Collecting battery data with Open Battery

Gareth L. Jones[1] and Peter G. Harrison[2]

1,2 Imperial College London, 180 Queen's Gate, London, SW7 2RH
`{gljones,pgh}@doc.ic.ac.uk`

Abstract

In this paper we present Open Battery, a tool for collecting data on mobile phone battery usage, describe the data we have collected so far and make some observations. We then introduce the *fluid queue* model which we hope may prove a useful tool in future work to describe mobile phone battery traces.

1998 ACM Subject Classification D.4.8 Queueing theory

Keywords and phrases battery model, battery data

Digital Object Identifier 10.4230/OASIcs.ICCSW.2012.75

1 Introduction and motivation

A recent Forrester study suggests that by 2016 a billion smartphones will be in use around the world [2]. Understanding battery behaviour and how devices are used (sometimes called *human battery interaction* [7]) is important to deliver improved performance in these devices.

Previous studies (e.g. Ferreira et al. [3]) have collected data under privacy agreements which do not allow the data to be shared outside the named researchers on the original proposal. This makes further work with the data hard. Data collected in our study is published under the PDDL on our website `http://www.openbattery.com/`, is in the public domain, and can be downloaded and redistributed freely.

In this paper we will make some observations about the data we have collected so far and then introduce the fluid queue modelling paradigm.

The authors of this paper previously published a result on how battery life of a device subject to random charging and discharging periods was affected by a power saving mode, implemented when power reserves fell below some threshold value [5]. In future work we intend to investigate fitting this model to our data.

2 Data collection

We have written an application for Android which logs battery usage data. The application listens for `ACTION_BATTERY_CHANGED` broadcasts and logs the battery state with timestamp each time the battery state changes. Data is saved locally and sent to our web server when the device is charging. A sample of the data collected is shown in Figure 1.

So far in this preliminary work, we have collected data for around 20 handsets for 3 months.

3 Observations on collected data

1. There is great variability in the number of data points logged. We observed a *Samsung GT-I9000* handset logging more than 1,000 data points a day (reporting regular small changes in voltage), while a *HTC Wildfire S A510e* logged nearer 100 data points a day.

2012 Imperial College Computing Student Workshop (ICCSW'12).
Editor: Andrew V. Jones; pp. 75–80
OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

