资料：

**A Context-Aware Approach for Dynamic GUI Testing of Android Applications:（动态生成输入）2015**

Random 没有考虑应用上下文 无法深入

Since GUI context of the application is not taken into account, random testing fails to dive deep into the program.

Model-based 需要静态预分析或为了模型学习修改应用

 However, they require a static pre-analysis phase to generate the GUI model, or need instrumentation to modify the application for model learning.

*Cadage* (stands for Context-Aware Dynamic Android Gui Explorer

It generates GUI inputs purely dynamically while taking the GUI context into consideration as well. During the whole process of the test, there are two things that highly influence the efficiency of testing: The first one is how we model the GUI state of applications under test and their transition relations dynamically to guide testing event selection. The second one is how we design corresponding strategies to select *excellent* GUI events using the targeted knowledge provided by the dynamic model.

广度优先搜索 尽快探索没处理过的GUI事件；给当前可触发事件优先级；用概率算法选择GUI输入作为测试事件

**A Combined Technique of GUI Ripping and Input Perturbation Testing for Android App （建立模型 生成输入）2015**

In this work two tools, called SlumDroid and GUIAnalyzer, are presented to support Android Application Testing (AAT). SlumDroid is a modified version of *AndroidRipper*  that implements the GUI Ripping Technique and creates a model of the Application Under Testing (AuT). This model is used by GUIAnalyzer to produce the perturbed text inputs that will be used in the new test session.

SlumDroid实现了GUI Ripping技术，建立模型，模型被GUIAnalyzer使用，来产生新的测试中要用的扰乱测试输入

Fuzz testing generates redundant events, thus it is not suited for generating inputs and events that require the human intelligence (e.g., playing and winning a game)  and cannot ensure effective failure detection.

随机测试问题：冗余事件因此不适于生成需要人类智力的输入和事件，并且不能确保有效的错误检测

The GUI Ripping Technique is implemented by a tool, called GUI Ripper, that constructs a navigation model of the application by dynamically interacting with its user interface and observing changes of the GUI state

GUI Ripper建立了一个应用的导航模型，通过1和它的用户界面动态交互2观察GUI的状态改变

SlumDroid与Android应用程序的用户界面元素进行交互并模拟物理交互

开发了称为GUIAnalyzer的单独工具来执行输入扰动。它接收GUITree（AuT的GUI模型）作为输入，并接收在翻录过程中捕获的屏幕截图。

**PATS: A Parallel GUI Testing Framework for Android Applications**

**（提高测试效率）2015**

基于主从模式的并行安卓GUI测试系统 动态分析被测应用，运行时并行生成测试用例 提高测试效率

在GUITAR中，所有测试用例都在中央控制器中生成，可能成为性能瓶颈。

PATS是完全分布式GUI测试框架设计，该设计在测试节点的集群上以细粒度模型并行化测试过程。也就是说，执行单个测试事件序列可能涉及一组分布式测试节点。每个测试节点都使用黑盒方法来生成测试序列的一部分。PATS中事件流图的生成是在控制器和所有从属节点的协作下完成的。从属节点负责分析分配的UI界面，并动态确定短期测试事件序列。协调器聚集这些短期事件序列，并将其分配给从属节点，以进行进一步测试并生成新的短期事件序列。

**Generating Test Cases for Android Applications through GUI Modeling, Usage Modeling, and Change Analysis（测试用例生成）2015**

1 GUI建模2通过app使用分析对使用建模3安卓app新版本的变更分析

测试用例是一个可执行操作的序列，带有被称为断言的预期输出

测试自动化程度低；现有的测试用例自动化方法缺乏对Android应用程序频繁变化的特性的考虑

GUI建模：现有工具；使用建模：Usage Agent后台监控，传给usage Modeler生成模型；变更分析：比较抽象语法树，应用程序差异数据是受更改影响的AST节点

上面三个组件的输出用于生成测试用例。具体来说，它们将组合成一个单独的模型，该模型将表示应用程序的行为(GUI模型和使用模型)和结构(更改分析)方面。每个事件序列对应于测试用例，因此表示将在Android设备上运行的可执行操作。

测试用例选择器选择要运行的测试用例，测试执行器在设备上运行，处理完后输出结果。

**GUI and Usage Model-Based Test Case Generation for Android Applications with Change Analysis （生成测试用例 提高代码覆盖率）2016**

基于模型的测试本来就很困难，因为随着GUI事件的执行，应用程序的当前状态可能会不断变化。另一方面，现有的测试技术只能用于一个应用程序版本，这使得自动化测试很难适应应用程序的变化。

使用1 基于现有自动化方法的GUI模型 2 终端用户的app使用信息 3 标记为新旧的两个源代码版本

1 GUI建模（用有限状态机给app行为建模） 2 用途建模3事件序列分析4测试用例生成（基于事件序列生成，这些事件序列的状态转换路径(来自行为模型)包含关于可在应用程序上执行的可执行操作的信息。这些操作被转换为JUnit类的形式，这些类使用Robotium5测试库执行。）5变更分析（受两个源代码版本之间变更影响的测试用例：检测源代码之间差异，确定受影响的方法，识别不同事件序列调用的受影响的方法，识别使用检测到的事件序列的测试用例

**Automatically Generating Test Scripts for GUI Testing（使用应用程序源代码和可执行文件的静态分析和动态分析来自动生成测试脚本 节省时间）**

测试自动化以减少工时

Selenium WebDriver自动化测试工具 但是开发团队需要很多工时来创建测试脚本。一般来说，通过测试自动化来减少工时的效果只有在运行回归测试超过三次之后才会出现。但是，在回归测试执行三次或更多次之前，由于屏幕上的目标(例如按钮或输入表单)发生了更改，测试脚本变得不可用。大量需要修改测试脚本

