

**Michalis Faloutsos
Aleksandar Kuzmanovic (Eds.)**

LNC8 8362

Passive and Active Measurement

**15th International Conference, PAM 2014
Los Angeles, CA, USA, March 10-11, 2014
Proceedings**

 **Springer**

Commenced Publication in 1973

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

Lancaster University, UK

Takeo Kanade

Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler

University of Surrey, Guildford, UK

Jon M. Kleinberg

Cornell University, Ithaca, NY, USA

Alfred Kobsa

University of California, Irvine, CA, USA

Friedemann Mattern

ETH Zurich, Switzerland

John C. Mitchell

Stanford University, CA, USA

Moni Naor

Weizmann Institute of Science, Rehovot, Israel

Oscar Nierstrasz

University of Bern, Switzerland

C. Pandu Rangan

Indian Institute of Technology, Madras, India

Bernhard Steffen

TU Dortmund University, Germany

Madhu Sudan

Microsoft Research, Cambridge, MA, USA

Demetri Terzopoulos

University of California, Los Angeles, CA, USA

Doug Tygar

University of California, Berkeley, CA, USA

Gerhard Weikum

Max Planck Institute for Informatics, Saarbruecken, Germany

Michalis Faloutsos
Aleksandar Kuzmanovic (Eds.)

Passive and Active Measurement

15th International Conference, PAM 2014
Los Angeles, CA, USA, March 10-11, 2014
Proceedings

Volume Editors

Michalis Faloutsos
University of New Mexico
Computer Science Department
Engineering Building II, Albuquerque, NM 87131, USA
E-mail: michalis@cs.unm.edu

Aleksandar Kuzmanovic
Northwestern University
EECS Department
2145 Sheridan Road, Evanston, IL 60208, USA
E-mail: akuzma@northwestern.edu

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-319-04917-5 e-ISBN 978-3-319-04918-2
DOI 10.1007/978-3-319-04918-2
Springer Cham Heidelberg New York Dordrecht London

Library of Congress Control Number: 2014930886

LNCS Sublibrary: SL 5 – Computer Communication Networks
and Telecommunications

© Springer International Publishing Switzerland 2014

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed. Exempted from this legal reservation are brief excerpts in connection with reviews or scholarly analysis or material supplied specifically for the purpose of being entered and executed on a computer system, for exclusive use by the purchaser of the work. Duplication of this publication or parts thereof is permitted only under the provisions of the Copyright Law of the Publisher's location, in its current version, and permission for use must always be obtained from Springer. Permissions for use may be obtained through RightsLink at the Copyright Clearance Center. Violations are liable to prosecution under the respective Copyright Law.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Welcome to the proceedings of the 2014 Passive and Active Measurement (PAM) Conference. The event, which was held in Los Angeles this year, focused on research in and the practice of Internet measurements. This was the 15th PAM. Following its genesis in 2000, the conference has maintained a strong workshop feel, providing an opportunity for the presentation of innovative and early work, with lively discussion and active participation from attendees.

In 2012 the conference broadened its scope, reflecting the widening uses of network measurement and analysis methods. The aim was to facilitate the understanding of the expanding role that measurement techniques play as they become building blocks for a variety of networking environments, application profiling, and cross-layer analysis. In 2014 we continued with this wider scope, although we did not neglect PAM's core topics.

PAM 2014 attracted 76 submissions. The papers came from academia and industry from around the world. It was especially pleasing to see the global nature of submissions.

The Technical Program Committee was chosen from a group of experts in Internet measurement, drawing on past contributors to PAM including distinguished academic and industrial researchers, but also with a group of first-time members. Additionally, we aimed to have a strong global representation on the committee, and achieved this with members from around the world.

The final program of 24 papers was selected after each submission was carefully reviewed by at least three members of the Program Committee (PC), at least one of whom rated themselves as knowledgeable with regard to the content of the paper. We were delighted with the quality of reviews – they were careful, insightful, and paid attention to detail. The reviews were followed by an extensive discussion phase. PAM has traditionally avoided a large PC meeting and the difficulties it creates for a global PC and instead uses on-line discussions. This year, these were impressively robust: Reviewers provided more than 350 comments on papers, some almost as detailed as the reviews themselves. Most of the final papers were then shepherded by PC members.

This year's conference also continued the selection criteria related to reproducible research, which was established in 2013. It is our belief that one of the most pressing issues in the field of Internet measurement research is the fact that many papers report on data sets that are never disclosed. Hence, PAM strongly encourages the authors to publish their data sets.

In addition, the PC selected seven papers to appear as posters at the conference, and these are included in this volume as extended abstracts. The final program included papers on a wide range of measurement topics, and included authors from 13 countries and five continents. Our most sincere thanks go to the PC members for their diligence and care in reviewing, discussing, and

shepherding the papers that appear here, and to Marcel Flores for organizing and maintaining the HotCRP site for us.

