**TRƯỜNG ĐẠI HỌC KIẾN TRÚC**

**KHOA CÔNG NGHỆ THÔNG TIN**

**ĐỒ ÁN TỐT NGHIỆP KỸ SƯ**

**NGÀNH: CÔNG NGHỆ THÔNG TIN**

**MÃ NGÀNH: 7480201**

**KHÓA: 2019 - 2024**

***ĐỀ TÀI***:

**TÌM HIỂU MỘT SỐ KỸ THUẬT SINH DỮ LIỆU KIỂM THỬ TỰ ĐỘNG CHO ANDROID**

|  |  |
| --- | --- |
| **SINH VIÊN THỰC HIỆN:** | **PHẠM THỊ THẢO** |
| **LỚP:** | **19CN2** |

**HÀ NỘI 12-2023**

**TRƯỜNG ĐẠI HỌC KIẾN TRÚC**

**KHOA CÔNG NGHỆ THÔNG TIN**

**ĐỒ ÁN TỐT NGHIỆP KỸ SƯ**

**NGÀNH: CÔNG NGHỆ THÔNG TIN**

**MÃ NGÀNH: 7480201**

**KHÓA: 2019 - 2024**

***ĐỀ TÀI***:

**TÌM HIỂU MỘT SỐ KỸ THUẬT SINH DỮ LIỆU KIỂM THỬ TỰ ĐỘNG CHO ANDROID**

|  |  |
| --- | --- |
| **GIÁO VIÊN HƯỚNG DẪN:** | **ThS. NGUYỄN XUÂN THU** |
| **SINH VIÊN THỰC HIỆN:** | **PHẠM THỊ THẢO** |
| **LỚP:** | **19CN2** |

**HÀ NỘI 12-2023**

# LỜI NÓI ĐẦU

**1. Lý do chọn đề tài**

Hiện nay trên thế giới có 5,16 tỷ người dùng Internet, nghĩa là 64,4% tổng dân số thế giới hiện đang trực tuyến. Và cũng theo Kepios, tính đến tháng 01/2023, toàn thế giới có 5.44 tỉ người dùng thiết bị di động (chiếm 68% dân số thế giới). Người chỉ dùng một thiết bị di động duy nhất đã dùng hơn 3% trong năm qua, với 168 triệu người dùng mới trong 12 tháng qua .

Android là một nền tảng nguồn mở thuộc sở hữu của Google, được nhiều thương hiệu điện thoại thông minh điều chỉnh và sử dụng. Trên cả năm 2023, thị phần hệ điều hành di động toàn cầu của Android giảm nhẹ xuống 70,27%, giảm 1,20% so với năm 2022. Trong khi đó, iOS của Apple tăng lên 29,01%, tăng 1,16% so với năm trước. Android bắt đầu chiếm vị trí dẫn đầu trên thị trường toàn cầu vào năm 2012, đánh bại iOS 3,37%. Từ năm 2009 đến nay, Android tăng trưởng hơn 67%, trong khi thị phần của iOS giảm khoảng 5%.

Kể cả những lập trình viên xuất sắc nhất cũng không thể lúc nào cũng xây dựng được một phần mềm không có lỗi. Lỗi tiềm ẩn trong mọi phần mềm và có thể gây ra những thiệt hại nghiêm trọng. Do đó, trong vòng đời phát triển phần mềm nói chung và ứng dụng di động nói riêng, kiểm thử là một hoạt động quan trọng, xuyên suốt, không thể bỏ qua, quyết định đến khả năng tiếp cận người dùng của sản phẩm.

Hoạt động kiểm thử có thể tiến hành một cách thủ công, tuy nhiên điều này sẽ mất thời gian, tốn chi phí và đôi khi không mang lại hiệu quả cao hoặc trong một vài trường hợp, kiểm thử thủ công là không thể thực hiện được. Vì vậy, cần thiết phải có một hình thức kiểm thử tự động để hỗ trợ và khắc phục các vấn đề này.

Kiểm thử tự động cũng có nhiều kỹ thuật với các mức độ tự động khác nhau. Đối với nhiều các công cụ kiểm thử vẫn cần có sự tham gia của kiểm thử viên vào trong quá trình. Kiểm thử viên sẽ phải xây dựng các kịch bản cho các công cụ kiểm thử thực thi. Đây là một công việc không hề đơn giản, tốn thời gian, chi phí và nhân lực.

Xuất phát từ tình hình trên, tôi chọn đề tài: **“*Tìm hiểu một số kỹ thuật sinh dữ liệu kiểm thử tự động cho Android*”** nhằm nghiên cứu, tìm hiểu các kỹ thuật sinh dữ liệu kiểm thử tự động cho Android; khảo sát các công cụ kiểm thử mã nguồn mở đã được triển khai sử dụng trên thế giới và trong nước, từ đó lựa chọn ra các công cụ phù hợp xây dựng thành kịch bản thử nghiệm để phân tích, đánh giá hiệu quả của công cụ. Các kết quả nghiên cứu này có thể được sử dụng để nâng cao hiệu quả, tiết kiệm chi phí, thời gian kiểm thử ứng dụng Android của tổ chức, cá nhân thực hiện kiểm thử.

**2. Tình hình nghiên cứu trong nước và trên thế giới**

***2.1. Tình hình nghiên cứu trên thế giới***

Đề cập đến kiểm thử tự động cho Android, trên thế giới đã có nhiều tổ chức, công ty đã xây dựng quy trình và công cụ đi kèm, trong đó phổ biến và được công nhận rộng rãi nhất có thể kể đến: UI Automator, Robotium, MonkeyRunner, Ranorex, Kobiton, Calabash.

