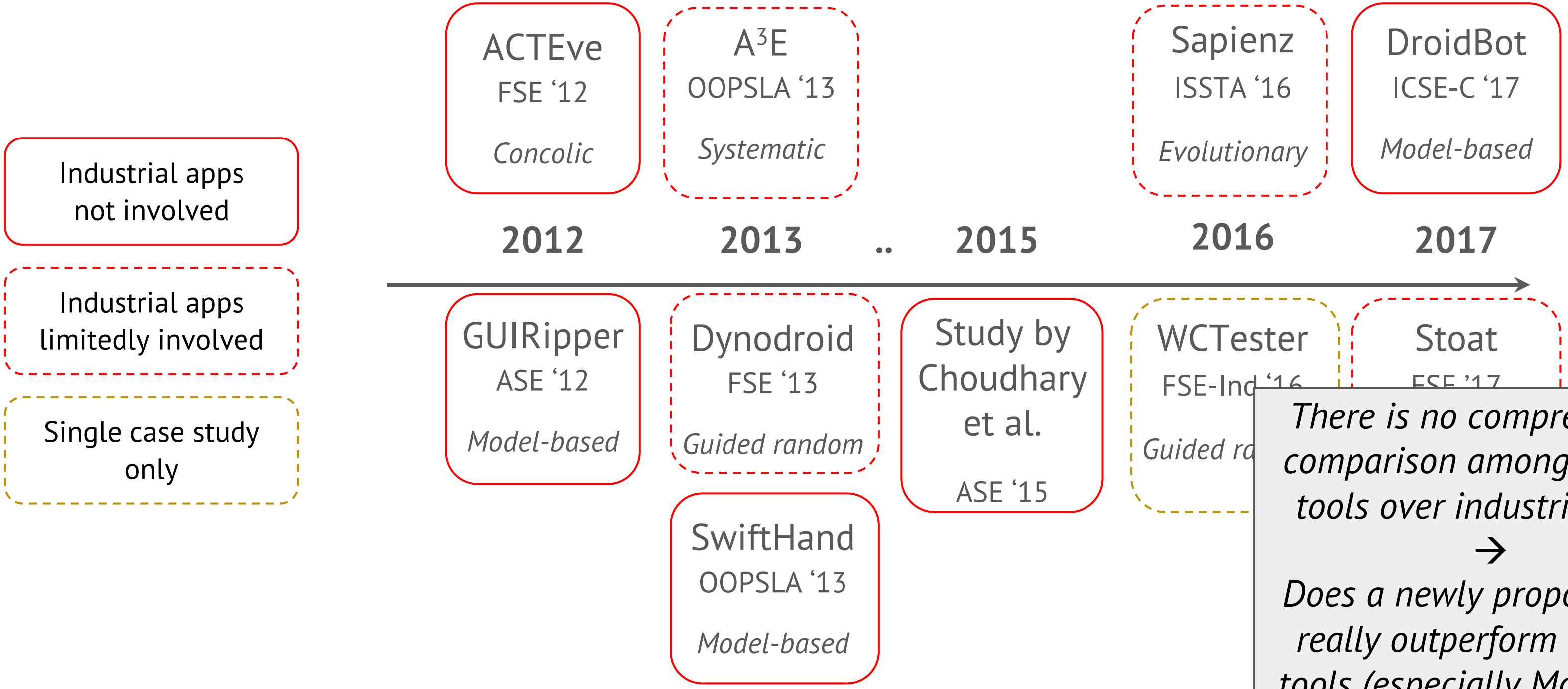
Study by Choudhary et al.
ASE '15

WCTester
FSE-Ind '16
Guided random

Stoat
FSE '17
Model-based evolutionary

How do these tools perform on industrial apps that people actually use everyday?

Android Test Generation Tools: Existing Evaluations



Industrial apps not involved

Industrial apps limitedly involved

Single case study only

2012

ACTEve
FSE ‘12
Concolic

GUIRipper
ASE ‘12
Model-based

2013

A^{3}E
OOPSLA ‘13
Systematic

Dynodroid
FSE ‘13
Guided random

SwiftHand
OOPSLA ‘13
Model-based

..

