Extend Your Journey: Introducing Signal Strength into Location-based Applications

Chih-Chuan Cheng and Pi-Cheng Hsiu
Research Center for Information Technology Innovation, Academia Sinica, Taipei 115, Taiwan, R.O.C.
Email: {charleycheng, pchsiu}@citi.sinica.edu.tw

***Abstract*—Reducing the communication energy is essential to facilitate the growth of emerging mobile applications. In this paper, we introduce signal strength into location-based applications to reduce the energy consumption of mobile devices for data reception. First, we model the problem of data fetch scheduling, with the objective of minimizing the energy required to fetch location-based information without adversely impacting user experience. Then, we propose a dynamic-programming algorithm to solve the fundamental problem and prove its optimality in terms of energy savings. We also provide an optimality condition with respect to signal strength fluctuations. Finally, based on the algorithm, we consider implementation issues. We have also developed a virtual tour system integrated with existing web applications to validate the practicability of the proposed concept. The results of experiments conducted based on real-world case studies are very encouraging.**

***Index Terms*—Energy-efficient optimization, signal strength, cellular data fetch scheduling, location-based applications**

I. Introduction

Recent years have witnessed a paradigm shift in personal computing. The popularity of mobile devices equipped with location-sensing technology has enabled the expansion of many existing information services by adding a location dimension. A variety of *location-based applications and services* have progressively permeated people's daily life, ranging from the services for directions or recommendations about nearby attractions to social interaction with friends via location sharing [19]. Location-based applications will become more diverse and pervasive due to the potential for a range of highly personalized and context-aware services [6]. However, the trend will lead to a significant boost in mobile data traffic and, consequently, result in further pressure on the limited battery capacity of mobile devices. Thus, reducing the communication energy is an imminent challenge in stimulating the development of emerging location-based applications.

Mobile applications can be classified into *delay-insensitive* and *delay-sensitive* applications. Delay-insensitive applications, like email and blogging, can afford to delay data transfer without significantly hurting user experience [3]. Various techniques that leverage this observation have been proposed to trade off transfer latency for energy savings. *Prediction-based offloading* predicts the future availability of WiFi connectivity, and delays data communications via 3G until WiFi becomes available if the delay is tolerable [2, 16]. The rationale behind the energy gains is that, compared to 3G, WiFi consumes much less energy per bit for data communications [15]; thus, WiFi is employed to improve the energy efficiency when available. On the other hand, it has been observed that a large fraction of the energy for data communication in 3G is wasted on the *tail energy* [4]. This phenomenon is resulted from that 3G does not switch from the high to the low power state immediately after the completion of each communication; instead, it waits for a period of time to alleviate the switching overhead, in case another communication will occur before long. The approaches that delay data transfer to reduce cumulative tail energy could be categorized into *batch scheduling* [5, 10] and *fast dormancy* [12, 13].

Delay-sensitive applications, such as video streaming and web mapping, in mobile environments are extremely sensitive to delay and thus rely on ubiquitous wireless connectivity [3]. For energy-efficient ubiquitous connectivity, the approach proposed in [15] leverages the complementary characteristics of WiFi and 3G by selecting the more efficient one based on context information. In [3], a fast switching mechanism was developed to enable fast switching to 3G once WiFi fails to deliver delay-sensitive data within the application's delay tolerance. In [14], a theoretic framework for exploring energy-delay tradeoffs was introduced to achieve balance in a wide spectrum between energy efficiency and data delay. Basically, these approaches rely on WiFi to improve energy efficiency and 3G to maintain ubiquitous connectivity.

Recently, it has been observed that signal strength has a direct impact on the communication energy consumption. The communication energy per bit when the signal is weak could be as much as six times more than that when the signal is strong [17]. This phenomenon has proved evident in both WiFi [11] and 3G [17]. The reason for such a phenomenon results mainly from the adaptive modulation and power control employed in wireless network systems. Based on the observation, it could be promising to exploit signal strength information to reduce the communication energy of mobile devices. In [17], a threshold-based approach was proposed for delay-insensitive applications, like email syncing, by deferring communications until a location with better signal strength. For delay-sensitive applications like video streaming, a schedule-based approach was presented, in a conceptual manner, to assign video frames into time slots (associated with different signal strengths) such that the communication energy is minimized. However, the approach relies on an accurate estimate of signal strength, and little attention has been paid to the impact of signal strength fluctuations on the feasibility of the derived solution.

In this paper, our major contribution is to introduce signal strength into location-based applications to reduce the energy

consumption of mobile devices for data reception. To validate the practicability of the concept, we developed a virtual tour system comprised of an on-line server and a mobile application program based on Android. Note that the concept, once proved practicable, could be extended and applied to other variants of location-based applications. First, we model the fundamental problem in the virtual tour system as a data fetch scheduling problem. Second, we propose a dynamic-programming algorithm to solve the fundamental problem. The solution involves scheduling the fetching of location-based information at appropriate locations so as to minimize the total energy consumption, without adversely impacting user experience in location-based applications. We prove that the algorithm is optimal in terms of energy savings and provide an optimality condition with respect to signal strength fluctuations. Third, we discuss technical implementation issues that arise when introducing signal strength into location-based applications for energy savings. Finally, we conducted a series of experiments in Taipei City for real-world case studies. The results show that an Android smartphone of HTC EVO 3D can achieve a significant energy reduction when accessing location-based applications. In addition, the observations based on the experiments provide further insights into improving energy efficiency of location-based applications by exploiting signal strength information.

The rest of this paper is organized as follows. Section II describes the system model and defines the problem. In Section III, we propose an optimal algorithm to solve the problem and provide an optimality condition with respect to signal strength accuracy. In Section IV, we discuss some system implementations. The experimental results are reported in Section V. Section VI contains some concluding remarks.

II. System Model and Problem Definition

Suppose that a mobile user carrying a smart mobile device is traveling from a source to a destination. Along the route, the user is provided with location-based information, such as local maps and directions, based on the geographical locations. In addition, the user may also be provided with entertainment information of interests, like travel recommendations. That is so called *social location-based services*. Recently, *augmented reality techniques* have also been introduced into location-based applications, like Layar, to further improve user experience. Such information is comprised of *objects*, e.g., map tiles, photos, and video clips. Some objects are requested by the user on demand and, thus, force the user to wait for immediate communication[1]. In contrast, some objects are supposed to be displayed along the route and fetched automatically.