***2.2. Tình hình nghiên cứu trong nước***

Báo cáo của DataAI & AppMagic cho biết, đến tháng 7 năm 2023, với 4,2 tỷ lượt tải của các ứng dụng phát hành bởi các nhà phát triển trong nước thì Việt Nam đã vươn lên đứng thứ 4 toàn cầu về lượt tải ứng dụng trên CH Play của Google. Việt Nam hiện có 7 ứng dụng với số lượng người dùng trên 10 triệu người và 11 ứng dụng có từ 5 - 10 triệu người dùng. Việt Nam cũng có 4 nhà phát triển ứng dụng lọt vào bảng xếp hạng top 50 công ty toàn cầu, có nhiều ứng dụng mới vượt mốc 100.000 lượt tải xuống năm 2023.

Tuy nhiên, thực tế cho thấy, ở Việt Nam, hầu hết các doanh nghiệp, cá nhân phát triển ứng dụng thực kiểm thử ứng dụng di động theo các phương pháp, công cụ của nước ngoài hoặc dựa trên "kinh nghiệm".

Vì vậy, nghiên cứu về kiểm thử tự động cho Android đang là một lĩnh vực hứa hẹn với nhiều tiềm năng phát triển trong tương lai. Các nghiên cứu này nhằm tạo ra các phương pháp và công cụ mới để giúp tăng cường chất lượng kiểm thử và tăng cường hiệu suất cho quy trình kiểm thử trong các dự án phần mềm.

**3. Mục đích nghiên cứu của đề tài**

Nghiên cứu, tìm hiểu các kỹ thuật sinh dữ liệu kiểm thử tự động cho Android; khảo sát các công cụ kiểm thử mã nguồn mở đã được triển khai sử dụng trên thế giới và trong nước, từ đó lựa chọn ra các công cụ phù hợp xây dựng thành kịch bản thử nghiệm để phân tích, đánh giá hiệu quả của công cụ.

**4. Nhiệm vụ nghiên cứu**

- Nghiên cứu, tìm hiểu tổng quan về hệ điều hành Android.

- Nghiên cứu, tìm hiểu về các phương pháp kiểm thử tự động.

- Nghiên cứu, khảo sát, tìm hiểu một số công cụ sinh đầu vào kiểm thử tự động mã nguồn mở cho ứng dụng Android.

- Xây dựng kịch bản thử nghiệm để phân tích, đánh giá hiệu quả của công cụ.

**5. Đối tượng nghiên cứu**

- Hệ điều hành Android và các ứng dụng cho hệ điều hành Android

- Công cụ mã nguồn mở kiểm thử tự động cho Android.

**6. Phạm vi nghiên cứu**

Các phương pháp kiểm thử tự động cho Android

**7. Phương pháp nghiên cứu**

- Phương pháp nghiên cứu tài liệu: Thu thập, nghiên cứu, phân tích tổng hợp các tài liệu, báo cáo khoa học về kỹ thuật sinh dữ liệu và công cụ kiểm thử tự động.

- Phương pháp tham khảo ý kiến chuyên gia: Tham khảo ý kiến của giáo viên hướng dẫn, các thầy cô giáo có nhiều kinh nghiệm trong lĩnh kiểm thử phần mềm.

- Phương pháp thực nghiệm: Lựa chọn các công cụ mã nguồn mở kiểm thử tự động cho Android, cài đặt, xây dựng kịch bản thử nghiệm để phân tích, đánh giá hiệu quả của công cụ.

**8. Dự kiến kết quả đạt được**

- Nghiên cứu về các kỹ thuật sinh dữ liệu kiểm thử tự động cho Android

- Quy trình kiểm thử tự động ứng dụng Android dựa trên các công cụ mã nguồn mở có sẵn

- Kịch bản thử nghiệm kiểm thử tự động một số ứng dụng Android dựa trên các công cụ mã nguồn mở và các kết quả phân tích, đánh giá hiệu quả của công cụ.

**9. Bố cục của đồ án tốt nghiệp**

Chương 1. Tổng quan về hệ điều hành Android

Chương 2. Các phương pháp kiểm thử tự động

Chương 3. Một số công cụ sinh đầu vào kiểm thử tự động cho Android

Chương 4. Thực nghiệm kiểm thử tự động cho ứng dụng Android.

# CHƯƠNG 1. TỔNG QUAN VỀ HỆ ĐIỀU HÀNH ANDROID

*Chương này giới thiệu nền tảng, kiến trúc, hoạt động của hệ điều hành Android; trình bày các vấn đề cơ bản về hệ thống tập tin trên hệ điều hành Android.*

## Giới thiệu chung về hệ điều hành Android

Hệ điều hành là một phần mềm hoặc một tập hợp các chương trình điều khiển mà các thiết bị điện toán sử dụng để quản lý và điều phối các tài nguyên phần cứng và phần mềm trên hệ thống. Nó cung cấp một giao diện giữa người dùng và phần cứng, cho phép người dùng tương tác với các thiết bị điện toán và thực hiện các nhiệm vụ thông qua các ứng dụng.

Android là một hệ điều hành dựa trên nền tảng Linux 2.6 được thiết kế dành cho các thiết bị di động có màn hình cảm ứng như điện thoại thông minh và máy tính bảng. Chiếc điện thoại đầu tiên chạy Android được bán vào năm 2008. Android có mã nguồn mở và Google phát hành mã nguồn theo Giấy phép Apache. Chính mã nguồn mở cùng với giấy phép không có nhiều ràng buộc đã cho phép các nhà phát triển thiết bị, mạng di động và các lập trình viên được tùy biến và phân phối Android một cách tự do.

## Các phiên bản của hệ điều hành Android

Lịch sử phiên bản của hệ điều hành Android bắt đầu với việc phát hành công khai phiên bản beta đầu tiên vào ngày 5 tháng 11 năm 2007.

Hệ điều hành Android đã trải qua nhiều bản cập nhật quan trọng từ khi ra mắt. Bản Android 1.0 ra mắt ngày 23/9/2008 khai sinh ra hệ điều hành di động này, sau đó, các phiên bản như CupCake, Donut, Ice Cream Sandwich, KitKat, Lollipop, Marshmallow, Nougat,… đem đến cải tiến về giao diện, hiệu suất và tính năng.

