Automated GUI Testing for Android News Applications

***Abstract*—Automated GUI (graphical user interface) testing tools have been used to help engineers test whether the software GUI is displayed correctly in different smartphones. However, due to different screen aspect ratios, the ratio of width to height, the same content of a mobile application (app) may have a different layout in different smartphones. As a result, the test oracle generated by traditional methods may not be reused for different smartphones and thus prolong the testing process. In this paper, we present a GUI testing tool, named FLAG (Fully Automatic mobile GUI testing), which aims to make the test oracle reusable without compromising test accuracy. In addition, the whole testing process, including generating test cases, simulating user gestures and verifying results, is automatically performed by FLAG without human interaction. News applications have been selected for our study not only because they are popular, but also because they support most commonly-used user gestures, such as tap, scroll, spread and pinch. In our experiment, we selected five commercial Android phones and one popular news apps to evaluate the effectiveness of the FLAG. Our experiment results show that the FLAG performs better than existing methods and can achieve an average accuracy of 95.20% in determining whether a test has passed or failed.**

***Index Terms*—GUI testing tools; test case generator; test oracle; Android phone**

I. Introduction

With mobile internet services becoming mature, the frequencies of people around the world reading mobile news through news applications (apps) have been increasing [1, 2]. Large television companies (i.e., NBC and FOX) and web portal (i.e., MSN and Yahoo) have all developed their own news apps for users to browse real-time news on their smartphones. A report by the Pew Research Center suggested that 33% of smartphone users frequently followed the latest news by using their phones [3]. As a result, ensuring the news apps works properly in different smartphones are crucial for large television companies, web portal and mobile app developers because the correctness of GUI display can affect users' experience and their selection of news apps. In addition, automated GUI (Graphical User Interface) testing tools are highly desirable so as to reduce time and cost of software development.

Automated GUI testing consists of simulating user gestures, such as tap, long press and scroll, and validating the changes in the GUI by a test oracle, which is a mechanism used for determining if an app displays correctly. In order to determine whether a test has passed or failed, given a user gesture input, the test oracle compares the GUI display of the app under test to the GUI display that app should have. Recently, some methods have been developed to perform automated GUI testing, such as Sikuli [4], SPAG-C [5] and GUICOP [6]. Sikuli[4] is an open source automated GUI testing solution that first redirects the screen of the app under test from a smartphone to a PC. The test oracle of Sikuli then compares the redirected screen to the prerecorded screenshot to verify the correctness of the app. Similarly, SPAG-C first adopted an external camera to capture the screenshot of the app under test. Three image comparison methods were then used by the test oracle to compare the captured screenshot with the prerecorded screenshot [5]. However, due to different screen aspect ratios, the ratio of width to height, the same text content of a news app may be displayed differently in different smartphones, such as breaking a line in a different place. As a result, the test oracle of Sikuli and that of SPAG-C may not be reused for different smartphones and thus prolong the testing process. GUICOP used the GUI layout information instead of screen images to determine whether a test has passed or failed [6]. However, if there is an issue with the device's screen driver and that issue caused the app to be displayed incorrectly, GUICOP will not detect the problem.

In this paper, given a news app that works correctly, we aim to find out if a given Android phone is capable of displaying the news app as expected. The given Android phone could be a prototype, a new type of smartphone, a mainstream smartphone, a smartphone with updated system software or so on. We focus on the accuracy and reusability of the test oracle. For this, we designed and implemented FLAG, a Fully Automatic mobile GUI testing tool, to make the test oracle reusable without compromising test accuracy. FLAG automatically performs the process of test case generation, user gesture simulation and screenshot verification without human interaction. FLAG first analyzes the GUI layout of the given news app and then generates test cases for all possible GUI operations. Next, based on the test cases, the screen aspect ratios and pixel resolution, FLAG simulates associated user gestures on the given Android phone. When verifying the correctness of GUI display, FLAG first identifies texts and images of the screenshot by the OCR (Optical

978-1-5386-7036-1/18/$31.00 ©2018 IEEE
DOI 10.1109/IS3C.2018.00013

Character Recognition) technology [7]. Then, the correctness of the texts are determined by the edit distance while that of the images are determined by SURF (Speeded Up Robust Features) comparison algorithm [8]. The test is to pass if both the text comparison and the image comparison are passed.

News applications have been selected for our study not only because they are popular, but also because they support most commonly-used user gestures, such as tap, scroll, spread and pinch. For example, tap gesture can be used to select a news article, scroll gesture can be used to move forward and go back a page, spread and pinch gestures can be used to enlarge and shrink an object. Therefore, FLAG is capable of testing not only news apps, but also other types of mobile apps, such as weather apps, comic apps, e-book apps, magazine apps, gallery apps and so on.