2015

Study by Choudhary et al.
ASE ‘15

2016

Sapienz
ISSTA ‘16
Evolutionary

WCTester
FSE-Ind ‘16
Guided ra

2017

DroidBot
ICSE-C ‘17
Model-based

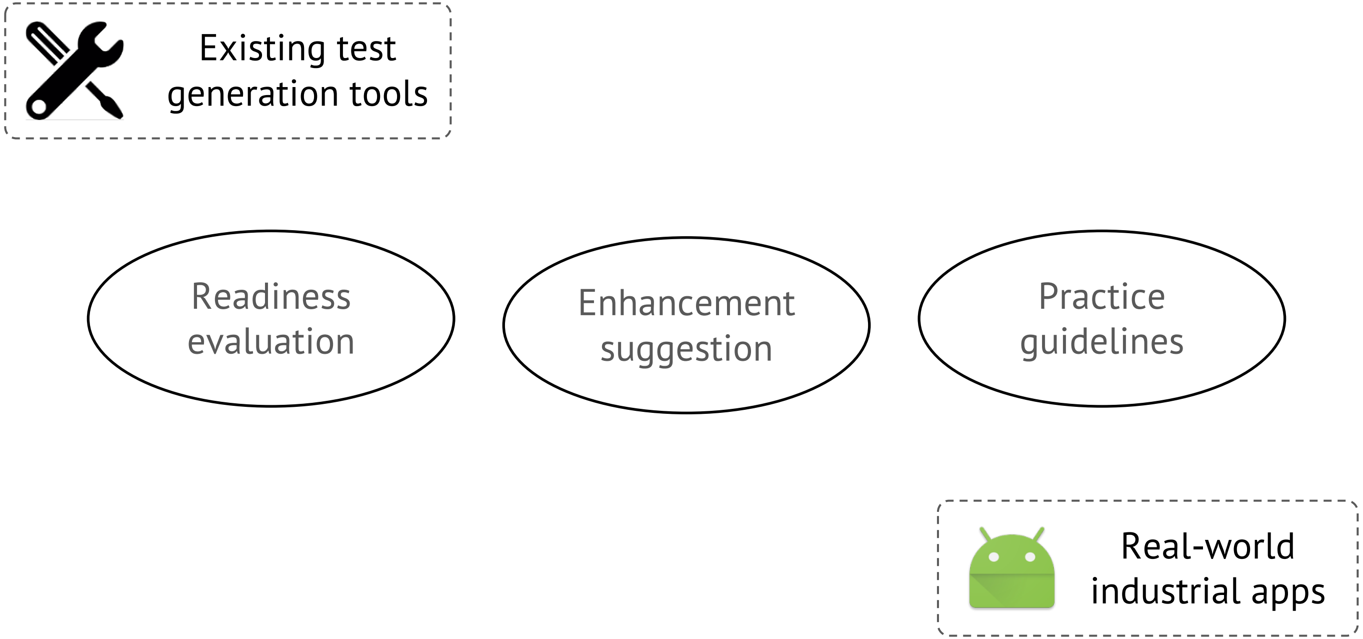
Stoat
FSE ’17

There is no comprehensive comparison among existing tools over industrial apps.

→

Does a newly proposed tool really outperform existing tools (especially Monkey) on industrial apps?