Các bản Android cập nhật sau này tập trung vào sự an toàn, quản lý nguồn pin và trải nghiệm người dùng., tăng cường quản lý thông báo và tính năng; mang đến trải nghiệm tùy chỉnh hóa mạnh mẽ. Đặc biệt chú trọng vào bảo mật và quyền riêng tư.



## Kiến trúc hệ điều hành Android

Hệ điều hành Android là một ngăn xếp các thành phần phần mềm, được chia thành 5 phần và 4 lớp chính:

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

**Hình 1.1 Kiến trúc hệ điều hành Android**



### Hạt nhân Linux (Linux Kernel)

Android sử dụng hạt nhân Linux làm hạt nhân của hệ điều hành, cung cấp các chức năng cơ bản như quản lý bộ nhớ, quản lý tài nguyên phần cứng và các phần cứng khác nhằm giao tiếp với các thiết bị ngoại vi dễ dàng hơn.

Hạt nhân Linux là mã nguồn mở, có nghĩa là mã nguồn có sẵn và có thể được sửa đổi phát triển bởi cộng đồng lập trình viên trên toàn thế giới. Sự phát triển linh hoạt của Linux và đã cho phép nó trở thành một trong những hệ điều hành phổ biến nhất trên nhiều nền tảng khác nhau.

### Thư viện (Libraries)

Android cung cấp một loạt các thư viện phần mềm để hỗ trợ các chức năng cơ bản như đồ họa, đa phương tiện, cơ sở dữ liệu, mạng và nhiều hơn nữa. Các thư viện này bao gồm quản lý bề mặt, khung phương tiện, SQLite, OpenGL|ES và nhiều thư viện khác.

### Thời gian chạy Android (Android Runtime)

Trong phiên bản Android trước 5.0 (Android 4.4 và cũ hơn), hệ điều hành Android sử dụng môi trường thực thi Dalvik (Dalvik Runtime). Tuy nhiên, từ phiên bản Android 5.0 trở đi, Android chuyển sang sử dụng môi trường thực thi ART (Android Runtime) làm môi trường thực thi mặc định. Thời gian chạy Android cũng cung cấp một tập các thư viện chính giúp các nhà phát triển ứng dụng Android có thể viết ứng dụng Android bằng Java.



### Bộ khung ứng dụng (Application Framework)

Bộ khung ứng dụng cung cấp các lớp và giao diện ứng dụng cho việc phát triển ứng dụng Android. Điều này bao gồm các thành phần như quản lý hoạt động, cung cấp nội dung, quản lý tài nguyên, hệ thống thông báo và các lớp khác.

Bộ khung ứng dụng Android cung cấp một cách tiếp cận chuyên sâu để phát triển các ứng dụng Android. Nó giúp nhà phát triển tận dụng các tính năng và dịch vụ của hệ điều hành Android một cách tiện lợi và hiệu quả.

### Ứng dụng (Application)

Lớp trên cùng của kiến trúc là ứng dụng. Android hỗ trợ việc phát triển, chạy các ứng dụng di động. Ứng dụng Android có thể được viết bằng Java hoặc Kotlin, được cài đặt và chạy trên các thiết bị di động Android. Đây là tầng giao tiếp với người sử dụng, nơi mà các ứng dụng tương tác và cung cấp các tính năng đa dạng.

Việc có thể tạo ra và cài đặt các ứng dụng đa dạng trên hệ điều hành Android đóng vai trò quan trọng trong việc tăng cường trải nghiệm người dùng và mang lại sự linh hoạt và sự tuỳ chỉnh cho thiết bị di động.

## Hoạt động của hệ điều hành Android

Hệ điều hành Android hoạt động theo mô hình mô hình mạng máy tính gồm có 2 thành phần chính đó là máy khách và máy chủ.Khi người dùng mở một ứng dụng, hệ điều hành sẽ tạo một tiến trình mới cho ứng dụng đó. Ứng dụng sẽ chạy trong tiến trình này cho đến khi người dùng đóng ứng dụng.

## Hệ thống tập tin trên hệ điều hành Android

Hệ điều hành Android sử dụng một hệ thống tập tin cung cấp một cấu trúc tổ chức cho việc lưu trữ, quản lý các tệp tin và thư mục trên thiết bị di động. Cho phép truy cập vào dữ liệu, tương tác với các tệp tin và thư mục thông qua các giao diện người dùng.

## Kết luận

Trên thực tế, chương 1: Tổng quan về hệ điều hành Android đã trình bày chi tiết về tổng thể của hệ điều hành Android. Chương này đã cung cấp cái nhìn tổng quan về lịch sử phát triển, nền tảng, kiến ​​trúc và các thành phần quan trọng của Android.

# CHƯƠNG 2. CÁC PHƯƠNG PHÁP KIỂM THỬ TỰ ĐỘNG

*Chương này sẽ tìm hiểu về khái niệm kiểm thử, các mức độ cũng như các kỹ thuật kiểm thử phần mềm, kiểm thử tự động. Ngoài ra, tìm hiểu một số phương pháp kiểm thử tự động cho ứng dụng Android.*



## Khái quát về kiểm thử phần mềm

### Khái niệm

Kiểm thử phần mềm là một quá trình bao gồm nhiều hoạt động nhằm đánh giá chất lượng các sản phẩm phần mềm và giảm thiểu rủi ro do lỗi gây ra trong quá trình vận hành khi đưa vào sử dụng thực tế. Các hoạt động kiểm thử này bao gồm các hoạt động xem xét đánh giá tài liệu, các bản thiết kế và bao gồm mã nguồn, các hoạt động này trong thực tế.

### Các mức độ kiểm thử phần mềm

Các mức kiểm thử là tập hợp các hoạt động kiểm thử được thực hiện ở một giai đoạn phát triển cụ thể của sản phẩm. Đây là một số mức kiểm thử cơ bản:

* Kiểm thử đơn vị (unit testing)
* Kiểm thử tích hợp (integration testing)
* Kiểm thử hệ thống (system testing)
* Kiểm thử chấp nhận (acceptance testing)

### Các kỹ thuật kiểm thử phần mềm

Kỹ thuật kiểm thử phần mềm giúp thiết kế các trường hợp kiểm thử tốt hơn. Vì kiểm thử toàn diện là không thể nên kỹ thuật kiểm tra thủ công sẽ giúp giảm số lượng các trường hợp kiểm thử được thực thi trong khi tăng phạm vi kiểm thử, giúp xác định các điều kiện kiểm tra khó nhận biết.