In our experiment, we selected five commercial Android phones and three popular new apps to evaluate the effectiveness of the FLAG. The Android phones included HTC M7, Samsung S2, Samsung S3, SONY TX, and Samsung S4, while the news app was MSN news app. We compared the accuracy of FLAG, Sikuli and SPAG-C in determining whether a screenshot is displayed correctly or not. Our experiment results show that FLAG can achieve an average accuracy of 95.20%, which is much higher than Sikuli's 52.65% and SPAG-C's 5.78%. FLAG is proved to be a practical solution to make the test oracle reusable without compromising test accuracy.

II. Methodology

A. FLAG overview

Figure 1 shows the architecture of FLAG. A news app A works correctly on the Android phone D. We aim to find out if a different Android phone D' can display the news app A correctly. In other words, D' is the device under test (DUT). The testing process are divided into two stages: the test case generation stage and the verification stage. Each of them is described as follows.

In the test case generation stage, shown in Figure 1(a), FLAG first adopts the uiautomator viewer [9] to analyze the GUI components currently displayed on the Android phone D. The obtained GUI information are sent to the Test Case Generator (TCG) for further analysis (Step 1). Based on the obtained GUI information, the TCG generates a test case including a set of GUI operations, such as tap, long press or scroll (Step 2). The GUI operations of the test case are sent to the Gesture Player to simulate user gestures (Step 3). Finally, the Test Oracle saves the screenshot generated by each simulated user gesture into a local storage (Step 4). The saved screenshots will be used in the the stage of verification to determine whether the test is passed or not. Because a news app usually has multiple level pages, the FLAG will continue to repeat step 1 to 4 until all pages are visited.

In the verification stage, shown in Figure 1(b), the GUI operations of the test case are sent to the Gesture Player to simulate the same user gestures on the Android phone D' (Step A). The screenshot generated by each user gesture is sent to the Test Oracle for verification (Step B). In order to determine if the Android phone D' displays correctly, the test oracle first loads the GUI image that app should display from the local storage, and then compares the loaded image with the screenshot (Step C).

B. Test case generator

The purpose of the Test Case Generator is to generate a test case, which is a set of GUI operations. A GUI operation can be a tap, press or scroll. In order to obtain the information of GUI components currently displayed on the Android phone D, FLAG adopts the uiautomator viewer [9] to analyze the news app A without the need to have access to the source code of A. A GUI component can be a menu list, a button, a text block or a scrollview. For each GUI component, FLAG first uses the Gesture Player to simulate the acceptable GUI operations of the the GUI component. Then, the screenshot of each GUI operation is stored by the test oracle in a local storage. Because a news app usually has multiple level pages, FLAG uses the depth-first search (DFS) method to visit each page and then perform appropriate GUI operations on every GUI component. The test case is completed until all GUI components are visited.

C. Gesture player

The purpose of the Gesture Player is to accurately reproduce the GUI operations of the test case as so to determine if the the device under test (i.e., Android Phone D') can display the news app A correctly. In this paper, a GUI operation is defined as a user gesture, such as tap, long press and scroll. In other words, we need to simulate the same user gestures at the right location of the screen of D' so as to generate the same display results. In this section, we first describe how the Gesture Player accurately simulates the three types of GUI operation; that is, tap, long press and scroll. We than introduce the method of coordinate system transformation.

For an Android phone, a GUI operation consists of a sequence of sub-operations. The sub-operations include press, move and release. In order to simulate a tap gesture, a press sub-operation and a release sub-operation should be sent to the Android kernel accordingly. For a long press gesture, we set the press duration of the press sub-operation at 5 sec. In order to simulate a scroll up gesture with a constant speed starting from the point (x, y) to the destination point with a distance d, a press sub-operation should be sent to the Android kernel first. Then, n move sub-operations are triggered accordingly to simulate the movement of a scroll gesture. The release sub-operation finally ends the scroll gesture.

D. Test oracle

The purpose of the Test Oracle is to check whether the device under test (i.e., Android Phone D') displays the news app A as expected or not. An intuitive method is to directly compare the screenshot image of D' with the corresponding image in the test oracle for similarity. However, the GUI layout

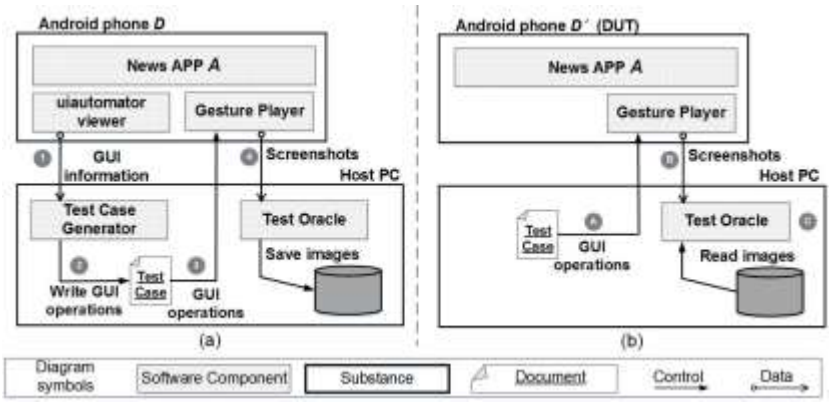

Fig. 1. System structure of FLAG and the device under test: (a) the test case generation stage and (b) the verification stage

