12-2007

# Location Update versus Paging Trade-Off in Cellular Networks: An Approach Based on Vector Quantization

Abhishek ROY
*Samsung Electronics*

Archan MISRA
*Singapore Management University*, archanm@smu.edu.sg

Sajal K. DAS
*University of Texas at Arlington*

**DOI:** https://doi.org/10.1109/TMC.2007.1059

## Citation

ROY, Abhishek; MISRA, Archan; and DAS, Sajal K.. Location Update versus Paging Trade-Off in Cellular Networks: An Approach Based on Vector Quantization. (2007). *IEEE Transactions on Mobile Computing*. 6, (12), 1426-1440. Research Collection School Of Information Systems.
**Available at:** https://ink.library.smu.edu.sg/sis_research/658

Published in IEEE Transactions on Mobile Computing, Volume 6, Issue 12, December 2007, Pages 1426-1440.
http://doi.ieeecomputersociety.org/10.1109/TMC.2007.1059

# Location Update versus Paging Trade-Off in Cellular Networks: An Approach Based on Vector Quantization

Abhishek Roy, *Member*, *IEEE*, Archan Misra, *Member*, *IEEE*, and Sajal K. Das, *Senior Member*, *IEEE*

**Abstract**—In this paper, we propose two information-theoretic techniques for efficiently trading off the location update and paging costs associated with mobility management in wireless cellular networks. Previous approaches always attempt to accurately convey a mobile's movement sequence and hence cannot reduce the signaling cost below the entropy bound. Our proposed techniques, however, exploit the rate distortion theory to arbitrarily reduce the update cost at the expense of an increase in the corresponding paging overhead. To this end, we describe two location tracking algorithms based on spatial quantization and temporal quantization, which first quantize the movement sequence into a smaller set of codewords and then report a compressed representation of the codeword sequence. Although the *spatial quantization* algorithm clusters individual cells into registration areas, the more powerful *temporal quantization* algorithm groups sets of consecutive movement patterns. The quantizers themselves are adaptive and periodically reconfigure to accommodate changes in the mobile's movement pattern. Simulation study with synthetic and real movement traces for both single-system and multisystem cellular networks demonstrate that the proposed algorithms can reduce the mobile's update frequency to 3-4 updates/day with reasonable paging cost, low computational complexity, storage overhead, and codebook updates.

**Index Terms**—Location management, update, paging, spatial and temporal quantization, information theory.

## 1 Introduction

The goal of this paper is to design a common adaptive location management technique that enables each individual *mobile node (MN)* to tune its mobility-related signaling load under a unified cellular access infrastructure. Such individualized adaptation of the signaling load is becoming increasingly important due to heterogeneity at various levels such as 1) *devices* (automobiles, personal medical monitors, laptops, cell phones, and so on), 2) *converged applications* like voice-over-IP, instant messaging, on-demand TV, and so forth, and 3) *access technologies* like 3G cellular, 802.11, and WiMax. In particular, we desire an *MN* to able to *efficiently trade off* the overheads associated with the two fundamental mobility-related operations, namely,

1. *location update*, whereby the *MN* proactively reports its current coordinates to the network, and
2. *paging*, where the network searches for the *MN* in a set of cells to which its location uncertainty can be confined.

- A. Roy is with the WiBro System Lab, Telecommunication R&D Center, Samsung Electronics, 416 Maetan-3dong, Yeongtong-Gu, Suwon-si, Geyonggi-do, 443-742, South Korea. E-mail: abhishek.roy@samsung.com, abhi_uta@yahoo.com.
- A. Misra is with the Event-Based Systems Department, IBM T.J. Watson Research Center, 19 Skyline Drive, Hawthorne, NY 10532. E-mail: archan@us.ibm.com.
- S.K. Das is with the Department of Computer Science and Engineering, University of Texas at Arlington, PO Box 19015, Arlington, TX 76019. E-mail: das@cse.uta.edu.

We use the term "trade-off" since the paging and update costs are mutually conflicting: A decrease in one results in a nonlinear increase in the other. In general, the precise degree of trade-off depends on the application requirements and characteristics of individual *MN*s. For example, a car may choose a higher update rate to obtain a smaller call setup latency (via a smaller network paging load). On the other hand, a wearable device might aim for very low update costs to conserve its battery.

In this paper, we thus envision a *user-specific* location management framework where the update and paging policies adapt to the movement and calling patterns of individual devices. Unlike existing per-user update (for example, distance based [1], dead reckoning [24], and velocity based [42]) and paging schemes (for example, profile based [32]), we propose a model-independent *information-theoretic* framework that can be tied to fundamental mathematical bounds. Although per-user schemes are known to result in lower signaling overheads than conventional registration-area-based static schemes, their practical adoption in large networks has been stymied by concerns on both computational and storage complexities. As telecommunications technologies evolve, per-user schemes are expected to become more attractive, especially if they can significantly reduce the signaling load across billions of heterogeneous *MN*s. In the nearer term, intelligent per-user location tracking may be deployed in small and medium-sized cellular networks such as smart homes/offices and campus local area networks (LANs). This paper takes a step in this direction, theoretically establishing the performance benefits of *MN*-specific trade-off between paging and location update costs and quantifying the resultant computational and storage complexities.

## 1.1 Our Contributions

Based on the information theory, we present two algorithms that permit a close-to-optimal trade-off between an individual *MN*'s location update and paging costs by adapting these costs to the *MN*'s movement pattern. Our approach is *model independent* in the sense that it makes no a priori assumptions on the movement behavior of an *MN*. We simply view the update process as one where the *MN* communicates its movement history by using techniques for a *universal compression of individual sequences* [18] to reduce the signaling overhead for any *arbitrary* movement statistics. In particular, we generalize the LeZi-Update [7] scheme, where an *MN* uses the Lempel-Ziv (LZ78) algorithm [43] to develop an online codebook for its observed movement pattern, which is nothing but a stochastic random sequence, and transmits the encoded sequence to the network. LZ78 can asymptotically achieve the lower bound, called *entropy* (see [12] for details), associated with the random sequence. As a corollary of Shannon's result [38], it is impossible for an *MN* to accurately (that is, without introducing errors) transmit its actual movement sequence with an overhead smaller than this entropy.

However, for certain mobile sources, even the entropy bound may not be *low enough*. For example, an entropy bound of 20-30 updates/day may be a significant load for a device such as an Active Badge [22]. To reduce the *MN*'s update rate below the entropy bound, we propose that an *MN* exploit the rate distortion theory [12] by introducing *distortion* in the update process, that is, by transmitting only a *quantized* (or lossy) version of its true movement pattern. We present two related but distinct forms of quantization:

- **Spatial.** The *MN* groups contiguous cells into a spatial cluster and reports its movement pattern (via intermittent location updates) only at cluster-level granularity.
- **Temporal.** The *MN* adaptively groups the *movement sequences*, where each sequence consists of a set of consecutive cells visited by the *MN*. It then reports to the network the movement pattern at the granularity of these "pattern-groups."

In both of these schemes, the *MN* observes its own changing cellular coordinates, stores this movement history locally, and intermittently updates the network with a coordinate sequence (that is, a segment of its recent movement history). Under spatial quantization, each coordinate represents a spatial cluster similar to conventional registration areas (RAs), whereas, under temporal quantization (TQ), each coordinate represents a group of cell sequences. A novelty of our work is that the mapping from the raw location or movement sequences to the update coordinates *is itself dynamically modified in response to changes in the MN's movement behavior*. At the network end, the *MN's reported* movement sequence is stored efficiently in a user-specific data structure called a *trie*. To compute a paging sequence, the network uses this information to estimate the *MN*'s residence probabilities in various cells.

The rest of the paper is organized as follows: Section 2 reviews the relevant related work. Section 3 introduces the mathematical notations and principle of using an adaptive vector quantization (VQ) for location tracking. Section 4 focuses on the *spatial* quantization approach, whereby groups of cells are clustered into *RAs* for a single cellular network. Section 5 explains the *TQ* approach, where movement sequences over time are clustered into common codewords. Subsequently, Section 6 presents the modifications to the spatial and TQ approaches required for a multisystem cellular network. Section 7 presents the simulation and experimental results on the effectiveness of our algorithms in trading off between the paging and update costs. We study their performance by using both synthetically generated movement profiles and *real-life traces from an 802.11-based campus LAN* [13]. Section 8 concludes the paper with pointers to future research.

## 2 Related Work

There exists a great deal of prior work on efficient location update and paging schemes in cellular networks. In general, the update schemes can be classified as *local* (where the update logic is distinct for different *MN*s) and *global* (where all *MN*s perform updates in an identical fashion). Moreover, both classes of update schemes can be either *static* (where the update strategy does not evolve dynamically with time) or *dynamic*. The current personal communication systems (PCS) networks use a static global scheme where the network is divided into zones called RAs such that the location uncertainty of an *MN* is confined to its current RA. Blanket paging [34] is then used within the last reported RA to locate the *MN*. Clearly, the larger the size of an RA, the lower the update rate (or cost) and the higher the paging cost. The reporting-center-based approach [3] is another static global scheme where an *MN* proactively updates its location only when visiting certain designated cells.

For local update strategies, the simplest schemes require the *MN* to generate an update after a specific threshold in time, distance, or movement has elapsed. In the distance-based approach [1], an *MN* updates only after moving sufficiently away from its last updated cell. In the time-based approach [4], an *MN* updates its location periodically, whereas, in the fractional movement-distance scheme [21], the *MN* updates only after a fixed number of cell transitions outside the proximity of its last update. Variations of these approaches include the dynamic timer-based approach [1], [33]. (In [33], the time-period between successive updates is varied based on the average time between an *MN*'s cell crossings, whereas, in [1], the location update process attempts to trade off between the update and paging costs and thus varies based on the call arrival rate.) A dynamic programming formulation varying the time between successive location updates was proposed in [27] to optimize the *joint* expected paging and location update costs.

*Profile-based* approaches represent an important class of local dynamic update schemes, where the network uses the past history of an *MN* to build its movement profile. In particular, this approach of using short or long-term movement histories of an *MN* to modify both the update and paging strategies was first proposed in [40]. Subsequently, a concrete profile-based scheme was analyzed in [32], where the *MN* would update its location only if it visited a cell not within its current profile. This profile was constructed from its past movement history and stored inside the network, which could then compute the residence probability of the *MN* in different cells. The

LeZi-Update approach [7], upon which our work in this paper is based, can be considered as a *path-based* and *model-independent* implementation of the profile-based approach as it generates location updates only when the *MN*'s path since the last update turns out to be unique. Alternative model-based approaches for location update include the one where the *MN*'s velocity is used to adapt its update rate [25] or the one where the *MN*'s velocity is grouped into a set of different classes and updates are generated whenever the *MN* movement speed changes from one class to another [42].

To minimize the paging overhead, several sequential paging strategies have been proposed in the literature as an alternative to the current blanket paging approach. They include such techniques as recently registered cell first [26], in which the *MN* is first paged in the cell where it last generated an update, shortest distance first [23], in which cells are paged progressively in expanding concentric rings centered at the cell of last update, and those using residence probabilities to page more probable cells first [35]. An important result in [35] shows that, in the absence of delay bounds, the paging cost within a single network is minimized by searching cells in the decreasing order of the *MN*'s residence probabilities. Alternative approaches to paging involve the use of an *MN*'s location profile [32] or velocity information [8], [42] to alter the residence probabilities within the paging region. More recent user-specific paging algorithms include the sectional paging technique [41] that assumes knowledge of the *MN*'s movement direction and the concurrent paging [19] and Bloom-filter-based [29] techniques, where the last two try to optimize the simultaneous paging of multiple *MN*s (assuming a finite number of paging channels).