We are also most grateful to the Steering Committee, Jelena Mirkovic, who was the local chair, and Jedidiah Crandall who served as the publicity chair. We hope that you enjoy the papers in these proceedings.

March 2014

Aleksandar Kuzmanovic
Michalis Faloutsos

Organization

Organizing Committee

Conference Chair

Michalis Faloutsos The University of New Mexico, USA

Program Chair

Aleksandar Kuzmanovic Northwestern University, USA

Local Chair

Jelena Mirkovic USC Information Sciences Institute

Publicity Chair

Jedidiah Crandall The University of New Mexico, USA

Steering Committee

Fabio Ricciato University of Salento, Italy
George Riley Georgia Institute of Technology, USA
Ian Graham Endace, New Zealand
Neil Spring University of Maryland, USA
Nevil Brownlee The University of Auckland, New Zealand
Nina Taft Technicolor Palo Alto Research Center, USA
Matthew Roughan University of Adelaide, Australia
Rocky K. C. Chang The Hong Kong Polytechnic University

Program Committee

Alan Mislove Northeastern University, USA
Alberto Dainotti CAIDA, USA
Arun Venkataramani UMass, USA
Bernhard Ager ETH Zurich, Switzerland
Bin Liu Tsinghua University, China
Bruce Maggs Akamai and Duke University, USA
Constantine Dovrolis Georgia Tech, USA
David Choffnes Northeastern University, USA
Dmitri Logiunov Texas A&M, USA
Fernando Silveira Technicolor, USA
Gabor Vattay Eotvos Larand University, Hungary
Han Song Narus Inc, USA

VIII Organization

Jelena Mirkovic	ISI, USA
Marios Iliofotou	Narus Inc., USA
Mark Allman	ICSI, USA
Matthew Luckie	CAIDA, USA
Michael Rabinovich	Case Western Reserve University, USA
Minaxi Gupta	Indiana University, USA
Myungjin Lee	University of Edinburgh, UK
Neil Spring	University of Maryland, USA
Paul Barford	Wisconsin Madison, USA
Rade Stanojevic	Telefonica, Spain
Sergey Gorinsky	IMDEA, Spain
Thomas Karagiannis	Microsoft, UK
Xenofontas Dimitropoulos	ETH Zurich, Switzerland
Youngseok Lee	CNU, Korea
Zhichun Li	NEC Labs, USA
Zhi-Li Zhang	University of Minnesota, USA

Sponsoring Institutions

University of New Mexico, USA

Table of Contents

Internet Wireless and Mobility

RadioProphet: Intelligent Radio Resource Deallocation for Cellular Networks	1
<i>Junxian Huang, Feng Qian, Z. Morley Mao, Subhabrata Sen, and Oliver Spatscheck</i>	
Mobile Network Performance from User Devices: A Longitudinal, Multidimensional Analysis	12
<i>Ashkan Nikraves, David R. Choffnes, Ethan Katz-Bassett, Z. Morley Mao, and Matt Welsh</i>	
Diagnosing Path Inflation of Mobile Client Traffic	23
<i>Kyriakos Zarifis, Tobias Flach, Srikanth Nori, David Choffnes, Ramesh Govindan, Ethan Katz-Bassett, Z. Morley Mao, and Matt Welsh</i>	
An End-to-End Measurement Study of Modern Cellular Data Networks	34
<i>Yin Xu, Zixiao Wang, Wai Kay Leong, and Ben Leong</i>	

Measurement Design, Experience and Analysis

A Second Look at Detecting Third-Party Addresses in Traceroute Traces with the IP Timestamp Option	46
<i>Matthew Luckie and kc claffy</i>	
Ingress Point Spreading: A New Primitive for Adaptive Active Network Mapping	56
<i>Guillermo Baltra, Robert Beverly, and Geoffrey G. Xie</i>	
On Searching for Patterns in Traceroute Responses	67
<i>Nevil Brownlee</i>	
Volume-Based Transit Pricing: Is 95 the Right Percentile?	77
<i>Vamseedhar Reddyvari Raja, Amogh Dhamdhere, Alessandra Scicchitano, Srinivas Shakkottai, kc claffy, and Simon Leinen</i>	

Performance Measurement

Dissecting Round Trip Time on the Slow Path with a Single Packet	88
<i>Pietro Marchetta, Alessio Botta, Ethan Katz-Bassett, and Antonio Pescapé</i>	
Is Our Ground-Truth for Traffic Classification Reliable?	98
<i>Valentín Carela-Español, Tomasz Bujlow, and Pere Barlet-Ros</i>	
Detecting Intentional Packet Drops on the Internet via TCP/IP Side Channels	109
<i>Roya Ensafi, Jeffrey Knockel, Geoffrey Alexander, and Jedidiah R. Crandall</i>	
The Need for End-to-End Evaluation of Cloud Availability	119
<i>Zi Hu, Liang Zhu, Calvin Ardi, Ethan Katz-Bassett, Harsha V. Madhyastha, John Heidemann, and Minlan Yu</i>	