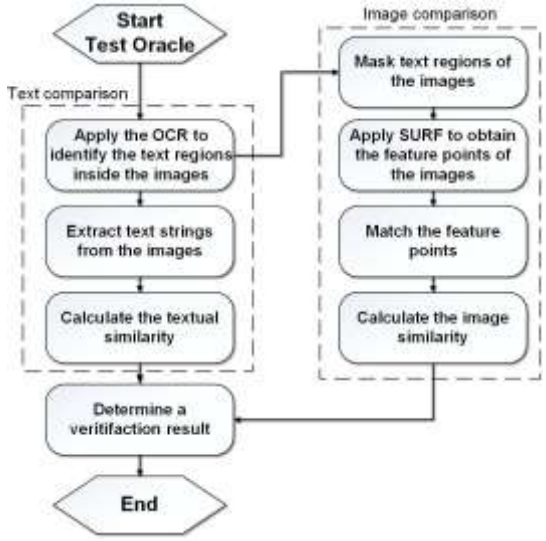

Fig. 2. The flow chart of the Test Oracle

of a smaller phone may be different depending on the size of a screen. In the case with a news app, a line of text may be displayed as two lines on a smaller phone. Using the similarity of the two images to determine whether the test has failed or passed may be incorrect. Therefore, we first identify the image regions and the text regions of the screenshot image of D'. We then verify the correctness of the image regions and text regions individually.

Figure 2 is the flow chart of the Test Oracle which takes the screenshot of D' and the corresponding image as input. We first apply the OCR (Optical Character Recognition) technology [10] to identify the text regions within the two images. For each image, we next remove the text regions from the image by masking it with white rectangles. The two masked images are then used for image comparison, in which we compare the feature points of the two images obtained by the SURF algorithm [8] to make a pass or fail decision. For the text region, we evaluate the content of each region to determine the similarity. A test will pass if both the image comparison and text comparison are passed.

III. Experiment

A. Experiment setup

In our experiment, we adopted a PC with an Intel I7 3.40 GHz CPU and 4GB as the host PC. In addition, five commercial smartphones of different specs were tested, including HTC M7, Samsung S2, Samsung S3, SONY TX, and Samsung S4. The application tested was MSN NEWS [11], with more than 500,000 downloads. In order to have a comprehensive evaluation, we had every Android phone take turns in generating the test case and test oracles. To make sure the experiment environment remained unchanged, the internet connection was cut off during the experiment, so that the apps would not automatically retrieve the latest news. Before the internet connection was cut off, the news pages content was downloaded and cached on each smartphone. Based on the cached news pages, the FLAG generated the test case.

We compared our solution with Sikuli [4] and SPAG-C [5] respectively. Sikuli redirected the screen of a smartphone to the host PC to determine the test is to pass or fail. In order to make a fair comparison, we did not change any default settings of Sikuli. SPAG-C used three image comparison methods, SURF [8], Image histogram [12], and Template matching [13] to determine if two images were the same. Two images were considered matched only when the results of all three methods indicated so.

B. The accuracy of the test oracles for the MSN News App

The purpose of this experiment was to evaluate the accuracy and false positives of different testing methods for the MSN news app. The five commercial Android phones are able to display the the MSN news app correctly. In order to have a comprehensive evaluation, for each testing method, we had every Android phone take turns in generating the test case

and test oracles. The other four Android phones, meanwhile, were the devices under test. We evaluated the effect of the Android phone on the accuracy and the false positive of the testing method. The news category selected for the experiment was "Top stories". There were 306 images; that is, 306 GUI operations were included in the test case. The first image was the home page image of this category. The average values were summarized in Figure 3, in which the x axis represents the three mobile GUI testing methods, each of which was used to test the five different Android phones. The y axis is the testing accuracy. According to the experiment results, FLAG has an accuracy of 97.22% and a false positive rate of 2.78%. In particular, for FLAG, when HTC m7 is selected to generate the test case and the test oracle. FLAG has an accuracy of 98.20% and a false positive rate of 1.80%. The false positives were induced by the OCR text identification. The lower the resolution of the images is, the lower the accuracy of the OCR text identification is, and the lower the accuracy of FLAG is.