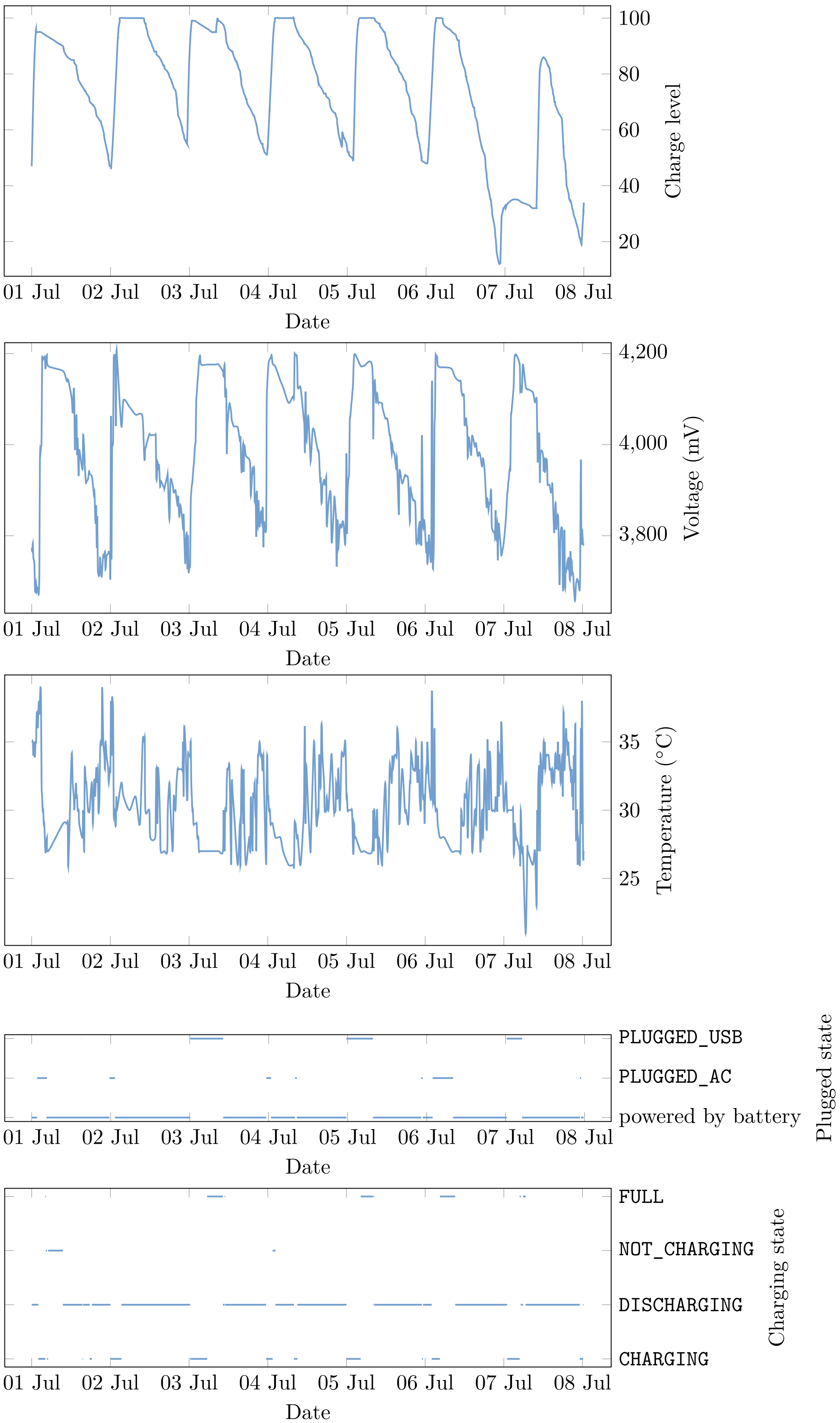

Figure 1 Sample data trace from July 1-7 2012 for device id `3fd6231afc7fec60`, a Galaxy Nexus. Throughout this trace the reported health was `GOOD`.

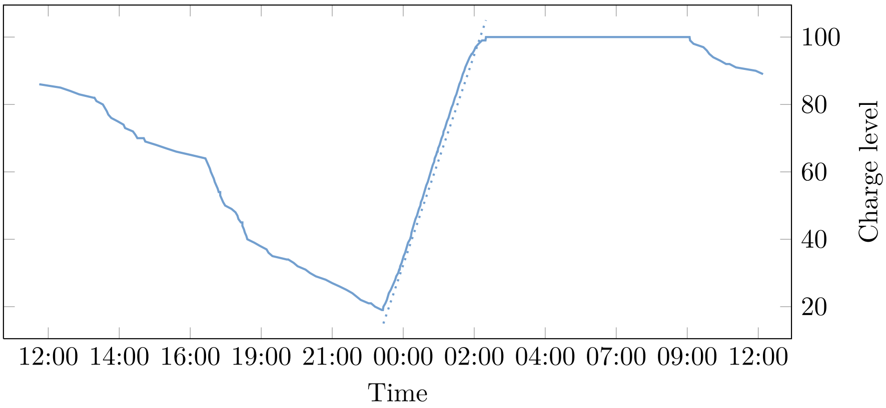

Figure 2 Linear charging/discharging period for Galaxy Nexus (device id `3fd6231afc7fec60`, 8–9 July 2012).

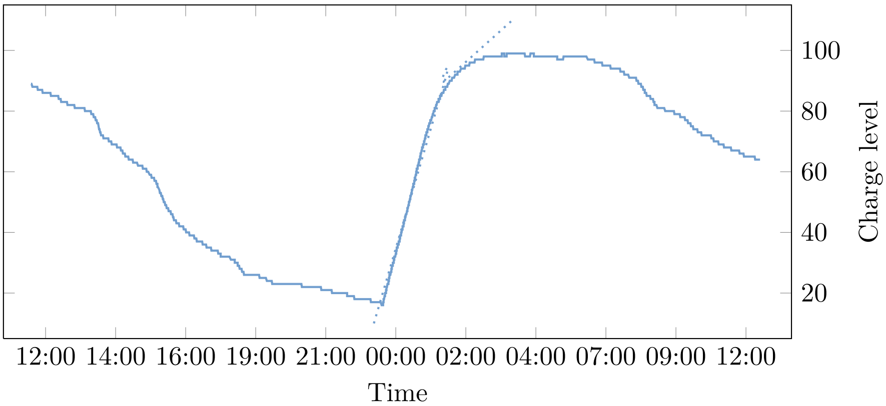

Figure 3 Non-linear charging for Nexus S (device id `3fe0f99cef3a49a8`, 11-12 July 2012) with two piecewise linear functions fitted.