Our Empirical Study: Motivations



Existing test generation tools

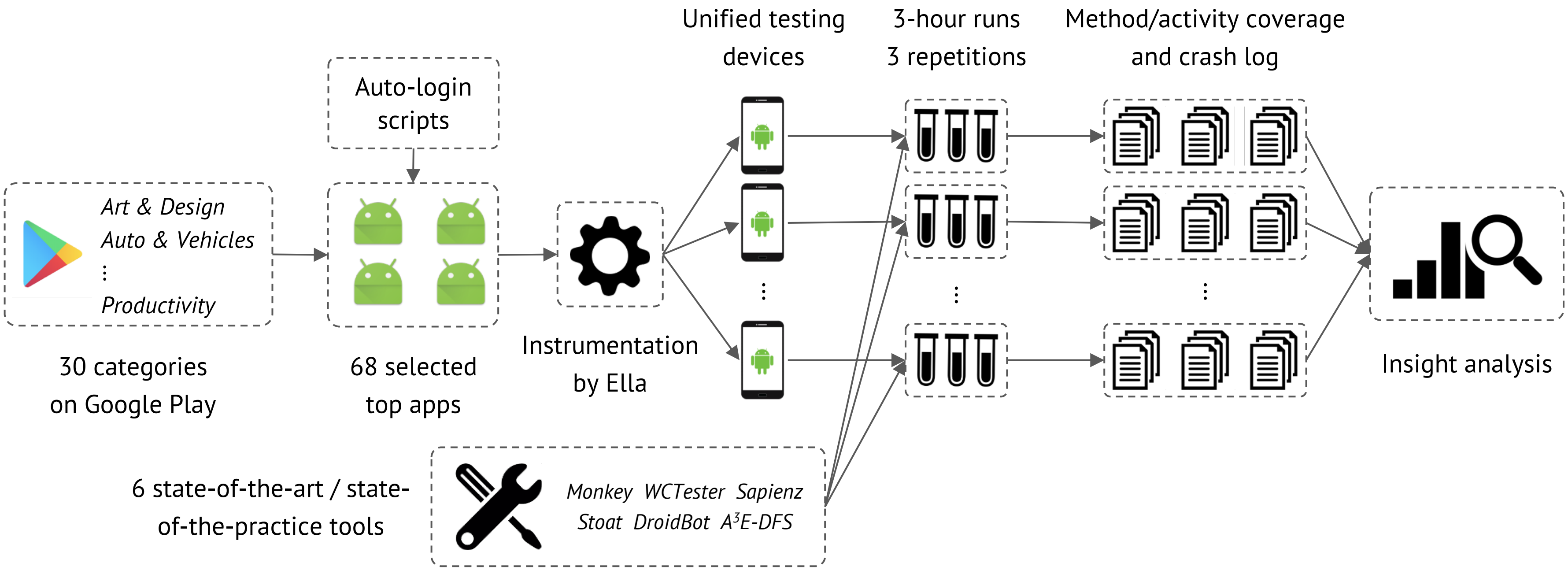
Readiness evaluation

Enhancement suggestion

Practice guidelines

Real-world industrial apps

Our Empirical Study: Methodology



Unified testing devices

3-hour runs
3 repetitions

Method/activity coverage and crash log

Art & Design
Auto & Vehicles
⋮
Productivity

Auto-login scripts

30 categories on Google Play

68 selected top apps

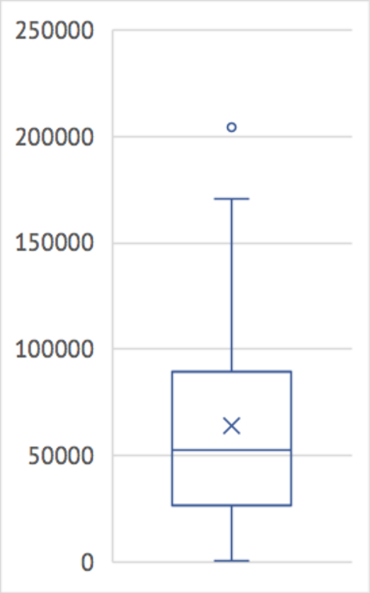
Instrumentation by Ella

Insight analysis

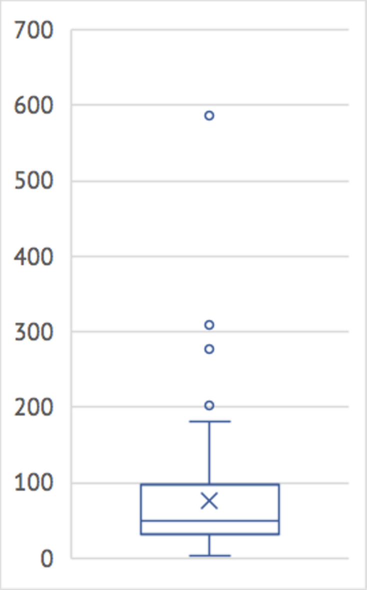
6 state-of-the-art / state-of-the-practice tools