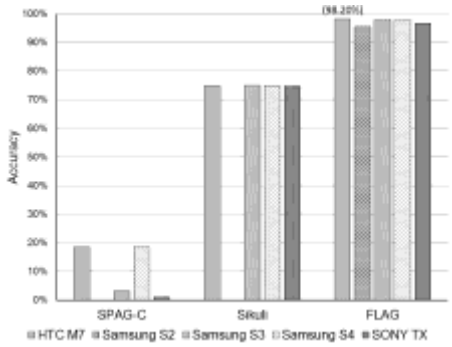

Fig. 3. The accuracy of the test oracles for the MSN news app

The results show that Sikuli failed to test Samsung S2. The reason was that Sikuli compared the the images without considering the text content. Two images with different layouts would not be considered similar by Sikuli. For Samsung S2, the number of characters displayed in a line was rather small because the width of Samsung S2 was shorter than other phones we tested. As a result, the same text paragraph was displayed in more lines on Samsung S2 than other phones. Therefore, the performance of Sikuli with Samsung S2 was bad. On the contrary, FLAG was not influenced by the image sizes. The accuracy of Sikuli was 59.86%, 37.36% much lower than that of FLAG. The false positive rate of Sikuli was 40.14%.

SPAG-C used three different image comparison methods to verify the test results. Every method had its own threshold for similarity test. The two images, the screenshot of the device under test and the test oracle, were considered as identical only when all three methods determined so. However, these thresholds must be set manually and the optimal settings of SPAG-C highly depended on the app to be tested. Our solution, on the other hand, determined the thresholds of similarity without human interaction. The experiment results showed that the test oracles of SPAG-C could not be used across platforms with the news apps because a test is said to pass if it passes the examination of the three different image comparisons. Similar images were considered different by the image comparison methods used by SPAG-C and resulted in wrong comparison results. The accuracy of SPAG-C with the MSN news app was only 8.26%, which was very low. The false positive rate of SPAG-C was 91.74%.

IV. Conclusion

In this paper, we designed and implemented FLAG to make the test oracle reusable without compromising test accuracy. The process starts with the Test Case Generator, which can generate test cases automatically; and the next is the Gesture Player, which can reproduce all the operations in the test cases, followed by the Test Oracle, which can compare the display from the tested device and the image from the Oracle to decide the test result. Our experiment results showed that FLAG can achieve an average accuracy of 95.20%, which is much higher than Sikuli's 52.65% and SPAG-C's 5.78%. FLAG is proved to be a practical solution to make the test oracle reusable without compromising test accuracy. In the future, we plan to further extend the functionality of FLAG in testing different types of apps.

V. Acknowledgment

The work was supported by Ministry of Science and Technology of Taiwan.

References

[1] F. Nel, "Where else is the money? a study of innovation in online business models at newspapers in britain's 66 cities," *Journalism Practice*, vol. 4, no. 3, pp. 360–372, 2010.
[2] O. Westlund, "Mobile news: A review and model of journalism in an age of mobile media," *Digital Journalism*, vol. 1, no. 1, pp. 6–26, 2013.
[3] A. Smith, "U.s. smartphone use in 2015," 2015. [Online]. Available: http://www.pewinternet.org/2015/04/01/us-smartphone-use-in-2015/
[4] T. Yeh, T.-H. Chang, and R. C. Miller, "Sikuli: Using gui screenshots for search and automation," in *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '09, 2009, pp. 183–192.
[5] Y.-D. Lin, J. Rojas, E. T.-H. Chu, and Y.-C. Lai, "On the accuracy, efficiency, and reusability of automated test oracles for android devices," *IEEE Transactions on Software Engineering*, vol. 40, no. 10, pp. 957–970, Oct 2014.
[6] F. Zaraket, W. Masri, M. Adam, D. Hammoud, R. Hamzeh, R. Farhat, E. Khamissi, and J. Noujaim, "Guicop: Specification-based gui testing," in *Proceedings of the 2012 IEEE 5th International Conference on Software Testing, Verification and Validation*, ser. ICST '12, 2012, pp. 747–751.
[7] S. Mori, H. Nishida, and H. Yamada, *Optical character recognition*. John Wiley & Sons, Inc., 1999.
[8] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, Jun. 2008.
[9] Android_Developers, "uiautomatorviewer," 2015. [Online]. Available: https://developer.android.com/training/testing/ui-testing/uiautomator-testing.html
[10] R. Smith, "An overview of the tesseract ocr engine," in *Proceedings of the Ninth International Conference on Document Analysis and Recognition - Volume 02*, ser. ICDAR '07, 2007, pp. 629–633.
[11] . Microsoft Corporation, "Msn news - breaking headlines," 2015.
[12] G. Pass and R. Zabih, "Comparing images using joint histograms," *Multimedia Syst.*, vol. 7, no. 3, pp. 234–240, May 1999.
[13] Opencv, "template matching." 2015.