Protocol And Application Behavior

Exposing Inconsistent Web Search Results with Bobble	131
<i>Xinyu Xing, Wei Meng, Dan Doozan, Nick Feamster, Wenke Lee, and Alex C. Snoeren</i>	
Modern Application Layer Transmission Patterns from a Transport Perspective	141
<i>Matt Sargent, Ethan Blanton, and Mark Allman</i>	
Third-Party Identity Management Usage on the Web	151
<i>Anna Vapen, Niklas Carlsson, Anirban Mahanti, and Nahid Shahmehri</i>	
Understanding the Reachability of IPv6 Limited Visibility Prefixes	163
<i>Andra Lutu, Marcelo Bagnulo, Cristel Pelsser, and Olaf Maennel</i>	

Characterization of Network Behavior

Violation of Interdomain Routing Assumptions	173
<i>Riad Mazloum, Marc-Olivier Buob, Jordan Augé, Bruno Baynat, Dario Rossi, and Timur Friedman</i>	
Here Be Web Proxies	183
<i>Nicholas Weaver, Christian Kreibich, Martin Dam, and Vern Paxson</i>	
Towards an Automated Investigation of the Impact of BGP Routing Changes on Network Delay Variations	193
<i>Massimo Rimondini, Claudio Squarcella, and Giuseppe Di Battista</i>	

Peering at the Internet's Frontier: A First Look at ISP Interconnectivity in Africa	204
<i>Arpit Gupta, Matt Calder, Nick Feamster, Marshini Chetty, Enrico Calandro, and Ethan Katz-Bassett</i>	

Network Security and Privacy

Assessing DNS Vulnerability to Record Injection	214
<i>Kyle Schomp, Tom Callahan, Michael Rabinovich, and Mark Allman</i>	
How Vulnerable Are Unprotected Machines on the Internet?	224
<i>Yuanyuan Grace Zeng, David Coffey, and John Viega</i>	
A Closer Look at Third-Party OSN Applications: Are They Leaking Your Personal Information?	235
<i>Abdelberi Chaabane, Yuan Ding, Ratan Dey, Mohamed Ali Kaafar, and Keith W. Ross</i>	
On the Effectiveness of Traffic Analysis against Anonymity Networks Using Flow Records	247
<i>Sambuddho Chakravarty, Marco V. Barbera, Georgios Portokalidis, Michalis Polychronakis, and Angelos D. Keromytis</i>	

Poster Abstracts

Scaling Bandwidth Estimation to High Speed Networks	258
<i>Qianwen Yin, Jasleen Kaur, and F. Donelson Smith</i>	
Scalable Accurate Consolidation of Passively Measured Statistical Data	262
<i>Silvia Colabrese, Dario Rossi, and Marco Mellia</i>	
A Needle in the Haystack - Delay Based User Identification in Cellular Networks	265
<i>Marco V. Barbera, Simone Bronzini, Alessandro Mei, and Vasile C. Perta</i>	
Understanding HTTP Traffic and CDN Behavior from the Eyes of a Mobile ISP	268
<i>Pedro Casas, Pierdomenico Fiadino, and Arian Bär</i>	
On Understanding User Interests through Heterogeneous Data Sources	272
<i>Samamon Khemmarat, Sabyasachi Saha, Han Hee Song, Mario Baldi, and Lixin Gao</i>	

Nightlights: Entropy-Based Metrics for Classifying Darkspace Traffic Patterns	275
<i>Tanja Zseby, Nevil Brownlee, Alistair King, and kc claffy</i>	
Distributed Active Measurement of Internet Queuing Delays.....	278
<i>Pellegrino Casoria, Dario Rossi, Jordan Augé, Marc-Olivier Buob, Timur Friedman, and Antonio Pescapé</i>	
Author Index	281

RadioProphet: Intelligent Radio Resource Deallocation for Cellular Networks

Junxian Huang¹, Feng Qian², Z. Morley Mao³,
Subhabrata Sen², and Oliver Spatscheck²

¹ Google Inc.

² AT&T Labs – Research

³ University of Michigan

Abstract. Traditionally, radio resources are released in cellular networks by statically configured inactivity timers, causing substantial resource inefficiencies. We propose a novel system **RadioProphet (RP)**, which dynamically and intelligently determines in real time when to deallocate radio resources by predicting the network idle time based on traffic history. We evaluate **RP** using 7-month-long real-world cellular traces. Properly configured, **RP** correctly predicts 85.9% of idle time instances and achieves radio energy savings of 59.1% at the cost of 91.0% of signaling overhead, outperforming existing proposals. We also implement and evaluate **RP** on real Android devices, demonstrating its negligible runtime overhead.