A comparative study of the trade-off between paging and update costs for different types of movement and call arrival patterns was first performed in [4]. For distance-based updates, a distance threshold of three cell crossings was shown to result in an "acceptable balance" between location update and paging costs. The performance of all the above schemes has also been analyzed in [5], which proposed a family of *topology-based* update strategies where the *MN*'s update decision depended only on its recent movement history. The D-MIP paging approach [9] is a rare adaptive paging scheme where an *MN* periodically computes its optimal paging area. The optimization is, however, static and aims at minimizing the cumulative paging and location update overhead. The trade-off between paging and update costs for a category of hierarchical location tracking strategies has been studied in [10] under the assumption of specific Markovian movement and calling patterns. More recent research has focused on minimizing the *sum* of the location and paging costs via optimal partitioning of cells into RAs. In [14], simulated annealing is used to derive an optimal clustering of cells into RAs. However, the optimization technique is offline and aims at minimizing the combined paging and location update costs. In contrast, our focus is to allow arbitrary trade-offs between paging and update costs.

The next section presents the proposed VQ-based framework for studying the trade-offs between the paging and update costs.

TABLE 1
Most Common Mathematical Symbols

| | |
|---|---|
| $X^n$ | Random infinite sequence representing $MN$'s movement ( $\mathcal{X}^n$: the sequence in a multi-system network) |
| $Y^n$ | A random infinite sequence representing the actual values sent to the network |
| $X_n$ | The $n^{th}$ element of sequence $X^n$ |
| $Y_n$ | The $n^{th}$ element of sequence $Y^n$ |
| $\mathcal{C}$ | Set of cells in a cellular network |
| $\mathcal{S}$ | Set of symbols (codewords) that an $MN$ transmits. |
| $N$ | Number of codewords in $\mathcal{S}$ (i.e., size of the codebook) |
| $RA_i$ | The group of cells mapped to the $i$th codeword $Q_i$. |
| $c$ or $c(i,j)$ | A cell in the network (with spatial coordinates $i$ and $j$) |
| $Q(c(i,j))$ | The codeword to which cell $c(i,j)$ is mapped; |
| $\mathcal{Q}^m$ | Codebook (quantizer) after $m$th iteration of Lloyd's algo. |
| $\varrho(c)$ | The $MN$'s residence probability in cell $c$ |

## 3 The Principle of VQ-Based Mobility Management

To understand our basic approach, let us first focus on the case of a single (homogeneous) cellular network consisting of a set $\mathcal{C} = \{a, b, c, \ldots\}$ of cells. By viewing these cells as symbols, it is easy to see that, as an *MN* moves across cell boundaries, its movement can be expressed as a random (possibly *infinite*) sequence $X^n = \{X_1, X_2, \ldots, X_n \ldots\}$ in the symbolic space, where each element $X_i$ is a symbol in the alphabet $\mathcal{C}$. Here, we assume that each cell in a provider network has a distinct identifier (in Section 7.8, we will discuss the resulting storage and computational overheads). For ease of reference, Table 1 lists the mathematical symbols most commonly used in this paper. A "bar" will usually represent the vector version of the corresponding symbol. For example, $\bar{RA}_i$ represents a set of movement sequences that are mapped to the $i$th codeword $\bar{Q}_i$, and $\bar{X}_n$ represents the vector-valued location of the *MN* (at time $n$) in a multisystem (heterogeneous) network.

We make only the very loose assumption that the *MN*'s movement is a *sample path* of an underlying random process with *some* (unknown and piecewise stationary) distribution. A key aspect of our update strategy is that the *MN* does not report each cell transition instantaneously; *rather, it reports subsequences of $X^n$ after varying intervals.* As a practical example, the sequence $\{a, b, a, c, b, c, \ldots\}$ implies that the *MN* actually moved from cell $a$ to $b$, back to $a$, and so on. The *MN* may update this sequence in batches as $\{(a, b), (a, c, b), c, \ldots\}$. The stochastic process $\chi$, which is a collection of all possible sequences $X^n$, is then associated with *an entropy bound* defined by

$$H(\chi) = \lim_{n\to\infty} H(X_n|X_1, X_2, \ldots, X_{n-1}),$$

where the entropy $H(\Lambda)$ of any discrete random variable $\Lambda$, with probability mass function $\varrho(\lambda)$, is defined by $H(\Lambda) = -\sum_{\lambda\in|\Lambda|} \varrho(\lambda)\log_2 \varrho(\lambda)$. As explained earlier, $H(\chi)$ is a lower bound on the signaling (in bits) needed on the average by an *MN* to convey the evolution of $X^n$ to a receiver (that is, the network controller) and is a measure of the "uncertainty" in the *MN*'s movement pattern. The LeZi-Update [3] algorithm uses the well-known, and widely used, adaptive LZ78 [43] compression technique to compress this cell-level movement sequence, thus achieving asymptotic closeness to the entropy bound $H(\chi)$.

We aim to reduce the *MN*'s update cost (the number of bits needed to inform the network of its movement) arbitrarily below the entropy bound. To reduce this overhead, the *MN* must transmit not the *real sequence* $X^n$ for $n \to \infty$ but, instead, some other *inaccurate* or *distorted* sequence $Y^n$. In practical terms, this means that the *MN* will actually not convey its true movement sequence $X^n = \{a, b, a, c, b, c, \ldots\}$ to the network but will instead update the network with an alternate movement sequence, say, $Y^n = \{a, c, c, c, b, a, \ldots\}$, called the *distorted* sequence, since some loss of accuracy occurs in replacing $X^n$ by $Y^n$. An element (or sequence) of $Y^n$ is also called the *codeword* for the corresponding element (or sequence) of $X^n$. Typically, the *MN* expresses multiple possible sequences with the same sequence $Y^n$. It is this reduction in the number of distinct patterns or symbols reported by the *MN* that lowers the update overhead. To determine the "best" symbol that should replace an element of $X^n$ resulting in $Y^n$, we first define a metric to quantify the cost of replacement. This metric is typically called the *distortion measure* and expresses the "distance" between the original sequence $X^n$ and the reported sequence $Y^n$ for any value of $n$.

**Definition 1.** *For any value of $n$, the distortion is measured as a distance between the original sequence $X^n$ and the reported sequence $Y^n$ through an appropriate function $d_n(.)$ that satisfies*

$$\begin{aligned} d_n(X^n, Y^n) &= 0, \text{ if } X^n = Y^n \\ &> 0 \text{ otherwise.} \end{aligned} \tag{1}$$

For our location management application, the distance function $d_n(.)$ refers to the sum of the distance between individual elements (cells) of $X^n$ and $Y^n$ (see Section 4.2).

Given any such distortion function $d(.)$ and the statistics of the stochastic process $\chi$, one can compute a *distortion rate function* $D(R)$. Intuitively, if the *MN* is constrained to transmit no more than $R$ bits per cell transition, then the distortion between any transmitted sequence $Y^n$, $n \to \infty$, and the real movement sequence $X^n$ is *at least* $D(R)$. Clearly, $D(R)$ is a decreasing function, with $D(H(\chi)) = 0$ implying that, if the update rate equals the *MN*'s entropy, then the network can receive the *MN*'s accurate movement history.

The *MN* can lower its update cost arbitrarily by choosing a progressively smaller $R$, thus incurring a larger $D$. To generate such a distorted sequence from $X^n$, the *MN* uses an approach based on VQ [20]. A vector quantizer $Q$ of dimension $k$ and size $N$ is a mapping from a vector (or "point") in $k$-dimensional space $\Re^k$ into a finite set $\mathcal{S}$ containing $N$ points, called *codewords*. A VQ implementation consists of an encoder ($\mathcal{E}$) that maps a point in $\Re^k$ to an indexed set $\mathcal{I}$ and a decoder that maps $\mathcal{I}$ to the set $\mathcal{S}$. We assume that both the transmitter ($MN$) and the receiver (network controller) have the same codebook. An *MN* first replaces its actual symbol sequence with the corresponding codeword and then transmits only the codeword's index: The network recovers the codeword by looking up the index in the same codebook.

**Definition 2 [20].** *The* basic rate ($R_Q$) *of a vector quantizer $Q$ is given by $R_Q = \log_2 N$, where $N = |\mathcal{S}|$ is the size of the codebook.*

Note that VQ is a very appealing and widely used technique due to the fundamental result [20] that for any given rate $R_Q$, as the block size $k \to \infty$, there exists a vector quantizer of dimension $k$ and size $N = 2^{R_Q k}$, with a resulting distortion very close to the ideal distortion rate bound $D_Q < D(R_Q) + \epsilon$ for any $\epsilon > 0$. For a more practical approach that reduces the complexity and yet achieves good performance for small values of $k$, we combine VQ with entropy (lossless) coding. Thus, the *MN* first generates $Y^n$, a sequence of codewords belonging to the set $\mathcal{S}$, and then transmits the LZ78- compressed representation of this sequence to the decoder (the network). It is well known [17] that the cascading of a vector quantizer with a conventional (lossless) entropy coding algorithm yields a transmission rate very close to the optimal rate distortion bound.

Our proposed *spatial* and *TQ* algorithms differ primarily in the choice of $k$. In spatial quantization, $k = 1$, and each independent location sample is quantized separately to generate the corresponding codeword. On the other hand, in TQ, $k > 1$, and multiple consecutive location samples are represented by a common codeword. To allow the quantizer to dynamically change in response to changes in the *MN*'s movement statistics, we shall use an adaptive version (AVQ) of the well-known Lloyd's *nearest neighbor* quantizer [20]. The Lloyd's algorithm is a standard iterative technique for VQ that converges by alternately 1) given a codebook, computing the partitioning of cells into codewords based on a *nearest neighbor* partitioning and 2) given a partitioning, defining a new set of codewords as the *centroid* of the current partitions.

In our schemes, the *MN* is thus responsible for four functions: tracking its own movement, storing its entire movement history efficiently (in compressed form) on its local disk, adaptively updating its quantizer, and transmitting the LZ-encoded updates to the network. The network controller receives the encoded updates, as well as modifications, to the quantizer codebook, and thus reconstructs and stores the *MN*'s (quantized) movement history.

## 4 Spatial Quantization Approach

Spatial quantization reduces the granularity of *each reported symbol* to a coarser value. Mathematically, the spatial quantization uses a VQ of dimension $k = 1$ such that each symbol is independently mapped to its nearest codeword. Each codeword can be viewed as a representation for a cluster of "adjacent" cells, similar to an RA. However, the number and set of cells that constitute an RA are adaptive, and the *MN* does *not* transmit an update at every RA crossing. As a practical illustration of this approach, assume that the set $\mathcal{C}$ of cells is partitioned into $N$ disjoint sets, where each set is the logical equivalent of an RA. Let $\mathcal{S}$ represent the quantized set $\{RA_1, RA_2, \ldots, RA_N\}$, where the quantization process $Q$ essentially maps each symbol in $\mathcal{C}$ to a unique symbol in $\mathcal{S}$. Let $Q(x) : x \to \mathcal{S}$ denote the mapping from a cell $x$ to its corresponding $RA$ and let $cell(RA_i)$ denote the set of cells comprising the RA $RA_i$.

### 4.1 Location Update

We present two slightly different versions of the spatial-quantization-based update algorithm (nonadaptive versions of such algorithms were presented in [37]). Fig. 1 illustrates a possible spatial quantization strategy, where $A$,

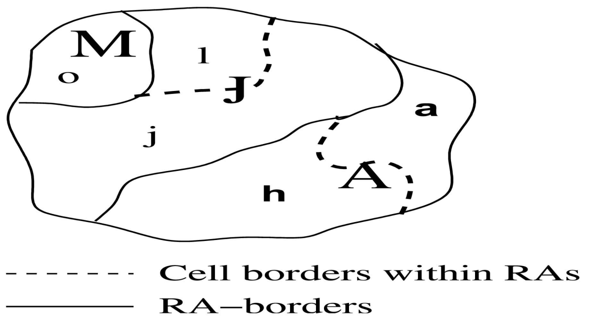


Fig. 1. Symbolic abstraction of cellular networks.

$J$, and $M$ are RAs consisting of cells $\{a, h\}$, $\{j, l\}$, and $\{o\}$, respectively.