Monkey WCTester Sapienz
Stoat DroidBot A^3E-DFS

Our Empirical Study: Codebase Statistics



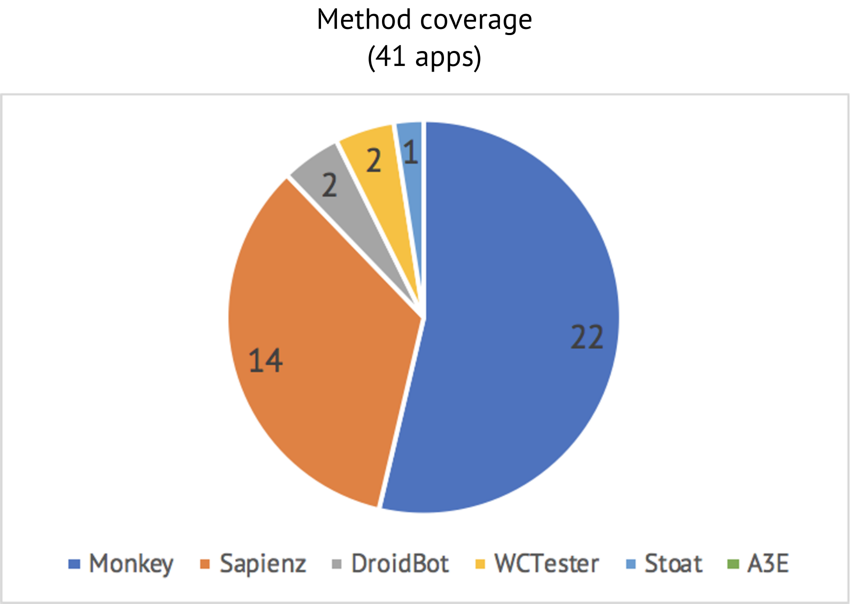
Method stats
(41 apps)

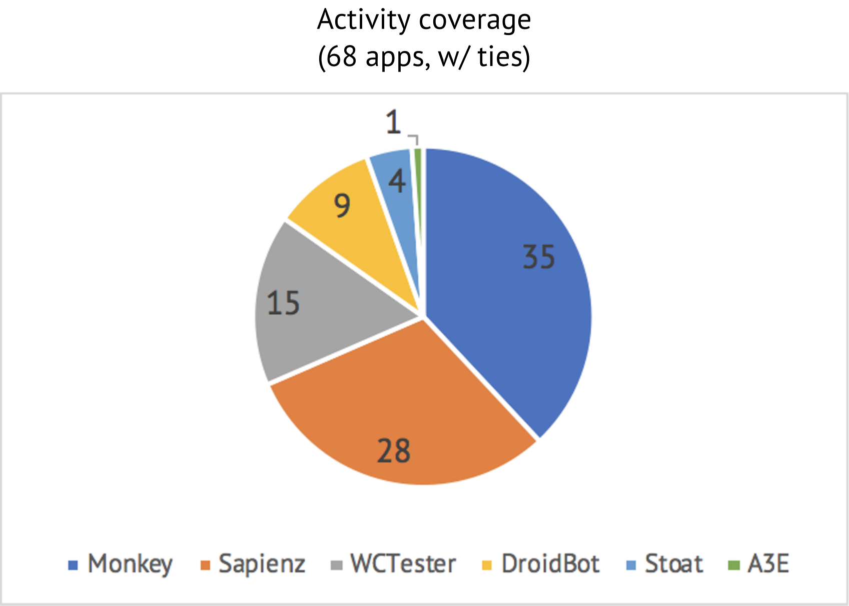


Activity stats
(68 apps)

Industrial apps are generally complex.