* Phân vùng tương đương (equivalence class partitioning)
* Phân tích giá trị biên (boundary value analysis)
* Bảng quyết định (decision table based testing)
* Đoán lỗi (error guessing)

### Kiểm thử tự động

Kiểm thử tự động là một kỹ thuật kiểm thử phần mềm thực hiện bằng cách sử dụng các công cụ kiểm thử tự động đặc biệt để thực hiện kiểm thử. Trong kiểm thử tự động, các công cụ kiểm thử sẽ thực hiện các bước kiểm thử theo kịch bản được xác định trước.

Một số phương pháp kiểm thử tự động cho ứng dụng Android:

* Kiểm thử dựa trên mô hình (Model based Testing)
* Ghi và phát lại (Record and Replay)
* Kiểm thử trên hệ thống (Systematic Testing)
* Kiểm thử mờ (Fuzzing)
* Kiểm thử ngẫu nhiên (Random Testing)
* Kiểm thử theo kịch bản (Scripted based Testing)

Quy trình kiểm thử ứng dụng bao gồm các bước sau:

*Bước 1*: Lựa chọn công cụ kiểm thử: Lựa chọn các công cụ kiểm thử phù hợp với nhu cầu và mục tiêu của dự án.

*Bước 2*: Chuẩn bị dữ liệu kiểm thử: Để cung cấp cho các công cụ kiểm thử.

*Bước 3*: Chạy kịch bản kiểm thử: Chạy kịch bản kiểm thử đã được thiết kế.

*Bước 4*: Phân tích và báo cáo kết quả: Phân tích kết quả kiểm thử và báo cáo cho các bên liên quan.

Sinh dữ liệu kiểm thử là một nhiệm vụ quan trọng trong kiểm thử phần mềm, giúp kiểm tra tính đầy đủ của ứng dụng bằng cách cung cấp một bộ dữ liệu đa dạng và đầy đủ. Việc tự động hóa việc sinh dữ liệu kiểm thử là một giải pháp để cải thiện hiệu quả và năng suất của kiểm thử phần mềm.

## 



### Phương pháp kiểm thử Fuzz Testing



#### Khái niệm Fuzz Testing

Fuzz testing là một phương pháp kiểm thử phần mềm tự động, tạo ra các đầu vào ngẫu nhiên, bao gồm các dữ liệu không hợp lệ, không đúng định dạng hoặc không mong muốn và sau đó đưa các đầu vào này vào hệ thống. Công cụ sẽ giám sát hệ thống để tìm kiếm các lỗi hoặc lỗ hổng và không đòi hỏi quyền truy cập vào mã nguồn nên khả năng tìm thấy lỗi nhanh chóng.

#### Các giai đoạn của kiểm thử Fuzz

*Giai đoạn 1: Xác định mục tiêu (Identify Target)*

*Giai đoạn 2: Xác định đầu vào (Identify the Inputs)*

*Giai đoạn 3: Sinh dữ liệu Fuzz (Generate Fuzzed Data)*

*Giai đoạn 4: Thực thi dữ liệu Fuzz (Excute Fuzzed Data)*

*Giai đoạn 5: Giám sát dữ liệu Fuzz (Monitor for exception Fuzzed Data)*

*Giai đoạn 6: Đăng lỗi và phân tích*

#### Phân loại kiểm thử Fuzz

Việc phân loại kiểm thử Fuzz có thể tùy thuộc vào các vector tấn công, các mục tiêu kiểm thử Fuzz khác nhau hay các phương pháp kiểm thử Fuzz khác nhau, v.v. Tuy nhiên phổ biến nhất là phân loại thành hai phương pháp như ở dưới đây:

* Kiểm thử fuzz dựa trên đột biến (Mutation based Fuzzing)
* Kiểm thử fuzz dựa trên thế hê ̣(Generation based Fuzzing)

#### Các lỗ hổng được phát hiện bởi kiểm thử Fuzz.

Kiểm thử Fuzz làm việc tốt nhất trong việc phát hiện ra lỗi về tràn bộ nhớ kịch bản hóa chéo trang, từ chối dịch vụ , lỗi chuỗi định dạng, chèn câu truy vấn,… Vı̀ thế với kiểm thử Fuzz người ta có thể kiểm tra sự an toàn của bất kỳ quá trình, dịch vụ, thiết bị, hệ thống hoặc mạng máy tı́nh,….

#### Ưu điểm và nhược điểm của kiểm thử Fuzz



### Phương pháp kiểm thử dựa trên mô hình (Model based Testing)








#### 2.2.2.1. Khái niệm kiểm thử dựa trên mô hình (model based testing)

Mô hình là mô tả hành vi của hệ thống. Kiểm thử dựa trên mô hình là một kỹ thuật kiểm tra, nơi mà hành vi thời gian chạy của một phần mềm kiểm thử được kiểm tra để chống lại những dự đoán được thực hiện bởi một đặc tả hoặc mô hình chính thức. Kiểm thử này có thể được áp dụng cho cả với phần cứng và phần mềm.



#### 2.2.2.2. Các loại kiểm thử dựa trên mô hình

* Ngoại tuyến/ tiên nghiệm: Sinh ra các bộ kiểm tra trước khi thực hiện chúng. Bộ kiểm thử chính là bộ kiểm tra hợp lệ của các trường, là tập hợp các ca kiểm thử.
* Trực tuyến/ trực tiếp: Sinh ra các bộ kiểm thử ngay trong quá trình thực hiện kiểm thử.