In the first approach, called *LeZi-RA*, if the *MN*'s cell-level movement sequence is denoted by

$$''ajlojhajlojajlojajo\ldots,''$$

then the *MN* generates the quantized RA-level sequence

$$Y^n = ''A, J, J, M, J, A, A, J, J, J, M, J, A, J, J, M, J, A, J, M, \ldots''$$

The online incremental Lempel-Ziv parsing results in "$A, J, JM, JA, AJ, JJ, JMJ, AJJ, MJA, \ldots$" Eventually, the *MN* transmits the resulting sequence $Y^n$ as an optimally coded sequence "$C(\omega_1)C(\omega_2)C(\omega_3)\ldots$," where $\omega_i$s are nonoverlapping distinct segments of RA sequences and $C(\omega)$ is the encoding for segment $\omega$. At the network (receiver) side, this compressed sequence is decoded to recover the sequence $Y^n$. The network uses $Y^n$ to infer the *MN*'s movement and determine the paging sequence.

The second approach, called *RA-LeZi*, is motivated by the observation that an *MN* would typically move across several consecutive cells belonging to the same cluster. Thus, it removes identical consecutive symbols in the quantized sequence, generating new values for $Y^n$ only when the *MN* changes its current RA. Thus, the lossy quantization now results in the sequence

$$''A, J, M, J, A, J, M, J, A, J, M, J, A, J, M.''$$

A subsequent application of the incremental parsing results in the phrase sequence

$$''A, J, M, JA, JM, JAJ, MJ, AJ, M\ldots''$$

As in the *LeZi-RA* scheme, the LZ78 entropy coding scheme is then used to generate and transmit the codewords to the receiver side. Although the removal of consecutive identical (quantized) symbols reduces the update sequence, a subsequent LZ78 compression further reduces the number of bits transmitted by the *MN*.

Both of the above algorithms have been described under the assumption that the quantizer has already been defined; that is, the mapping of cells to the constituent RAs is already available at both the *MN* and the decoder. We now explain how this quantizer is formed and updated.

## 4.2 Efficient Quantizer Design

We first define a distortion function by assuming that the cells of the network are organized in any arbitrary layout, except that they are now identified by two coordinates $(i, j)$, where $i$ and $j$ refer to the coordinate values along the $x$ and $y$ axes. It is important to note that our coordinate system is virtual and does not assume or use any geometric location information available in the base stations. *We also do not assume the cells to be regular or have the same size, and we merely define a logical coordinate space that helps us cluster adjacent cells in the same RA.* As we would like the RAs to be compact, we shall arbitrarily define the distance (in this logical coordinate system) between cells $(i_1, j_1)$ and $(i_2, j_2)$ to be the euclidean distance. When a $c(i, j)$ is substituted by its quantized measure $Q(c(i, j))$, the resulting distortion $D_w(.)$ is then

$$D_w(c, Q_c) = \sum_{c \in Q_c} \Big[ (c(i) - Q(c(i)))^2 + (c(j) - Q(c(j)))^2 \Big] \times \varrho(c), \tag{2}$$

where $\varrho(c)$ is the *MN*'s residence probability in cell $c(i, j)$.

Both variants of our spatial quantization algorithm start with an initial codebook containing the set of RAs constructed by assuming a uniformly distributed residence probability in each cell. As the *MN*'s movement history becomes gradually available, the composition of *RAs* is *gradually modified* to reflect the true movement statistics. To make the time $(\tau)$ between two successive codebook modifications a function of the *MN*'s mobility pattern, the codebook is modified only when a new *LZ78*-coded update is actually generated by the *MN*. In other words, whenever the *MN* quantized location changes from $RA^n$ to $RA^{n+1}$ (thus generating the $Y_n$th element of $Y^n$), it checks if the sequence $Y_{k+1}, \ldots, Y_{n+1}$ (where $Y_k$ resulted in the previous update) has not been encountered before. If so, the *MN* must perform an update to the system, as well as recompute the quantizer codebook. Initially, when update sequences are small (that is, the user's profile is building up), the codebook gets frequently updated. After the codebook is adequately cognizant of the user's mobility profile, updates and codebook modifications occur with decreasing frequency.

The quantizer codebook is computed using the Lloyd's *nearest neighbor* algorithm [20], with the iterative process terminating when the percentage change in the cumulative distortion falls below a threshold $\epsilon$. The algorithm essentially clusters points in a 2D space, obtaining as the codeword the centroid of the corresponding Voronoi region (see [20] for details). Formally, the codebook is represented by the set of codewords $\{Q_\imath\}$, $i = 1, \ldots, N$, with all cells in the region such that $RA_\imath$ maps to the codeword $Q_\imath$ satisfying the following conditions:

$$RA_k = \{c : D_w(c, Q_k) \leq D_w(c, Q_l);\ \forall l \neq k\}, \tag{3}$$

where

$$Q_k(i, j) = \frac{\sum_{c \in RA_k} c(i, j) \times \varrho(c)}{\sum_{c \in RA_k} \varrho(c)}.$$

Intuitively, *(3) indicates that the ith codeword $Q_i(.)$ is the centroid of $RA_k$; that is, the $(x, y)$-coordinates of the codeword is a weighted mean of the $(x, y)$-coordinates of the constituent cells.*

Whenever there is a change in the encoder, the *MN* communicates the new codebook to the network, transmitting only the *changes to the Voronoi regions* for efficiency.

```
1. $m=1$, $D_w^0(Q_\imath^m)=0$, dictionary:= NULL, $\bar{w}_\imath :=$ NULL;
2. Construct an initial codebook $\mathcal{Q}_m=\{RA_\imath^m\}$;
3. loop
4.   Wait for symbol (cell) $x_\imath$;
5.   Quantize $x_\imath$ to get $Y_\imath = Q_\imath(x_\imath)$;
6.   if($\bar{w}.Y_\imath$ in dictionary)
7.     $\bar{w} := \bar{w}.Y_\imath$;
8.   else
9.     for (each cell $c$) do
10.        Estimate $\mathcal{Q}^{m+1}$ using centroid condition
           $Q_\imath(i,j) \approx \frac{\sum_{\forall\imath} c(\imath)p_c}{\sum_{R_\imath} p_c}$
11.        Find optimal partition using condition
           $RA_k = \{c : D_w(c,Q_k) \le D_w(c,Q_l); \forall l \ne k\}$
12.        $D_w^{m+1}(Q_\imath) = \sum_{k=1}^{N} \sum_{c\in cell(Q_k)} \{(c(i)-Q_k(i))^2$
           + $(c(j)-Q_k(j))^2\}p_c$;
13.        if $\left(\frac{D_w^{m+1}(Q_\imath)-D_w^m(Q_\imath)}{D_w^m(q_\imath)} \ge \epsilon\right)$
14.           Set $m := m+1$ and go to 5;
15.    end-for;
16.    Compute differential codebook-updates
       $\left(\delta_i^{m+1}, \Delta_i^{m+1}\right)$ using the two relations:
       $\delta_\imath^{m+1} = RA_\imath^{m+1} - RA_\imath^m$, $\Delta_\imath^{m+1} = RA_\imath^m - RA_\imath^{m+1}$;
17.    Transmit differential update to the decoder;
18.    Encode index $<\bar{w}>, Y_\imath$;
19.    add $\bar{w}.Y_\imath$ to dictionary;
20.    $\bar{w}$ := NULL;
21.  end-if;
22. forever;
```

Fig. 2. Encoder of spatial quantization.

Formally, let $RA_\imath^t$ represent a codeword region (RA) at time $t$ and let $RA_\imath^{t+1}$ represent the same region at time $t+1$, where $t$ denotes the instant when the codebook is modified. Then, the *MN* transmits the differential information ($\delta_\imath^{t+1}$, $\Delta_\imath^{t+1}$) of $RA_\imath$, where $\delta_\imath$ denotes the cells that newly map to the $\imath$th codeword, and $\Delta_\imath$ denotes the previous cells that no longer map to this codeword:

$$\begin{aligned}\delta_\imath^{t+1} &= RA_\imath^{t+1} - RA_\imath^t = \{c : c \in RA_\imath^{t+1} \text{ and } c \notin RA_\imath^t\} \\ \Delta_\imath^{t+1} &= RA_\imath^t - RA_\imath^{t+1} = \{c : c \in RA_\imath^t \text{ and } c \notin RA_\imath^{t+1}\}.\end{aligned} \tag{4}$$

Upon reception of ($\delta_\imath^{t+1}$, $\Delta_\imath^{t+1}$), the decoder can reconstruct its own RA information $RA_\imath^{t+1}$. As an example, consider the layout of Fig. 1. Now, assume that the Lloyd algorithm results in the new partitions given by $A = \{a, h\}$, $J = \{j\}$, and $M = \{l, o\}$. The encoder can then compute $\delta_M^{t+1} = \{l\}$ (since earlier, $M = \{o\}$) and $\Delta_J^{t+1} = \{l\}$ (since earlier, $J = \{j, l\}$) and transmit it to the decoder. The decoder refreshes the codebook to get $Q_M^{t+1} = \{o\} \cup \{l\} - \emptyset = \{l, o\}$ (and $Q_J^{t+1} = \{o\}$). Fig. 2 describes the LeZi-RA encoder (the RA-LeZi algorithm has an intermediate step, between steps 5 and 6, to remove the self-transitions). Fig. 3 describes the corresponding decoder logic.

### 4.3 The Paging Process

The system stores the received sequence in a trie and then uses the observed movement pattern to construct the cell-level residence probabilities. Fig. 4 shows a sample representation of the different phrases ($\psi$) and their frequencies in the trie, where the frequencies are updated for every prefix of every suffix of $\psi$ [7]. Using the Prediction by Partial Match (PPM) blending algorithm [11], the network first

```
1. Initialize dictionary:= NULL
2. Construct an initial codebook
   $\mathcal{Q}_m = \{RA_\imath^m \ : \imath = 1, \ldots, N\}$;
3. loop
4.   Receive differential codebook information
       ($\delta_\imath^{m+1}$,$\Delta_\imath^{m+1}$);
5.   Update the codebook using:
     $RA_\imath^{m+1} = RA_\imath^m \cup \delta_\imath^{m+1} - \Delta_\imath^{m+1}$;
6.   Decode the entropy-coded phrase;
7.   add phrase to dictionary;
8.   Increment frequency for every prefix
             of every suffix of phrase;
8. forever
```

Fig. 3. Corresponding decoder at the system.

computes the unconditional probabilities for each of the symbols in the trie. Each element of the trie represents a cluster of cells in the spatial quantization. Thus, to compute the unconditional *cell* residence probabilities, the computed probability of each trie element (or RA) is then distributed equally among the constituent cells. For example, the unconditional probability $\varrho(RA_\imath)$ is distributed among these cells as $\varrho(x) = \frac{\varrho(RA_\imath)}{|RA_\imath|}$ $\forall x \in cell(RA_\imath)$. The *MN* is then paged in the decreasing order of cell residence probabilities. The complete algorithm is described in Fig. 5.