Our Empirical Study: Code Coverage Statistics

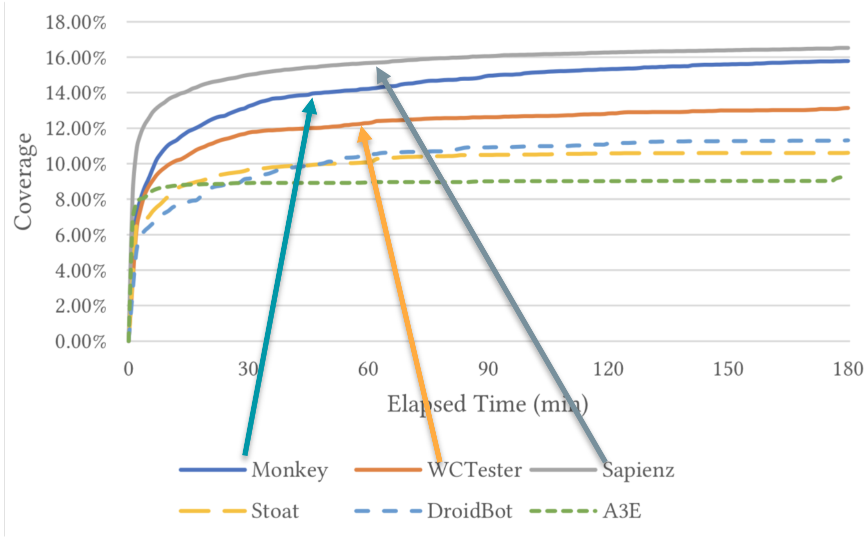




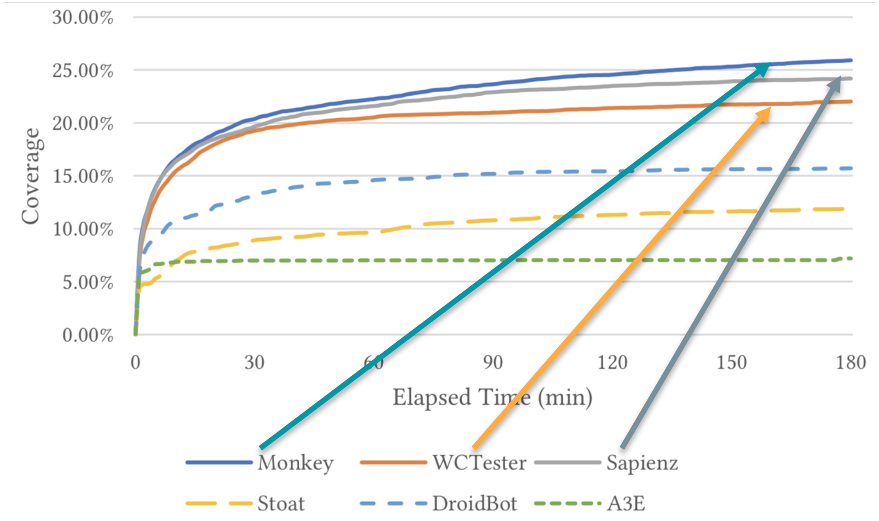
# of apps on which a tool achieves the highest code coverage

Monkey achieves the highest code coverage on most industrial apps.

