AMUSE: Empowering Users for Cost-Aware Offloading with Throughput-Delay Tradeoffs

Youngbin Im*, Carlee Joe-Wong†, Sangtae Ha‡, Soumya Sen‡, Ted "Taekyoung" Kwon*, Mung Chiang‡

*School of Computer Science and Engineering, Seoul National University, Seoul, Korea
†Program in Applied and Computational Mathematics, Princeton University, Princeton, NJ, USA
‡Department of Electrical Engineering, Princeton University, Princeton, NJ, USA
Emails: ybim@mmlab.snu.ac.kr, {cjoe, sangtaeh, soumyas, chiangm}@princeton.edu, tkkwon@snu.ac.kr

***Abstract*—Mobile users face a tradeoff between cost, throughput, and delay in making their offloading decisions. To navigate this tradeoff, we propose AMUSE (Adaptive bandwidth Management through USer-Empowerment), a practical, cost-aware WiFi offloading system that takes into account a user's throughput-delay tradeoffs and cellular budget constraint. Based on predicted future usage and WiFi availability, AMUSE decides which applications to offload to what times of the day. To practically enforce the assigned rate of each TCP application, we introduce a receiver-side TCP bandwidth control algorithm that adjusts the rate by controlling the TCP advertisement window from the user side. We implement AMUSE on Windows 7 tablets and evaluate its effectiveness with 3G and WiFi usage data obtained from a trial with 25 mobile users. Our results show that AMUSE improves user utility.**

I. Introduction

The growing popularity of mobile devices has recently caused a surge in data usage, driven by applications such as mobile video, cloud services, and online magazines. According to [1], mobile traffic is growing at 78% annually and is expected to account for more than 54% of IP traffic in 2015. To cope with this unprecedented demand, many wireless Internet Service Providers (ISPs) have adopted tiered (usage-based) pricing plans to discourage heavy usage. In addition, they have begun to shift some traffic to other networks, offering free WiFi hotspots and femtocells to offload their 3G traffic [2], [3]. However, though beneficial for ISPs, these new data plans and offloading measures do not always match the interests of their customers [4], [5].

A. Empowering User Decisions

An end user is interested in using his cellular (e.g., 3G) data[1] plan as efficiently as possible. While offloading to WiFi saves users money on their data spending, they must also take into account WiFi's intermittent availability and higher throughput performance. At some times, e.g., while out shopping, a user does not have immediate WiFi access and must wait for WiFi connectivity. The user then faces a tradeoff: she can wait for WiFi access, saving money and experiencing higher throughput performance, or she can consume 3G data immediately but pay for this lower-throughput traffic. The exact structure of this tradeoff varies for different types of traffic. To fully exploit the benefits of WiFi offloading, a user must balance these competing cost, throughput quality, and delay tradeoffs for different applications.

Most users will not manually balance these three competing factors in making offloading decisions. Thus, we propose a user-side, automated WiFi offloading system called AMUSE (Adaptive bandwidth Management through USer Empowerment) that intelligently offloads user traffic from 3G to WiFi networks according to users' preferences. AMUSE utilizes WiFi access prediction and application usage prediction to decide how long application sessions should wait for WiFi, taking into account the amount of 3G bandwidth they will receive should WiFi not be available after they wait. In order to build such a system, we also require a way to automatically enforce the 3G throughput rates for those application sessions that wait for WiFi access. In solving these problems, AMUSE makes the following contributions:

1) We develop a system for cost-aware WiFi offloading that exploits a user's delay tolerances for different applications and makes offloading decisions satisfying her throughput-delay tradeoffs and 3G budget constraints.
2) To enforce AMUSE's bandwidth allocation decisions for each application, we implement a practical receiver-side[2] rate control algorithm for TCP.
3) We surveyed 100 participants in the U.S. to evaluate users' tradeoff between the cost of 3G usage and their willingness to wait for WiFi access. We incorporate the resulting cost-throughput-delay tradeoff estimates into our model, and evaluate AMUSE's performance using both these results and 3G and WiFi usage data collected from 25 mobile users.