Let the objects to be displayed along the route be denoted by $O = \{o_1, o_2, \cdots, o_m\}$. In practical implementations, objects are usually of various sizes and fetched in an atomic manner. Let each object $o_i \in O$ be of size s_i (bytes). Moreover, to avoid the overheads incurred by maintaining which objects have been (or have not been) dispatched to certain users, objects are normally dispatched in some predefined order, referred to as the *dispatch constraint* in our scheduling problem.

[1]We do not consider the objects requested on demand because their energy consumption depends on whether, and where, the user downloads them.

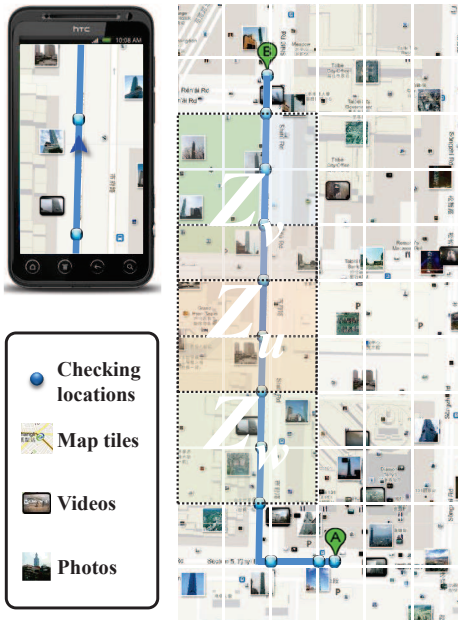


Fig. 1. An example location-based application

On the other hand, along the route exists a set of *checking locations*, $P = \{p_1, p_2, \cdots, p_n\}$, at which location-based information updates may be triggered. The checking locations are usually situated at the intersections of the route and the boundaries of map tiles, shown by the blue bullets in Figure 1. It is assumed that the user will visit the checking locations in order; otherwise, a new route is derived once the user deviates from the original route. At a checking location p_u, a *local scene*, Z_u, comprising some objects will appear on the device screen. Consequently, for the route, there is a corresponding set of local scenes $\mathbb{Z} = \{Z_1, Z_2, \cdots, Z_n\}$, where two adjacent local scenes may be overlapped. To ensure user experience, the corresponding local scene should be available before the user arrives at each checking location, except the source whose local scene is deemed to be requested on demand. Thus, the *availability constraint* is imposed on our scheduling problem. Finally, the time required to download a given amount of information depends on the downlink data rate. Because the data rate has its limitation and the user will pass through a checking location in a finite time, the amount of information that can be fetched at a location is also limited. Thus, each checking location should be associated with a maximum fetch size, reflected by the *fetch constraint* in our scheduling problem.

It requires time and consumes energy to fetch location-based information. It has been noticed that the data rate and the communication energy have strong relationship with signal strength. The relationship can be established with a monotonic function using regression-based techniques [20]. Figure 2 shows the downlink data rate (bytes per second), as well as the energy cost (joules per byte), with respect to signal strength measured based on an Android smartphone of HTC EVO 3D in practice. The signal strength may vary from location to location. Nevertheless, as observed in [17], the signal strength at a location is generally stable over time, which makes it

possible to measure and collect the signal strengths at the checking locations. As a result, the route is associated with a set of maximum fetch sizes, $C = \{c_1, c_2, \cdots, c_n\}$, and a set of energy costs, $E = \{e_1, e_2, \cdots, e_n\}$, each of which corresponds to a checking location. In addition to the energy consumption for data communication, there is a non-ignorable energy cost, known as tail energy and denoted as $\tilde{e}$, after each communication. More details about the estimation of energy costs and the determination of maximum fetch sizes will be discussed in Section IV. In this paper, we do not assume signal strength of one hundred percent accuracy. On the contrary, we prove that the algorithm is optimal in terms of energy savings if the accuracy satisfies the optimality condition given in Theorem 2.

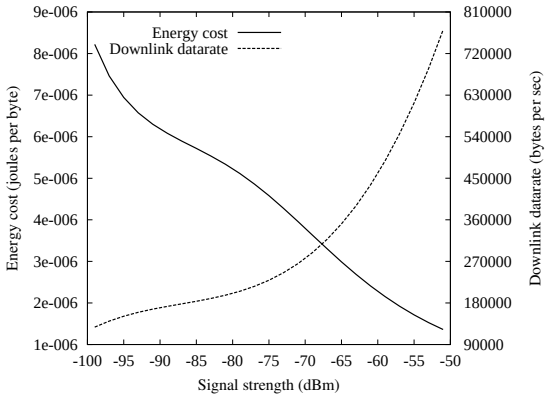


Fig. 2. Energy and data rate models

On the basis of the above descriptions, the communication energy required to fetch the same object may be different at different locations. This raises a technical problem: which objects should be fetched at which checking locations such that the total communication energy[2] is minimized, without adversely impacting the user experience? The solution involves scheduling the fetching of objects at appropriate locations. A mapping of a set of checking locations, P, to a set of objects, O, is called a fetch schedule $\sigma : u \in [1, n] \rightarrow \{o_i \in O\}$. A fetch schedule is *feasible* if the aforementioned scheduling constraints, namely the dispatch, availability and fetch constraints, are satisfied. Next, we formally define the *data fetch scheduling problem*:

Instance: A set of checking locations $P = \{p_1, p_2, \cdots, p_n\}$, where each location $p_u \in P$ is associated with an energy cost e_u, a maximum fetch size c_u, and a local scene Z_u; a set of objects $O = \{o_1, o_2, \cdots, o_m\}$, where each object $o_i \in O$ is associated with a size s_i, and a tail energy cost $\tilde{e}$ for each communication.

Objective: A feasible fetch schedule σ that minimizes the total energy consumption for data reception,

$$\sum_{\substack{\forall p_u \in P \\ \sigma(u) \neq \{\phi\}}} \left(\tilde{e} + \sum_{\forall o_i \in \sigma(u)} s_i \times e_u \right).$$

[2]Data retransmission is not considered in the problem definition because of its uncertainty. We assess its effect in real-world experiments in Section V.

III. Energy-efficient Cellular Data Fetch Schedule Optimization

This section proposes a dynamic-programming algorithm to solve the data fetch scheduling problem. First, we present the algorithm in Section III-A. Then, we analyze the time complexity of the algorithm and prove its optimality in Section III-B. Finally, in Section III-C, we provide an optimality condition with respect to signal strength accuracy to show the robustness of the algorithm.

