BreezChirp: Energy Efficient Wi-Fi Bandwidth Estimator for Smartphones

Jian Li*, Jin Xiao*, Huu Nhat Minh Nguyen‡, James Won-Ki Hong*, Raouf Boutaba*†

*Division of IT Convergence Engineering, POSTECH, Pohang, Korea
email: {gunine, jinxiao, jwkhong, rboutaba}@postech.ac.kr

†David R. Cheriton School of Computer Science, University of Waterloo, Canada
email: rboutaba@cs.uwaterloo.ca

‡Faculty of Computer Science and Engineering, Hochiminh City University of Technology, Vietnam
email: nhatminhhcmut@gmail.com

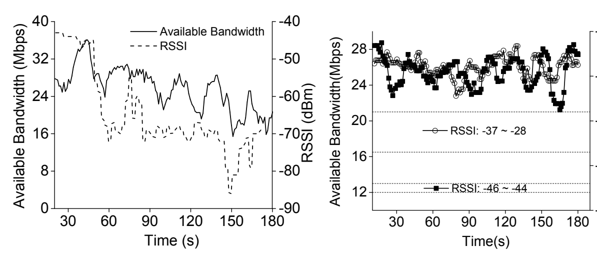
***Abstract*—Mobile data service is a rapidly growing business sector today. Available application bandwidth is an essential metric for managing applications and user experience. However, existing bandwidth measurement tools are ill-suited for the wireless environment due to large variance in environmental conditions, mobility, and energy consumption constraints. In this paper, we modify a known available bandwidth measurement technique - pathChirp to the wireless environment by utilizing the energy management features of the Wi-Fi communication device on smartphones. Our resulting tool BreezChirp is effective and energy conserving. We implemented BreezChrip on smartphones and evaluate its performance through field experiments.**

***Index Terms*—Energy Saving, Smartphones, Measurement**

I. Introduction

The proliferation of mobile applications on smartphones necessitates effective network resource management. Compared with wired network, wireless network has varying conditions and consequently understanding the current network condition plays a key role in wireless resource management. To this end, available application bandwidth is an often used network resource metric for managing mobile applications. In this paper, we focus on Wi-Fi networks due to their popularity in supporting mobile applications. Although measuring application bandwidth is a well investigated problem in wired networks, it remains an open problem in wireless environment due to: one, the wireless channel exhibits high variance, making precise measurement difficult; two, terminal mobility requires frequent update of bandwidth measurements; three, energy efficiency is of utter importance to smartphones. Issue one and two call for prolonged and frequent bandwidth measurements to ensure information precision and fidelity, while issue three strives for short and infrequent measurement. We therefore appear to have arrived at a logical impasse.

One simple method of getting around this is to rely on passive rather than active measurement. Indeed in Wi-Fi network, a metric called Received Signal Strength Indicator (RSSI) is used to quantify the power level present in received radio signal. Why not infer the application bandwidth based on RSSI value? We conducted experiments to quantify this assertion. Figure 1 shows the results obtained from a simple one terminal one access point setup. It is clear that the correlation between RSSI and application bandwidth is not strong enough for modeling purpose. Moreover, the presence and activities of other mobile terminals in the same Wi-Fi zone further obfuscate this correlation (e.g., channel contention, interference and etc.).



(a) Relations between Application Bandwidth and RSSI (wide RSSI range)

(b) Relations between Application Bandwidth and RSSI (narrow RSSI range)

Fig. 1. Measured Application Bandwidth under Different RSSI Ranges

Therefore, there is a reliance on active measurement, which is by far the most effective method for wired network. Unfortunately, existing active measurement tools (such as pathChirp) does not do well in wireless environment [1][2]. In general, the measurement duration needed to obtain reasonable precision in Wi-Fi is much longer than wired network. This observation, combined with concerns over communication overhead and energy consumption have resulted in the lack of an effective and efficient wireless application bandwidth measurement tool for smartphones.