1 Introduction

Cellular networks employ a specific radio resource management policy distinguishing them from wired and Wi-Fi networks. Previous studies [5][10][8] have shown that in cellular networks, the origin of low resource efficiency comes from the way resources are *released*. To avoid high signaling load, radio resources are only released after an idle time (also known as the “tail time” or T_{tail}) controlled by statically configured inactivity timers. During the tail time, energy is essentially wasted by the radio interface.

Without knowing when network traffic will occur, long tail timer settings (*e.g.*, 11.6 seconds configured by an LTE network [8]) are essentially a conservative way to ensure low signaling overhead, which is known to be a bottleneck for cellular networks. Given that application behaviors are not random, using a statically configured timer is clearly suboptimal. A smaller static timer value helps reduce radio energy, but is not an option due to the risk of overloading cellular networks caused by signaling load increase.

An attractive alternative is to configure the timer dynamically — adaptively performing radio resource release signaled by the handset by monitoring the traffic and accommodating different traffic patterns. But the key challenge is determining when to release resources, which essentially comes down to accurate and efficient *prediction of the idle time period*. Clearly, the best time to do so is when the handset is about to experience a long idle time period, otherwise the incurred resource allocation overhead (*i.e.*, signaling load) might be unacceptably high. Therefore, accurate and efficient prediction of the idle time period is a critical prerequisite for dynamic timer schemes.

This paper proposes **RadioProphet (RP)**, a practical system running on a handset that makes dynamic decisions to deallocate radio resources based on accurate and efficient prediction of network idle times. It makes the following contributions.

First, **RP** utilizes standard online machine learning (ML) algorithms to accurately predict the network idle time, and performs resource deallocation only when the idle time is sufficiently long. We explored various ML algorithms and prediction models with tunable parameters, with the main contribution of using a measurement-driven approach to find robust and easy-to-measure features, whose complex interaction with the network idle time can be automatically discovered by the ML algorithms. The model is validated using seven-month-long traces collected from real users (§5).

Second, we implement **RP** on a real Android smartphone to demonstrate its negligible energy and CPU overhead. In contrast, all previous proposals [10][4][7] only perform trace-driven simulation. To reduce the runtime overhead, **RP** strategically performs *binary* prediction (*i.e.*, whether the idle time is short or long) at the granularity of a traffic *burst* consisting of a packet train sent or received in a batch. Compared to fine-grained prediction of the precise value of packet inter-arrival time, our proposed approach is much more efficient while yielding similar optimization results.

Third, we overcome critical limitations of previously proposed approaches, *i.e.*, RadioJockey [4] and MakeIdle / MakeActive [7] are only applicable to background applications without user interaction, with the ideal usage scenario of RadioJockey for a single application only. With multiple concurrent applications, it suffers from low prediction accuracy with increased overhead. In contrast, **RP** is specifically designed for both foreground and background traffic. Since its prediction is based on the aggregate traffic of all apps, **RP** incurs no additional overhead for supporting concurrent apps.

Fourth, we conduct comprehensive measurement of **RP** using real-world smartphone traces (7 months from 20 users). The overall prediction accuracy is 85.9%. **RP** achieves radio energy saving by 59.1%, at the cost of 91.0% additional signaling overhead in LTE networks, significantly outperforming previous proposals. To achieve the same energy saving, the additional signaling overheads incurred by MakeIdle [7] and naïve fast dormancy [1] are 305% and 215%, respectively. The maximal energy saving achieved by RadioJockey [4] is only 27% since it is only applicable to background traffic.

Paper Organization. We provide sufficient background in §2 before giving an overview of the **RadioProphet (RP)** system in §3. We detail how we select relevant features for idle time prediction in §4, and then systematically evaluate **RP** in §5. In §6, we describe related work before concluding the paper.

2 Background

In cellular networks, there is a radio resource control (RRC) state machine that determines radio resource usage based on application traffic patterns, affecting device energy consumption and user experience. Conceptually similar RRC state machines exist in different types of cellular networks from 2G to 4G LTE. In 3G UMTS networks, there are usually three RRC states [11]: idle, low-power state, and high-power state. In 4G LTE networks, there are only two RRC states: idle and active [8]. Note that **RP** works for any type of RRC state machine with fast dormancy (described soon) support.

State Transitions. There are two types of state transitions. State promotions switch from a low-power state to a high-power state. They are triggered by user data transmission in either direction. State demotions go in the reverse direction, usually triggered

by inactivity timers configured by the radio access network (RAN). For example, for a commercial LTE network [8], at the active state, the RAN resets the timer to a constant threshold $T_{tail}=11.6$ seconds whenever it observes any data frame. If there is no user data transmission for T_{tail} seconds, the timer expires and the state is demoted to idle. Similar timers exist in 3G networks (*e.g.*, 12 seconds [11]).