A. Algorithm Description

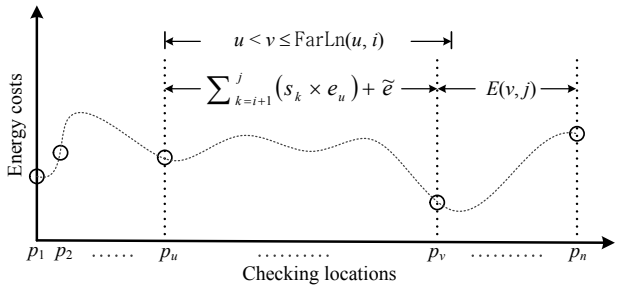


Fig. 3. An illustration to the dynamic-programming formula

The basis of the proposed algorithm is the recursive formula given in Equation (1). Before discussing the formula, we define some terms. Let $\texttt{LastOt}(v)$ denote the maximum subindex of the objects in local scene Z_v, i.e., $\texttt{LastOt}(v) = \max\{j \mid o_j \in Z_v\}$. Let $\texttt{FarOt}(u, i)$ be the maximum subindex of the objects that we can fetch at p_u from o_{i+1}, i.e., $\texttt{FarOt}(u, i) = \max\{j \leq m \mid \sum_{k=i+1}^{j} s_k \leq c_u\}$. Similarly, let $\texttt{FarLn}(u, i)$ be the maximum subindex of the checking locations reachable from p_u if we fetch as many objects as possible at p_u from o_{i+1}, i.e., $\texttt{FarLn}(u, i) = \max\{v \leq n \mid \sum_{k=i+1}^{\texttt{LastOt}(v)} s_k \leq c_u\}$. The recursive formula allows us to compute $E(u, i)$, which is defined as the minimum energy required to reach p_n from p_u when the first i objects have been available on the device already. Accordingly, our objective is to compute $E(1, 0)$. We delineate the three possible cases in Equation (1):

1) If $u = n$, then $E(u, i)$ is set at 0. That is, location p_u is the destination p_n; thus, no energy is required to fetch objects to reach p_n from p_u.
2) If $u \geq \texttt{FarLn}(u, i)$, then $E(u, i)$ is set at ∞. That is, it is impossible to reach a checking location farther than p_u, even if objects are downloaded from o_{i+1} sequentially until the maximum fetch size c_u is achieved. Thus, the energy consumption is set at ∞ to indicate there is no feasible fetch schedule to reach p_n from p_u.
3) Otherwise, if $u < \texttt{FarLn}(u, i)$, there exists a checking location p_v such that $u < v \leq \texttt{FarLn}(u, i)$, as shown by the example in Figure 3. Suppose that no objects will be fetched before p_v. To ensure that p_v is reachable from p_u, local scene Z_v should be available on the device before the user leaves p_u. Thus, if object o_j is the last one fetched at p_u, then the possible values of j should be in the range from $\max(i, \texttt{LastOt}(v))$ to $\texttt{FarOt}(u, i)$. We consider all the possible values to derive a candidate solution to $E(u, i)$. If $j = i$, it means

$$E(u,i) = \begin{cases} 0, & \text{if } u = n; \\ \infty, & \text{else if } u \geq \texttt{FarLn}(u,i); \\ \min\limits_{\substack{u<v\leq \texttt{FarLn}(u,i) \\ \max(i,\texttt{LastOt}(v))\leq j\leq \texttt{FarOt}(u,i)}} \left\{ \begin{array}{ll} E(v,i), & \text{if } j = i; \\ E(v,j) + \tilde{e} + \sum_{k=i+1}^{j} s_k \times e_u, & \text{if } j \neq i. \end{array} \right\} & \text{otherwise.} \end{cases} \quad (1)$$

that no objects are fetched at p_u. By definition, the minimum energy required to reach p_n from p_v, with the first i objects available on the device, is $E(v,i)$. On the other hand, if $j \neq i$, the energy consumption involves the energy required to fetch objects from o_{i+1} to o_j at p_u, i.e., $\sum_{k=i+1}^{j} s_k \times e_u + \tilde{e}$, and then the minimum energy required to reach p_n from p_v, i.e., $E(v,j)$. Note that the derived solution is based on the presupposition that no objects will be fetched before p_v. To relax the presupposition, by considering all the possible values of v, $E(u,i)$ is set as the minimum of the candidate solutions derived.

Algorithm 1

Input: A set of n checking locations P, where each p_u in P is associated with an energy cost e_u, a maximum fetch size c_u, and a local scene Z_u; a set of m objects O, where each o_i in O is of size s_i; and a tail energy cost $\tilde{e}$

Output: The minimum energy consumption for a feasible fetch schedule σ

1: **for** $u \leftarrow 1$ **to** n **do**
2: **for** $i \leftarrow 0$ **to** m **do**
3: $T[u,i] \leftarrow -1$
4: **return** $E(1,0)$

Procedure $E(u,i)$
5: **if** $T[u,i] \geq 0$ **then**
6: **return** $T[u,i]$
7: **if** $u = n$ **then**
8: $T[u,i] \leftarrow 0$
9: **else if** $u \geq \texttt{FarLn}(u,i)$ **then**
10: $T[u,i] \leftarrow \infty$
11: **else**
12: $T[u,i] \leftarrow \infty$
13: **for** $v \leftarrow u+1$ **to** $\texttt{FarLn}(u,i)$ **do**
14: **for** $j \leftarrow \max(i, \texttt{LastOt}(v))$ **to** $\texttt{FarOt}(u,i)$ **do**
15: **if** $j = i$ **then**
16: $T[u,i] \leftarrow \min(T[u,i], E(v,i))$
17: **else**
18: $T[u,i] \leftarrow \min(T[u,i], E(v,j) + \tilde{e} + \sum_{k=i+1}^{j} s_k \times e_u)$
19: **return** $T[u,i]$

Algorithm 1 implements the dynamic-programming formula in Equation (1) recursively. Once derived, the solution to each subproblem $E(u,i)$ is stored in a corresponding entry $T[u,i]$ in a two-dimensional table T. At the beginning of the algorithm, each table entry is initialized as -1 to indicate that the corresponding subproblem has not been solved yet (Lines 1-3). At the end, the algorithm returns the solution to $E(1,0)$, which involves Procedures $E(u,i)$ to be invoked recursively (Line 4). Whenever Procedure $E(u,i)$ is invoked, the procedure simply returns the previously derived solution stored in entry $T[u,i]$ if the entry has been updated (Lines 5-6). Otherwise, the solution to the subproblem is derived based on the presented dynamic-programming formula and returned (Lines 7-19).