#### 2.2.2.3. Các mô hình khác nhau trong kiểm thử

* Máy trạng thái hữu hạn
* Biểu đồ trạng thái
* Ngôn ngữ mô hình hóa thống nhất (UML)

#### 2.2.2.4. Ưu điểm và nhược điểm của kiểm thử dựa trên mô hình

## Kết luận

Model based Testing (kiểm thử dựa trên mô hình) và kiểm thử Fuzz testing là hai kỹ thuật kiểm thử phần mềm phổ biến. Kiểm thử dựa trên mô hình sử dụng mô hình của ứng dụng để tạo ra các trường hợp kiểm thử, trong khi kiểm thử kiểm thử Fuzz testing sử dụng các kịch bản được viết trước để kiểm tra ứng dụng.

# CHƯƠNG 3. MỘT SỐ CÔNG CỤ SINH ĐẦU VÀO KIỂM THỬ TỰ ĐỘNG CHO ANDROID

*Chương này tìm hiểu chi tiết về một số công cụ sinh đầu vào kiểm thử tự động cho Android được sử dụng để tạo ra các kịch bản kiểm thử, dữ liệu đầu vào hoặc điều khiển kiểm thử một cách tự động, giúp mở rộng phạm vi và tăng cường khả năng phát hiện lỗi.*



## Công cụ kiểm thử ngẫu nhiên – Monkey Testing Tool



### Tổng quan chung về Monkey Testing Tool

#### Khái niệm về Monkey Testing Tool

Công cụ kiểm thử Monkey là một phần của SDK Android do Google phát triển, là một công cụ phần mềm được sử dụng để thực hiện kiểm thử tự động trên ứng dụng di động Android. Công cụ này được thiết kế để tạo ra các sự kiện và hành vi ngẫu nhiên trên màn hình của thiết bị di động. Với sự tích hợp có sẵn trong Android Studio, Monkey testing tool là công cụ hữu ích để các lập trình viên thử nghiệm ứng dụng trong quá trình phát triển.

#### Ưu điểm của Monkey Testing

#### Nhược điểm của thử nghiệm Monkey

#### Cách sử dụng cơ bản của Monkey Testing Tool

Khởi động Monkey testing tool từ dòng lệnh trên máy tính hoặc từ tập lệnh, kịch bản có sẵn và gõ trực tiếp các lệnh. Cú pháp cơ bản là:

$ adb shell monkey [options] <event-count>

### Kiểm thử Fuzz với Monkey

Bước 1: Xác định hệ thống mục tiêu: truyền các tham số về gói và danh mục của ứng dụng cần thực thi quá trình kiểm thử vào trong dòng lệnh

Bước 2: Xác định đầu vào: chọn và đặt tốc độ tỉ lệ sự kiện mong muốn xuất hiện được tạo ra trong quá trình thực thi kiểm thử.

Bước 3: Sinh dữ liệu kiểm thử: sau khi xác định hệ thống mục tiêu và các thành phần, Monkey tạo đầu vào thử nghiệm, là những sự kiện đáp ứng các yêu cầu cụ thể.

Bước 4: Thực hiện kiểm thử: Các lệnh ADB được truyền tới thiết bị kiểm thử và thực thi các thao tác sử dụng một cách hoàn toàn tự động.

Bước 5: Giám sát hành vi hệ thống: trong quá trình thực thi, các sự kiện sinh ra đều được lưu lại dưới dạng các tệp tin log.

Bước 6: Đăng lỗi và phân tích: Monkey ghi lại những điểm bất thường trong hoạt động của ứng dụng và tạo ra các thông báo lỗi trên màn hình điều khiển.

## Công cụ kiểm thử dựa trên mô hı̀nh – Droidbot



### Tổng quan chung về Droidbot

#### 3.2.1.1. Tìm hiểu về Droidbot

Droidbot là một công cụ sinh đầu vào kiểm thử mã nguồn mở dựa trên giao diện người dùng (UI) cho các ứng dụng Android. Được phát triển bởi Yuanchun Li, một nghiên cứu sinh tại Học viện phần mềm, Đại học Bắc Kinh. Droidbot được thiết kế với nguyên tắc hỗ trợ việc tạo ra các đầu vào kiểm thử dựa trên mô hình với yêu cầu tối thiểu.

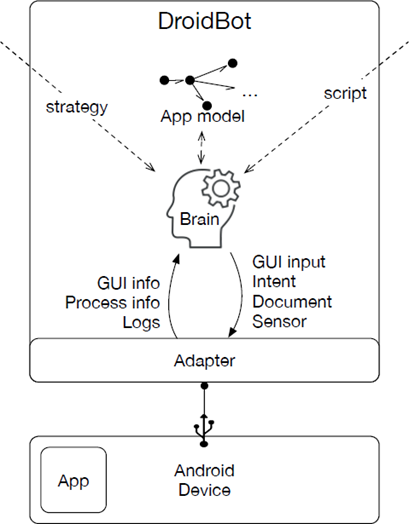
Droidbot cung cấp một bộ sinh đầu vào theo hướng dẫn UI, dựa trên mô hình chuyển đổi trạng thái được tạo ra trong quá trình thực thi. Mặc định, Droidbot sử dụng chiến lược tham ăn theo chiều rộng để sinh ra các đầu vào.

Droidbot là một công cụ nhẹ, nó chỉ mô hình hóa các trạng thái đã được khám phá dựa trên bộ công cụ kiểm tra/gỡ lỗi tích hợp sẵn của Android, cho phép Droidbot hoạt động trên bất kỳ ứng dụng trên hầu hết các thiết bị tùy chỉnh.

Mã nguồn của Droidbot được chia sẻ công khai và lưu trữ trực tiếp trên GitHub: [https://github.com/honeynet/droidbot](file:///C:\Users\Asus\AppData\Roaming\Microsoft\Word\BaocaoDATN4.docx)

#### 3.2.1.2. Kiến trúc của Droidbot

Để kiểm tra một ứng dụng trên một thiết bị, Droidbot yêu cầu thiết bị phải được kết nối thông qua cầu nối gỡ lỗi (ADB). Thành phần đầu tiên của Droidbot ở đây là mô đun Adapter được sử dụng để trừu tượng hóa của thiết bị và ứng dụng thử nghiệm. Adapter cũng hoạt động và đóng vai trò như là một cầu nối giữa môi trường kiểm thử và thuật toán kiểm thử. Mô đun Brain nhận thông tin của thiết bị và ứng dụng trong thời gian chạy rồi gửi các đầu vào kiểm thử được sinh ra đến Adapter.