In our research, we found that in utilizing the existing power saving schemes of 802.11 protocol and in understanding the application traffic flow, it is possible to obtain an effective and energy-efficient bandwidth measurement method for

This research was supported by the MSIP(Ministry of Science, ICT & Future Planning), Korea, under the ITRC(Information Technology Research Center) support program supervised by the NIPA(National IT Industry Promotion Agency) (NIPA-2013-H0301-13-3002)

smartphones. We call the resulting method Battery REsource Efficient measurement System (BreezChirp), as the idea of chirp for bandwidth measurement first came from pathChirp [3]. The advantage of using a chirp over more traditional packet train is that chirp's exponential increasing inter-packet dispatch technique is ideal for sampling a wide range of bandwidth values within a relatively short period of time. This allowed us to adapt the chirp size based on the Power Saving Mode (PSM) characteristics of 802.11. We compensate the loss of accuracy with full chirp measurement when the Wi-Fi device is in Continuous Active Mode (CAM). Consequently, BreezChirp is able to obtain good measurement accuracy, is energy conserving and is fully implementable on smartphones.
In this paper, we first introduce the chirp idea of pathChirp and discuss its performance and enhancement needed in Wi-Fi environment. We then present BreezChirp and discuss its implementation details and performance tuning. Finally, we show the experiment results we have observed by deploying BreezChirp on smartphones.
The remainder of this paper is organized as follows. Section II presents related works on active available bandwidth measurement techniques, and we study the performance of a prominent method, pathChirp in wireless environment in Section III. In Section IV, we present BreezChirp. Section V reports on the experiment results. Section VI concludes the paper.

II. Related work

Active measurement of available bandwidth in literature generally falls under one of two techniques: packet dispersion and self-induced congestion.
For packet dispersion technique, two or more packets are sent back-to-back to estimate capacity of a network path based on inter-packet arriving rate. This is because the time dispersion between probe packets is related to the bottleneck link capacity. pathrate [4] and CapProbe [5] are two well-known representative tools utilizing this technique. In presence of crossing traffics however, these techniques become ineffective with high error rate. Furthermore, They introduces significant communication overhead and has long convergence time.
The key idea of self-induced congestion approach is that if the probing rate exceeds the available bandwidth, then the probe packets become queued, and consequently an increased transfer time is required. Therefore, with varying probing rate, we can estimate available bandwidth by analyzing the packet delay at the receiver through congestion detection. The proposed techniques in literature generally differ in the design of the rate controller used for the probing packet train. For instance, Pathload [6] uses constant bit rate of packet train, TOPP [7] uses linearly increasing probing rate of packet pairs, while pathChirp [3] uses exponentially increasing probing rate of packet train. Pathload tries to adaptively vary the rates of consecutive probing based on the long Constant Bit-Rate (CBR) packet trains. The measurement duration is long and the communication overhead is large. Instead of using the packet trains, TOPP employs non-uniform spaced packet pairs to estimate the available bandwidth. Because TOPP relies on paired packets, it does not have delay correlation information typically obtained from packet trains, and consequently the measurement precision suffers. pathChirp combines the idea of probe train with non-uniform spaced packet pairs into what they call a chirp, which is a sequence of probing packets with exponentially decreasing inter-packet spacing. pathChirp has been shown to perform very well in wired networks and is relatively resource conscious compared with Pathload in that the self-induced congest state is short lived, and overall fewer number of trains are needed to obtain a good bandwidth measurement.
To date, few works on wireless bandwidth measurement are reported. Among them are Wireless bandwidth estimation tool (Wbest) [1] and DietTOPP [8]. Wbest proposed analytical model in packet dispersion technique to improve accuracy and convergence time specifically for wireless network. However, the measurement accuracy is poor in the presence of cross traffic. DietTOPP is a simplified version of TOPP with reduced complexity. Similar to TOPP, DietTOPP also injects substantial among of probing traffic into the network.