Once the entire table has been derived, a corresponding feasible schedule σ can be constructed by tracing the table according to the dynamic-programming formula as follows: We begin with the table entry $T[1,0]$ by having indexes $u = 1$ and $i = 0$, and examine each entry $T[v,j]$, where $u < v \leq \texttt{FarLn}(u,i)$ and $\max(i, \texttt{LastOt}(v)) \leq j \leq \texttt{FarOt}(u,i)$, until we find an entry that minimizes the solution to subproblem $E(u,i)$. When such an entry, say $T[w,k]$, is found, we schedule objects, $o_{i+1}, o_{i+2}, ..., o_k$, to be fetched at location p_u. We then start with the discovered entry by having $u = w$ and $i = k$, and repeat the above process recursively until $u = n$. Because we have to determine object sets for n locations at most, and each determination takes $O(mn)$ time according to the time complexity analysis in Lemma 1, the construction of a feasible schedule σ based on table T can be completed in $O(mn^2)$ time.

B. Properties

In this section, we analyze the time complexity of Algorithm 1 and prove that it is an optimal algorithm for solving the data fetch scheduling problem.

Lemma 1: Algorithm 1 can be implemented to run in $O(m^2n^2)$ time.

Proof: First, we show how to implement the three functions, $\texttt{LastOt}()$, $\texttt{FarOt}()$, and $\texttt{FarLn}()$, such that each call to any of them takes constant time $O(1)$. For $\texttt{LastOt}()$, we construct a table by a linear search on the objects of every local scene, and then store the maximum subindex in a corresponding table entry. With the table constructed in $O(mn)$, each call to the function requires only $O(1)$ time. Similarly, two respective tables can be constructed, in a more elaborate manner, for the other two functions in $O(mn)$ time.

Then, the time complexity of the implementation depends on the number of table entries in T and the time required to derive the solution to a subproblem. The table contains $O(mn)$ entries, each of which is initialized and will never change once stored with the solution to the corresponding subproblem. The solution to each subproblem $E(u,i)$ is derived by referring to $O(mn)$ entries at most. When an entry $T[v,j]$ is referred to, deriving a candidate solution to $E(u,i)$ takes $O(1)$ time. Note that $\sum_{k=i+1}^{j} s_k$ can be computed in $O(1)$ time by simply adding s_j into $\sum_{k=i+1}^{j-1} s_k$, because the latter term has been computed when entry $T[v,j-1]$ was referred to. Therefore, deriving the solution to a subproblem takes $O(mn)$ time. In

summary, table T can be constructed in $O(m^2n^2)$ time. ■

Theorem 1: Algorithm 1 solves the data fetch scheduling problem optimally.

Proof: This theorem follows directly from the correctness of the dynamic-programming formula. We prove its correctness by mathematical induction on the distance to the destination p_n. As the induction basis, if the distance is 0, no energy is required to reach p_n from itself. Thus, the formula is correct. For the induction hypothesis, suppose that the formula is correct for any checking location closer to p_n than p_u is. We show that the formula is also correct for p_u.

Without loss of generality, we assume that the first i objects, for $1 \leq i \leq m$, have been available already. If $u \geq \texttt{FarLn}(u,i)$, there is no feasible fetch schedule to reach p_n from p_u, so the energy consumption is deemed to be ∞. Otherwise, let us consider any checking location p_v reachable p_n from p_u on the way to p_n, i.e., $u < v \leq \texttt{FarLn}(u,i)$. Suppose that no objects will be fetched before p_v and o_j is the last one fetched at p_u. To ensure that p_v is reachable from p_u and the fetch constraint is satisfied, $\max(i, \texttt{LastOt}(v)) \leq j \leq \texttt{FarOt}(u,i)$. We consider all the possible values of j to derive a candidate solution to $E(u,i)$. If $j = i$, it means that no objects are fetched at p_u, and the minimum energy required to reach p_n from p_u is equal to that required to reach from p_v. Since p_v is closer to p_n than p_u is, by the induction hypothesis, $E(v,i)$ is the minimum energy consumption. For the other cases, $j \neq i$, the energy consumption is the sum of the energy required to fetch the $(j-i)$ objects at p_u and the energy required to reach p_n from a closer location p_v, i.e., $E(v,j) + \tilde{e} + \sum_{k=i+1}^{j} s_k \times e_u$. By considering all the possible values of v, $E(u,i)$ is set as the minimum energy derived, and the theorem follows. ■

C. An Optimality Condition

The fetch schedule derived by Algorithm 1 is proved to be optimal with respect to the estimated signal strength. We do not expect that the estimated strength is exactly equal to the real signal strength. In this section, we consider signal strength fluctuations and give an optimality condition[3]. The rationale behind our analysis is as follows. Given an input instance I, where the energy costs and the maximum fetch sizes are set based on the estimated signal strength, Algorithm 1 can be applied to derive an optimal fetch schedule on I. Suppose that $\hat{I}$ is the input instance with respect to the real signal strength. If the schedule derived on I is also an optimal fetch schedule on $\hat{I}$, then the derived schedule remains optimal even if it is derived based on the estimated signal strength, not the real signal strength.

To derive the optimality condition, we construct a complete directed graph G based on the dynamic-programming formula when the formula is applied to instance I. G contains $n \times m$ vertices, each of which is a pair of (u,i) corresponding to a subproblem $E(u,i)$ in the formula. Every pair of vertices is connected by a symmetric pair of directed edges. Each edge from vertex (u,i) to vertex (v,j) has a weight $\delta^{(u,i)}_{(v,j)}$ representing the difference in energy consumption between $E(u,i)$ and $E(v,j)$. Specifically, $\delta^{(u,i)}_{(v,j)} = \infty$ if $v \leq u$ or $v > \texttt{FarLn}(u,i)$; otherwise, $\delta^{(u,i)}_{(v,j)} = 0$ if $i = j$, and $\delta^{(u,i)}_{(v,j)} = \tilde{e} + \sum_{k=i+1}^{j} s_k \times e_u$ if $i \neq j$. Clearly, each path in G corresponds to a fetch schedule on I, and the length of a shortest path from $(1,0)$ to (n,m) in G is equal to the amount of energy consumption $E(1,0)$.

Figure 4 provides a simple example which is constructed based on an input instance as follows. Consider six objects to be displayed on a route with four checking locations, where the respective sizes of the objects are 45, 60, 50, 40, 60, and 100 Kbytes, and the local scenes of the checking locations are $\mathbb{Z} = \{\{o_1\}, \{o_1, o_2\}, \{o_2, o_3, o_4\}, \{o_4, o_5, o_6\}\}$. Suppose, based on the estimated signal strength, that the energy costs at the four locations are $E = \{5.7 \times 10^{-6}, 5.4 \times 10^{-6}, 5 \times 10^{-6}, 5.1 \times 10^{-6}\}$ joules, and the maximum fetch sizes are $C = \{180, 190, 200, 194\}$ Kbytes. The tail energy cost is assumed to be 0.5 joules. In Figure 4, the two values on each edge represent the estimated weight and the real weight, respectively. For the sake of clarity, we do not draw the edges with an infinite weight and the vertices connected by only such edges, but draw the dashed edge as an example indicating that vertex $(4,6)$ is unreachable from vertex $(1,0)$. Let r^* denote the shortest path that corresponds to the fetch schedule derived by Algorithm 1. For example, $\mathrm{r}^* \equiv (1,0) \rightarrow (2,2) \rightarrow (3,4) \rightarrow (4,6)$ in G and its length is 3.38.