Our Empirical Study: Code Coverage Trends



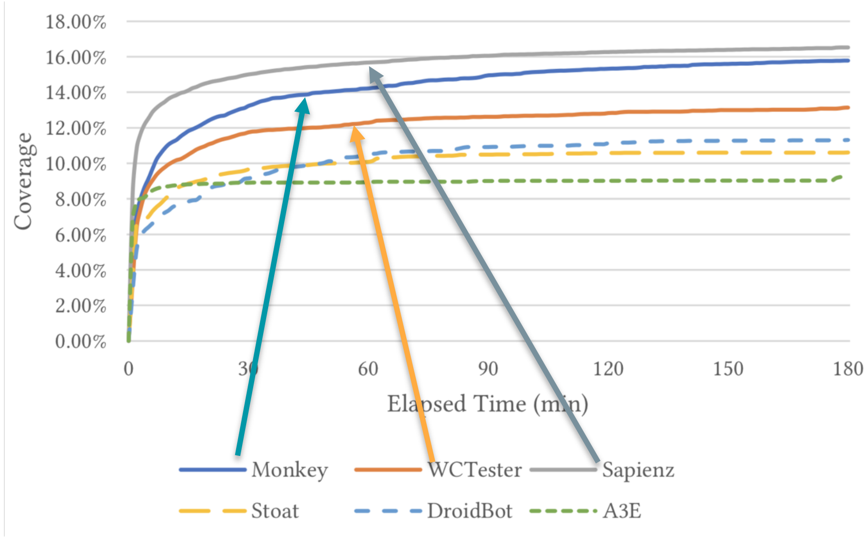
Average Method Coverage Percentages



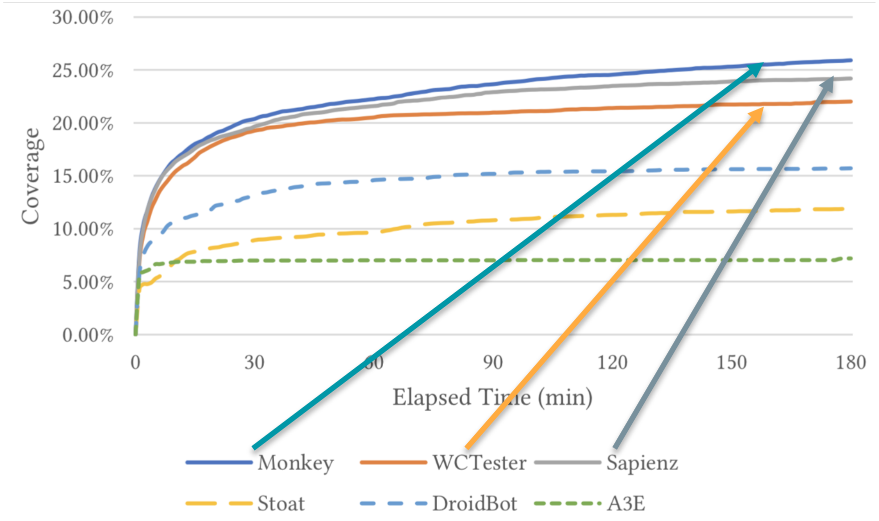
Average Activity Coverage Percentages

Monkey, Sapienz, and WCTester constantly have higher average code coverage percentages than other tools.