III. pathChirp Overview and Experimental Study

A. pathChirp Overview

pathChirp use m number of packet trains called chirps. In each chirp, N exponentially spaced probe packets reside. The relative queueing delay between chirp sender (measurement host) and receiver (measurement server) is measured based on the sequence of arrival times at the receiver, and available bandwidth is estimated accordingly. Each probe packet has identical packet size $\mathbf{P}$ bytes and the inter-packet time Δ between two consecutive packets is exponentially decreased by a spread factor γ. As Δ decreases over the course of a chirp, the probe intensity increases which eventually lead to congestion, called excursion. When an excursion occurs, pathChirp tries to calculate packet rate $\mathbf{R}_k$ through equation $\mathbf{R}_k = \mathbf{P}/\Delta_k$, where k denotes packet index and $\mathbf{R}_k$ denotes available bandwidth. However, not all excursions are indication of congestion. Transient excursions may occur due to variance in the underlying communication channels. pathChirp filters such transient excursions out by applying statistical analysis on the packet delays over a certain period of time, called a sliding window $\mathbf{W}$.

B. Experiment Study

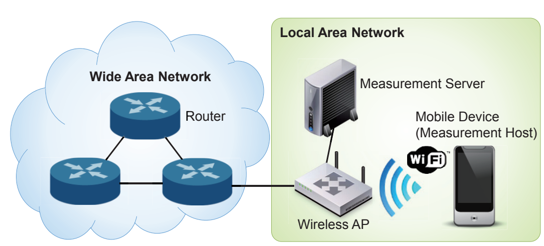
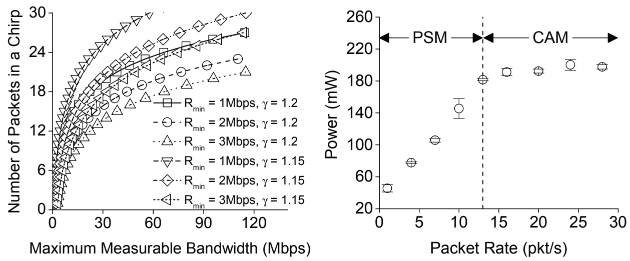


Fig. 2. Testbed Setup for Studying pathChirp

To analyze the performance of pathChirp in Wi-Fi environment, we constructed a simple testbed as depicted in Figure 2. The testbed is comprised of three components: smartphone client (measurement host), wireless AP and measurement server. The Measurement server is connected to the AP through 1 Gbps Ethernet. The original pathChirp is implemented in C and runs on top of linux machine (measurement host and server). We ported the client program to Android smartphones using Android's Native Development Kit (NDK). The pathChirp server program is unchanged and resides on the measurement server.



(a) The relation between minimum measurable bandwidth $\mathbf{R}_{min}$, spread factor γ and number of packets generated per second

(b) The Relation between Energy Expenditure and Number of Packets Transmitted Per Second

Fig. 3. The Relation between Energy Expenditure and pathChirp Parameters

pathChirp has a number of tunable parameters, among which spread factor γ and measurable bandwidth range (defined by maximum measurable bandwidth $\mathbf{R}_{max}$ and minimum measurable bandwidth $\mathbf{R}_{min}$) are of particular importance to us because they jointly determine the rate of probe packet generations. Based on these three parameters, Inter-spacing time Δ is determined, which is exponentially decreasing across packets in the same chirp train.

For instance, assuming we have N packets per chirp, then the first inter-spacing time Δ_2 will have $T\gamma^{N-2}$ ($\gamma > 1$ strictly), while the last inter-spacing time Δ_N will have value T. The following equation is used to determine all Δ within a chirp:

$$\mathbf{\Phi} = \sum_{i=2}^{N} \Delta_i = \sum_{i=2}^{N} T\gamma^{N-i} = T\sum_{i=2}^{N} \gamma^{N-i} \tag{1}$$

Each chirp has identical cumulated inter-spacing time, i.e., $\mathbf{\Phi} = 1$ (second). Therefore N varies with γ, and hence varied probing packet generation rate. Since the packet rate directly relates to energy consumption and communication overhead, we want to quantify the relation among γ, $\mathbf{R}_{min}$, $\mathbf{R}_{max}$, and N. Accordingly, we have performed series of experiments based on the following setup: for differing the measurable bandwidth range, we linearly increase the value of $\mathbf{R}_{max}$ from 3 Mbps to 120 Mbps, and assign 1 Mbps, 2 Mbps and 3 Mbps to $\mathbf{R}_{min}$ respectively. Moreover, we choose 1.15 and 1.2 for γ and conduct six sets of experiments. In order to track the packet generation rate, we use a packet count program and compute metrics on generated packets per second. Each set of experiments is performed multiple times and the averaged value is computed and shown in Figure 3(a). On Figure 3(a), we can observe that with the same measurable bandwidth range (the line for the same $\mathbf{R}_{min}$), as value of γ increases, smaller number of packets are generated, while with the same γ value, smaller range of measurable bandwidth (which has larger $\mathbf{R}_{min}$) generates smaller number of packets.