### 4.4 Dynamically Adjusting the Update Rate ($R$)

In the description above, the number ($N$) of codewords was assumed to be a known constant. Although the nominal update rate (per generated symbol) of an $N$ codeword quantizer is $\log_2 N$, the actual update rate (in terms of bits *per unit time*) depends on the actual movement pattern of the *MN*, which directly affects the symbol generation rate. Now, an *MN* may actually wish to control its *average* update rate. Since the movement frequency of the *MN* cannot be predicted a priori, the average update rate may be controlled by *dynamically* (over medium time scales) altering the value of $N$ and recomputing the codebook. Accordingly, $N$, which is the size of the codebook or the number of distinct clusters, is a slowly varying parameter that an *MN* adaptively varies to control its update rate. Let $\rho_T$ be the target (desired) update rate and let $\rho_{N_{curr}}$ be the

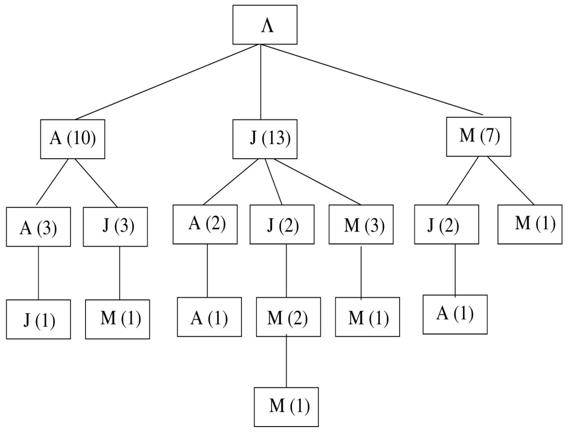


Fig. 4. Trie with spatial quantization in a single-system network.

```
1. Initialize $i := 0$, $Pr[\psi] := 0$, $h$:= highest order
2. Initialize escape probability $Pr_h^{(e)} := 1$
3. While ($i \leq h$)
4.       Search for $\psi$ at order $h-i$
5.       if ($\psi$ found)
6.          Compute $(h-i)$th order probability ($Pr_{h-i}[\psi]$)
7.       else
8.            $Pr_{h-i}[\psi] := 0$
9.       end-if
10.      Compute the escape probability
         $Pr_{h-i}^{(e)}$ at order ($h-i$)
11.      Estimate the combined probability as
         $Pr[\psi] := Pr[\psi] + \Pi_{j=h}^{h-i} Pr_j^{(e)} \times Pr_{h-i}[\psi]$
12.      $i := i+1$
13. end-while
14. Compute the probability of individual RAs
    based on their relative weights in the phrase
15. Probe RAs in decreasing order of
    normalized residence probabilities
16. Within each RA, page its cells in any random order
    (as their residence probabilities are identical)
```

Fig. 5. Paging with spatial quantization in a single-system network.

observed (current) update rate for a codebook of size $N_{curr}$. Since the rates are proportional to $\log_2 N$, we update the new size of the codebook according to

$$N_{new} = \lfloor (N_{curr})^{\gamma} \rfloor, \tag{5}$$

where $\gamma = \frac{\rho_T}{\rho_{N_{curr}}}$ is the ratio of the desired to the current rates. By combining VQ with an adaptive choice of the size of the codebook, we *essentially allow the MN to determine the near-optimal strategy of the location update that adheres to its desired update rate.*

## 5 TQ Approach

One of the drawbacks of the spatial quantization approach is that it does not exploit the correlation within *movement sequences*. If the codewords can represent certain temporal sequences rather than simply spatial clusters, then it will almost certainly outperform the spatial quantizer [12]. This correlation can be easily captured by *VQ*. In this case, a contiguous block of data (cell locations) from the infinite sequence $X^n$ is captured (in an online fashion) and quantized into a designated set of $N$ codewords. Of course, the paging algorithm (Fig. 5) must now be modified to reflect the fact that each received codeword represents *one of the several movement sequences that map to the codeword.*

### 5.1 Location Update Strategy

As before, the mobility of the user in the cellular network is represented by the sequence of cells $\bar{X}^n = \{X_1, X_2, \ldots\}$. Now, instead of quantizing each individual sample, we group $k$ consecutive samples into a single block and then quantize the entire block. In other words, we consider nonoverlapping $k$-length subsequences of $\bar{X}_n$, which can be represented by $\bar{X}(k) = \{x_m, x_{m+1}, \ldots, x_{m+k}\}$. Each of these $k$-length subsequences forms an input vector of dimension $k$. Every such vector $\bar{X}(k)$ is quantized (approximated) to a *code vector* $\bar{Q}$ with the objective of achieving *minimum distortion*. We thus form Voronoi regions in $k$-dimensional space, where each Voronoi region ($\mathcal{V}_j$), with its component input vectors, is represented by a code vector. Each codeword $\bar{Q}_i$ for $i \in \{1, \ldots, N\}$ thus represents a set of $k$-element vectors $\bar{X}$. Returning to our original example from Section 4.1, if the user's movement is given by "$ajlojhajlojajlojajo\ldots$," then for $k = 2$, the symbols are grouped into blocks (vectors) $\bar{X}_1 = \{a, j\}$, $\bar{X}_2 = \{l, o\}$, $\bar{X}_3 = \{j, h\}$, $\bar{X}_4 = \{j, a\}$, $\bar{X}_5 = \{j, l\}$, $\bar{X}_6 = \{o, j\}$, $\ldots$. The new vector sequence now becomes $\bar{X}_1\bar{X}_2\bar{X}_3\bar{X}_1\bar{X}_2\bar{X}_4\bar{X}_5\bar{X}_6\bar{X}_1\ldots$. Now, let the Voronoi regions $\bar{RA}_1$ and $\bar{RA}_2$ consist of the set of vectors $\{\bar{X}_1, \bar{X}_2, \bar{X}_4\}$ and $\{\bar{X}_3, \bar{X}_5, \bar{X}_6\}$ and be represented by the code vectors $\bar{Q}_1$ and $\bar{Q}_2$, respectively. Thus, the corresponding update sequence $\bar{Y}^n$ now becomes $\bar{Q}_1\bar{Q}_1\bar{Q}_2\bar{Q}_1\bar{Q}_1\bar{Q}_1\bar{Q}_1\bar{Q}_2\bar{Q}_2\bar{Q}_1$.

For a distortion measure, we utilize the weighted euclidean distance, where the distance between any vector $\bar{X}_i$ and its quantized estimate $\bar{Q}_i$ is the sum of the distance between the components of $\bar{X}_i$ and $\bar{Q}_i$. The weight is the *user's normalized residence probability in the particular $k$-dimensional vector (subsequence)*, that is, the probability of the *MN*s exhibiting the particular movement sequence. The quantization process itself uses the multidimensional version of Lloyd's algorithm to partition the underlying vectors among the set of computed codewords and transmit the codebook updates whenever necessary. Equations (2) and (3) are thus modified to account for the vectors $\bar{X}$, its probability $p_{\bar{X}}$, code vectors $\bar{Q}$, and the corresponding regions $\mathcal{V}$ in the $k$-dimensional space as follows:

$$\begin{aligned} D_w(\bar{X}_i, \bar{Q}_i) &= \|\bar{X}_i - \bar{Q}_i\|^2 \times \varrho(\bar{X}_i), \\ \bar{RA}_k = \{\bar{X} : D_w(\bar{X}, \bar{Q}_k) &\leq D_w(\bar{X}, \bar{Q}_l); \forall l \neq k\}, \\ \bar{Q}_k &= \frac{\sum_{\bar{X} \in \bar{RA}_k} \bar{X} \times \varrho(\bar{X})}{\sum_{\bar{X} \in \bar{RA}_i} \varrho(\bar{X})}. \end{aligned} \tag{6}$$

Fig. 6 provides a pseudocode for the encoder of the TQ-based location update scheme (AVQ followed by entropy coding). As in Section 4.4, the *MN* can adjust its location update rate by appropriately changing $N$, the number of distinct codewords permitted. The corresponding decoding operation at the network side is almost the same as in Fig. 3, with the only change that each individual symbol now represents a vector of dimension $k$. In general, the TQ-based update strategy outperforms the scalar (spatial) quantization approach, as shown in a recent result [31].

**Result 1.** *For any simple distortion measure within $[0, \infty)$, in the worst case, the cascading of the VQ (TQ) with dimension $k > 2$ and entropy coding results in a transmission rate very close to the rate distortion bound and arbitrarily lower than that achieved by a similar scalar quantizer ($k = 1$) such as the spatial quantizer described in Section 4.1.*

### 5.2 Paging Scheme

The paging process associated with the TQ scheme starts with a similar traversal on the probability trie to obtain the unconditional residence probabilities $\varrho(\bar{Y})$ of the individual code vectors $\bar{Y}$. Since the trie also contains the probabilities of different input vectors $\bar{X}$ associated with every code vector $\bar{Y}$, we now distribute $\varrho(\bar{Y})$ among the vectors $\bar{X}$ belonging to the same region $\Omega$ in such a way that every vector $\bar{X}_j$ receives a fraction of $\varrho(\bar{Y}_i)$ inversely proportional to its distance $d(\bar{X}_j, \bar{Y}_i)$ from the code vector $(\bar{Y}_i)$ of that region. The residence probability of each individual cell (symbol) $x_i$ is computed by distributing the probability

```
1.  Set $m=1$, $D_w^0(\bar{RA}_i^m)=0$, dictionary:=NULL, $\bar{w}_i$ :=NULL;
2.  Construct an initial codebook $\mathcal{Q}_m=\{\bar{RA}_i^m\}$;
3.  loop
4.  Get a sequence of cells $\{X_1,X_2,\dots,X_n, n\to\infty\}$;
5.  Group $k$-cells to get a block $\bar{X}=\{X_1,X_2,\dots,X_k\}$;
6.  Quantize $\bar{X}$ to get $\bar{Y}_i$;
7.    if($\bar{w}.\bar{Y}_i$ in dictionary)
8.      $\bar{w}:=\bar{w}.\bar{Y}_i$;
9.    else
10.     for (each code-vector) do
11.         Estimate the new codebook $\mathcal{Q}^{m+1}$ using:
            $\bar{Q}_i^m \approx \frac{\sum_{\bar{RA}_i}\bar{X}\varrho(\bar{X}}{\sum_{\bar{RA}_i}\varrho(\bar{X})}$;
12.         Find optimal partition using condition:
            $\bar{RA}_k=\{\bar{X}: D_w(\bar{X},\bar{Q}_k^m)\le D_w(\bar{X},\bar{Q}_l^m);\forall l\ne k\}$
13.         $D_w^{m+1}(\bar{X}_i,\bar{Q}_i)=||\bar{X}_i-\bar{Y}_i||^2\times\varrho(\bar{X}_i)$,
            where $||\bar{X}||^2=\sum_{j=1}^k x_j^2$;
14.         if $\left(\frac{D_w^{m+1}(\bar{Q}_i)-D_w^m(\bar{Q}_i)}{D_w^m(\bar{Q}_i)}\ge\epsilon\right)$
                Set $m:=m+1$ and go to Step 7;
15.     end-if;
16.     End-for;
17.     Compute the differential updates $\left(\bar{\delta}_i^{m+1},\bar{\Delta}_i^{m+1}\right)$
        in codebook using two relations:
        $\bar{\delta}_i^{m+1}=\bar{RA}_i^{m+1}-\bar{RA}_i^m$, $\bar{\Delta}_i^{m+1}=\bar{RA}_i^m-\bar{RA}_i^{m+1}$;
18.     Transmit differential update to decoder;
19.     Encode index$<\bar{w}>,\bar{Y}_i$;
20.     Add $\bar{w}.\bar{Y}_i$ to dictionary;
21.     $\bar{w}$ := NULL;
22.   end-if;
23. forever;
```

Fig. 6. Encoder of TQ-based location update.

mass of the vectors among its component cells and normalizing it with the number of occurrences of cells in the entire temporal domain. Formally, we can say that

$$\varrho(\bar{X}_j)=\frac{\varrho(\bar{Y}_i)}{d(\bar{X}_j,\bar{Y}_i)},\ \forall\bar{X}_j\in\mathcal{V}_i;\ \varrho(x_i)=\sum_{\forall\bar{X}_j}\varrho(\bar{X}_j). \qquad (7)$$

As in the case of the spatial quantization in a single cellular network, the network now issues paging requests in the decreasing order of $\varrho_{x_i}$ with ties broken randomly.