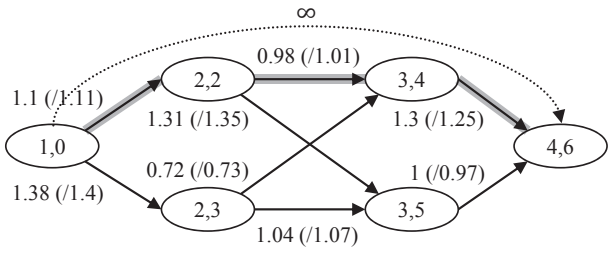


Fig. 4. Complete directed graphs G (/$\hat{G}$) constructed based on the dynamic-programming formula with respect to estimated (/real) signal strength

Next, we construct another complete directed graph $\hat{G}$ by replacing each weight $\delta^{(u,i)}_{(v,j)}$ in G with weight $\hat{\delta}^{(u,i)}_{(v,j)}$ computed based on the real signal strength. Let $\Delta\delta^{(u,i)}_{(v,j)}$ denote the difference between the real weight and the estimated weight, i.e., $\hat{\delta}^{(u,i)}_{(v,j)} = \delta^{(u,i)}_{(v,j)} + \Delta\delta^{(u,i)}_{(v,j)}$. Let $\hat{\delta}_{\mathrm{r}}$ and δ_{r} be the lengths of a path r in $\hat{G}$ and in G, respectively, and $\Delta\delta_{\mathrm{r}}$ be their difference, i.e., $\hat{\delta}_{\mathrm{r}} = \delta_{\mathrm{r}} + \Delta\delta_{\mathrm{r}}$. Moreover, let $\mathbf{R}$ be the set of all the possible paths from $(1,0)$ to (n,m) in the graphs. The following theorem gives an optimality condition of the proposed algorithm.

Theorem 2: The fetch schedule derived by Algorithm 1 based on the estimated signal strength remains optimal in the

[3]Note that even though the optimality condition is not satisfied, the accuracy of signal strength will only affect the amount of energy saved, not the user's experience if the scheduled objects can be fetched successfully at every checking location.

real environment if $\Delta\delta_{\mathrm{r}^*} - \Delta\delta_{\mathrm{r}} \leq \delta_{\mathrm{r}} - \delta_{\mathrm{r}^*}$, $\forall\ \mathrm{r} \in \mathbf{R}$.

Proof: In the theorem, r^* denotes the shortest path that corresponds to the fetch schedule derived by Algorithm 1. If the above optimality condition holds, then by simple substitution we have $\hat{\delta}_{\mathrm{r}^*} \leq \hat{\delta}_{\mathrm{r}}, \forall\ \mathrm{r} \in \mathbf{R}$. This derived inequality means that r^* is a shortest path in graph $\hat{G}$ with respect to the real signal strength. Thus, the fetch schedule corresponding to r^* remains optimal in the real environment, and the proof is completed. ■

Let us revisit the previous input instance. Signal strength only affects the estimation of the energy costs and the maximum fetch sizes of an input instance. Suppose that the real energy costs are $\hat{E} = \{5.8\times10^{-6}, 5.7\times10^{-6}, 4.7\times10^{-6}, 5.2\times10^{-6}\}$ joules, and the maximum fetch sizes should be $\hat{C} = \{169, 180, 247, 191\}$ Kbytes. Figure 4 shows the complete directed graph $\hat{G}$ constructed by replacing the estimated weights in G with the real weights. Compared with any other path from $(1,0)$ to $(4,6)$ in $\hat{G}$, path $(1,0) \rightarrow (2,2) \rightarrow (3,4) \rightarrow (4,6)$ of length 3.37 is a shortest path. The path is just the shortest path r^* from $(1,0)$ to $(4,6)$ in G. For this example, it can be examined that the optimality condition holds. Note that the length of r^* in G is not necessary equal to the length in $\hat{G}$. However, following the fetch schedule that corresponds to r^* will lead to the minimum energy consumption of 3.37 joules, which is optimal in the real environment.

IV. Implementation Remarks

In this section, we discuss some implementation issues that arise when introducing signal strength into location-based applications for energy savings. We have developed a virtual tour system that integrates Google Maps, Panoramio, and YouTube to provide tour information. In addition, it maintains a database that stores signal strength information. The collection of signal strength could be done in several ways, depending on the circumstances. In crowded downtowns, signal strength could be collected and updated frequently via application programs installed by mobile users, like OpenSignalMaps. The signal strength information in suburbs could be measured by street cars, like the way Google collects street views. For mountain areas, appropriate propagation models, along with the locations of base stations, could be adopted to estimate signal strength. In our prototype system, the second approach was adopted to measure the signal strength in two small-scale regions in Taipei City for case studies. Once this concept is embraced gradually by location-based service providers, some entities might have intention to collect and provide such information.

We have also developed an augmented reality application program for Android devices to access the tour guide service. When a user starts his journey, the program sends the server a request specifying the geographic coordinates of the source and destination, as well as the demanded types of information. Upon receiving the request, the server recommends a route to the user, and determines an optimal fetch schedule by running the proposed algorithm. With such a design, mobile devices are exempted from additional computational overheads, and the time required to determine a fetch schedule only ranges from dozens to hundreds of milliseconds on the server. Obviously, all the input parameters required by the algorithm can easily be acquired, except the energy costs and the maximum fetch sizes that are related to signal strength. In the remainder of this section, we focus on how we estimated the energy costs and determined the maximum fetch sizes at checking locations.

