A Context-driven Energy Assessment for Energy-aware Development of Mobile Sensing Applications

Seungchul Lee
School of Computing, KAIST
seungchul@nclab.kaist.ac.kr

ABSTRACT

Energy efficiency is a key factor in developing mobile sensing applications. Despite its importance, such laborious and time-consuming energy evaluation process makes developers reluctant to perform the measurement during the development. To this end, we propose a novel context-driven energy assessment system, which utilizes the unique characteristic of MSA's energy consumption pattern. We implemented the energy emulation environment, which supports accurate energy estimation in the development environment and repetitive evaluations for the same real-life situations. The evaluation shows that our environment achieves a reasonable energy estimation accuracy of 95.9%.

1. INTRODUCTION

Dorothy is a mobile application developer. She wants to develop an energy efficient mobility tracker application leveraging accelerometers and GPS. She first designs a mobility recognition logic and implements it. She then goes outside and walks, runs, rides a bicycle, and takes a bus with observing the battery level drop. It turns out that the drop is larger than her expectation; thereby, she looks into the code to find rooms for energy optimization. She then repetitively modifies, walks, observes, determines, and so on.

Energy efficiency is one of the primary objectives in the development of mobile sensing applications (MSAs). Despite its importance, developers have difficulties in understanding the energy consumption of MSAs, due to the repetitive and laborious evaluation process. On top of burden in a single iteration – implementation, observation, and determination, developers need repetitive iterations on energy evaluation in various execution scenarios. It is because MSAs by nature have diverse energy consumption patterns depending on the user's behavior and situation, i.e., user context [4]. For example, Dorothy's mobility tracker could consume more energy when she takes a bus than walks, since it tends to request a GPS location more frequently when her moving speed gets faster. To understand the precise energy consumption behavior, it is inevitable to evaluate it under numerous real-world user contexts.

Our preliminary study with seven professional MSA developers shows that even in industry, developers suffer from the difficulty of the energy evaluation. The reason is threefold. First, a repetitive measurement is extremely laborious and burdensome. A startup company developer said, *"Normally we install our MSA on smartphones, monitor the battery levels for a day, find issues, and repeat it for a few more days."* Second, developers need to test their MSAs under diverse situations in which they potentially operate. A developer of a beacon-based advertisement application mentioned, *"To evaluate our MSA for usage situations, we manually stay near the beacon for 10 seconds and walk away, stay for 20 seconds and walk away, and so on."* He added, *"We newly found that the MSA had a bug when a user quickly moved from a subway station to the street. But we didn't debug it because we had no idea how to handle it."* Third, lack of supporting tools leads to hesitation from considering energy efficiency during the development.

In this regard, we propose a novel context-driven energy assessment system for MSAs. Its key idea is to execute the MSAs under diverse user contexts and analyze the energy consumption in a context-driven manner. For example, given Dorothy's MSA, the system predicts its energy consumption under various mobility conditions including stationary and driving. It then analyzes the overall energy consumption of the MSA considering mobility patterns of target users such as businessperson who usually commute by bus. Developers could use the system to achieve energy optimization for a potential user pool. It is also possible to compare the energy efficiency among similar MSAs by analyzing per-context energy consumption. This information can further assist users in choosing which MSA to use, based on the user's real-life patterns.

2. CHALLENGES AND ROADMAP

To realize the proposed system we need to overcome the following four challenges. 1) The system should be able to estimate the accurate energy consumption of MSAs in the development environment. 2) It should support the repetitive energy assessment in the same real-life situation. 3) The system should complete the overall energy estimation process in a short time, considering the huge number of execution environments. 4) The system is required to support various kinds of MSAs.

To handle the challenges, we set up the three-step research process. In the first step, we have constructed the energy emulation environment to solve the first two challenges. The environment takes an MSA and target user contexts as inputs, then provides emulation outputs, i.e., energy analysis results, to developers. Since MSAs are running on pre-collected user contexts, developers need not be in the actual execution environment. It also enables running different versions of MSAs in the identical execution environment. The next step is an acceleration of the emulation, to alleviate the huge cost of time and resource usages for the emulation over numerous user contexts. Specifically, we plan to achieve it by reducing

MobiSys'17 Ph.D. Forum, June 19, 2017, Niagara Falls, NY, USA

ACM ISBN 978-1-4503-4957-4/17/06.

DOI: http://dx.doi.org/10.1145/3086467.3086473

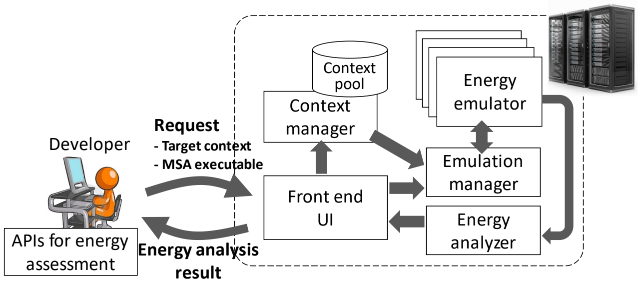

Figure 1: Overview of the emulation environment

the duration of per-emulation processes and the number of emulation processes. In the third step, we will define and implement the specification, collection, and management schemes of input user contexts that is generally applicable to various types of MSAs.