## 6 Adaptive Location Management in a Multisystem Network

The single-system network analyzed in the last two sections is a generic representation of a variety of cellular networks, including the second-generation (*2G*) and third-generation (*3G*) wide-area cellular networks, as well as extended campus-wide Wireless LANs (WLANs), where each *MN* has only one active radio interface. However, our model of quantization-based trade-off can also be extended to a multisystem heterogeneous integrated network architecture (as shown in Fig. 7) introduced in [28]. Such a network captures the emerging vision of the so-called fourth-generation (*4G*) wireless environments, where an *MN* may possess multiple physical or software-defined radio interfaces and seamlessly switch between or *concurrently use* loosely coupled multiple cellular access systems, typically owned by multiple service providers, with varying coverage areas (indoor/outdoor), transmission ranges, and bandwidth. For example, an *MN* could be using a wide-area cellular interface to transmit voice and a separate WLAN infrastructure to transmit data for a single collaboration application. Location management in such a loosely coupled multisystem network gives rise to the following additional challenges:

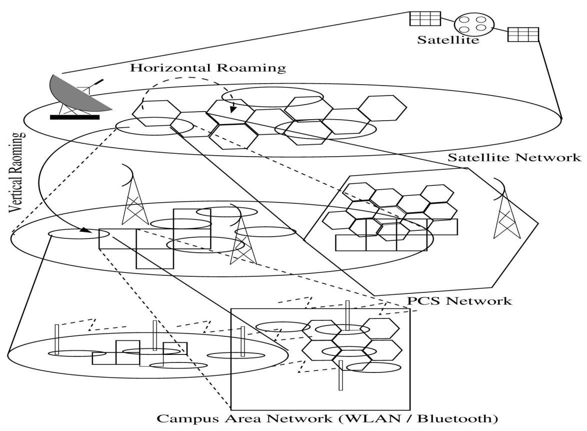


Fig. 7. A typical multisystem heterogeneous cellular network.

- From a location update perspective, an *MN*'s movement may now result in cell changes in more than one subnetworks. Moreover, as infrastructures such as WLAN hotspots provide only discontinuous coverage, an *MN* may often be outside the coverage area of some subnetworks.
- For paging, the subnetworks must accommodate the possibility of an *MN* actively communicating on one interface (with one subnetwork) but being in passive mode on other interfaces. In particular, the paging process should exploit the possibility of "indirect paging," where the overhead of multiple paging messages in a subnetwork can be avoided by requesting the *MN* to register through some other subnetwork where the *MN* is currently active [28].

Mathematically, we can model such a network as consisting of $\eta$ independent subnetworks $\{SU_1, SU_2,\cdots,SU_\eta\}$. Each subnetwork $SU_i$ consists of a set of $|SU_i|$ cells such that $C_i^j$ represents its $j$th cell. To allow for the subnetwork with discontinuous coverage areas, let each subnetwork have a separate virtual cell $\phi$ representing the disconnected state. Also, let $PG_i$ denote the paging cost/message and $LU_i$ denote the update cost/message in $SU_i$.

At any instant, the *MN*'s location may be represented by a vector-valued random variable $\bar{X}$ of dimension $\eta$, where the $i$th element in the vector represents the current cell in the subnetwork $SU_i$. As an example, if $\eta=3$, then the vector $\bar{x}=[C_1^2, C_2^3, C_3^\phi]$ implies that the *MN* is currently in a location formed by the intersection of the cells $C_1^2$ (belonging to $SU_1$), $C_2^3$ (belonging to $SU_2$), and $C_3^\phi$ (the entire set of dead zones, that is, physical areas with no connectivity, in $SU_3$). The movement of the *MN* can then be viewed as a stochastic vector-valued random sequence $\chi=\{\mathcal{X}^n, n\to\infty\}$. The *MN* must keep the network informed of the evolution of the random vector sequence $\chi$, where each vector is of dimension $\eta$.

### 6.1 Spatial Quantization

For spatial quantization, we split the entire set of location tuples $\bar{X}$ into $N$ disjoint sets. Thus, each set or RA consists of one or more $\eta$-element vectors. Partitioning the entire vector space into $N$ disjoint sets essentially forms $N$ quantized regions $\{\vartheta_1, \vartheta_2, \ldots, \vartheta_N\}$ such that the quantization process maps every vector $\bar{X}$ to its nearest possible region $\vartheta_l$. Every quantized region $\vartheta$ is now represented by the corresponding codeword $\bar{U}$. Formally, the mapping is given by $\bar{U} = Q(\bar{V}) : \bar{V} \rightarrow \vartheta_l$. In other words, the user's location information is now transmitted in a quantized form.

The update algorithm is then a logical modification of the single-system update algorithm presented in Section 4.1, with Lloyd's iterative codebook construction algorithm now applied to points in the $\eta$-dimensional space. At the decoder (network) end, the trie-traversal approach outlined earlier in Fig. 5 is used to first construct the probabilities for various symbols in the trie. Each symbol now represents a particular combination of location tuples. Thus, the trie traversal provides unconditional residence probabilities $\varrho(\bar{U})$ of the code vector $\bar{U}$ representing a particular region $\vartheta$ (that is, a group of $\eta$-element cell tuples). This probability is now equally distributed among all the $\eta$-dimensional vectors $\bar{X}$ constituting that region. Next, to estimate unconditional probabilities for each *cell* within its own subnetwork, we treat each element of a vector-valued symbol $\bar{X}$ as having the corresponding probability mass and then normalize these values over all the elements (cells) in the system (an approach detailed in [28]). Formally,

$$\varrho(\bar{X}_k) = \frac{\varrho(\bar{U}_j)}{\Im(\bar{X}_k)}, \ \forall \bar{X}_k \in \vartheta_k, \tag{8}$$

$$\varrho^i_{S(i)} = \sum_{k \ : \ c^i_{S(i)} \in \bar{X}_k} \varrho(\bar{X}_k), \tag{9}$$

where $\Im(\bar{X})$ represents the *type* [12] or number of elements of $\bar{X}$. Although (8) computes the probabilities of individual cell tuples, (9) subsequently computes the probability of a particular cell by adding the probabilities for all the tuples that it is a member of.

Given the unconditional residence probabilities for each cell, the network then computes the paging sequence (by using the heuristic defined in [36]) in the decreasing order of the *conditional residence probabilities*, that is, iteratively selecting the cell that has the highest residence probability, conditional on the *MN* being simultaneously absent in all previous cells of the paging sequence.

### 6.2 TQ

Recall that the user's mobility in a multisystem cellular network is an $\eta$-valued vector sequence. The TQ of this vector sequence involves the grouping of $k$ contiguous vector samples $\bar{X}$ into a block $\bar{\Gamma}$ and then the partitioning of the resulting $\eta \times k$-dimensional space of $(\bigcup_{i=1}^{\eta_s} |S_i|)^k$ possible $k$-element vectors into a set of $N$ distinct regions. At a fundamental level, the quantization approach is very similar to the TQ of a single-system network, except that each element $\bar{X}$ of each vector is itself vector valued. Thus, the *MN* uses a similar approach to obtain the code vectors (by minimizing the weighted distortion measure), performs LZ78 compression, and transmits the compressed and coded sequence to the network. The paging process also uses similar trie-traversal schemes to compute unconditional probabilities of $k$-length code vectors. This probability of a code vector is now distributed among the vectors of its region in proportion to its normalized probability mass. The greedy paging process and the residence probability estimation of individual cells in each subnetwork are computed using (8), as in the last section.

## 7 Performance Study

Our spatial quantization and TQ algorithms are designed to illustrate the fundamental trade-offs in any *abstract cellular environment*. Our focus is not to develop specific protocols for a specific technology like *3G* systems or the universal mobile telecommunication system (UMTS) but, rather, to illustrate the benefits accruing from a per-user trade-off of location versus paging costs. We use both simulated movement models and real-life data [13] to quantify the performance benefits of our proposed algorithms vis-a-vis the existing location management scheme in PCS networks and the previously proposed nonadaptive LeZi-Update algorithm [7]. In particular, we study the potential reduction in the update cost versus the corresponding increase in the paging cost, as well as the storage and computational overhead on the *MN*. To study the performance of our algorithms, we implemented a discrete-event simulator, which takes as input a network topology and movement/calling trace and computes the resulting update and paging loads. The Lloyd quantizer is assumed to converge when the fractional change in distortion is less than $\epsilon = 0.05$.

### 7.1 Simulation Parameters

For simulations with synthetic data, we generated a trace of a single user's movement and calling patterns according to an underlying statistical distribution:

1. The single-system topology is a grid consisting of 500 cells, representing a large campus network or a city-wide wireless mesh environment, with the degree of connectivity (number of neighboring cells) varying between 2 and 3 for the corner cells, between 3 and 5 for "edge" cells, and between 4 and 8 for "inner" cells.
2. The multisystem cellular network consists of three distinct subnetwork types: satellite (largest cell size), wide-area cellular (intermediate cell size), and WLAN (multiple disconnected hot spots). The relative update costs for WLANs and satellite networks are respectively set to three and 10 times that of the PCS network. Similarly, the relative paging costs of PCS and satellite networks are set to four and nine times that of the WLANs.
3. An *MN* is assumed to have randomly chosen "home" and "work" places. Although the home cell stays the same, each user's work cell changes once every $T \in \{14, 28, 42\}$ days. The *MN* movement pattern differs between weekdays and holidays/weekends. The residency time in a cell is normally distributed (negative values are ignored).
4. The call arrival process is Markov-modulated with three distinct states, each with its own Poisson arrival rate $\zeta$ and normally distributed holding times, with mean $\mu$ and variance $\sigma$. The states represent weekday day-times ($\zeta = 0.2$ calls/hour, $\mu = 10$ minutes, and $\sigma = 3$ minutes), weekday evenings ($\zeta = 0.3$ calls/hour, $\mu = 20$ minutes, and

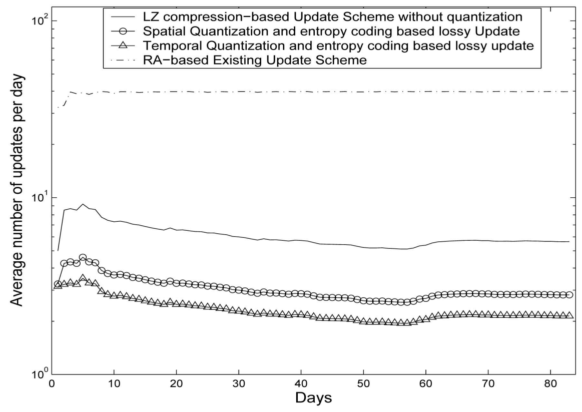


Fig. 8. Number of updates in single-system cellular PCS networks.

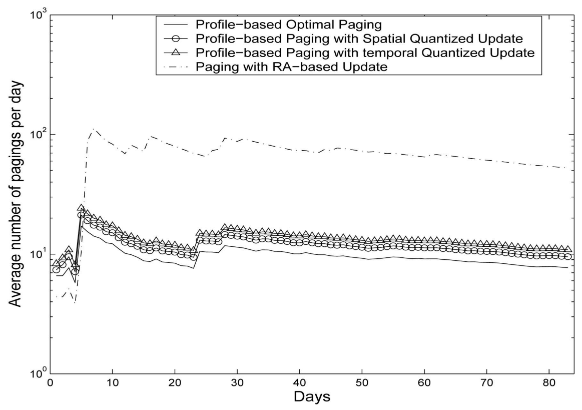


Fig. 9. Paging in single-system cellular PCS networks.

$\sigma = 5$ minutes), and weekends ($\zeta = 0.5$ calls/hour, $\mu = 30$ minutes, and $\sigma = 7$ minutes). The simulation results are computed over a continuous period of 12 weeks.

## 7.2 Performance in Single-System Cellular Network

We first investigate the update and paging overheads in a single-system cellular PCS network with a codebook size of $N = 5$. Fig. 8 shows that the proposed spatial-quantization-based and TQ-based update schemes result in a significantly lower update cost, compared to the existing RA-based update strategy or the LeZi-Update scheme. More precisely, the update costs of our spatial and temporal schemes are only $\sim 8$ percent and $\sim 6$ percent, respectively, of the existing RA-based strategy and $\sim 50$ percent and $\sim 39$ percent, respectively, of the existing LeZi-Update scheme. Furthermore, the TQ scheme outperforms the spatial scheme by capturing the movement patterns (or temporal correlations) in blocks of samples. Note that the "update" cost also includes the overhead due to the transmission of any changes to the codebook. With these simple-to-implement approaches, *the MN generates only five to six updates a day*, in contrast to static non-user-specific RA-based schemes, where the update rate is $\sim 30/\text{day}$.