A. Energy Cost Estimation

The energy cost (joules per byte) at a checking location is defined as the mobile device's receiving power (watts) divided by the downlink data rate (bytes per second). The downlink data rate has a strong relationship with the signal strength. To plot their relationship, we installed the application program developed by OpenSignalMaps on an HTC EVO 3D smartphone to measure the signal strengths and data rates at a variety of locations in Taipei City. We gathered over 3000 pairs of such data within the coverage of 3G/3.5G signals provided by Chunghwa Telecom. Then, we applied the *polynomial regression method* [20] to the gathered data and modeled the relationship with a monotonic function, as shown in Figure 2. It is no doubt that the more (and diverse) the data gathered, the more accurate the monotonic function, and the less the effect due to signal fluctuation. Furthermore, we observed that the signal strength at a location is generally stable over time, which also agrees with the phenomenon observed in [17]. Based on our measurement, the signal strength at a checking location is close to the expected signal strength with a standard deviation up to 4 dBm, and the standard deviations are smaller than 2 dBm at most locations.

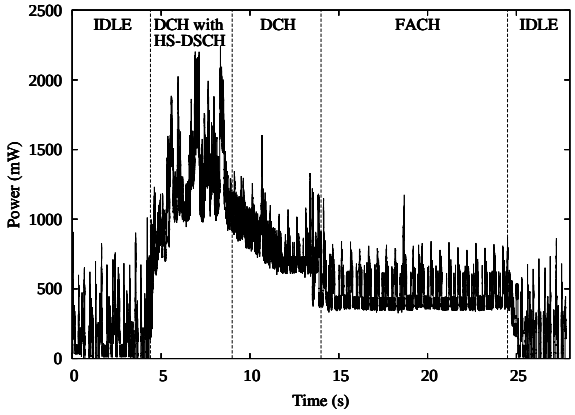


Fig. 5. The power consumption along with the state transition of 3G/3.5G

The receiving power depends mainly on the communication chip adopted by the mobile device. Fortunately, the accuracy of the power model will only affect the amount of energy saved, not the user experience; therefore, other device models could also benefit even if their accurate power models have not be acquired. The receive mode of 3G/3.5G has four/five states, and the state transition adheres to the *radio resource control protocol* included in the 3GPP standard. In practice, we used the power monitor produced by Monsoon Solutions to measure the receiving power of HTC EVO 3D smartphones. Figure 5 shows the receiving power of each state during an ICMP

ping. The radio interface is initiated in CELL_IDEL, which consumes almost no power. Then, it transits to CELL_DCH with HS-DSCH, a state supporting high-speed data downlink, and consumes 1050mW when remaining in the state. Thus, the energy cost at a location can be computed by dividing 1.05W by the downlink data rate there. After that, the interface starts to release the radio resources, resulting in a state demotion, and lasts in CELL_DCH with receiving power of 590mW until an inactive timer of 5 seconds expires. Again, the radio interface remains in CELL_FACH until another expiration of 12 seconds, and eventually returns to the idle state. The power consumption in CELL_FACH is 310mW. Thus, the tail energy cost can be computed by $0.59 \times 5 + 0.31 \times 12 = 6.67$ (joules).

B. Maximum Fetch Size Determination

The maximum fetch size (bytes) at a checking location is defined as the downlink data rate (bytes per second) multiplied by the time (second) the mobile device stays in the *effective region* covered by the same signal strength around that location. The effective region depends on the distance between the mobile device and the base station, surrounding obstacles, interference from other cells, among others [8]. To capture the changes of signal strength along a route, we installed a mobile application program, called RF Signal Tracker, to measure the signal strengths and their effective regions around the checking locations along the route. For example, the red circles in Figure 6(a) indicate the effective regions of the corresponding checking locations. Note that the effective radiuses may be different from location to location, varying from 19 meters to 75 meters, depending on the signal strengths and surrounding circumstances. Based on our measurement, the effective radiuses centered at checking locations along a route are 45 meters on average.

(a) Effective regions (b) The mobile platform

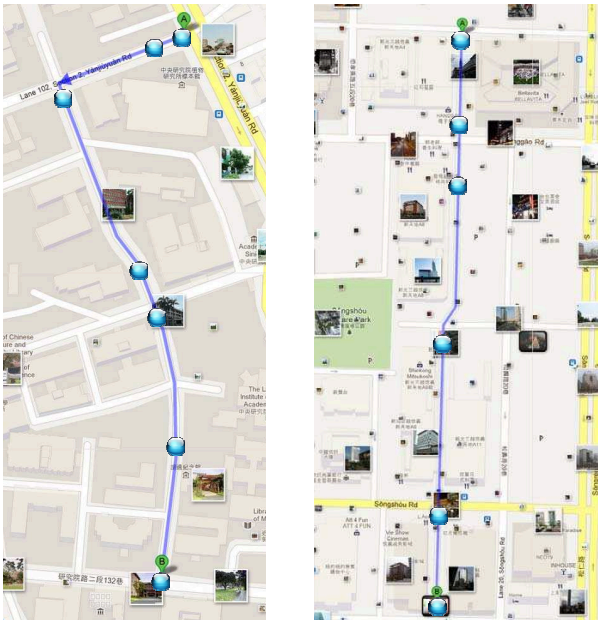
Fig. 6. Experimental equipments

The time a mobile device stays in an effective region depends on the user's movement speed. Based on some statistics, the velocities of pedestrians [9], bicyclers [7], and drivers [18] in urban areas are roughly 83, 216, and 667 meters per minute, respectively. In other words, the respective users would take roughly 65, 25, and 8.1 seconds to pass through an effective region with a radius of 45 meters for example. The maximum fetch sizes can be computed accordingly. To ensure that the scheduled objects can always be fetched successfully before the user leaves the effective region, we would prefer underestimating the maximum fetch sizes slightly. We simulate different velocities by varying the maximum fetch sizes and discuss the impact on energy savings in Section V.

V. Performance Evaluation

A. Experimental Setup

To demonstrate the efficacy of introducing signal strength into location-based applications, we conducted a series of experiments in Taipei City. In the experimental platform, shown in Figure 6, the mobile application program was installed on an Android smartphone of HTC EVO 3D, equipped with a 3.5G communication subsystem, and the energy consumption was measured by the Power Monitor of Monsoon Solutions. We evaluated the performance of the proposed algorithm, denoted as OPT, in terms of the energy consumption required for data reception. To show the performance gain, we compared OPT with the native approach, denoted as NATIVE, adopted by Google Maps. We also estimated the energy consumption based on our system model, denoted as OPT-THEORY, to gain further insights into the gap between theoretical estimation and experimental measurement.