State promotions incur long “ramp-up” delays of up to several seconds during which tens of control messages are exchanged between the handset and the RAN for resource allocation. Excessive state promotions increase the signaling overhead at the RAN and degrade user experience, especially for short data transfers [3][10]. On the other hand, state demotions incur *tail times* (T_{tail}) causing waste of radio resources and handset energy [5]. During the tail time, no data is transferred but the handset radio power is much higher than that at the idle state (*e.g.*, 1060mW vs 11mW for LTE [8]).

Fast Dormancy. Why are tail times necessary? First, the overhead of resource allocation (*i.e.*, state promotions) is high and tail times prevent frequent allocation and deallocation of radio resources. Second, the RAN has no easy way of predicting the network idle time of a handset, so it conservatively appends a tail to every network usage period. This naturally gives rise to the idea of letting the handset actively request for immediate resource release. Based on this intuition, a feature called Fast Dormancy has been included in 3GPP since Release 7 [1][2]. It allows a handset to send a control message to the RAN to immediately demote the RRC state to idle (or a hibernating state) without experiencing the tail time. Fast dormancy is supported by many handsets [2]. It can dramatically reduce the radio resource and the handset energy usage with the potential penalty of increased signaling load when used aggressively [3][10].

3 The RadioProphet (RP) System

The static tail times are the root cause of low resource efficiency in cellular networks. **RP** leverages fast dormancy to dynamically determine when to release radio resources.

Challenge 1: trading off between resource saving and signaling load. The best time to perform resource deallocation is when the handset is about to experience a long idle time period t . If t is longer than the tail time, deallocating resources immediately saves resources without any penalty of signaling load (*i.e.*, state promotions). Otherwise, doing so incurs an additional state promotion. Balancing such a critical tradeoff requires predicting the idle time between data transfers so that fast dormancy is only invoked when the idle time is sufficiently long.

Challenge 2: handling both foreground and background traffic. Idle time prediction is particularly difficult for applications involving user interactions. Previous systems, such as RadioJockey [4] and MakeActive [7], simply avoid this by only handling traffic generated by applications running in the background.

Challenge 3: trading off between prediction accuracy and system performance. **RP** is a service running on a handset with limited computational capabilities and more importantly, limited battery life. So we need to minimize the overhead without sacrificing much of the prediction accuracy.

To address **Challenge 1**, we establish a novel machine-learning-based framework for idle time prediction. Besides measuring the effectiveness and efficiency of a wide-range of ML algorithms, our key contribution is addressing the hard problem of selecting discriminating features that are relevant to idle time prediction. Based on extensive measurement, we find that strategically using a few simple features (*e.g.*, packet direction and size) leads to high prediction accuracy (§4). To address **Challenge 2**, we designed a general prediction framework that works for the aggregated (possibly concurrent) traffic containing both foreground and background traffic. In contrast, previous systems such as RadioJockey have the ideal usage case for a single app. Further, we leverage the screen status [9], which indicates whether a user is interacting with the device, to customize the prediction for screen-on and off traffic. Such a novel approach can better balance the aforementioned tradeoff between resource saving and signaling load. To address **Challenge 3**, **RP** performs *binary* prediction at the granularity of a traffic *burst* consisting of a train of packets. In other words, we find that the knowledge of whether the inter-burst time (**IBT**) is short or long (determined by a threshold) is already accurate enough for guiding the resource deallocation. Such an approach is much more efficient while yielding similar accuracy compared to the expensive approach of predicting the precise value of packet inter-arrival time.

RP consists of three components: a traffic monitor, an **IBT** prediction module, and a Fast Dormancy (FD) scheduler. The monitor inspects network traffic (only examines packet headers) and extracts lightweight features for each burst in an online manner. The features are then fed into the **IBT** prediction module, which trains models to predict the **IBT** for the current burst. Then, the FD scheduler makes decision on whether to invoke fast dormancy based on the **IBT** prediction result.

For **IBT** prediction, we formulate the traffic pattern as follows. The traffic is a sequence of packets $\{P_i\}(1 \leq i \leq n)$ in both directions. Let the timestamp of P_i be t_i . Using a burst threshold **BT**, the packets are grouped into *bursts*, *i.e.*, $\{P_p, P_{p+1}, \dots, P_q\}$ belongs to a burst B if and only if: (1) $t_{k+1} - t_k \leq \mathbf{BT}$ for $\forall k \in \{p, \dots, q-1\}$, (2) $t_{q+1} - t_q > \mathbf{BT}$, and (3) $t_p - t_{p-1} > \mathbf{BT}$. We define the inter-burst time **IBT** of burst B to be the time gap following this burst, *i.e.*, $t_{q+1} - t_q$. We use a short **IBT** threshold called **SBT** to classify an **IBT**, *i.e.*, if **IBT** \leq **SBT**, the burst is *short*, otherwise, it is *long*.