The savings in the update cost comes at the expense of a higher network paging cost. Fig. 9 demonstrates that the paging costs incurred by the spatial quantization and TQ schemes are bounded by $\sim 1.8$ times and $\sim 2.1$ times that of the profile-based paging cost incurred in LeZi-Update. However, the two proposed schemes still result in only $\sim 17$ percent and $\sim 20$ percent, respectively, of the existing *blanket* paging cost associated with the RA-based updates in current PCS systems.

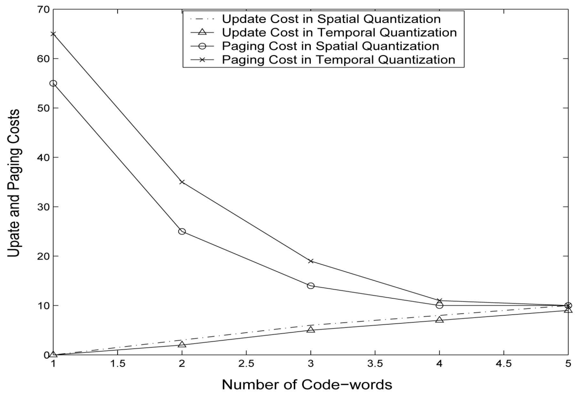


Fig. 10. Variation in update paging costs with codewords.

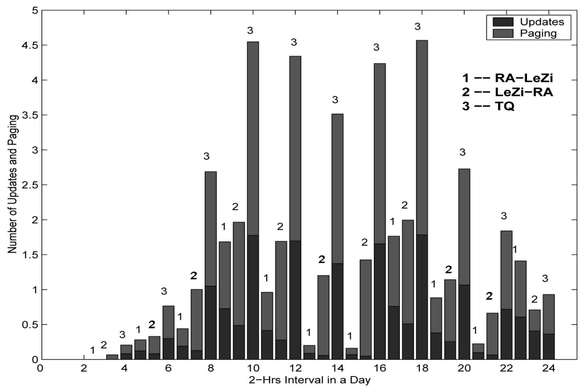


Fig. 11. Variation in update and paging costs over the 24-hour day.

The principal motivation of this work is to make the location tracking scheme adaptive such that each *MN* can independently adjust its location update cost based on its individual preferences. In our quantization-based approaches, *the size of the codebook* $(N)$ *and the block size* $k$ *(for TQ) are tuning knobs that each individual MN can use to trade off between its update cost and the network's paging overhead.* Fig. 10 shows that, for a fixed block size of $k = 5$ with a decrease in the codebook size $(N)$, the update cost associated with the spatial quantization or TQ decreases nonlinearly, obviously at the expense of an increase in the paging cost.

To further understand the behavior of our update strategy, we plotted in Fig. 11 the variation in the update rate over the course of a day. As we can see, the update rate shows a spike during the hours of 7 and 9 a.m. (morning rush hour) and 3 and 5 p.m. (evening rush hour). On the other hand, the paging load is uniformly high between 8 a.m. and 6 p.m. when the user receives the largest fraction of calls. This graph also shows that the relative performance of the RA-LeZi, LeZi-RA, and TQ schemes remains unchanged over the diurnal variations in the mobile user's behavior.

### 7.2.1 *Performance Sensitivity*

We have also investigated the sensitivity of our VQ-based approach (in terms of both the location update and paging

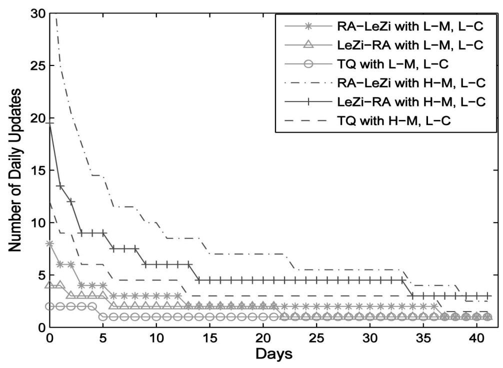


Fig. 12. Variation in update overhead for different call and mobility profiles.

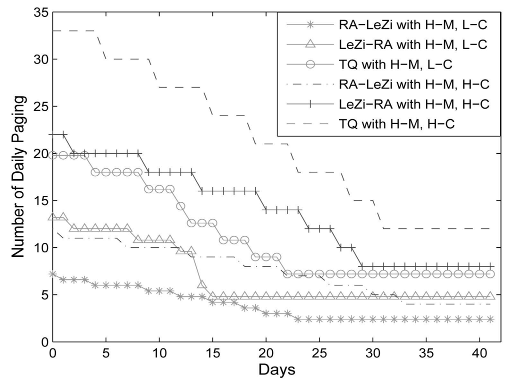


Fig. 13. Variation in paging overhead for different call and mobility profiles.

overhead) to changes in the call-mobility ratio. To illustrate this, we simulated the behavior of the LeZi-RA, RA-Lezi, and TQ algorithms for three different operating points: 1) low mobility, low call rate (L-M, L-C), 2) high mobility, low call rate (H-M, L-C), and 3) high mobility, high call rate (H-M, H-C). In the H-M scenarios, the average cell residency is half of the L-M scenario. Similarly, the call rate $\zeta$ for the H-C scenario is twice that of the L-C case. The (L-C, L-M) scenario uses the parameters outlined in Section 7.1. Fig. 12 shows the average number of updates/day (over the duration of the simulation) for the (L-M, L-C) and (H-M, L-C) cases (only the L-C case is plotted, as the location update frequency is essentially insensitive to the call arrival rate). Clearly, whereas the number of updates/day is expectedly higher for the H-M scenario, our quantization algorithms quickly *adapt* to the movement pattern and reduce the update load to less than five updates/day across all mobility rates. Similarly, in Fig. 13, only the H-M case is plotted, as the paging overhead is largely independent of the rate of movement (depends only on the movement *statistic*). We observe the progressive reduction in the paging cost as the system learns the true paths followed by the *MN*. In line with previous results, the TQ approach results in the smallest update overhead but highest paging overhead. *Our results also demonstrate that, unlike fixed-threshold schemes, the signaling overhead in quantization-based approaches varies across users, depending on their individual call and mobility rates.*

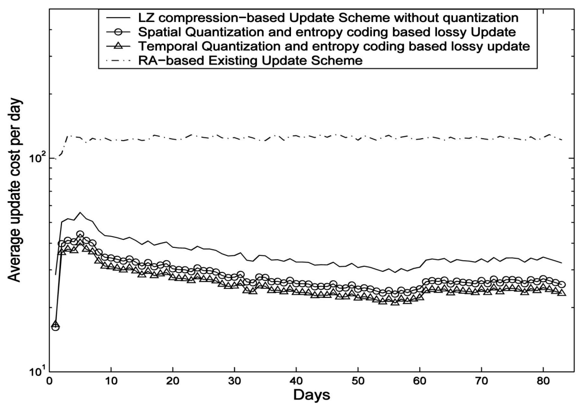


Fig. 14. Update cost in multisystem cellular networks.

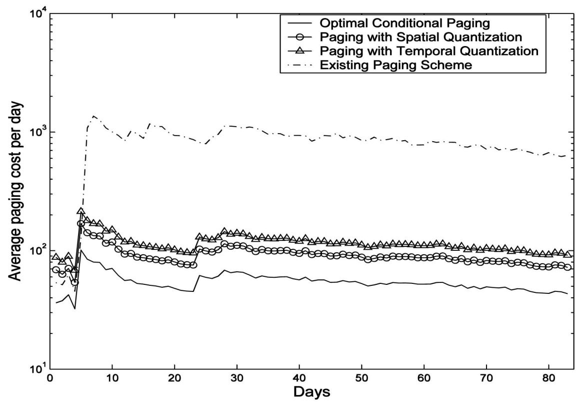


Fig. 15. Paging cost in multisystem cellular networks (simulated).

## 7.3 Performance on Multisystem Cellular Networks

In a multisystem heterogeneous cellular network, different individual subnetworks have differential signaling costs for update and paging. Hence, to get a fairer performance comparison, the update and paging costs are computed by weighing the actual number of update and paging messages in a particular subnetwork $S_i$ with its corresponding costs $LU_i$ and $PG_i$. Fig. 14 demonstrates that our proposed spatial-quantization-based and TQ-based update strategies have update overheads of only $\sim 25$ percent and $\sim 23$ percent of the existing movement-based update scheme and $\sim 88$ percent and $\sim 79$ percent of the lossless compression strategy (that is, the multisystem LeZi-Update [28]). These results demonstrate how lossy compression techniques can reduce the update cost by an order of magnitude compared to the RA-based strategy, even in a heterogeneous cellular environment.

The corresponding effect on the total paging cost is shown in Fig. 15. The proposed spatial and TQ strategies result in a paging cost of only $\sim 1.52$ times and $\sim 1.75$ times more than that of the paging cost provided by the near-optimal profile-based greedy paging scheme. It should be noted that, even with this increased uncertainty, these two schemes also result in only $\sim 17.8$ percent and $\sim 20.4$ percent, respectively, of the existing paging cost associated with movement-based updates in multisystem cellular networks. This reduction is due to the ability to customize the location profile for each *MN* based on the observations of its true movement pattern.

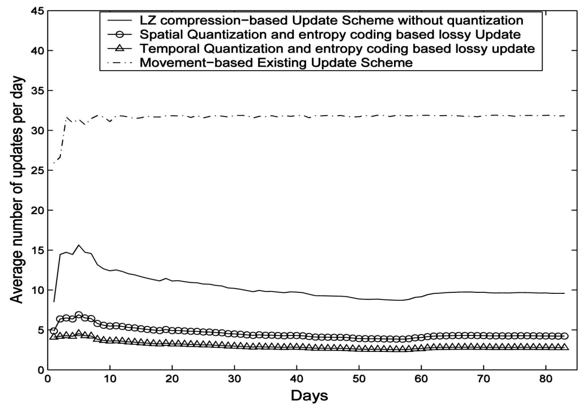


Fig. 16. Number of updates in 802.11 cellular network (real traces).

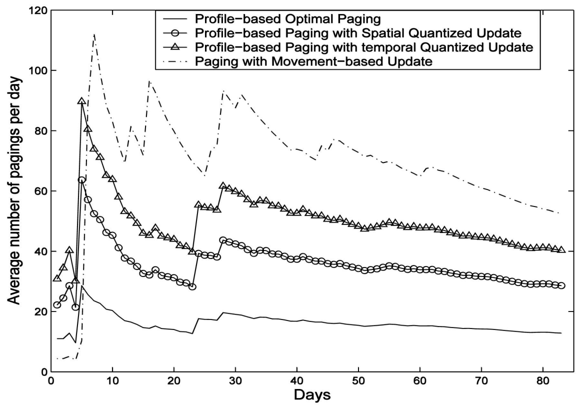


Fig. 17. Paging in cellular network (real traces).

## 7.4 Performance with Real Movement Traces

We have also studied the performance of our algorithms on the real-life movement traces obtained from the Dartmouth Campus environment [13]. Note that these traces are only from a single-system extended 802.11 deployment (not a wide-area cellular network). Earlier studies such as [39] on WLAN traces have concluded that the *MN*'s movement can be adequately described by relatively low-order Markov chains. Fig. 16 demonstrates that the spatial-quantization-based and TQ-based techniques result in only $\sim \frac{1}{5}$ and $\sim \frac{1}{7}$ of the update cost incurred in existing movement-based updates and $\sim \frac{1}{3}$ and $\sim \frac{2}{5}$ of the update traffic generated by the LeZi-Update scheme. Thus, the strategies can achieve a significantly lower update cost even in real movement traces. Obviously, as Fig. 17 shows, whereas the resultant paging costs are higher (almost two and three times that of the optimal paging strategy), they are still only 40 percent of the existing cluster paging scheme.