According to the power model documented in [9], the Wi-Fi device mode, Continuous Active Mode (CAM) or Power Saving Mode (PSM), is crucial in determining the smartphone energy consumption rate due to Wi-Fi communication. Device in PSM consumes only a fraction of energy compared with CAM mode. The mode switching is in turn determined by the packet generation rate. However as we can see in Figure 3(b), even with a large value of γ and low value of $\mathbf{R}_{min}$, pathChirp switches the network device into CAM mode. Now, if we can control the packet generation rate within the mode switching threshold, we can save significant amount of energy. This is the key rationale behind BreezChirp. However, the measurement efficiency suffers significantly when we do this. Hence novel technique is needed to compensate for this loss of efficiency.

In the following section, we present BreezChirp which constraints the chirp to operate under PSM mode with an adaptive sliding window technique, and compensate the loss of efficiency with application traffic adaptive full chirps. In doing so, we arrive at a novel bandwidth measurement tool that is energy efficient, effective, and smartphone deployable.

IV. Adaptive Range Projection Algorithm

In this section, we discuss the design of BreezChirp. BreezChirp achieves energy saving by significantly reduce the bandwidth probing range of a chirp. As we have shown in Figure 3(b), Wi-Fi energy expenditure for smartphones is achieved by keeping the network device in PSM mode, and the mode switching parameter is the total number of packet generated per second. Based on experiments, We observed that 8 packets per second is the threshold value that triggers CAM to PSM switch. Conversely, 13 packets per second is the threshold value that triggers PSM to CAM switch. Therefore, if we can control the chirp packet generation rate per second to below this threshold, we can achieve energy efficiency. However, this in practice severely constrains the bandwidth probing range of a chirp. If there is a drastic change in the environment in terms of available bandwidth (e.g., due to mobility, interference, cross traffic, etc.), BreezChirp cannot efficiently shift its measurement range within reasonable time frame. Therefore, full chirp is needed to compensate for this loss of efficiency. Again as we observe in Figure 3(b), when the network device is in CAM mode, increasing the packet rate does not significantly increase energy expenditure. Hence BreezChirp schedules a full chirp measurement to coincide with application-level traffic activities, by detecting whether the network device is already in CAM mode or not. Thus the design of BreezChirp involves two modes: *limited* mode (with a narrow sliding window), efficient for measuring moderate bandwidth changes in the environment; and *normal mode* (full chirp), efficient in capturing drastic bandwidth changes.

As γ, the spread factor, moderates the packet generation rate. We want to be able to constrain its value to fit into the PSM threshold. The way pathChirp computes available bandwidth is as follows: $\mathbf{R}_k = \mathbf{P}/\Delta_k$ is used to determine the packet rate and when channel congestion occurs at a particular packet rate, we obtain the available bandwidth. We can rewrite the equation with respect to γ as,

$$\mathbf{R}_{k+1} = \frac{\mathbf{P}}{T\gamma^{N-(k+1)}} = \frac{\mathbf{P}}{T\gamma^{N-k}} \times \gamma = \mathbf{R}_k \times \gamma \qquad (2)$$