Our Empirical Study: Code Coverage Trends



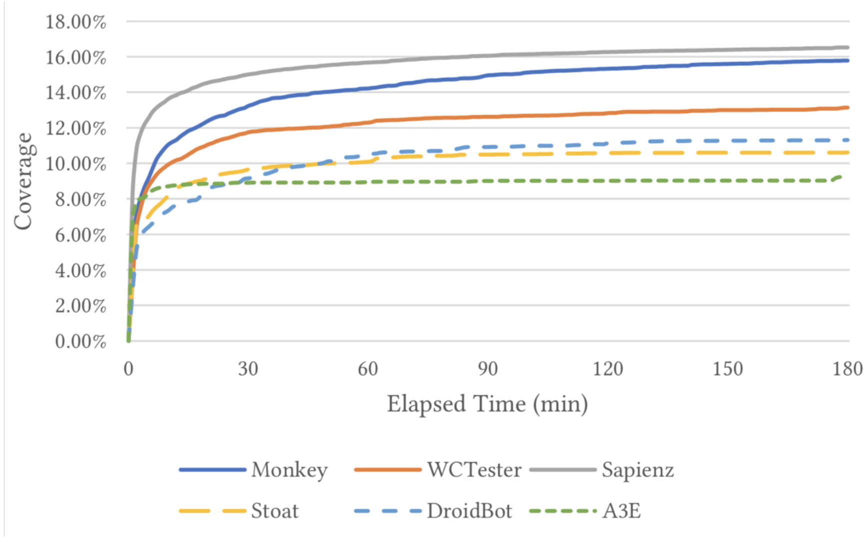
Average Method Coverage Percentages



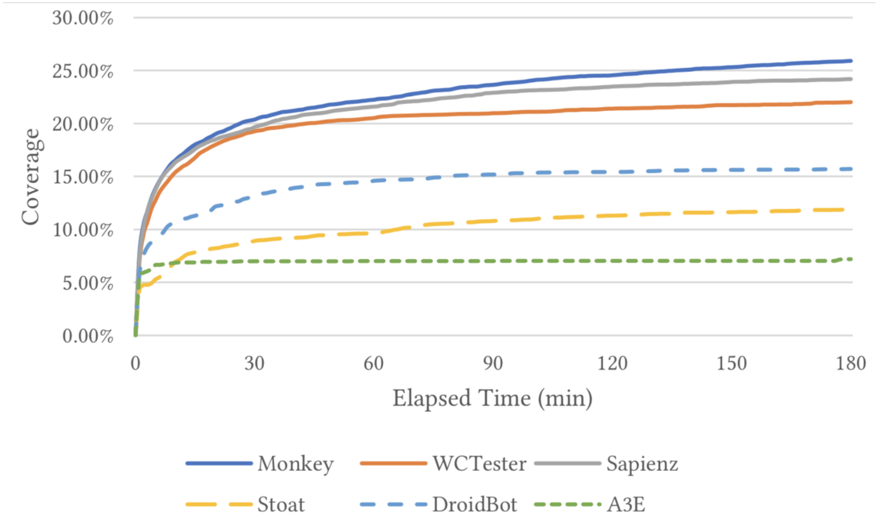
Average Activity Coverage Percentages

Sapienz has higher average method coverage percentages than Monkey, with advantages reduced over time.