The **IBT** prediction module trains a model based on historical traffic information, which consists of an array of bursts $\{B_1, \dots, B_m\}$. Each B_i is a vector $(f_1, f_2, \dots, f_t, ibt_i)$ where $\{f_1, \dots, f_t\}$ is the list of features of B_i and ibt_i is the **IBT** following burst B_i observed by the traffic monitor. Whenever there is an idle time of **BT**, *i.e.*, a new burst appears, the prediction process starts. The feature vector of the current burst $\{f_1, \dots, f_t\}$ is generated and fed to the prediction module, which predicts whether the **IBT** following the current burst is short or long. If short, no change is made and the handset stays in the tail, since a packet is likely to appear soon. Otherwise, the FD scheduler invokes fast dormancy to save energy. The prediction model is customized for each handset, and is dynamically updated to adapt to the recent traffic pattern.

4 Feature Selection

We describe the measurement dataset before studying the feature selection in §4.2.

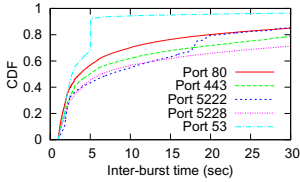


Fig. 1. **IBT** distributions of bursts whose last packets have specific port numbers

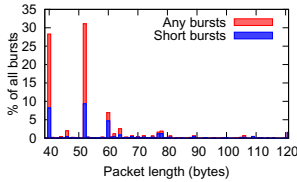


Fig. 2. Distributions of bursts grouped by packet length of the last packet

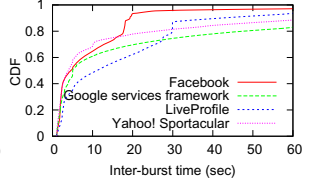


Fig. 3. **IBT** distributions of bursts whose last packets are associated with specific apps

4.1 The UMICH Dataset

The measurement data used in this study, which we call the UMICH dataset, is collected from 20 students at University of Michigan for seven months. The students were given Motorola Atrix (11 of them) or Samsung Galaxy S smartphones (9 of them) running Android. Our custom data collection software continuously runs in the background and collects three types of data. (1) Packet traces (only headers are used in this study). (2) The process name responsible for sending or receiving each packet. (3) Other system information such as screen status. Over the seven months (May to Dec 2011) we collected 152 GB data. Although both cellular and Wi-Fi traces were collected, in this study, we only use cellular traces, which contribute to 57.8% of the total traffic volume.

4.2 Measurement Driven Feature Selection for Burst Classification

We use a measurement-driven approach to derive features for the prediction model by analyzing the correlation between various features and the **IBT**. First, to predict whether an **IBT** is short or long, we look at the burst *right before* the **IBT**, since we observe that the correlations between the **IBT** and earlier bursts' features are much weaker. Second, the features are extracted from the *last three* packets of a burst. This is because in most cases, bursts are small (53% of bursts consist of no more than 3 packets), and even for large bursts, we can usually tell their nature based on the last three packets, *e.g.*, TCP three-way handshake. Third, we only inspect packet headers since examining payload incurs much higher overhead and also because traffic is increasingly being encrypted.

The lightweight features of the last three packets¹ used by **RP** are listed below: (1) packet direction, (2) server port number, (3) packet length (including header), (4) protocol field in IP header, (5) TCP flags field in TCP header (0 if not TCP), and (6) application name associated with the packet. These features are selected empirically so that they are most relevant to **IBT** based on our measurement. We show three features below as examples. We start our analysis with **BT** = 1s and **SBT** = 3s. Later we explore how different **BT** and **SBT** settings affect our results in a quantitative manner (§5.4).

Port Number. Figure 1 shows **IBT** distributions of the top 5 ports ordered from top to bottom in the legend, *e.g.*, 80 is the most popular port, across all users. **IBT** distributions

¹ If a burst contains less than three packets, all features for the missing packet(s) have a value of 0.

of different ports clearly differ, especially for port 53, whose sudden jump at $\mathbf{IBT} = 5$ seconds corresponds to the DNS retransmission timeout on Android. We also observe clusters of \mathbf{IBT} values for many other ports. For example, most bursts over port 5222 have a 20-second \mathbf{IBT} corresponding to the keep-alive periodicity of Facebook.