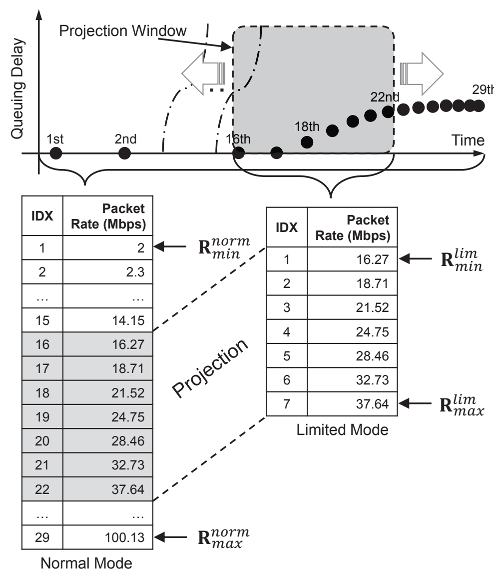


Fig. 4. Example of Adaptive Range Projection

In Equation 2, k is the index number of a packet within a chirp, and the next packet rate $\mathbf{R}_{k+1}$ is computed by multiplying γ to the current packet rate $\mathbf{R}_k$. As γ increases, the inter-packet gap of two consecutive packets increases accordingly, which in turn reduces the measurement granularity. Because of this inverse relation between γ and measurement granularity, we cannot increase γ without bound. This led us to choose the most appropriate value of γ with which we can preserve maximum measurement granularity while generating the least number of packets. Through numerous experiments, with measurement bandwidth range $0 \sim 100$ Mbps, we found that 29 is the least number of packets, and we denote the spread factor that yields 29 packets as γ_{opt}. However, 29 packets per second will put the network device into CAM mode which we do not want, we therefore need to also adjust the measurable bandwidth range to further reduce the packet number from 29 to 13. Since the measurable bandwidth range covered by 13 packets is the subset of that covered by 29 packets, we term this subset a *projection*. BreezChirp's *limited* mode utilizes the technique of *adaptive range projection*, which is illustrated in Figure 4. It operates like a sliding window that dynamically adjusts to a projection that encapsulates the available bandwidth. *Limited* mode uses the least amount of packets $\mathbf{N}_{lim}$ (13) to measure a narrow range of bandwidth ($\mathbf{R}_{min}^{lim}$ - $\mathbf{R}_{max}^{lim}$ Mbps).

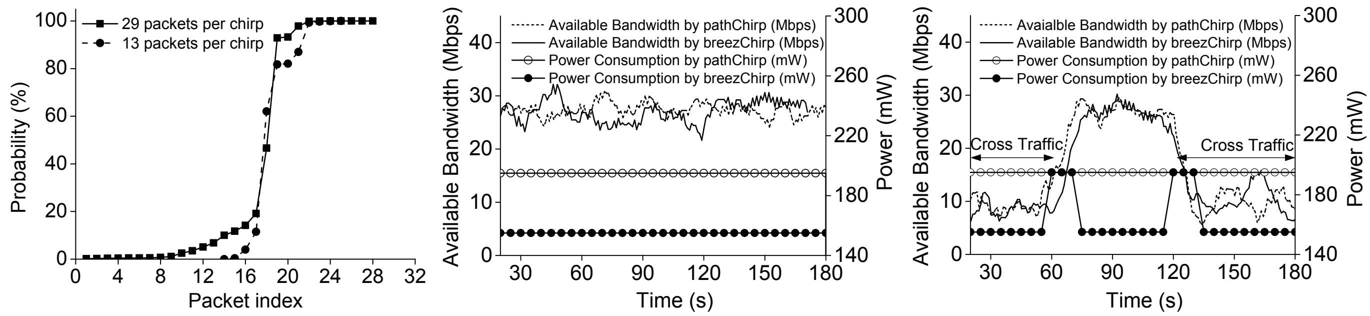
The pseudo code of the adaptive range projection algorithm is given in Algorithm 1. The algorithm is comprised of three parts: 1) Divide the $\mathbf{N}_{lim}$ into three partitions - $\mathbf{N}_{lim}^{up}$, $\mathbf{N}_{lim}^{mid}$ and $\mathbf{N}_{lim}^{low}$; 2) Find the packet rate $\mathbf{R}_{appr}$ which is closely approximated to current measurement result $\mathbf{R}_{curr}$ and assign $\mathbf{N}_{lim}^{mid}$ as the index of $\mathbf{R}_{appr}$; and 3) Find the new measurement range through calculating $\mathbf{R}_{min}^{lim}$ as well as $\mathbf{R}_{max}^{lim}$ by dividing spread factor $\mathbf{N}_{lim}^{up}$ times to $\mathbf{R}_{appr}$, or multiplying spread factor $\mathbf{N}_{lim}^{low}$ times to $\mathbf{R}_{appr}$. If the new measurement result is larger than the previous result, the algorithm would move the projection window forward to preserve the increasing trend. On the contrary, the algorithm would move the projection window backward to preserve the decreasing trend (see Figure 4). In this way, *limited* mode adapts the measurement range slowly according to the newest measurement result, and is ideal for accommodating small to moderate changes in the environment with respect to available bandwidth.