B. Components of AMUSE

Figure 1 gives an overview of AMUSE's components and architecture. AMUSE's User Interface interacts directly with the user, displaying the offloading decisions made as well as the user's app-level usage history. The user may also set her preferences, e.g., the maximum budget for 3G usage and delay tolerances for different applications. The Bandwidth Optimizer

[1]While our systems apply to any form of cellular data, e.g., 3G or LTE networks, we frame our discussion in terms of 3G data.

[2]We assume that download traffic makes up most of users' usage, so that the receiver is synonymous with the user.

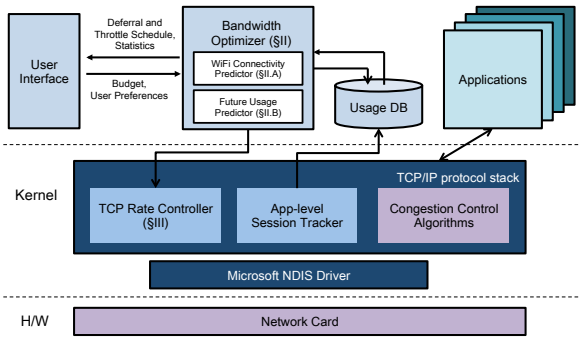

Fig. 1. System implementation architecture.

makes offloading decisions for the user, given the preferences set on the User Interface. It consists of three components: the Future Usage Predictor, the WiFi Connectivity Predictor, and a utility maximization algorithm. To calculate the expected utility of waiting for WiFi, the Bandwidth Optimizer predicts the probabilities of WiFi access at future times, along with the usage volume of different application sessions. The TCP rate controller on end-user devices enforces the 3G bandwidth allocations and offloading decisions made by AMUSE's Bandwidth Optimizer. The app-level session tracker measures the actual usage for each application. These usage data are then used to update AMUSE's prediction modules, and are displayed to the user on the User Interface.

II. Related Work

Recent studies of 3G and WiFi usage traces, e.g., [6], showed that offloading 3G traffic to WiFi can significantly benefit mobile ISPs. A similar study [7] of WiFi connectivity data showed that the high predictability of WiFi access improved the offloading of several example applications. Other works have proposed different practical offloading algorithms: Wiffler [8], for instance, proposes an online algorithm for offloading WiFi traffic. While Wiffler does consider different applications' delay tolerances, it does not take into account the available 3G bandwidth should WiFi not be available, in contrast to AMUSE. Win-Coupon [9] takes a slightly different perspective and proposes a reverse-auction scheme to incentivize users to offload their traffic so as to decrease the congestion experienced by all users.

One of AMUSE's unique features is its use of receiver-side TCP advertisement windows to control application-specific 3G bandwidth from the user side. While several commercial applications (e.g., [10], [11]) provide user-side application rate control, most require users to manually specify the desired rates. AMUSE provides automated bandwidth rates and, by conforming to TCP interactions, avoids the TCP timeouts common to existing user-side rate control applications.

III. Bandwidth Optimizer

In this section, we describe the individual components of AMUSE's bandwidth optimization algorithm. In the discussion below, we first introduce practical algorithms to predict WiFi access and application-specific usage (Sections III-A and III-B). We then incorporate these predictions into a mathematical allocation framework in Section III-C and propose a heuristic algorithm for computing AMUSE's bandwidth allocations and offloading decisions in Section III-D.

To consider a user's different delay tolerances on different applications, we group a user's traffic into different application types, e.g., streaming, browsing, and downloads. For practical implementability, we assume that only the most heavily used (e.g., top five) applications are considered, and denote these collectively as a set J. We suppose that the day is divided into n discrete periods of time, e.g., 24 hours, and for each period, we predict both WiFi access and application usage volumes.

Given these predictions, a user must decide which applications to offload when, subject to a maximum 3G usage budget. By delaying sessions to future periods, users may gain WiFi access and the ability to offload; however, if WiFi is not available, the user must send these sessions over 3G, which has a finite bandwidth capacity that must be shared among the different application types. AMUSE therefore computes a 3G bandwidth allocation when deciding whether to wait for WiFi. We formulate this decision as a multiple choice knapsack problem, and propose a heuristic solution algorithm.