3. CONSTRUCTION OF EMULATION ENVIRONMENT

In our prior research, we implemented the emulation environment for the context-driven energy assessment [3]. Figure 1 describes the overall structure. It has four main components: the front end UI, the emulation manager, energy emulators, and the energy analyzer. The operation flow is as follows. First, a developer configures user contexts in which the emulator executes MSAs, along with the target MSA and its settings. Upon receiving the input, the emulation manager identifies the required number of energy emulators and assigns the inputs to each emulator. The energy emulator executes the application under the given user context and sends the result to the energy analyzer. The analyzer collects the outputs from each emulator and extracts meaningful energy-related information. It then provides the developer with the energy evaluation result.

We evaluated the energy estimation accuracy of our environment using five MSAs. We chose three commercial MSAs (Accupedo, Pedometer2.0, SleepBot), one open-source MSA (Android-pedometer), and one research MSA, iTrack. iTrack is an energy efficient location tracker application that records a user's location only when the user is moving. We executed each MSA under a one-hour long real-world usage scenario. We used Nexus 5 phones to collect sensor streams and energy consumption data. We took the measured energy consumption of the Monsoon power monitor as the ground truth. As a result, our environment achieves an average accuracy of 95.9% (Figure 2). We plan to extend the evaluation to include more smartphone models. It is well-known that heterogeneity of mobile devices would degrade the energy estimation accuracy and such problem happens even within the same device model [7].

4. RELATED WORK

A number of research works have been proposed to predict the energy consumption of mobile applications. It is mainly divided into two, direct estimation of energy consumption in mW scale [2, 6] and indirect estimation by providing hardware usage information [1]. However, they are based on direct execution, making it difficult for developers to evaluate the energy consumption in a repetitive manner and with various execution environments. Recently, there have been a few works, which predict applications' energy consumption based on emulation such as WattsOn [5]. However, they are not proper in predicting the energy consumption of MSAs. They do not consider sensor-related energy information that is critical in case of MSAs. Unlike such tools, our system can predict energy consumption of MSAs accurately by reflecting their context-dependent energy characteristics.

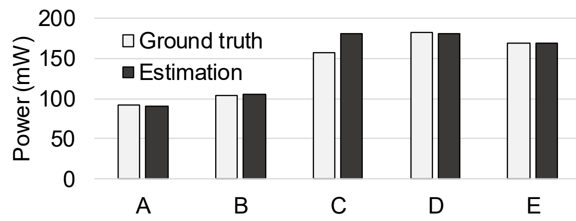

Figure 2: Energy estimation; A: Accupedo, B: Pedometer2.0, C: SleepBot, D: Android-pedometer, E: iTrack

5. CONCLUSION

We proposed a novel context-driven energy assessment system. We first identified the limitation and challenges of the current energy estimation process. As the first step to overcome the challenges, we implemented the energy emulation environment. We plan to extend our environment to equip with the emulation acceleration and supports for diverse types of MSAs.

6. ACKNOWLEDGEMENTS

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP: Ministry of Science, ICT & Future Planning) (No. 2017R1C1B101-0619).

7. REFERENCES

[1] Android batterystats and battery historian. https://developer.android.com/studio/profile/battery-historian.html. Accessed: Apr 12, 2017.

[2] D. Li, S. Hao, W. G. Halfond, and R. Govindan. Calculating source line level energy information for android applications. In *Proceedings of the 2013 International Symposium on Software Testing and Analysis*, pages 78–89. ACM, 2013.

[3] C. Min, S. Lee, C. Lee, Y. Lee, S. Kang, S. Choi, W. Kim, and J. Song. Pada: power-aware development assistant for mobile sensing applications. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 946–957. ACM, 2016.

[4] C. Min, C. Yoo, I. Hwang, S. Kang, Y. Lee, S. Lee, P. Park, C. Lee, S. Choi, and J. Song. Sandra helps you learn: the more you walk, the more battery your phone drains. In *Proceedings of the 2015 ACM international joint conference on Pervasive and ubiquitous computing*, pages 421–432. ACM, 2015.

[5] R. Mittal, A. Kansal, and R. Chandra. Empowering developers to estimate app energy consumption. In *Proceedings of the 18th annual international conference on Mobile computing and networking*, pages 317–328. ACM, 2012.

[6] A. Pathak, Y. C. Hu, M. Zhang, P. Bahl, and Y.-M. Wang. Fine-grained power modeling for smartphones using system call tracing. In *Proceedings of the sixth conference on Computer systems*, pages 153–168. ACM, 2011.

[7] G. P. Srinivasa, R. Begum, S. Haseley, M. Hempstead, and G. Challen. Separated by birth: Hidden differences between seemingly-identical smartphone cpus. In *Proceedings of the 18th International Workshop on Mobile Computing Systems and Applications*, pages 103–108. ACM, 2017.