(a) Route@campus (b) Route@downtown

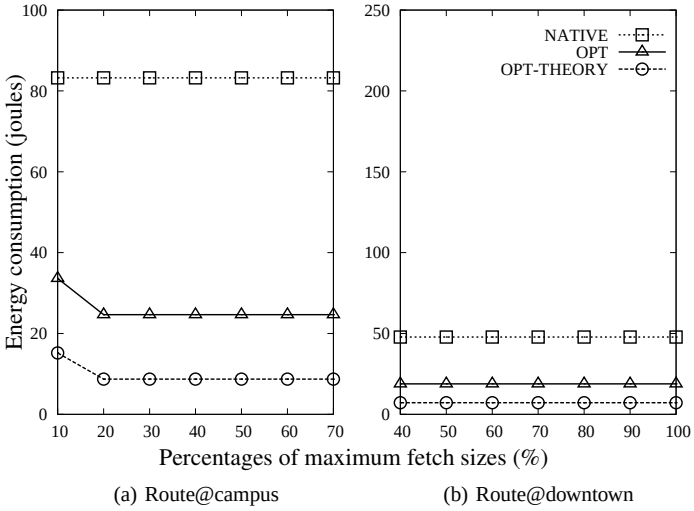
Fig. 7. Two real-world case studies in Taipei City

We investigated two routes with diverse characteristics, as shown in Figure 7. Route@campus is a path in Academia Sinica located in the suburb of Taipei City, while Route@downtown represents a crowded street in an urban area like the Xinyi District. The location-based information is relatively sparse along Route@campus (i.e., 54 objects including 24 maps tiles, 7 street views, 22 photos, and 1 video)

and dense along Route@downtown (i.e., 239 objects including 21 map tiles, 1 street view, 214 photos, and 3 videos). The respective sizes of a Google map tile (/street view), Panoramio photo, and YouTube video are roughly 10 (/30), 30, and 4000 Kbytes. To explore the impact of the amount of information, we considered three scenarios. LBS3 provided all the three applications, Google Maps, Panoramio, and YouTube; LBS2 provided the first two and LBS1 provided only the first one.

The signal is relatively weak at the checking locations along Route@campus (i.e., -77, -75, -78, -86, -79, -91, and -91 dBm, with -82dBm on average) and strong along Route@downtown (i.e., -65, -72, -78, -76, -58, and -60 dBm, with -68dBm on average). The corresponding data rate at each checking location was derived based on the model in Figure 2, and the maximum fetch size was computed based on the corresponding effective region and the velocity of pedestrians (83 meters per minute). To simulate a variety of velocities (e.g., bicyclers and drivers), we also underestimated the maximum fetch sizes and explored the impact of different percentages of underestimation.

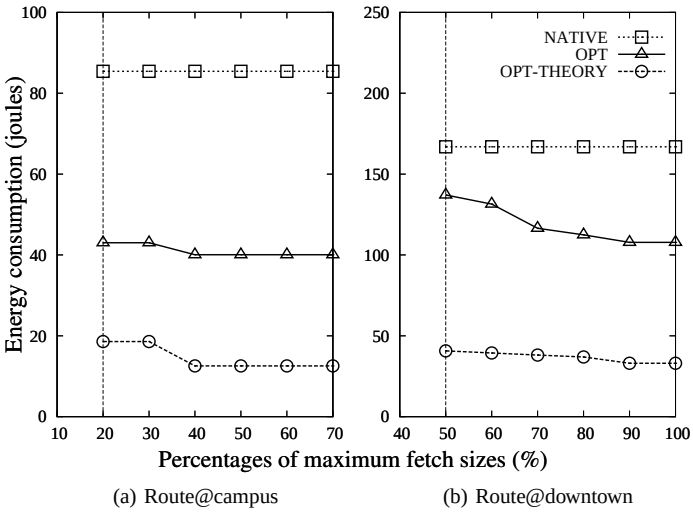
B. Experimental Results



(a) Route@campus (b) Route@downtown

Fig. 8. Energy consumption required by LBS1

Figures 8(a) and 8(b) show, respectively, the energy consumption required by LBS1 along Route@campus and Route@downtown. The maximum fetch sizes have no impact on the performance of NATIVE, because it always fetches the corresponding objects when a local scene is needed. In contrast, for OPT, it is expected that the energy consumption will decrease as the maximum fetch sizes increase, because larger fetch sizes imply more flexibility. Interestingly, the results are not exactly as expected. It is because LBS1 comprises only a few small objects. Consequently, most objects can be fetched at a few specific locations with stronger signal. This also explains why the energy consumption required by LBS1 is less along Route@downtown than along Route@campus. The results show that, with the information of signal strength, OPT reduces the communication energy by 59-70% along Route@campus and about 61% along Route@downtown, compared to that consumed by LBS1 under NATIVE. We also observed a gap that cannot be ignored between OPT and OPT-THEORY. One reason for the gap is that the measured signal strength and energy model could not be one hundred percent accurate. Other reasons could include the overhead of packet headers, the round-trip time of requests, and the retransmission of data, which are not considered in our system model. Finally, there are feasible schedules even if the maximum fetch sizes are very small. It implies that LBS1 is applicable to mobile users with a variety of velocities.

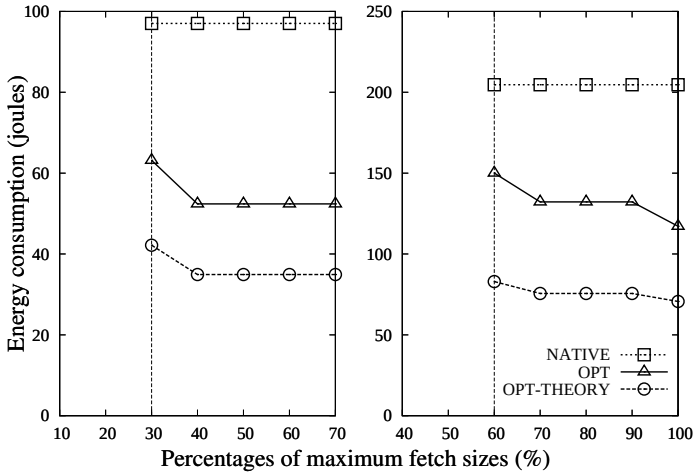


(a) Route@campus (b) Route@downtown

Fig. 9. Energy consumption required by LBS2

Figures 9(a) and 9(b) show the respective energy consumption required by LBS2 along the two routes. The energy consumption under OPT decreases as the maximum fetch sizes increase, and the decrease is more manifest for Route@downtown than for Route@campus. This is because, along Route@downtown, LBS2 comprises many objects and the signal strength varies more significantly. When the maximum fetch sizes are larger, OPT has more flexibility to schedule objects based on signal strength, resulting in more energy savings. In addition, there are much more objects along Route@downtown than along Route@campus; thus, the energy consumption is more along Route@downtown even if the signal is stronger. We observed a large discrepancy between OPT and OPT-THEORY. The main reason is that LBS2 contains many photos, and each request for a photo requires a round-trip time during which the radio interface remains in the high power state. The discrepancy could be reduced by combining the photos of a local scene into a large object from the perspective of system implementation, or by considering the round-trip time in the system model although the problem definition will become more complex. Despite the disregard of the round-trip time, OPT still achieves great energy savings. The results show that the communication energy is reduced by 49-53% along Route@campus and 18-35% along Route@downtown, depending on the maximum fetch sizes. Finally, as can be seen in the figures, no feasible schedule exists when the maximum fetch sizes are set at small. Thus, LBS2 might be more applicable to pedestrians

and bicyclers than drivers.