A. Predicting WiFi Connectivity

Since WiFi availability is heavily location-dependent, we predict the probabilities of WiFi access by combining user location prediction with the probabilities of WiFi access at different locations. We use a training set of empirical WiFi access data to estimate time-dependent WiFi access probabilities at each location, which are updated over time. For a location l, we denote the probability of WiFi access during period k as $v_k(l)$. We use L_k to denote the set of observed locations in period k.

Given the time- and location-specific probabilities $v_k(l)$, we then predict *overall* WiFi access by incorporating predictions of users' future locations. We define w_k to be the overall WiFi probability in period k. We use a second-order Markov chain for the location prediction, which has been shown to be highly accurate [12]. Algorithm 1 summarizes the calculation of overall WiFi access probabilities. We use the notation $p_l^{k+2}(l_k l_{k+1})$ to denote the probability that a user is at location $l \in L_{k+2}$ during period $k+2$, given his locations l_k in period k and l_{k+1} in period $k+1$. To calculate these p^k, we define $N^k(s)$ as the number of times that s is observed, where s is a sequence of locations observed that ends in period k; the observed location in each period k is denoted by λ_k.

B. Predicting Future Usage

At the beginning of each day, we use previous data to predict the size $s_j(k)$ of each application type $j \in J$'s usage in each period k. To accommodate the dependence of session size on the amount of bandwidth allocated, our definitions of session "size" depend on the application: for fixed-volume application sessions such as downloads, in which the volume (MB) does not depend on the available bandwidth, we define the session size as its volume. For fixed-time sessions such as streaming, in which the volume does depend on the bandwidth, we define the size as the time to complete. We use J_v to denote the set

Algorithm 1: Computation of WiFi access probabilities over the rest of the day in period i.

if $i = 1$ **then**
  for $k \leftarrow 1$ **to** n **do**
    $w_k \leftarrow \sum_{l \in L_k} v_k(l) \frac{N^k(l)}{N}$, N is the number of days of data.
    `// Calculate WiFi probabilities for the next n periods.`
if $i > 1$ **then**
  for $k \leftarrow 2$ **to** n **do**
    forall the $l \in L_k$, $l_{k-1} \in L_{k-1}$, $l_{k-2} \in L_{k-2}$ **do**
      if $N^{k-1}(l_{k-2}l_{k-1}) > 0$ **then**
        $p_l^k(l_{k-2}l_{k-1}) \leftarrow \frac{N^k(l_{k-2}l_{k-1}l)}{N^{k-1}(l_{k-2}l_{k-1})}$
      else
        $p_l^k(l_{k-2}l_{k-1}) \leftarrow \frac{N^k(l_{k-1}l)}{N^{k-1}(l_{k-1})}$
    $w_k \leftarrow \sum_{l \in L_k} p_l^k(\lambda_{k-2}\lambda_{k-1}) v_k(l)$

of fixed-volume application types, and J_t the set of fixed-time application types.

We estimate the future usage $s_j(k)$ by taking a moving average of the observed usage sizes $\sigma_j(k)$ of application j in period k over some fixed number of days. In updating our usage estimates, we modify the moving-average calculation to take into account our deferral recommendations. Since a user may delay application usage to another time in order to offload it to WiFi, we "shift the usage back" in order to evaluate and detect changes in the underlying usage pattern over the day.

C. User Utility Maximization

1) Utility Functions: To mathematically formulate the user's offloading decision problem, we need a concrete measure of the user's tradeoffs between cost, throughput, and delay. Thus, for a given application type j in period i, we derive expressions for users' *utility* of completing those application sessions over 3G and over WiFi. This utility is determined by the per-volume price p of 3G, the amount of time t the session is deferred, the bandwidth speed r at which the session is completed, and the size s of the session. We use $U_j(p, t, r, s)$ to denote the utility of application $j \in J$.