Algorithm 1: Adaptive Range Projection Algorithm

input : stable measurement result $\mathbf{R}_{curr}$ and number of packets for limited mode $\mathbf{N}_{lim}$
output: minimum packet rate $\mathbf{R}_{min}^{lim}$ and maximum packet rate $\mathbf{R}_{max}^{lim}$

if $\mathbf{R}_{curr} < \mathbf{R}_{min} \cup \mathbf{R}_{curr} > \mathbf{R}_{max}$ **then**
  $\mathbf{R}_{min}^{lim} = \mathbf{R}_{min}^{norm}$
  $\mathbf{R}_{max}^{lim} = \mathbf{R}_{max}^{norm}$
  go to normal mode
else
  $\mathbf{N}_{lim}^{mid} = 1$
  $\mathbf{N}_{lim}^{up} = \lceil \mathbf{N}_{lim}/2 \rceil - 1$
  $\mathbf{N}_{lim}^{low} = \mathbf{N}_{lim} - \mathbf{N}_{lim}^{up} - \mathbf{N}_{lim}^{mid}$
  $\mathbf{R}_{appr} = \mathbf{R}_{min}^{lim}$
  while $\mathbf{R}_{appr} < \mathbf{R}_{curr} - \epsilon$ **do**
    $\mathbf{R}_{appr} = \mathbf{R}_{appr} * \gamma$
  $\mathbf{R}_{max}^{lim} = \mathbf{R}_{appr}$
  $\mathbf{R}_{min}^{lim} = \mathbf{R}_{appr}$
  for $i = 1, 2, ..., \mathbf{N}_{lim}^{up}$ **do**
    $\mathbf{R}_{max}^{lim} = \mathbf{R}_{max}^{lim} * \gamma$
  for $i = 1, 2, ..., \mathbf{N}_{lim}^{low}$ **do**
    $\mathbf{R}_{min}^{lim} = \mathbf{R}_{min}^{lim}/\gamma$

However if the bandwidth change is drastic in the environment, adaptive range projection would not work well. We can draw similarity to how an observer tracks a moving target in the distance with a binocular. When the target is moving slowly, the observer can track the target through his binocular, which has precise focus but very limited scope. However, if the target all of a sudden vanishes from within the scope (e.g. a sudden dash). The observer will use his eyesight to locate the target's new location rather than trying to search with the binocular which is too slow to adapt. Similarly, we need a full chirp to find the available bandwidth when the bandwidth change is too drastic. BreezChirp's *normal* mode does exactly that. Energy efficiency is achieved by: 1) Use $\mathbf{N}_{norm}$ (29) to measure the available bandwidth in full measurable bandwidth range ($\mathbf{R}_{min}^{norm}$ - $\mathbf{R}_{max}^{norm}$ Mbps). As we discussed before, 29 packets appears to be the least number of packet per second needed to obtain a good measurement in Wi-Fi environment; 2) Whenever



(a) PDF of congestion with 29 and 13 packets in a chirp

(b) Measured Available Bandwidth and Energy Consumption Comparison without Cross Traffic

(c) Measured Available Bandwidth and Energy Consumption Comparison with Cross Traffic

Fig. 5. Comparison Result of Measured Available Bandwidth Accuracy and Energy Consumption from pathChirp and BreezChirp

possible, we schedule full chirp when there is active application traffic (i.e., the network device is already in CAM mode).