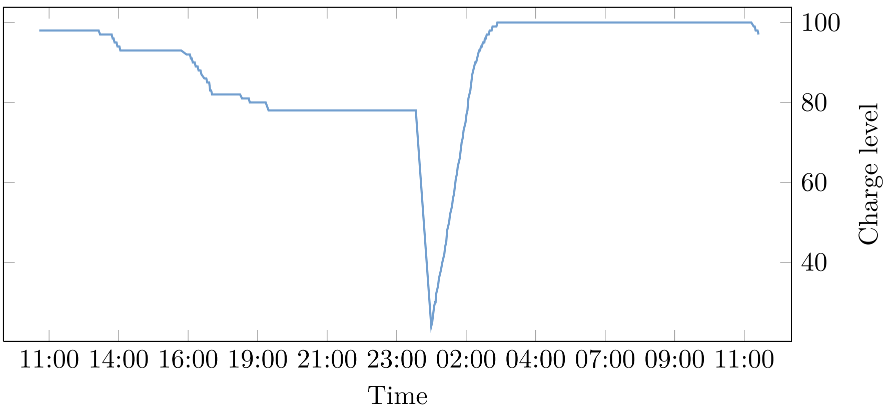

Figure 4 Erroneously reported discharging for *asus Transformer Prime TF201* (device id `3fcb2812d3d9e9b8`, 9-10 July 2012). From 19:00 to 23:20 the battery level was reported in the operating system as at a constant value.

`HEALTH` $\in$ {`UNKNOWN`, `GOOD`, `OVERHEAT`, `DEAD`, `OVER_VOLTAGE`, `UNSPECIFIED_FAILURE`, `COLD`}.
`PLUGGED` $\in$ {powered by battery, `PLUGGED_AC`, `PLUGGED_USB`}
`STATUS` $\in$ {`UNKNOWN`, `CHARGING`, `DISCHARGING`, `NOT_CHARGING`, `FULL`}

Figure 5 `BatteryManager` health, plugged and status values.

State	Average change	Average filling/emptying time
AC charging	+30 pp/hour	3 hours 15 minutes to fully charge
USB charging	+20 pp/hour	5 hours to fully charge
discharging	−3 pp/hour	33 hours until fully drained

Table 1 Average values computed for Galaxy Nexus

2. Generally reported charging rates are reasonably linear throughout the charging period such as for the Galaxy Nexus shown in Figure 2. However, data recorded for five Nexus S handsets shows linear charging up to 85%, with a non-linear portion up to 100%. As shown in Figure 3 we could reasonably approximate this with a second linear rate. Unsurprisingly we have less data for the lower end of battery charging (an earlier version of the logging application required manual restart), but are now aware that charging rates need to be level dependent.
3. There are also logging problems with data we need to consider. An *ASUS Transformer Prime TF201* (device id `3fcb2812d3d9e9b8`) has misreported the battery level as remaining constant for a few hours before dropping 50 percentage points or more in a matter of seconds as shown in Figure 4. This is not a bug with our tool, but with the levels that the battery hardware is reporting to the operating system.
4. Different handsets report their charging state differently. The values documented in the `BatteryManager` class are shown in Figure 5, but not all handsets report in the same way. For example, the *HTC Wildfire* (device id `3fe37029cced541a`) never reports itself `DISCHARGING`, only `CHARGING`, `NOT_CHARGING` or `FULL`.

4 Fluid queues

Fluid queues are a particular type of stochastic process which we hope will prove to be a good model for the charging and discharging behaviour seen in our data.

A fluid queue is a bivariate stochastic process (J_t, X_t) where J_t describes the background state and X_t the charge level. J_t is a Markov chain on the state space

{`AC charging`, `USB charging`, `discharging`}.

With each of these states we associate a rate of change which determines the rate at which X_t changes. In Table 1 we show parameters estimated from a 4 month trace from a Galaxy Nexus (device id `3fd6231afc7fec60`). Average charging rates for this device are broadly similar over all time periods, but average discharging rates varied significantly from 1 to 11 percentage points per hour (pp/hour).