## 7.5 Storage Overhead

Two other important performance metrics that we study for our adaptive algorithms are the storage and computational overhead. Let us first study the storage overhead of the adaptive location management strategies which require an *MN* to maintain dictionaries for its observed movement pattern. Intuitively, the storage overhead depends on the codebook size and is different for the proposed spatial quantization and TQ schemes. Fig. 18 compares the evolution of the maximum size of the codebook at the *MN*'s encoder for our quantization

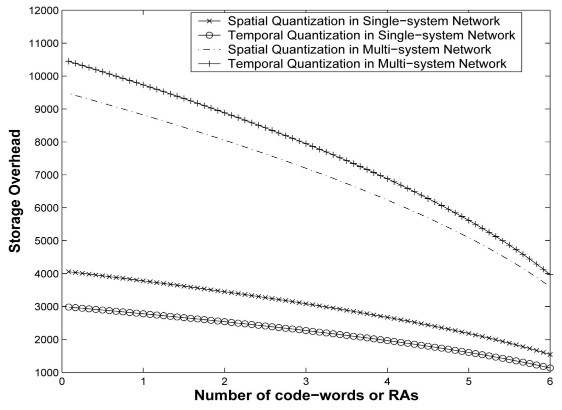


Fig. 18. Storage overhead in location tracking.

TABLE 2
Average Number of Mathematical Operations per Day

| System | Cellular LeZi | Spatial Quant | Temporal Quant |
|---|---|---|---|
| Single-System Cellular | 21783 | 22357 | 23783 |
| Real IEEE 802.11 | 20359 | 21453 | 22343 |
| Multi-System Cellular | 40475 | 42389 | 44321 |

strategies over both the single-system and multisystem wireless networks with varying numbers of codewords. This figure reveals that, for single-system wireless networks, the spatial quantization and TQ schemes result in a storage requirement of 1-3 Kbytes and 1.5-4 Kbytes for a codebook size of 1-6 codewords. The storage requirement for multisystem wireless networks is in the range of 4-9 Kbytes and 5-11 Kbytes for the spatial quantization and TQ schemes, respectively.

## 7.6 Computational Complexity and MN's Energy Cost

For practical use, the savings in communication costs on the *MN* should not be entirely negated by its additional algorithmic computational overhead. Clearly, the quantization-based approaches increase the complexity of the *MN*, which must 1) intermittently run the Lloyd quantization algorithm and 2) store a more elaborate dictionary. To study this issue, we computed the average number of numerical operations performed daily by the *MN* for the LeZi-Update and the spatial quantization and VQ algorithms. Table 2 shows the results. Note that the spatial quantization and TQ schemes respectively require only $\sim 5$ percent and $\sim 10$ percent more mathematical operations in all cases as compared with the existing LeZi-Update algorithm that does not support quantization.

More important than the raw computational overhead (on the *MN*) is the combined computation and communication costs. Recent studies (for example, [6]) have shown how overly complicated algorithms on an *MN* can negate the savings in the communication overhead by the added penalty in the computational and memory overheads. Although an exact enumeration of overheads is device-specific, we can obtain a rough enumeration of the cost for our algorithms by utilizing the microbenchmarks in [6] (obtained using an 802.11b wireless card attached to a PDA using a 22-MHz StrongARM processor). The experiments in

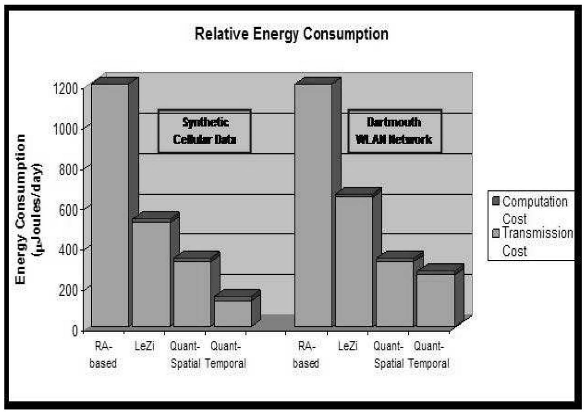


Fig. 19. Relative energy consumption of *MN* for various schemes.

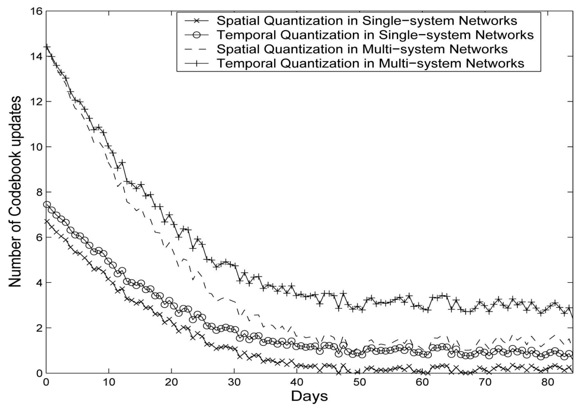


Fig. 20. Dynamics of codebook update.

[6] revealed a computational cost of $\approx 0.86\text{nJ/operation}$ and a transmission cost of $\approx 1\mu\text{J/bit}$. By combining these costs with the update (Figs. 8 and 16) and computational traffic (Table 2), we plot the approximate *MN*'s total energy consumption/day in Fig. 19 (for both the simulated data and the Dartmouth traces). Here, we pessimistically assume that the current RA-based update strategy incurs zero computation cost. As the figure shows, the increase in the computational complexity for our quantization algorithms is more than compensated for by the reduced location update overhead, *resulting in almost $\frac{1}{5}$ of the overall energy consumption compared with the current RA-based location update mechanism*.

## 7.7 Codebook Update

We now study how frequently the codebook needs to be updated. Clearly, a large number of codebook updates would reduce the efficiency of the entire strategy by adding extra overhead. However, as the system becomes cognizant of the user mobility profiles, the update intervals would get longer. Intuitively, in the steady state, the codebook (with sufficiently long memory) associated with a stationary process will remain fixed, resulting in a rate asymptotically $\rightarrow 0$. Fig. 20 reveals that the number of codebook updates for both quantization schemes (for a single-system cellular network) is initially $\sim 7/\text{day}$ but tends to 0 after a period of 30 days. For an integrated multisystem network, the number of codebook updates is initially $\sim 14$. However, as the system learns the *MN*'s movement pattern, this number reduces and saturates to very low values of $\sim 2$.

## 7.8 Practical Networks and Overhead Implications

Although our quantization algorithms reduce the update overhead via the per-user adaptation of RAs/codewords, they impose additional storage and computational overhead on both the *MN* and the network. In particular, the *MN* stores its *entire movement history* locally until it runs out of storage (at which point the trie is flushed and the entire learning process restarts). We now provide some numerical calculations to quantify the additional costs of the per-user mobility management for two candidate networks: 1) a wide-area national PCS cellular network with 10,000,000 users and 2) an extended-area campus network with 10,000 users.

Although the total number of cells for a national PCS provider is not publicly available, data in [15], [16], and [30] suggests that the *total* number of base stations (among all providers) in the UK/Spain is 30,000-40,000. Extrapolating on the basis of population and considering the number of distinct providers, we infer that a single US cellular provider is likely to have *not more* than 100,000 distinct cells. In current Global System for Mobile Communications (GSM) systems, each distinct cell ID occupies 56 bits, consisting of a 40-bit Location Area Identifier (LAI) and a 16-bit Cell Identifier (CI). Now, even if each RA were to consist of only four cells, the number of distinct codewords (25,000) could be encoded in 15 bits ($2^{15} > 25{,}000$), and the maximum upper bound on the quantizer mapping (for the entire US) would be $15 \times 25{,}000 \times 4 \times 56 \text{ bits} \approx 10.5$ Mbytes corresponding to 25,000 RAs, each storing four cells. Similarly, if the 802.11-based campus environment consists of 200 access points (APs), each having a unique 48-bit ID, and each RA has four cells (APs), then the storage overhead on the client would be $\approx 9.6$ Kbytes.

The movement history itself is maintained in a single hash-based trie. Even if a user visits 50 *new* cells/day (for example, a highly unpredictable traveling salesman), his *quantized* movement history over an entire year would still be 128 Kbytes. Even allowing for the additional overheads of indexing, the total storage overhead would thus be not more than $\sim$ 3-4 Mbytes/year, which is not significant, since cell phones today come equipped with 1 to 2-Gbyte drives. Similarly, in a campus environment, where a user is assumed to switch 100 APs/day, the total annual movement history could be stored in a trie of $\sim 220$ Kbytes. Note that the *MN*'s movement is efficiently stored as a trie, enabling easy insertion and retrieval, resulting in a constant computational overhead for trie insertion or matching, irrespective of the size of the cellular network.

The use of per-user profiles also significantly adds to the storage and processing costs at the network end. The processing overhead occurs only when the paging sequence must be determined from a trie and can be performed by specialized hardware or optimized software. For the national cellular network, the storage overhead for each user/year equals that of storing the movement trie and is thus $\sim 4$ Mbytes. For 10,000,000 users, this translates into a total storage requirement of 40,000 Gbytes (the corresponding storage requirement for the campus network is a relatively trivial 2.2 Gbytes). Clearly, even with distributed storage, this is a significantly large storage requirement.

TABLE 3
Approximate Storage Overheads for Per-User Profiles

| | **National network** (10 million users, 100,000 cells) | **Campus network** (10,000 users 200 cells) |
|---|---|---|
| **Map storage** (overhead at $MN$) | 10 MB | 9.6 KB |
| **Movement storage** (overhead at $MN$) | 128 KB | 220 KB |
| **Storage overhead** (at network) | 40,000 GB | 2.2 GB |

Table 3 summarizes the main storage costs (at both the *MN* and network) for both of our sample network scenarios. Overall, our results show that online per-user adaptive mobility management algorithms are already practical for environments such as extended campuses and smart homes/offices (where the scheme may be used for indoor location management at room-level granularity). However, the analysis demonstrates that, for large tier-1 cellular providers, *the usage of per-user movement profiles may impose significantly high storage requirements.* For large cellular networks, one way to circumvent this storage challenge may be to *group users with similar movement statistics* and store their aggregate history in a single trie. Our work in this paper may thus be viewed as an initial effort outlining the techniques for an effective paging-versus-update trade-off. In the future, we aim to develop such techniques for groups of users, thus improving scalability.

## 8 Conclusion

In this paper, we have presented an effective framework that trades off between the location update and paging costs for wireless environments such as wide-area cellular networks and extended campus networks. Unlike the current static schemes, where the same RA and paging sequence is applied to each *MN*, our approach creates per-user movement histories, in effect customizing the location update and paging sequence for each user to its observed mobility pattern. We have proposed two schemes based on the rate distortion theory, which view the trade-off as a process of introducing distortion between an *MN*'s true and reported movement sequences, thereby reducing the rate (cost) of the location update. Both schemes utilize a combination of adaptive *MN*-driven quantization of the movement sequence, followed by an $LZ78$-based adaptive compression. Simulation results demonstrated that the proposed algorithms can reduce the update cost to very low values (often 3-4 updates/day), with only modest increases in the paging overhead. Our algorithms can be implemented with relatively low storage overhead (at most 10-12 Mbytes) on the *MN*, even in large-scale cellular networks, and can result in a five- or sixfold decrease in the *MN*'s overall energy consumption. However, per-user profiles may present a scalability concern in tier-1 (national) cellular provider networks, where the individual profile of millions of devices must be stored separately.

To make the schemes practical in large cellular environments, our future research will investigate how we can extend the quantization approach to operate on the aggregate movement patterns, thus freeing the network from having to store *MN*-specific movement histories. Similarly, future research may suggest alternative quantization techniques, preferred from the standpoint of implementation efficiency and storage overhead. Finally, the impact of privacy on these algorithms is also an open issue.

## References

[1] I.F. Akyildiz and J.S.M. Ho, "Dynamic Mobile User Location Update for Wireless PCS Networks," *ACM Wireless Networks,* vol. 1, no. 2, pp. 187-196, July 1995.

[2] I.F. Akyildiz and W. Wang, "A Dynamic Location Management Scheme for Next-Generation Multitier PCS Systems," *IEEE Trans. Wireless Comm.,* vol. 1, no. 1, pp. 178-189, Jan. 2002.

[3] A. Bar-Noy and I. Kessler, "Tracking Mobile Users in Wireless Communication Networks," *IEEE/ACM Trans. Information Theory,* vol. 39, no. 6, pp. 1877-1886, Nov. 1993.