The switching between *limited* mode and *normal* mode is implemented as follows: at initial stage, we have no clue as to where the bandwidth range for *limited* mode should reside, and therefore we obtain a stable measurement from *normal* mode. We obtain a stable measurement result through observing the variance of measurement results in a period **W**, and if the variance is smaller than the predefined threshold, then we regard the measurement result as stable result. Once we have a stable result, *limited* mode is activated by setting the projection such that the mid-bracket projection covers the observed bandwidth measurement. In *limited* mode, we continuously update the measurement range until a new measurement result is out of all of the measurement brackets, then we schedule a *normal* mode. In this way, BreezChirp is able to provide efficient Wi-Fi bandwidth measurement while conserving energy.

V. Evaluation

To evaluate the performance of BreezChirp, we implemented both pathChirp and BreezChirp on Samsung Galaxy S3, and performed experiments based on the one terminal one access point setup shown in Section III. First we examine the impact of reducing packet numbers on measurement accuracy. Figure 5(a) shows the probability of congestion being detected by a chirp on a given packet index in the chirp. The available bandwidth of the measured channel is indicated by the sharp rise in congestion probability. When the available bandwidth is within the bandwidth measurement range, 13 packets per second is sufficient. Figure 5(b) shows the measured available bandwidth and energy consumption with pathChirp and BreezChirp when the channel condition is relatively stable. BreezChirp performs as well as pathChirp but at a fraction of the energy consumption. We then significantly alter the available bandwidth by introducing a cross traffic from another terminal. As we can see in Figure 5(c), BreezChirp again performs well. We observe a short lag when the environment transition occurs because a switch from *limited* mode to *normal* mode have occurred in BreezChirp. Overall, BreezChirp can conserve around 20.6% power consumption compared to pathChirp, and the energy conservation rate reduces as more environment transition occurs.

VI. Conclusion

In this paper, we presented BreezChirp as an efficient and energy conserving Wi-Fi bandwidth estimation tool for smartphones. By analyzing the bandwidth measurement technique of chirp from pathChirp, in understanding the CAM and PSM - two Wi-Fi operational modes' behavior of the network device, and leveraging the PSM's energy saving feature, we are able to develop a novel bandwidth measurement tool. As we have shown in experiment, BreezChirp works as well as pathChirp but can save significant amount of energy. However, we find that the measurement accuracy of pathChirp is not as stable as its performance in wired networks, and we will make improvement on this as a future work.

References

[1] M. Li, M. Claypool, and R. Kinicki, "Wbest: A bandwidth estimation tool for ieee 802.11 wireless networks," in *Local Computer Networks, 2008. LCN 2008. 33rd IEEE Conference on*, oct. 2008, pp. 374 – 381.

[2] M. Bredel and M. Fidler, "A measurement study of bandwidth estimation in ieee 802.11 g wireless lans using the dcf," in *NETWORKING 2008 Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet*, 2008, pp. 314–325.

[3] V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, and L. Cottrell, "pathchirp: Efficient available bandwidth estimation for network paths," in *Passive and Active Measurement (PAM)*, 2003.

[4] C. Dovrolis, P. Ramanathan, and D. Moore, "What do packet dispersion techniques measure?" in *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, 2001, pp. 905 – 914.

[5] R. Kapoor, L.-J. Chen, L. Lao, M. Gerla, and M. Y. Sanadidi, "Capprobe: a simple and accurate capacity estimation technique," in *Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, 2004, pp. 67 – 78.

[6] M. Jain and C. Dovrolis, "End-to-end available bandwidth: measurement methodology, dynamics, and relation with tcp throughput," *IEEE/ACM Transactions on Networking (TON)*, vol. 11, no. 4, pp. 537 – 549, aug 2003.

[7] B. Melander, M. Bjorkman, and P. Gunningberg, "A new end-to-end probing and analysis method for estimating bandwidth bottlenecks," in *Global Telecommunications Conference, 2000. GLOBECOM '00. IEEE*, 2000, pp. 415 – 420.

[8] A. Johnsson, B. Melander, and M. Björkman, "Diettopp: A first implementation and evaluation of a simplified bandwidth measurement method," *Proceedings of Swedish National Computer Networking Workshop*, 2004.

[9] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, and L. Yang, "Accurate online power estimation and automatic battery behavior based power model generation for smartphones," in *Proceedings of the 8th IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*, 2010, pp. 105–114.