Our Empirical Study: Code Coverage Trends



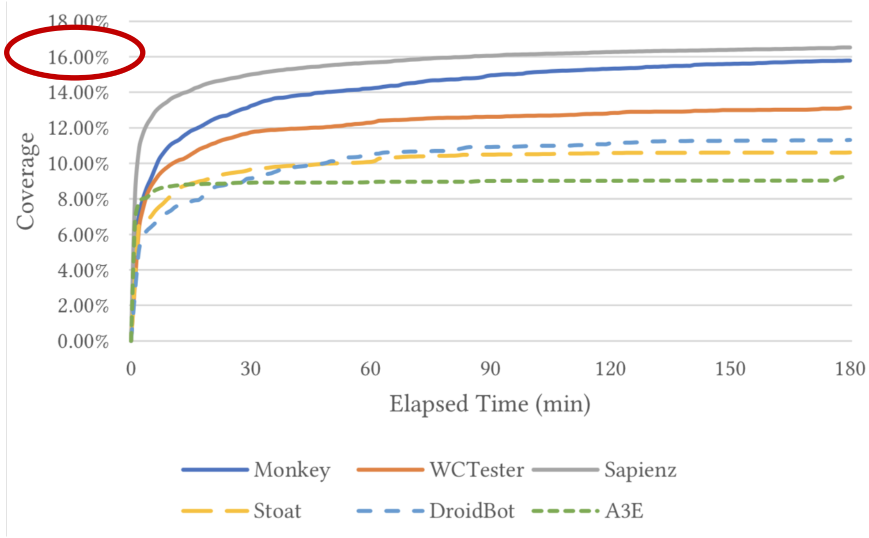
Average Method Coverage Percentages



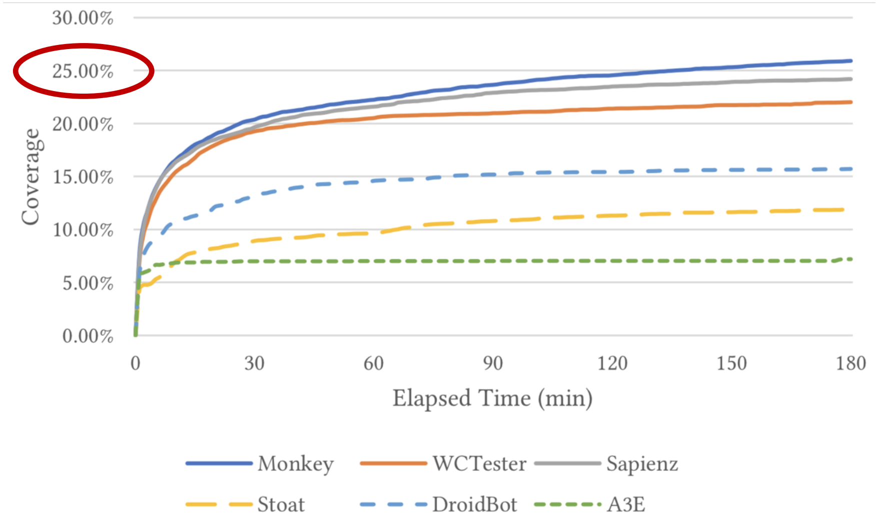
Average Activity Coverage Percentages

Activity coverage is generally higher than method coverage.

Our Empirical Study: Code Coverage Trends



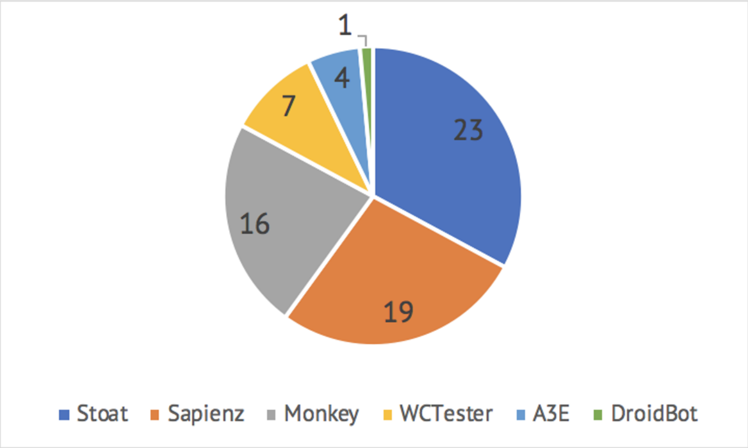
Average Method Coverage Percentages



Average Activity Coverage Percentages

There is still much space for improvements on testing industrial apps.

Our Empirical Study: Unique-Crash Statistics



# of apps on which a tool achieves the highest number of unique crashes
(totaling 68 apps, w/ ties)

Stoat, Sapienz, and Monkey trigger the highest numbers of unique crashes on most industrial apps.

Our Empirical Study: Case Study On The App *Photo*

Stoat

21 unique crashes

Mainly triggering `NullPointerException` during activity starting

Monkey / Sapienz

Both 20 unique crashes

Mainly triggering `ArrayIndexOutOfBoundsException` and `StackOverflowError`

System-level event injection could be helpful for revealing hidden issues.

Our Empirical Study: Case Study On The App *Wattpad*

Sapienz

77 unique crashes

Mainly triggering `SQLiteException` by accessing non-existent tables

Other tools

No more than 2 unique crashes

Crafting special conditions can be helpful for reaching corner cases.

Our Empirical Study: Choosing Tools For Tasks

Method Coverage

Monkey + Sapienz

>90% joint contribution

Activity Coverage

Monkey + Sapienz/Stoat

Good complements

Crash Triggering

Stoat + Monkey/Sapienz

Good complements

Apps Sharing Similar Functions with WeChat

WCTester

Our Empirical Study: Human Efforts

Non-trivial human efforts required for all tools except Monkey.

Our Empirical Study: Threats of Validity

Scope of study subjects

Indeterminism of experiments

Reliability of the infrastructure

Summary

For industry users,

Monkey is still a desirable choice,

e.g., due to its good usability and competitive testing effectiveness.

For research community,

Industrial apps deserve more consideration,

e.g., a newly proposed tool should also be compared with existing tools over industrial apps.

Questions?

This work was supported in part by National Science Foundation under grants no. CNS-1513939 and CNS-1564274.

Summary

For industry users,

Monkey is still a desirable choice,

e.g., due to its good usability and competitive testing effectiveness.

For research community,

Industrial apps deserve more consideration,

e.g., a newly proposed tool should also be compared with existing tools over industrial apps.