**Automated model-based Android GUI testing using multi-level GUI comparison criteria（GUI模型生成） 2016**

基于模型的安卓GUI测试

大多数现代的Android应用程序都包含大量动态构建的GUI，这使得精确行为建模更加具有挑战性

一组多层次的GUI比较标准(GUICC)，它为GUI模型生成提供了多个抽象级别的选择

用多层次GUICC进行基于模型的测试比基于活动的GUI模型生成效率更高

通过灵活地操作GUICC，多层GUICC可以缓解现有单层GUICC在现实Android应用行为建模中固有的状态爆炸问题。

问题：准确建模，GUI中动态行为导致不一致的模型生成或状态爆炸问题

解决：明确定义GUI比较标准来指定行为抽象的级别

GUICC是用来区分不同GUI状态的与GUI相关的信息

基于活动可能会丢失状态信息，只关注页面变化会存储冗余信息

GUICC的选择会显著影响基于模型的GUI测试中行为建模的结果，但是更高级别的GUICC并不总是意味着会有更多优化的解决方案。

GUICC的选择会显着影响基于模型的GUI测试中行为建模的结果，但是更高级别的GUICC并不总是意味着会有更多优化的解决方案。

与基于活动的GUI模型相比，基于多层GUICC的模型测试可以在代码覆盖率和错误检测能力方面生成更有效的GUI模型。

**DetReduce: Minimizing Android GUI Test Suites for Regression Testing（减少冗余）2018**

自动化测试工具不擅长生成能用于增量开发和回归测试的简洁的回归测试套件

通过一个自动化的Android GUI测试工具，从生成的大型测试套件中为Android应用程序创建一个小型回归测试套件

由自动化工具生成的测试用例可以包含冗余用户操作的子序列，可以删除这些子序列以获得更小的测试套件。

（最近提出了几种自动化GUI测试技术。这些技术包括基于学习的测试、基于模型的测试、遗传编程、模糊测试和基于静态分析的方法。这些技术的主要目标是实现良好的代码和屏幕覆盖(即覆盖应用程序的所有不同屏幕)，并找到常见的bug，如崩溃和无响应。这些技术中的大多数是将自动生成的用户输入或操作序列注入应用程序，持续数十小时。）

（测试套件的运行时间与测试套件的大小成线性关系。但是，回归测试应该是快速的，这样它们才能在开发过程中经常使用。因此，在回归测试中很难使用这样的测试套件）

目标：合理时间内生成测试套件，测试套件运行时间比较短，但能获得和原来的测试套件接近的代码和屏幕覆盖率

已有的最小化GUI测试套件的技术只能处理输入测试套件很小的情况或者没有减少每个测试用例的大小

**From Manual Android Tests to Automated and Platform Independent Test Scripts（创建独立于平台的测试脚本）2016**

允许测试人员轻松地为应用程序创建独立于平台的测试脚本，并在多个设备和操作系统版本上自动运行生成的测试脚本。该技术不需要修改被测试的应用程序或运行时系统，通过(1)拦截测试人员与应用程序的交互，(2)为测试人员提供一种直观的方式来指定预期结果，然后将其编码为测试准则

**Practical GUI Testing of Android Applications via Model Abstraction and Refinement**

在测试过程中利用运行时信息来动态地优化模型

提高了模型的精度，从而显著提高了测试的有效性

**A Comprehensive Investigation of Modern Test Suite Optimization Trends, Tools and Techniques**

***References：***

Haowen Zhu, Xiaojun Ye, Xiaojun Zhang and Ke Shen, “A Context-Aware Approach for Dynamic GUI Testing of Android Applications”, 2015 IEEE 39th Annual Computer Software and Applications Conference

Gennaro Imparato, “A Combined Technique of GUI Ripping and Input Perturbation Testing for Android App”, 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering

Hsiang-Lin Wen, Chia-Hui Lin, Tzong-Han Hsieh, and Cheng-Zen Yang, “PATS: A Parallel GUI Testing Framework for Android Applications”, 2015 IEEE 39th Annual Computer Software and Applications Conference

J. L. San Miguel and S. Takada, “Generating Test Cases for Android Applications through GUI Modeling, Usage Modeling, and Change Analysis”, C3S2E 2015 July 13-15, 2015, Yokohama, Japan

J. L. San Miguel and S. Takada, “GUI and Usage Model-Based Test Case Generation for Android Applications with Change Analysis”, Mobile! 2016 Proceedings of the 1st International Workshop on Mobile Development Pages 43-44

Young-Min Baek and Doo-Hwan Bae, “Automated model-based Android GUI testing using multi-level GUI comparison criteria”, 2016 31st IEEE/ACM International Conference on Automated Software Engineering (ASE)

Wontae Choi, Koushik Sen, George Necula, and Wenyu Wang. 2018. “DetReduce: Minimizing Android GUI Test Suites for Regression Testing.” In Proceedings of ICSE ’18: 40th International Conference on Software Engineering, Gothenburg, Sweden, May 27-June 3, 2018 (ICSE ’18), 11 pages.

Riccardo Coppola, Maurizio Morisio and Marco Torchiano. 2017. “Scripted GUI Testing of Android Apps: A Study on Diffusion, Evolution and Fragility.” In Proceedings of PROMISE , Toronto, Canada, November 8, 2017, 10 pages.

DOI: 10.1145/3127005.3127008