Hình 3.1: Kiến trúc tổng quan của Droidbot



### Kiểm thử dựa trên mô hình với Droidbot

Bước 1: Mô hình hóa: Droidbot lấy thông tin giao diện người dùng từ ứng dụng kiểm thử: đối với mỗi giao diện người dùng, Droidbot ghi lại ảnh chụp màn hình và cây phân cấp nhằm tạo ra một mô hình của ứng dụng kiểm thử dựa trên thông tin được giám sát ngay trong quá trình chạy.

Bước 2: Lựa chọn yêu cầu kiểm thử: Droidbot tích hợp bốn thuật toán thăm dò khác nhau là naive depth-first, naive breadth-first, greedy depth-first và greedy breadth-first bên cạnh lựa chọn khám phá bằng Monkey.

Bước 3: Sinh dữ liệu kiểm thử: Các loại đầu vào kiểm thử được Droidbot hỗ trợ bao gồm đầu vào giao diện, tải lên tài liệu và dữ liệu cảm biến. Droidbot cung cấp một danh sách cho việc truy xuất thông tin từ thiết bị và gửi đầu vào tới thiết bị.

Bước 4: Cụ thể hóa quá trình kiểm thử: Droidbot truy xuất thông tin của thiết bị/ứng dụng từ thiết bị và gửi đầu vào kiểm thử tới thiết bị thông qua ADB.

Bước 5: Thực thi kiểm thử: Các sự kiện sinh ra bởi Droidbot thông qua ADB sẽ được thực thi trên thiết bị kiểm thử. Đồng thời cũng trong quá trình thực thi thử nghiệm, các thông tin sẽ được giám sát chặt chẽ, lưu lại và hiển thị ngay trên màn hình.

## Kết Luận.

Monkey testing và Droidbot là hai kỹ thuật kiểm thử tự động được sử dụng để phát hiện lỗi trong ứng dụng Android. Monkey testing là kỹ thuật đơn giản, dễ triển khai nhưng có thể không hiệu quả trong việc phát hiện các lỗi có thể dự đoán được. Droidbot là kỹ thuật phức tạp hơn, đòi hỏi kiến thức về ứng dụng nhưng có thể hiệu quả hơn trong việc phát hiện các lỗi.

# CHƯƠNG 4. THỰC NGHIỆM KIỂM THỬ TỰ ĐỘNG CHO ỨNG DỤNG ANDROID

*Ở chương này sẽ tiến hành xây dựng ứng dụng thực hiện kiểm thử tự động cho ứng dụng Android với công cụ Droidbot. Quá trình này sẽ thực thi việc cài đặt, kiểm thử ứng dụng, ghi lại số liệu về thời gian thực hiện, số lỗi. Từ đó, rút ra được nhận xét về ưu, nhược điểm của Droidbot.*



## Lựa chọn công cụ kiểm thử

Droidbot là một công cụ kiểm thử tự động mã nguồn mở dựa trên mô hình cho các ứng dụng Android, sử dụng các mô hình của ứng dụng để tạo ra các sự kiện ngẫu nhiên, giúp tăng khả năng phát hiện lỗi.

## Triển khai thử nghiệm



### Cài đặt Droidbot

* Phiên bản Python (cả 2 và 3 đều được hỗ trợ)
* Java
* Android SDK
* Thêm platform\_tools thư mục trong SDK Android vào PATH
* Androguard
* Networkx
* Pillow

Cài đặt Droidbot: sao chép mã nguồn : <https://github.com/honeynet/droidbot>

Thực hiện cài đặt Droidbot bằng lệnh: pip install -e .

Sử dụng Samsung Galaxy J6 (SM-J600G/DS), hệ điều hành Android Pie 9.0 để cài đặt các ứng dụng và tiến hành kiểm thử.



### Kiểm thử tự động ứng dụng Android

#### 4.2.2.1. Cài đặt các ứng dụng Android

Tìm và tải xuống bản apk mã nguồn mở của các ứng dụng.

|  |  |  |
| --- | --- | --- |
| **STT** | **TÊN ỨNG DỤNG** | **LINK MÃ NGUỒN** |
| 1 | **A Photo Manager** | <https://f-droid.org/en/packages/de.k3b.android.androFotoFinder/> |
| 2 | **Clock Screensaver** | <https://f-droid.org/en/packages/systems.sieber.fsclock/> |
| 3 | **Coloring Book** | <https://f-droid.org/en/packages/eu.quelltext.coloring/> |
| 4 | **Dooz** | <https://f-droid.org/en/packages/io.github.yamin8000.dooz/> |
| 5 | **Fossify Calendar** | <https://f-droid.org/en/packages/org.fossify.calendar/> |
| 6 | **MemoPad** | <https://f-droid.org/en/packages/org.zakky.memopad/> |
| 7 | **Pocket Broomball** | <https://f-droid.org/en/packages/com.simondalvai.pocketbroomball/> |
| 8 | **Souvenirs** | <https://f-droid.org/en/packages/fr.nuage.souvenirs/> |
| 9 | **TextPad** | <https://f-droid.org/en/packages/com.maxistar.textpad/> |
| 10 | **Zen** | <https://f-droid.org/en/packages/zen.meditation.android/> |

**Bảng 4. 1: Các ứng dụng mã nguồn mở**

Các ứng dụng này đều được lấy mã nguồn từ một trang ứng dụng https://f-droid.org/. F-Droid là một danh mục có thể cài đặt gồm các ứng dụng dành riêng cho nền tảng Android với sự tải xuống đơn giản, dễ dàng duyệt, cài đặt và theo dõi các bản cập nhật trên thiết bị.