[4] A. Bar-Noy, I. Kessler, and M. Sidi, "Mobile Users: To Update or Not to Update," *Wireless Networks,* vol. 1, no. 2, pp. 175-185, July 1995.

[5] A. Bar-Noy, I. Kessler, and M. Naghshineh, "Topology-Based Tracking Strategies for Personal Communication Networks," *Mobile Networks and Applications,* vol. 1, no. 1, pp. 49-56, Jan. 1996.

[6] K. Barr and K. Asanovic, "Energy Aware Lossless Data Compression," *Proc. ACM MobiSys,* May 2003.

[7] A. Bhattacharya and S.K. Das, "LeZi-Update: An Information-Theoretic Approach for Personal Mobility Tracking in PCS Networks," *Wireless Networks,* vol. 8, no. 2, pp. 121-137, 2002.

[8] Y. Birk and Y. Nachman, "User Direction and Elapsed-Time Information to Reduce the Wireless Cost of Locating Mobile Users in Cellular Networks," *Wireless Networks,* vol. 1, no. 4, pp. 403-412, Dec. 1995.

[9] C. Castellucia, "Extending Mobile IP with Adaptive Individual Paging: A Performance Analysis," *ACM Mobile Computing and Comm. Rev.,* vol. 5, no. 2, pp. 15-26, 1999.

[10] I.R. Chen and B. Gu, "A Comparative Cost Analysis of Degradable Location Management Algorithms in Wireless Networks," *The Computer J.,* vol. 45, no. 3, 2002.

[11] J.G. Cleary and I.H. Witten, "Data Compression Using Adaptive Coding and Partial String Matching," *IEEE Trans. Comm.,* vol. 32, no. 4, pp. 396-402, Apr. 1984.

[12] T.M. Cover and J.A. Thomas, *Elements of Information Theory.* John Wiley & Sons, 1991.

[13] http://cmc.cs.dartmouth.edu/data/dartmouth, Mar. 2005.

[14] I. Demirkol, C. Ersoy, M.U. Caglayan, and H. Delic, "Location Area Planning in Cellular Networks Using Simulated Annealing," *Proc. IEEE INFOCOM,* vol. 1, pp. 13-20, 2001.

[15] Deployment of Mobile Telephony Infrastructure Hindered, http://strategis.ic.gc.ca/epic/internet/inimr-ri.nsf/en/gr111954e.html, June 2006.

[16] Digital Wireless Basics, http://www.privateline.com/PCS/Frequencies.htm#anchor5612950, Aug. 2006.

[17] N. Farvardin and J. Modestino, "Optimum Quantizer Performance for a Class of Non-Gaussian Memoryless Sources," *IEEE Trans. Information Theory,* vol. 30, no. 3, pp. 485-498, May 1984.

[18] M. Feder, N. Merhav, and M. Gutman, "Universal Prediction of Individual Sequences," *IEEE Trans. Information Theory,* vol. 38, no. 4, pp. 1258-1270, 1992.

[19] R. Gau and Z. Haas, "Concurrent Search of Mobile Users in Cellular Networks," *IEEE/ACM Trans. Networking,* vol. 12, no. 1, pp. 117-130, Feb. 2004.

[20] A. Gersho and R.M. Gray, *Vector Quantization and Signal Processing.* Kluwer Academic Publishers, 1992.

[21] V. Casares-Ginerl and P. Garcia-Escallel, *On the Fractional Movement-Distance Based Scheme for PCS Location Management with Selective Paging.* Springer-Verlag, 2005.

[22] A. Harter and A. Hopper, "A Distributed Location System for the Active Office," *IEEE Network,* vol. 8, no. 1, pp. 62-70, Jan./Feb. 1994.

[23] J.S.M. Ho and I. Akyildiz, "Mobile User Location Update and Paging under Delay Constraints," *ACM/Kluwer Wireless Networks,* vol. 1, no. 4, pp. 413-425, Dec. 1995.

[24] A. Jagoe, *Mobile Location Services: The Definitive Guide.* Scitech Publishing, 2003.

[25] B. Liang and Z.J. Haas, "Predictive Distance-Based Mobility Management for PCS Networks," *Proc. IEEE INFOCOM,* Mar. 1999.

[26] G.L. Lyberopoulos, J.G. Markoulidakis, D.V. Polymeros, D.F. Tsirkas, and E.D. Sykas, "Intelligent Paging Strategies for Third Generation Mobile Telecommunication Systems," *IEEE Trans. Vehicular Technology,* vol. 44, no. 3, pp. 543-553, Aug. 1995.

[27] U. Madhow, M. Honig, and K. Steiglitz, "Optimization of Wireless Resources for Personal Communications Mobility Tracking," *IEEE/ACM Trans. Networking,* vol. 3, no. 6, pp. 698-707, Dec. 1995.

[28] A. Misra, A. Roy, and S.K. Das, "An Information Theoretic Framework for Optimal Location Tracking in Multi-System 4G Wireless Networks," *Proc. IEEE INFOCOM,* vol. 1, pp. 286-297, 2004.

[29] P. Mutaf and C. Castelluccia, "Hash-Based Paging and Location Update Using Bloom Filters," *Mobile Networks and Applications,* vol. 9, no. 6, pp. 627-631, 2005.

[30] http://www.ofcom.org.uk/static/archive/ra/topics/mpsafety/school-audit/stewqa.htm#bs2, 2007.

[31] A. Orlitsky, "Worst-Case Rate of Scalar vs. Vector Quantization," *IEEE Trans. Information Theory,* vol. 48, no. 6, pp. 1393-1409, June 2002.

[32] G.P. Pollini and I. Chih-Lin, "A Profile-Based Location Strategy and Its Performance," *IEEE J. Selected Areas in Comm.,* vol. 15, no. 8, pp. 1415-1424, 1997.

[33] C. Rose, "Minimizing the Average Cost of Paging and Registration: A Timer-Based Method," *Wireless Networks,* vol. 2, pp. 109-116, 1996.

[34] C. Rose and R. Yates, "Ensemble Polling Strategies for Increased Capacity in Mobile Communications Networks," *Wireless Networks,* vol. 3, no. 2, pp. 159-167, May 1997.

[35] C. Rose and R. Yates, "Minimizing the Average Cost of Paging under Delay Constraints," *Wireless Networks,* vol. 1, no. 2, pp. 211-219, 1995.

[36] A. Roy, A. Misra, and S.K. Das, "The Minimum Expected Cost Paging Problem for Multi-System Wireless Networks," *Proc. IEEE/ACM Workshop Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt '04),* 2004.

[37] A. Roy, A. Misra, and S.K. Das, "A Rate-Distortion Framework for Information-Theoretic Mobility Management," *Proc. IEEE Int'l Conf. Comm. (ICC '04),* vol. 27, no. 1, pp. 3937-3941, 2004.

[38] C. Shannon, "A Mathematical Theory of Communication," *Bell Systems Technical J.,* vol. 27, pp. 379-423 and 623-656, 1948.

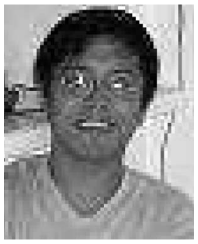
[39] L. Song, D. Kotz, R. Jain, and X. He, "Evaluating Location Predictors with Extensive Wi-Fi Mobility Data," *Proc. IEEE INFOCOM,* vol. 23, no. 1, pp. 1415-1425, 2004.

[40] S. Tabbane, "An Alternative Strategy for Location Tracking," *IEEE J. Selected Areas in Comm.,* vol. 13, no. 5, pp. 880-892, June 1995.

[41] T. Tung and A. Jamalipour, "Adaptive Location Management Strategy Combining Distance-Based Update and Sectional Paging Techniques," *Proc. 36th Hawaii Int'l Conf. System Sciences (HICSS '03),* Jan. 2003.

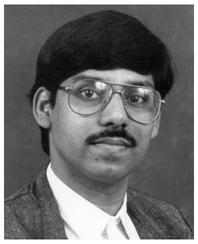
[42] G. Wan and E. Lin, "A Dynamic Paging Scheme for Wireless Communications Systems," *Proc. ACM MobiCom,* 1997.

[43] J. Ziv and A. Lempel, "Compression of Individual Sequences via Variable-Rate Coding," *IEEE Trans. Information Theory,* vol. 24, no. 5, pp. 530-536, Sept. 1978.



**Abhishek Roy** received the BE degree in computer science and engineering from Jadavpur University, Calcutta, in 2000 and the MS degree (with Best MS Research Award) from the University of Texas at Arlington, USA, in 2002. He is currently working as an assistant manager in the WiBro System Laboratory, Telecommunication R&D Center, Samsung Electronics, Suwon, South Korea. He has about five years of experience in different aspects of R&D in wireless telecommunication. His research interests include modeling and analysis of mobility management, quality of service (QoS), and voice over IP (VoIP) in 3G and 4G wireless systems. He has published with more than 15 international conferences and five international journals. He is a member of the IEEE.

**Archan Misra** received the BTech degree in electronics and communication engineering from the Indian Institute of Technology (IIT), Kharagpur, India, in July 1993 and the PhD degree in electrical and computer engineering from the University of Maryland, College Park, in May, 2000. He is a research staff member at the IBM T.J. Watson Research Center, Hawthorne, New York, where he works on algorithms and protocols for high-speed wireless networking, middleware for presence-based context-aware computing, and network management technologies for converged networks. He is presently on the editorial boards of the *IEEE Wireless Communications Magazine* and the *Journal of Pervasive and Mobile Computing* and he also chairs the IEEE Computer Society's Technical Committee on Computer Communications (TCCC). His current research projects include middleware for Session Initiation Protocol (SIP)-based "rich presence" in converged networks, high-performance wireless mesh networks, and sensor data management techniques for healthcare and remote monitoring scenarios. He has published extensively in the areas of wireless networking and pervasive computing and is a coauthor on papers that received the Best Paper Awards at the Fifth ACM Workshop on Wireless Mobile Multimedia (WOWMOM 2002) and the 20th IEEE Military Communications Conference (MILCOM 2001). He is a member of the IEEE.



**Sajal K. Das** received the BSc degree (with honors) in physics from Narendrapur Ramkrishna Mission, University of Calcutta, in 1980, the BTech degree in computer science from the University of Calcutta in 1983, the MS degree in computer science from the Indian Institute of Science, Bangalore, in 1984, the PhD degree in computer science from Washington State University, Pullman, in 1986, and the PhD degree in computer science from the University of Central Florida, Orlando, in 1988. He is a distinguished scholar professor of computer science and engineering and is the founding director of the Center for Research in Wireless Mobility and Networking (CReWMaN) at the University of Texas at Arlington (UTA). He is also a visiting professor at the Indian Institute of Technology, Kanpur, an honorary professor at Fudan University, Shanghai, and a visiting scientist at the Institute of Infocomm Research (I2R), Singapore. He is the editor in chief of the *Journal of Pervasive and Mobile Computing* and serves as an associate editor of the *IEEE Transactions on Mobile Computing*, *IEEE Transactions on Parallel and Distributed Systems*, *ACM/Springer Wireless Networks*, and *Journal on Emergent Parallel and Distributed Systems*. He serves on the IEEE Computer Society's TCPP and TCCC executive committees and on the advisory boards of several cutting-edge companies. He served as the general or program chair and a TPC member of numerous IEEE and ACM conferences. His research interests include the design and modeling of smart environments, mobile and pervasive computing, resource and mobility management in wireless networks, ad hoc and sensor networks, mobile internet architectures and protocols, security, distributed and grid computing, applied graph theory, and game theory. He has published more than 400 papers in leading journals and conferences, holds five US patents, and has coauthored the book *Smart Environments: Technology, Protocols and Applications* (John Wiley, 2005). He has received numerous awards, including five Best Paper Awards in conferences like ACM MobiCom '99, the Fourth Annual IEEE International Conference on Pervasive Computing and Communications (PerCom 2006), the Third ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM 2000), the 15th International Conference on Information Networking (ICOIN 2001), and the 11th Workshop on Parallel and Distributed Simulation (PADS 1997). He is a senior member of the IEEE.