Packet Length. Figure 2 plots the distributions of last packet lengths of bursts with short \mathbf{IBT} ($\mathbf{IBT} < \mathbf{SBT}$) and all bursts. Most bursts end with small packets, *i.e.*, 84.59% have their last packets ≤ 100 bytes, as a large packet is typically in the middle of a burst. We observe high correlation for a few packet lengths values. For example, for 121 bytes, 93.04% bursts have short \mathbf{IBTs} . The machine learning algorithms could automatically discover these rules for prediction.

Applications. In Figure 3, the legend shows the sorted list of apps contributing the largest amount of bursts with Facebook ranked at top 1. The differences in \mathbf{IBT} values are clear across apps. We also observe that for some apps, their periodic transfer behaviors contribute to clusters of specific \mathbf{IBT} values, *e.g.*, Facebook and LiveProfile. The application information can be very efficiently obtained (*e.g.*, on Android [11]).

5 Implementation and Evaluation

5.1 Implementation

Trace-Driven Evaluation. We implement simulators of \mathbf{RP} , MakeIdle [7], and RadioJockey [4] on a desktop (3.16 GHz Xeon CPU with 16GB memory) using Matlab. They work with an RRC state machine simulator (§5.2). We use them to evaluate the accuracy and resource savings of \mathbf{RP} under various configurations (§5.3, §5.4), as well as to compare \mathbf{RP} with other optimization techniques (§5.5), using the UMich trace.

Implementation on Real Android Phone. We also implement the full \mathbf{RP} system on a Samsung Galaxy S3 phone running Android 4.0.4 to evaluate its running overhead (§5.6). A modified TcpDump program is used as the traffic monitor. The \mathbf{IBT} prediction module is implemented as a native Android application running in the background.

5.2 Evaluation Methodology

We use three metrics to evaluate \mathbf{RP} : prediction accuracy, saved radio energy, and increased signaling load. The accuracy is defined as the number of bursts whose immediate \mathbf{IBT} (short or long) are correctly predicted divided by the total number of bursts in the input trace. The radio energy, denoted as E , is the energy consumed by the handset radio interface. It is one of the most significant components for the overall energy usage of a handset, along with screen and CPU energy [11]. We build an RRC state machine simulator, which takes as input a packet trace and employs the LTE radio energy model derived in our previous work [8] to calculate E (using a UMTS model [11] yields qualitatively similar results). The signaling load, denoted as S , is quantified by the number of state promotions, each incurring a fixed number of signaling messages [4]. S is also computed by the RRC state machine simulator.

Assume when a specific user trace is evaluated without any optimization performed (no fast dormancy), E and S are calculated to be E_d and S_d , respectively. When \mathbf{RP}

Table 1. Impact of α , β on the prediction accuracy (**PerUserDynamic** model)

$\alpha =$	100	500	1000	2000	5000
$\beta = 1$	81.5%	83.7%	84.2%	82.4%	80.2%
$\beta = 2$	80.1%	81.4%	82.9%	82.0%	80.0%
$\beta = 5$	79.8%	80.9%	81.4%	81.0%	79.3%
$\beta = 10$	79.4%	80.0%	80.9%	80.0%	79.0%
$\beta = 20$	78.9%	79.6%	80.2%	79.5%	78.7%

Table 2. Summary of prediction models

Name	Description	Accuracy
PerUserDynamic	Use most recent α bursts of a user to predict next β bursts for that user	84.2%
PerUserStatic	Use a fixed set of n bursts of a user to train a fixed model for that user	80.8%
AllUserStatic	Use a fixed set of k bursts of all users to train a fixed model for all users	77.5%

is used, the resulting E and S become E' and S' , respectively. We define $\Delta(E) = (E_d - E')/E_d$ and $\Delta(S) = (S' - S_d)/S_d$ (usually both are positive). They correspond to the reduction of the radio energy and the increase of the signaling load brought by **RP**, respectively. **RP**'s goal is to maximize $\Delta(E)$ while minimizing $\Delta(S)$.

5.3 Prediction Model Comparison

In **RP**, we use recent traffic information of a user to train a model, denoted as **PerUserDynamic**. Specifically, for each user, the most recent α bursts are used to predict the next β bursts. We study the impact of α, β in Table 1, using the Ensemble Bagging [6] learning algorithm as an example (number of trees set to 20). If α is too small, there is not enough training data for learning; if α is too large, the user is more likely to switch to new applications that generate different traffic patterns so previously learned rules may not be useful. Based on Table 1, we choose $\alpha = 1000$ and $\beta = 1$ that maximize the accuracy. In practice, α and β could also be dynamically adjusted.