(a) Route@campus (b) Route@downtown

Fig. 10. Energy consumption required by LBS3

Figures 10(a) and 10(b) show the energy consumption required by LBS3 along the two routes, respectively. LBS3 contains some videos of larger sizes, in addition to maps and photos. As can be seen in the figures, the videos lead to a significant increase in the energy consumption. More energy savings, however, can be achieved by OPT if the videos can be scheduled to some locations more appropriate than where they are fetched under NATIVE. The results show that the communication energy is reduced by 35-46% along Route@campus and 27-43% along Route@downtown. Moreover, we observed that the gap between OPT and OPT-THEORY is significantly reduced, compared with that in Figure 9. It is because the additional energy consumption incurred by the round-trip time is amortized by the energy consumed by the videos. This also evidences the significant impact of the round-trip time on the communication energy. Finally, there will be no feasible schedule if the maximum fetch sizes are not sufficient to fetch videos. Thus, LBS3 might be applicable to only pedestrians.

VI. Conclusion

In this paper, we introduce signal strength into location-based applications to reduce the communication energy of mobile devices. The rationale behind the reduction is that the communication energy is much more when the signal is weak than when it is strong. To prove the concept, we have developed a virtual tour system, where the key technology is to schedule the fetching locations of objects based on signal strength, without adversely impacting user experience. We propose a dynamic-programming algorithm to derive optimal schedules in terms of energy savings, and provide an optimality condition with respect to signal strength accuracy. To evaluate the improvement in energy efficiency, we conducted a series of experiments along two routes of diverse characteristics in Taipei City. The results show that an HTC EVO 3D smartphone can achieve energy savings of 46-70% and 35-60% for pedestrian users along the two routes, respectively.

In the future, we will integrate Taiwan's signal database acquired from OpenSignalMaps [1] into our virtual tour system, and release the mobile application program in an Android marketplace to identify more issues in this research direction. We will also extend this concept to other variants of location-based applications.

Acknowledgement

This work was supported in part by the National Science Council under grant No. 101-2219-E-002-002.

References

[1] OpenSignalMaps. http://opensignalmaps.com/.
[2] G. Ananthanarayanan and I. Stoica. Blue-Fi: Enhancing Wi-Fi Performance using Bluetooth Signals. In *Proc. of the ACM MobiSys*, pages 249–262, 2009.
[3] A. Balasubramanian, R. Mahajan, and A. Venkataramani. Augmenting Mobile 3G using WiFi. In *Proc. of the ACM MobiSys*, pages 209–222, 2010.
[4] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani. Energy Consumption in Mobile Smartphones: A Measurement Study and Implications for Network Applications. In *Proc. of the ACM IMC*, pages 280–293, 2009.
[5] M. Calder and M. K. Marina. Batch Scheduling of Recurrent Applications for Energy Savings on Mobile Phones. In *Proc. of the IEEE SECON*, pages 1–3, 2010.
[6] S. Dhar and U. Varshney. Challenges and Business Models for Mobile Location-based Services and Advertising. *Commun. ACM*, 54(5):121–128, 2011.
[7] A. El-Geneidy, K. J. Krizek, and M. Iacono. Predicting Bicycle Travel Speeds along Different Facilities using GPS Data: a Proof of Concept Model. In *Proc. of the Annual Meeting of the Transportation Research Board*, 2007.
[8] H. Holma and A. Toskala. *WCDMA for UMTS: HSPA Evolution and LTE*, pages 175–221. John Wiley & Sons, Inc., 2007.
[9] R. L. Knoblauch, M. T. Pietrucha, and M. Nitzburg. Field Studies of Pedestrian Walking Speed and Start-Up Time. *Transportation Research Record*, pages 27–38, 1996.
[10] H. Liu, Y. Zhang, and Y. Zhou. TailTheft: Leveraging the Wasted Time for Saving Energy in Cellular Communications. In *Proc. of the ACM MobiArch*, pages 31–36, 2011.
[11] A. J. Nicholson and B. D. Noble. BreadCrumbs: Forecasting Mobile Connectivity. In *Proc. of the ACM MobiCom*, pages 46–57, 2008.
[12] F. Qian, Z. Wang, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. Characterizing Radio Resource Allocation for 3G Networks. In *Proc. of the ACM IMC*, pages 137–150, 2010.
[13] F. Qian, Z. Wang, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. TOP: Tail Optimization Protocol For Cellular Radio Resource Allocation. In *Proc. of the IEEE ICNP*, pages 285–294, 2010.
[14] M.-R. Ra, J. Paek, A. B. Sharma, R. Govindan, M. H. Krieger, and M. J. Neely. Energy-delay Tradeoffs in Smartphone Applications. In *Proc. of the ACM MobiSys*, pages 255–270, 2010.
[15] A. Rahmati and L. Zhong. Context-Based Network Estimation for Energy-Efficient Ubiquitous Wireless Connectivity. *IEEE Trans. on Mobile Computing*, 10(1):54–66, 2011.
[16] N. Ristanovic, J.-Y. Le Boudec, A. Chaintreau, and V. Erramilli. Energy Efficient Offloading of 3G Networks. In *Proc. of the IEEE MASS*, pages 202–211, 2011.
[17] A. Schulman, V. Navda, R. Ramjee, N. Spring, P. Deshpande, C. Grunewald, K. Jain, and V. N. Padmanabhan. Bartendr: A Practial Approach to Energy-aware Cellular Data Scheduling. In *Proc. of the ACM MobiCom*, pages 85–96, 2010.
[18] A. Studer, M. Luk, and A. Perrig. Efficient Mechanisms to Provide Convoy Member and Vehicle Sequence Authentication in VANETs. In *Proc. of the IEEE SecureComm*, pages 422–432, 2007.
[19] S. J. Vaughan-Nichols. Will Mobile Computing's Future Be Location, Location, Location? *IEEE Computer*, 42(2):14–17, 2009.
[20] S. Weisberg. *Applied Linear Regression*, pages 115–146. Wiley/Interscience, 3 edition, 2005.