To derive reasonable utility function estimates, we conducted an online survey of over 100 users, primarily students, faculty and staff from U.S. universities. For each application (email, browsing, video, social networking, downloads), we gave participants the cost to complete one application session over 3G, as well as the speed of WiFi relative to 3G. We then asked the participants how long they would wait for WiFi access instead of immediately completing the session over 3G.

We find that the following functional form provides a good fit for our data:

$$\begin{cases} U_j(b,t,r,s) = C_j \exp\left(-\nu + r\nu - \mu t\right) - \eta b & j \in J_t \\ U_j(b,t,r,s) = C_j \exp\left(-(s/r)\nu - \mu t\right) - \eta b & j \in J_v, \end{cases} \quad (1)$$

where U_j denotes the parameterized utility function for application types j; b denotes the cost of each session (ps if $j \in J_v$ and prs if $j \in J_t$); and C_j, μ, ν, and η are nonnegative parameters that depend on j. For ease of notation, in the discussion below we denote the utility as $U_j(b, t, r, s) = U_j(p, t, r, s)$.

2) Users' Optimization Problem: We now use the utility functions (1) to formulate the user's optimization problem. To represent possible 3G and WiFi bandwidth speeds, we normalize the volume units so that the fixed per-second WiFi speed equals 1. The 3G speed γ is chosen from a finite subset of possibilities Γ. For each $\gamma \in \Gamma$ and period $k \geq i$, we define the indicator variables $c_i^j(k, \gamma)$ to be 1 if a session of type j is deferred from period i to period k and assigned 3G speed γ, and 0 otherwise.

We obtain the optimization problem

$$\max_{c_i^j(k,\gamma)} \sum_{i=1}^{n} \left[\sum_{j \in J} \left(\sum_{k \geq i} \left(\sum_{\gamma \in \Gamma} \Big(w_k U_j\big(0, k-i, 1, s_j(i)\big) + (1-w_k) U_j\big(p, k-i, \gamma, s_j(i)\big) \Big) c_i^j(k,\gamma) \right) \right) \right] \quad (2)$$

$$\text{s.t. } p \sum_{i=1}^{n} \left[\sum_{j \in J_v} \left(\sum_{k \geq i} \sum_{\gamma \in \Gamma} c_j^i(k,\gamma)(1-w_k) s_j(i) \right) + \sum_{j \in J_t} \left(\sum_{k \geq i} \sum_{\gamma \in \Gamma} c_j^i(k,\gamma)(1-w_k)\gamma s_j(i) \right) \right] \leq B \quad (3)$$

$$(1-w_l) \sum_{i \leq l} \sum_{j \in J} \sum_{\gamma \in \Gamma} c_j^i(l,\gamma)\gamma \leq (1-w_l)\beta \, \forall \, l \quad (4)$$

$$\sum_{k \geq i} \sum_{\gamma \in \Gamma} c_j^i(k,\gamma) = 1 \, \forall \, j \in J; \; i = 1, 2, \ldots, n \quad (5)$$

$$c_j^i(k,\gamma) \in \{0, 1\}.$$

The user's objective is to maximize the sum of the expected utility from WiFi and 3G in period k, over all (original) periods i and application types j. The constraint (5) ensures that each application j in period i is deferred to only one period k (we may have $k = i$), with 3G speed γ. This optimization is performed subject to two constraints: a budget constraint on expected 3G usage (3), and capacity constraints on the 3G bandwidth in each period (4).

We assume that the user specifies a maximum monthly budget $\overline{B}$ for 3G usage. We then calculate a *daily budget* B, taking into account both the number of days remaining in the month (denoted by m) and the amount of budget B_r that has not yet been spent. To allow the user some flexibility, we multiply the average usage B_r/m by the factor $\exp\left(1 - m^{-1}\right)$. The 3G bandwidth capacity constraints (4) ensure that the sum of the bandwidth allocated to each application in a given period does not exceed the fixed maximum bandwidth β.

D. Online Algorithm

Algorithm 2 presents our online algorithm, along with the WiFi and app usage predictions (Sections III-A and III-B). While various algorithms exist to compute solutions of the knapsack problem, we use a Lagrange-multiplier based solution [13] that has relatively small computational overhead and

Algorithm 2: Bandwidth allocation over a day.