Table 2 compares the **PerUserDynamic** model with two other models, **PerUserStatic** (a fixed model for each user) and **AllUserStatic** (a fixed model for all 20 users). For fair comparison, we use the same ML algorithm (Ensemble Bagging) as used in Table 1. We set $\alpha = 1000$ and $\beta = 1$ for the **PerUserDynamic** model as discussed previously, and use $n = 10,000$ for **PerUserStatic** and $k = 10,000$ for **AllUserStatic** (n and k defined in Table 2). Similar to Table 1, n and k are empirically selected to yield good prediction accuracies. We observe that **PerUserDynamic** has higher prediction accuracy than the other two models, suggesting that it is necessary to have a dynamic model for each user whose traffic pattern may be different from others.

5.4 Selecting Burst Thresholds

We study the impact of **BT** and **SBT** (previous evaluations use **BT** = 1s and **SBT** = 3s). In Table 3, S_0 to S_4 correspond to representative (**BT**, **SBT**) pairs. We find that aggressively using a short **SBT** (S_1) can significantly increase $\Delta(S)$. Among all settings, S_4 yields the highest $\Delta(E)/(1 + \Delta(S))$ value (the average radio energy saving per unit of signaling load). It quantifies how well the balance between $\Delta(E)$ and $\Delta(S)$ is handled.

As mentioned in §3, configuring screen-on and off settings differently may yield better optimization results, as screen-off traffic is usually generated by background apps without user interaction, leading to statistically longer **IBT**. Therefore a more aggressive

Table 3. Impact of **BT** and **IBT** (Classification Tree with **PerUserDynamic** model, $\alpha=1000$, $\beta=1$)

Settings (unit: sec)	Accuracy	$\Delta(E)$	$\Delta(S)$	$\frac{\Delta(E)}{(1+\Delta(S))}$
S_0 BT : 1 SBT : 3	82.65%	52.10%	101.64%	0.26
S_1 BT : 1 SBT : 2	84.80%	56.69%	158.99%	0.22
S_2 BT : 1 SBT : 4	81.94%	49.07%	83.34%	0.27
S_3 BT : 0.5 SBT : 3	84.71%	53.74%	100.36%	0.27
S_4 BT : 1.5 SBT : 3	85.39%	58.85%	93.75%	0.30
S_5 BT : 1/1.5 off/on SBT : 2.5/3 off/on	85.88%	59.07%	91.01%	0.31

Table 4. Performance and accuracy of different ML algorithms

ML Algorithm	Prediction time (Training time)	Accuracy
Naïve Bayes	2.5 ms 6.4 ms	76.1%
Classification Tree	5.9 ms 136.9 ms	85.9%
Ensemble Bagging	106.6 ms 626.1 ms	87.4%

setting (smaller **BT** and **SBT**) can be applied to screen-off traffic without incurring much signaling overhead. In Table 3, S_5 is such a screen-aware setting. Compared with S_4 , S_5 saves more energy with less signaling overhead incurred. In fact, S_5 achieves results comparable to the optimal scenario to be shown in Table 5. This also indicates that dynamically changing **BT** and **SBT** can help improve the effectiveness of **RP**.

5.5 Comparing Fast Dormancy Based Resource Optimization Approaches

Table 5 compares various optimization techniques using the UMICH dataset.

Basic fast dormancy. We set T_{tail} to a fixed value smaller than its original value.

RadioJockey [4] uses system calls to predict the end-of-session (EOS) for background app without user interaction, with the ideal usage scenario for a single app. Given that we do not have system call traces in our dataset, we make two assumptions in our simulation: (1) we use end-of-burst to approximate end-of-session, (2) RadioJockey has high prediction accuracies (90% and 100%) for both single and concurrent apps (although in reality, it performs worse when concurrent apps exist). A key limitation of RadioJockey is it does not handle foreground traffic and only works when the screen is idle (see §6), so we only apply RadioJockey to screen-off traffic².

Makeldle [7] computes a wait time T_{wait} that maximizes the energy saving if T_{tail} is set to T_{wait} for the previous M packets, it then applies this T_{wait} for the next N packets. The range we search for the optimal T_{wait} is [0.5, 11.5] seconds, as suggested by the authors. Since no recommendations have been made for the values of M and N , we empirically select different combinations of (M, N) pairs.

RadioProphet : we explore three off-the-shelf machine learning algorithms with the **PerUserDynamic** model ($\alpha=1000$ and $\beta=1$): Naïve Bayes, Classification Tree, and Ensemble Bagging. Their performance and accuracy are summarized in Table 4³.

We now discuss the results in Table 5. “Fast dormancy 1s” is an aggressive approach incurring unacceptable signaling overhead. “Fast dormancy 3s” reduces $\Delta(S)$ with less energy saving as expected. For both approaches, their $\Delta(E)/(1 + \Delta(S))$ values

² We configured short screen timeout for the 20 phones so screen-off is good approximation for screen-idle.

³ The performance numbers in Table 4 correspond to the execution time of the scripts written in Matlab on desktop. Our real implementation on the S3 smartphone uses C++ so it is much more efficient (§5.6).