A single exponential distribution holding time in each state is unlikely to describe the traces well, but extra states can be added within the fluid queue model to give a phase-type distribution fit to our data.

The process X_t is continuous and piecewise linear with the rate determined by the process J_t. The process is bounded above and below by the capacity of the battery ($0 \leq X_t \leq B$ for all t).

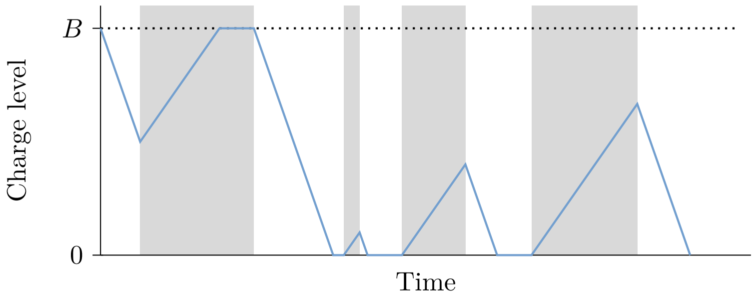

Figure 6 Sample trace from a fluid queue. The grey highlights represent time in charging periods and the white background periods when the device was discharging.

A sample trace from a fluid queue model with just two states {`charging`, `discharging`} is shown in Figure 6.

The *busy period* of the fluid queue is the time period between instants when the buffer is empty. The busy period is a stochastic quantity because it is determined by the sequence of charging and discharging period durations.

The fluid queue model has seen significant attention in the literature and the stationary distribution and busy period are known for infinite buffer models [4]. We will require an extension of the model introduced here where charging/discharging rates are dependent on the charge level and the buffer is of finite capacity.

Authors of this paper published the busy period for a model with level-dependent rates in a recent paper [5]. The Laplace–Stieltjes transform of the busy period distribution was computed, from which moments can be computed analyically by differentiation and numerical inversion can quickly compute particular values (e.g. 95th percentile). Extending this result to a finite buffer with numerous emptying states remains as future work.

5 Motivation and future goals

Smart phone user feedback on battery life is currently very crude. Android handsets typically warn the user of low battery at 15% and 5%, irrespective of how long the battery life is likely to last (on some handsets this might be 8 or more hours).

Some apps already exist to give users a clearer idea of how long their battery will last (like *Battery Monitor Widget* [1]), though they do not offer the user time-based alerts. A significanlty more elaborate system 'CABMAN' has been suggested [8] where the device would make decisions about power usage based on user position and proximity to the next charging session.

Our theoretical model may be can be improved by considering the 'phantom recharging' effects descibed by the KiBaM model [5, 6]. We see voltage recoveries during discharge periods in our collected data and intend to investigate the effect.

In the longer term we seek to investigate how the same device performs over time and between different users and quantitatively qualify the degradation in battery performance over time. This information will be of interest to both users and manufacturers of Android devices.

References

1 Battery monitor widget by 3c `http://www.3c71.com/android/?q=node/1`.

2 Forrester research mobile adoption forecast, 2012 to 2017 (US). 2012.
3 Denzil Ferreira, Anind K. Dey, and Vassilis Kostakos. Understanding human-smartphone concerns: a study of battery life. In *Pervasive'11 Proceedings of the 9th international conference on Pervasive computing*, pages 19–33, 2011.
4 Tony Field and Peter G. Harrison. Busy periods in fluid queues with multiple emptying input states. *Journal of Applied Probability*, 47:474–497, 2010.
5 Gareth L. Jones, Peter G. Harrison, Uli Harder, and A. J. Field. Fluid Queue Models of Battery Life. In *IEEE 19th International Symposium on Modelling, Analysis Simulation of Computer and Telecommunication Systems*, July 2011.
6 J. F. Manwell and J. G. McGowan. Lead acid battery storage model for hybrid energy systems. *Solar Energy*, 50(5):399–405, 1993.
7 Ahmad Rahmati, Angela Qian, and Lin Zhong. Understanding human-battery interaction on mobile phones. In *Proceedings of the 9th international conference on Human computer interaction with mobile devices and services*, pages 265–272.
8 Nishkam Ravi, James Scott, Lu Han, and Liviu Iftode. Context-aware battery management for mobile phones. In *Sixth Annual IEEE International Conference on Pervasive Computing and Communications, 2008. PerCom 2008*, pages 224–233, 2008.