ABHIYANTRIKI
An International Journal of Engineering & Technology
(A Peer Reviewed & Indexed Journal)
Vol. 3, No. 5 (May, 2016) http://www.aijet.in/ eISSN: 2394-627X

Automated Interpretation for Smartphone Applications

Jagtap Prajakta P*
Department of Computer Engineering
Govt. College of Engineering & Research
Avasari, Pune, India

Yarole Jayashri Y
Department of Computer Engineering
Govt. College of Engineering & Research
Avasari, Pune, India

Chimate Yamuna S
Department of Computer Engineering
Govt. College of Engineering & Research
Avasari, Pune, India

Bhor Priyanka V
Department of Computer Engineering
Govt. College of Engineering & Research
Avasari, Pune, India

Abstract

Smartphone applications energy efficiency is vital, but many Android applications are facing serious energy inefficiency problems. Locating these problems is labor-intensive and automated diagnosis is highly essential. However, a key challenge is the lack of a decidable criterion that facilitates automated judgment of such energy problems. Our work aims to address this challenge. Two common causes of energy problems: missing deactivation of sensors or wake locks, and cost-ineffective use of sensory data. With these findings, we propose an automated approach to diagnosing energy problems in Android applications. Our approach explores an application's state space by systematically executing the application using Java Pathfinder (JPF). It monitors sensor and wake lock operations to detect missing deactivation of sensors and wake locks. It also tracks the transformation and usage of sensory data and judges whether they are effectively utilized by the application using our state-sensitive data utilization metric. In this way, our approach can generate detailed reports with actionable information to assist developers in validating detected energy problems. We built our approach as a tool, Greendroid, on top of JPF. Technically, we addressed the challenges of generating user interaction events and scheduling event handlers in extending JPF for analyzing Android applications. It successfully located real energy problems in these applications, and additionally found new unreported energy problems that were later confirmed by developers.

Keywords*: Smartphone application, Energy inefficiency, Automated diagnosis, Sensory data utilization, Green computing.*

****Author for correspondence*** jagtapprajaktap@gmail.com

1. Introduction

Locating energy problems in Android applications is difficult. After studying 66 real bug reports concerning energy problems, we found that many of these problems are intermittent and only manifest themselves at certain application states. Reproducing these energy problems is labour

intensive. Developers have to extensively test their applications on different devices and perform detailed energy profiling. To figure out the root causes of energy problems, they have to instrument their programs with additional code to log execution traces for diagnosis. Such a process is typically time consuming. This may explain why some notorious energy problems have failed to be fixed in a timely fashion. In this work, we set out to mitigate this difficulty by automating the energy problem diagnosis process. A key research challenge for automation is the lack of a decidable criterion, which allows mechanical judgment of energy inefficiency problems.

2. Literature Survey

The smartphone application market is growing rapidly. Up until July 2013, the one million Android applications on Google Play store had received more than 50 billion downloads. Many of these applications leverage smartphones' rich features to provide desirable user experiences. For example, Google Maps can navigate users when they hike in the countryside by location sensing. However, sensing operations are usually energy consumptive, and limited battery capacity always restricts such an application's usage. As such, energy efficiency becomes a critical concern for smartphone users. In this work, we set out to mitigate this difficulty by automating the energy problem diagnosis process. A key research challenge for automation is the lack of a decidable criterion, which allows mechanical judgment of energy inefficiency problems. By examining their bug reports, commit logs, bug-fixing patches, patch reviews and release logs, we made an interesting observation. Although the root causes of energy problems can vary with different applications, many of them (over 60 percent) are closely related to two types of problematic coding phenomena, missing sensor or wake lock deactivation. To use a smartphone sensor, an application needs to register a listener with the Android OS. The listener should be unregistered when the concerned sensor is no longer being used. Similarly, to make a phone stay awake for computation, an application has to acquire a wake lock from the Android OS. The acquired wake lock should also be released as soon as the computation completes. Forgetting to unregister sensor listeners or release wake locks could quickly deplete awfully charged phone battery [5]. Smartphone sensors probe their environments and collect sensory data. These data are obtained at high energy cost and therefore should be utilized effectively by applications. Poor sensory data utilization can also result in energy waste.

3. Mathematical Modeling

- Input: Identify the Inputs as, I = L, Wd, Md, BLd, Ap

Where, L = List of Sensors, Wd = Wi-Fi-data, network data, Md = mobile, BLd = Bluetooth data, Ap = Aplications running

- Output: Identify the outputs as, O = l, ws, ms, bs, Ss

Where l = list of sensors closed by greendroid, ws = Wi-Fi's current state, ms = mobile network's current state, bs = bluetooth's current state, Ss = Sensor's current state, Ap=Running application state

- Identify the Constraints as, =
  1. More the no. of sensors on device more output is observable.
  2. If applications like whatsapp are installed then it's not valid to close Internet connection.

1st Module: Sensors Module

U= L,Sd, Ss

Where L=List of Sensors, Sd=Sensory data, Ss=Sensors current state.

2nd Module: Wi-Fi or Mobile Network

H= Wd,Md,ws,ms

Where Wd=Wi-Fi data, Md=mobile network data, ws=Wi-Fi's current state, ms=mobile networks current state.