$i \leftarrow 1$ // The current period is denoted by i.
$B \leftarrow (B_r/m)\exp\left(1-m^{-1}\right)$ // Calculate the budget for the day.
Calculate WiFi probabilities using Algorithm 1 with $i = 1$.
Calculate predicted usage over all n periods using a moving average.
Allocate bandwidth by approximately solving (2-5).
for $i \leftarrow 2$ **to** n **do**
  $B \leftarrow B - S_{i-1}$ // Remaining daily budget, given the spending S_{i-1} in period $i-1$.
  Update WiFi probabilities using Algorithm 1.
  Update bandwidth allocations by re-solving (2-5) for the remaining $n-i+1$ periods.

generally returns good approximations to the optimum. At the beginning of each day, the user computes an initial solution, given estimates of the w_k and $s_j(k)$. As the user consumes data over the day, we update both the remaining daily budget B and our predictions of future WiFi connectivity $\{w_k\}$. The new optimization problem over the remainder of the day can then be solved by taking the existing solution as the initial point of our Lagrange multiplier algorithm.

IV. Implementation

We implemented an AMUSE prototype on Windows 7 tablets with the system architecture shown in Fig. 1. We used the Windows Filtering Platform (WFP) to track application usage and implemented a user-side TCP rate control algorithm to control each application's download rate. The AMUSE prototype displays both the total usage and the usage of individual applications on a daily, weekly, and monthly basis, as well as the current upload and download rates. We also provide user interfaces from which the user can, if he so chooses, set the download rate of each application, as well as his billing starting date and data plan (e.g., 2GB per month).

We use TCP ACK clocking to shape the incoming/downloading rates of TCP traffic, by modifying the TCP advertisement window size (rcv_wnd) field in each ACK packet using the WFP driver. While we could instead adjust the round-trip time (RTT) of each flow (i.e., stretching each ACK packet), that approach increases the overall response time. Modifying the advertisement window size does not increase the RTT of each flow, making it suitable for all TCP applications. Unlike current traffic control tools, our proposed control mechanism does not forcibly drop incoming packets, which can induce such undesirable side effects as frequent TCP timeouts.

Algorithm 3 presents the pseudo code of our implementation. The algorithm periodically calculates the traffic throughput for each application in each period. If the throughput for a given period is smaller than the target bandwidth ($target\_BW$), we increase the advertisement window size by an amount (inc) proportional to the deficit throughput. Similarly, if the throughput is larger than the target bandwidth, we decrease the size of the advertisement window by an amount proportional to the surplus throughput (- inc). Depending on the increase/decrease of the advertisement window, the TCP sender will increase or decrease the rate of the traffic accordingly. Here, we multiply the deficit/surplus bandwidth

Algorithm 3: Receiver-side TCP rate control.

Initialization:
  $target\_BW \leftarrow$ // Desired bandwidth (bytes/sec)
  $min\_adv\_win \leftarrow 512\ (bytes)$
  $adv\_win \leftarrow min\_adv\_win$
  $last\_check\_time \leftarrow current\_time\ (sec)$
  $check\_period \leftarrow 0.2\ (sec)$
  $bytes \leftarrow 0\ (bytes)$
// accumulated received bytes for current period
  $\alpha \leftarrow 0.5$ // smoothing factor
For each TCP data and ACK packet:
begin
  if *a data packet is received* **then**
    $bytes \leftarrow bytes + packet\_len$
    if $current\_time - last\_check\_time > check\_period$ **then**
      $interval \leftarrow current\_time - last\_check\_time$
      $throughput \leftarrow bytes/interval$
      $inc \leftarrow adv\_win * \frac{target\_BW - throughput}{target\_BW} * \alpha$
      $adv\_win \leftarrow adv\_win + inc$
      if $adv\_win > rcv\_buf\_size$ **then**
        $adv\_win \leftarrow rcv\_buf\_size$
      else if $adv\_win < min\_adv\_win$ **then**
        $adv\_win \leftarrow min\_adv\_win$
      $last\_check\_time \leftarrow current\_time$
      $bytes \leftarrow 0$
  if *an ACK packet is ready to be sent* **then**
    set the advertisement window of the ACK to adv_win

TABLE I
Basic rate control test using Iperf. Parentheses denote the standard deviations.