#### 4.2.2.2. Hoạt động kiểm thử tự động

Thực hiện kiểm tra tự động với các ứng dụng đã tải.

Với việc kiểm tra tự động, mỗi ứng dụng có thể được thực hiện một lần hoặc nhiều lần với số lượng các sự kiện trong mỗi lần kiểm thử là khác nhau (500 sự kiện và 1000 sự kiện).

Cú pháp cơ bản là:

droidbot –a <đường dẫn apk> -o <đường dẫn sinh báo cáo> -count <số lượng sự kiện: 500/1000> -grant\_perm



## Phân tích, đánh giá kết quả thu được.

Droidbot sinh các sự kiện dựa trên mô hình, sử dụng chiến lược và thuật toán duyệt theo chiều rộng. Do đó, các sự kiện được sinh ra lần lượt theo những luồng nhất định. Điều này có thể giúp Droidbot phát hiện được các lỗi phổ biến trong ứng dụng Android. Tuy nhiên, Droidbot cũng có thể gặp khó khăn khi gặp các giao diện phức tạp, chẳng hạn như giao diện có trường nhập thông tin chứa các yêu cầu đặc biệt hoặc giao diện mà các thành phần thông tin không hiển thị sẵn trên màn hình.

Tuy nhiên, thời gian thực hiện của Droidbot khá lâu. Bởi vì với mỗi sự kiện, Droidbot sẽ lưu lại kịch bản thực hiện, ảnh chụp màn hình và lưu lại các luồng giao diện đã đi qua.

Để cải thiện hiệu quả của Droidbot, các nhà phát triển có thể thực hiện các biện pháp sau:

* Sử dụng các công nghệ mới để giảm thời gian thực hiện
* Tăng cường khả năng xử lý giao diện phức tạp

Với những cải tiến này, Droidbot sẽ trở thành một công cụ kiểm thử tự động hiệu quả hơn.

# KẾT LUẬN

Sau khi nghiên cứu và tìm hiểu về đề tài “**Tìm Hiểu Một Số Kỹ Thuật Sinh Dữ Liệu Kiểm Thử Tự Động Cho Android**”, luận văn đã đạt được những kết quả như sau:

Trước hết, nghiên cứu này đóng góp vào việc đưa ra cái nhìn tổng quan về quá trình kiểm thử tự động. Luận văn đã trình bày một cách hệ thống và đầy đủ các khái niệm, thuật ngữ của quá trình kiểm thử tự động. Đặc biệt, luận văn đã nhấn mạnh tầm quan trọng của sinh dữ liệu kiểm thử tự động trong quá trình kiểm thử tự động cho ứng dụng Android.

Bên cạnh đó, luận văn đã phân tích chi tiết ưu và nhược điểm của hai kỹ thuật sinh dữ liệu kiểm thử tự động là kiểm thử Fuzz và kiểm thử dựa trên mô hình, giúp người đọc có cái nhìn tổng quan và chi tiết hơn về từng kỹ thuật, từ đó có thể lựa chọn kỹ thuật phù hợp cho mục đích sử dụng.

Luận văn đã thực hiện thí nghiệm kiểm thử tự động với các ứng dụng mã nguồn mở khác nhau. Kết quả thu được từ thực nghiệm, bao gồm số lượng lỗi và thời gian thực thi của mỗi lần kiểm thử tự động nhằm phân tích và đánh giá hiệu suất của phương pháp kiểm thử cũng như công cụ kiểm thử tự động Droidbot trong thực tế.

Tuy nhiên, luận văn vẫn còn một số những hạn chế như: Số lượng công cụ kiểm thử và số lượng ứng dụng được lựa chọn chưa được đa dạng và đầy đủ các lĩnh vực. Luận văn chỉ nghiên cứu một số công cụ kiểm thử tự động phổ biến và một số ứng dụng mã nguồn mở trong một số lĩnh vực nhất định. Do đó, kết quả nghiên cứu có thể chưa phản ánh đầy đủ hiệu quả của các kỹ thuật sinh dữ liệu kiểm thử tự động trong thực tế. Luận văn sử dụng dữ liệu sẵn có để thực hiện thí nghiệm kiểm thử tự động. Tuy nhiên, số lượng dữ liệu này có thể còn hạn chế và chưa đại diện đầy đủ cho mọi trường hợp. Điều này có thể ảnh hưởng đến kết quả thực nghiệm.

Việc triển khai các mô hình sinh dữ liệu kiểm thử tự động vào thực tế và đánh giá sự hiệu quả của chúng trong môi trường thực tế là một thách thức quan trọng. Luận văn chưa đề cập đến vấn đề này một cách đầy đủ.

Giải pháp là có thể mở rộng phạm vi nghiên cứu bao gồm nhiều công cụ kiểm thử tự động khác nhau và nhiều ứng dụng mã nguồn mở trong các lĩnh vực khác nhau. Điều này sẽ giúp phản ánh đầy đủ hơn hiệu quả của các kỹ thuật sinh dữ liệu kiểm thử tự động trong thực tế. Các nhà nghiên cứu có thể thu thập thêm dữ liệu hoặc xây dựng các mô hình sinh dữ liệu mới để tăng cường số lượng dữ liệu sẵn có. Điều này sẽ giúp cải thiện độ chính xác của kết quả thí nghiệm.

Nghiên cứu các giải pháp, triển khai các mô hình sinh dữ liệu kiểm thử tự động vào thực tế và đánh giá sự hiệu quả trong môi trường. Lựa chọn các công cụ thicchs hợp để phát triển và cải tiến hơn nữa nhằm phục vụ cho công việc. Điều này sẽ giúp đảm bảo rằng các mô hình này có thể được sử dụng hiệu quả trong công việc kiểm thử phần mềm, kiểm thử tự động.

# TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | S. Kemp, "DataReportal," DataReportal – Global Digital Insights, 4 2 2023. [Online]. Available: https://datareportal.com/reports/digital-2023-global-overview-report. |
| [2] | A. T. i. t. C. o. B. stores, "Android vs. Apple Market Share: Leading Mobile OS," BankMyCell, [Online]. Available: https://www.bankmycell.com/blog/android-vs-apple-market-share/. |
| [3] | T. Hamilton, "UIAutomatorViewer tutorial: Inspector for Android testing," Guru99, 5 1 2024. [Online]. Available: https://www.guru99.com/uiautomatorviewer-tutorial.html. |
| [4] | L. T. H. Thuong, "MỘT SỐ CÔNG CỤ KIỂM THỬ TỰ ĐỘNG ỨNG DỤNG TRÊN ANDROID," Viblo, [Online]. Available: https://viblo.asia/p/mot-so-cong-cu-kiem-thu-tu-dong-ung-dung-tren-android-naQZRYPvKvx. |
| [5] | Baodientuvtv, "Tiềm năng phát triển của ngành thiết kế ứng dụng di động tại Việt Nam," vtv.vn, 1 9 2023. [Online]. Available: https://vtv.vn/cong-nghe/tiem-nang-phat-trien-cua-nganh-thiet-ke-ung-dung-di-dong-tai-viet-nam-20230823222518989.htm. |
| [6] | T. Le, "Lịch sử và chặng đường phát triển của Android qua các phiên bản," Trang thông tin dành cho tín đồ công nghệ, 11 11 2021. [Online]. Available: https://genk.vn/lich-su-va-chang-duong-phat-trien-cua-android-qua-cac-phien-ban-phan-1-thoi-ky-dau-cua-android-20150608204249527.chn. |
| [7] | "Tìm hiểu thông tin chi tiết về hệ điều hành Android," 3 10 2023. [Online]. Available: https://m.45cm.com/blog/he-dieu-hanh-android-133.html. |
| [8] | T. Anh, "Android là gì? Hệ điều hành android là gì? Ưu, nhược điểm của Android," HOSTVN, 9 8 2019. [Online]. Available: https://digilux.vn/android-la-gi/. |
| [9] | "Tìm hiểu về hệ điều hành Android: Khái niệm, ưu nhược điểm, các phiên bản hiện tại," Công nghệ mới nhất - Đánh giá - Tư vấn thiết bị di động, 18 1 2023. [Online]. Available: https://didongviet.vn/dchannel/he-dieu-hanh-android/. |
| [10] | D. T. N. T. T. N. X. T. Kieu Linh, "Nhìn lại các phiên bản Android từ 1.0 đến 14 (Phần 1)," HoangHaMobile, 18 4 2023. [Online]. Available: https://hoanghamobile.com/tin-tuc/nhin-lai-cac-phien-ban-android-tu-1-0-den-14-phan-1/. |
| [11] | D. T. N. T. T. N. X. T. Kieu Linh, "Nhìn lại các phiên bản Android từ 1.0 đến 14 (Phần 2)," HoangHaMobile, 19 4 2023. [Online]. Available: https://hoanghamobile.com/tin-tuc/cung-nhin-lai-cac-phien-ban-android-tu-1-0-den-14-phan-2/. |
| [12] | "Android - Kiến trúc," [Online]. Available: https://www.tutorialspoint.com/android/android\_architecture.htm. |
| [13] | N. Đào, "Kiểm thử phần mềm là gì?," TESTING VN, 17 11 2022. [Online]. Available: https://www.testing.vn/kiem-thu-phan-mem/. |
| [14] | N. Trang, "Các kỹ thuật kiểm thử phần mềm," Viblo, [Online]. Available: https://viblo.asia/p/cac-ky-thuat-kiem-thu-phan-mem-gGJ599eG5X2. |
| [15] | C. Q.-L. a. M. J. Abel Méndez-Porras, "Automated Testing of Mobile Applications: A Systematic Map and Review," p. 1. |
| [16] | "What is Fuzz Testing and how does it work?," Synopsys, [Online]. Available: https://www.synopsys.com/glossary/what-is-fuzz-testing.html#:~:text=Definition,as%20crashes%20or%20information%20leakage. |
| [17] | K. C. n. t. t. -. Đ. h. D. Tân, "Bài 01 - Sơ lược về Kiểm tra mờ," Bài viết sưu tầm, [Online]. Available: https://kcntt.duytan.edu.vn/Home/ArticleDetail/vn/128/2461/bai-01-so-luoc-ve-fuzzing-testing. |
| [18] | T. T. H. Trang, "Tìm hiểu về Fuzz testing," Viblo, [Online]. Available: https://viblo.asia/p/tim-hieu-ve-fuzz-testing-YWOZrDzv5Q0. |
| [19] | L. T. Hà, "Giới Thiệu Về kiểm thử fuzzing & tool test," Viblo, [Online]. Available: https://viblo.asia/p/gioi-thieu-ve-kiem-thu-fuzzing-tool-test-bWrZngPOlxw. |
| [20] | T. Hamilton, "What is model based testing?," Guru, 30 September 2023. [Online]. Available: https://www.guru99.com/model-based-testing-tutorial.html. |
| [21] | S. Guillaume, "The model-based testing," Matelo, 20 October 2022. [Online]. Available: https://matelo-testing-software.com/blog/model-based-testing-en/the-model-based-testing/. |
| [22] | "Giao Diện Người Dùng/ trình tập ứng dụng monkey : Android studio," Android developers, [Online]. Available: https://developer.android.com/studio/test/other-testing-tools/monkey?hl=vi. |
| [23] | Đ. T. Duyên, "Tìm Hiểu Về Monkey Testing (phần 1)," Viblo, [Online]. Available: https://viblo.asia/p/tim-hieu-ve-monkey-testing-phan-1-1Je5EGaAZnL. |
| [24] | Z. Y. Y. G. X. C. Yuanchun Li, "DroidBot: A Lightweight UI-Guided Test Input". |
| [25] | "Droid - Free and Open Source Android App Repository," [Online]. Available: https://f-droid.org/en/. |