3rd Module: Bluetooth Module

BL= BLd,bs

Where BLd=Bluetooth data, bs=bluetooth's current state

4th Module: Application Module

Ap=Aid.Cs

Where Aid=Application ID, Cs=Applications current state.

- Functions : Identify the functions as F

F = Detect Sensors (),
Detect Wi-Fi or Mobile Network (), Detect Bluetooth (), Detect screens current state (), Check Apps State () Detect Sensors (h) =P ::takes the list of sensors.

P = h — h takes the list of sensors.

Detect Wi-Fi or Mobile Network (d) =A :: takes the Wi-Fi and mobile networks current state.

A = d — d takes the Wi-Fi and mobile networks current state.

Detect Bluetooth (c) =B :: takes bluetooth's current state. B = c — c takes bluetooth's current state Detect screens current state(s) =D :: takes the screens current state.

D = s — s takes the screens current state

Check Apps State (I,T) = G::takes applications current state and ID

G=I—I takes applications id, T—T takes applications current state.

4. System Architecture

Basically, in this paper we introduce our system architecture. So we can clearly figure out basic idea about this problem definition. These system architectures help us to enhance the technologies currently available for processing image usually in medical field. The basic idea of this project is to develop an application / software to detect the energy problems in android applications. This project aims to develop accurate determination of tumor in the brain. The project is aimed to be cheap and as useful as possible.

5. Conclusion

We proposed an approach for automated energy problem interpretation in Android applications. Our approach systematically explores an applications state space, automatically analyzes its sensory data utilization, and monitors the usage of sensors and wake locks. It helps developers

locate energy problems in their applications and generates actionable reports, which can greatly ease the task of reproducing energy problems as well as fixing them for energy optimization. We implemented our approach into a tool Green Droid on top of JPF. In future, we plan to study more Android applications and identify other common causes of energy problems. For example, we found from our study that a non-negligible proportion (about 16 percent) of energy problems was caused by network issues (e.g., energy-inefficient data transmission).We are going to study these issues to further extend our approach. By doing so, we expect that our research will help advance energy efficiency practices for wider range of Smartphone applications, and thus potentially benefit millions of Smartphone users.

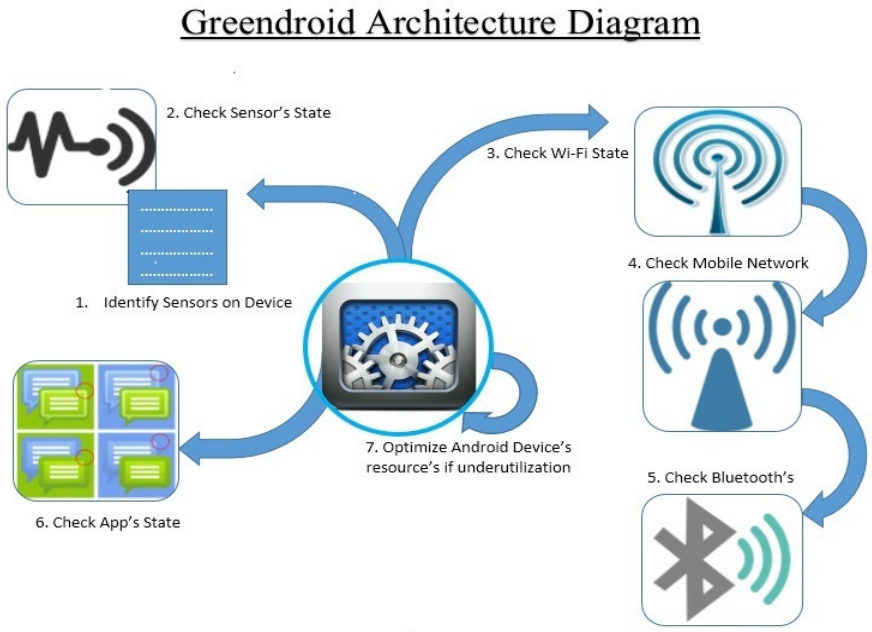

Fig.: Architecture diagram of greendroid

Acknowledgement

It gives us great pleasure in presenting the preliminary project report on 'Automated interpretation for smartphone applications'. We would like to take this opportunity to thank our internal guide Mrs. Nemade S. B. and Prof. Danny J. Pereira, Head of Computer Engineering Department, Govt. College of Engineering and Research, Avasari (Kd) for giving us all the help and guidance we needed. We are really grateful to them for their kind support and valuable suggestions. We would also like to thank our Faculty Members & Lab Assistants for extending their warm support towards our project.

References

[1] RS Pressman. *Software engineering (3rd Ed.): A Practitioner's Approach*. New York: McGraw-Hill, 1992.
[2] P Kulkarni. *Knowledge Innovation Strategy*. Pune: Bloomsbury Publication, 2015.
[3] S Anand, M Naik, MJ Harrold, & H Yang. Automated concolic testing of smartphone apps. Proc. 20th Intl Symp. on Foundations of Soft. Engr. (FSE 12), ACM, 2012, pp. 59:1-59:11.
[4] Android Activity Lifecycles URL. Available at http: //developer.android.com/guide/components/ activities.html
[5] Google Code. Available at http://code.google.com/