Target rate	100 Kbps	500 Kbps	1,024 Kbps
Ethernet	103.8 (0.42)	506.2 (0.42)	1031.2 (1.81)
WiFi	83.14 (3.63)	459 (6.46)	902.4 (21.67)
3G	95.28 (1.52)	474.7 (11.86)	896 (47.28)

by a ratio α, in order to reduce the oscillation of throughput due to the drastic window size changes. To verify Algorithm 3 in practice, we ran `Iperf` over Ethernet, WiFi, and 3G networks with various target bandwidths. The experimental results are shown in Table I.

V. Experimental Evaluation

To evaluate the effects of AMUSE's Bandwidth Optimizer (Algorithm 2) on users' offloading experience, we collected 3G and WiFi usage and mobility data from users. We then simulate the performance of AMUSE's bandwidth optimizer, taking the recorded usage data as the historical usage, and compare with two other known offloading algorithms.

We recruited 25 iPhone users to participate in our trial. We recorded participants' 3G and WiFi usage, WiFi availability, and user location at a ten minute granularity by implementing a usage monitoring app and installing it on their iPhones. The data were recorded for one week. Since some of our 25 users exhibited very similar traffic patterns, we choose eight representative users' data on which to run the AMUSE simulation. We use three days of data as each user's usage history, and run the simulation assuming hour-long periods. Users' monthly budgets for 3G data usage are chosen from a truncated normal distribution between $20 and $40. We compare AMUSE's performance to two baseline algorithms: on-the-spot offloading, which offloads only if WiFi is immediately available, and delayed offloading, which waits up to one hour for WiFi [6].

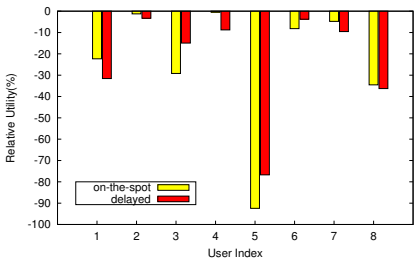

Fig. 2. Distribution of utility function values compared with AMUSE.

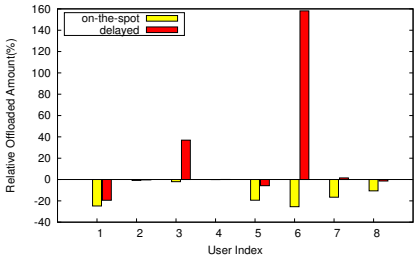

Fig. 3. Traffic offloaded compared to AMUSE.

Figure 2 compares each user's utility under AMUSE to that under our benchmark algorithms. For each user, both benchmark algorithms decrease the utility. On average, the utility of on-the-spot offloading is 24% less than that of AMUSE, while that of delayed offloading is 23% lower. AMUSE yields higher utility values than on-the-spot offloading due to offloading more traffic onto WiFi: Fig. 3 shows the amount of traffic offloaded under both benchmark algorithms, as compared to AMUSE. We see that for all users, the amount of traffic offloaded is larger with AMUSE than it is with on-the-spot offloading; thus, AMUSE leverages the delay tolerance of some sessions by allowing them to wait for WiFi access. Users then save money: Fig. 4 compares users' amount spent under the two benchmark algorithms to that spent with AMUSE. Users consistently spend over 20% more with on-the-spot offloading, and on average increase their spending by 37% compared with AMUSE.

Though AMUSE does offload some delay-tolerant traffic onto WiFi, it trades off between reducing users' spending by offloading traffic and completing some sessions immediately due to their intolerance of delay. Figure 3 shows that delayed offloading offloads more traffic than AMUSE for users 3, 6, and 7: AMUSE sends some sessions over 3G without waiting for WiFi, allowing users to spend more and delay less. On the other hand, delayed offloading offloads less traffic than AMUSE for users 1, 2, 4, 5, and 8: AMUSE allows delay-tolerant traffic to wait more than an hour for WiFi. We see from Fig. 4 that this additional wait for WiFi reduces these users' spending; the resulting gain in utility offsets the loss in utility from delaying the session.

VI. Conclusion

In this paper, we propose AMUSE, a cost-aware WiFi offloading system that maximizes the *end user's* utility under her 3G budget constraints. AMUSE consists of two main components: a bandwidth optimizer and TCP rate controller. By predicting future usage and WiFi availability, the bandwidth

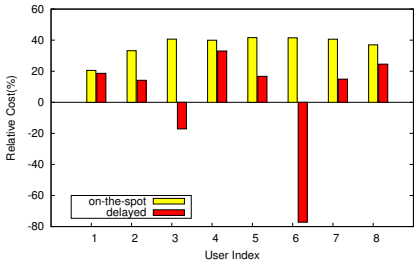

Fig. 4. Distribution of amount spent for different users.

optimizer chooses how long an application should wait for WiFi access, as well as a 3G data rate should WiFi not be available. These choices are optimized so as to balance the user's tradeoffs between the cost of sending an application over 3G, the higher throughput received over WiFi, and the delay inherent in waiting for WiFi. The TCP rate controller practically enforces the 3G rates chosen for each application by controlling the TCP advertisement window from the user side. We prototyped AMUSE on Windows 7 tablets and evaluated its performance with mobile traces from 25 users. Our results show that AMUSE can improve data users' utility.

Acknowledgments

The ICT at Seoul National University provides research facilities for this study. This work (Grants No.C0018176) was supported by Business for Cooperative R&D between Industry, Academy, and Research Institute funded Korea Small and Medium Business Administration in 2012. Carlee Joe-Wong was supported by the NDSEG fellowship. Part of the work reported here was supported by NSF CNS-1117126.

References

[1] "Cisco Visual Networking Index: Forecast and methodology, 2011 - 2016," May 2012, http://www.tinyurl.com/VNI2011.
[2] T. Ricker, "AT&T making tourists even more annoying with free Times Square WiFi," Engadget, May 2010, http://www.engadget.com/2010/05/25/atandt-making-times-square-tourists-even-more-annoying-with-free-w/.
[3] V. Chandrasekhar, J. Andrews, and A. Gatherer, "Femtocell networks: A survey," *IEEE Communications Mag.*, vol. 46, no. 9, pp. 59–67, 2008.
[4] "Smart Data Pricing Forum," July 31 2012, http://scenic.princeton.edu/SDP2012.
[5] S. Sen, C. Joe-Wong, S. Ha, and M. Chiang, "Incentivizing time-shifting of data: A survey of time-dependent pricing for Internet access," *IEEE Communications Mag.*, vol. 50, no. 11, pp. 91–99, 2012.
[6] K. Lee, I. Rhee, J. Lee, S. Chong, and Y. Yi, "Mobile data offloading: How much can WiFi deliver?" in *Proc. of ACM CoNEXT*, 2010.
[7] A. J. Nicholson and B. D. Noble, "BreadCrumbs: Forecasting mobile connectivity," in *Proc. of ACM MobiCom*, 2008.
[8] A. Balasubramanian, R. Mahajan, and A. Venkataramani, "Augmenting mobile 3G using WiFi," in *Proc. of ACM Mobisys*, 2010.
[9] X. Zhuo, W. Gao, G. Cao, and Y. Dai, "Win-Coupon: An incentive framework for 3G traffic offloading," in *Proc. of IEEE ICNP*, 2011.
[10] "NetLimiter," http://www.netlimiter.com/.
[11] "SoftPerfect Bandwidth Manager," http://www.softperfect.com/.
[12] L. Song, D. Kotz, R. Jain, and X. He, "Evaluating next-cell predictors with extensive Wi-Fi mobility data," *IEEE Transactions on Mobile Computing*, vol. 5, no. 12, pp. 1633–1649, 2006.
[13] M. Moser, D. Jokanovic, and N. Shiratori, "An algorithm for the multidimensional multiple-choice knapsack problem," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 80, no. 3, pp. 582–589, 1997.