Wireless Networks 9, 7–20, 2003

Architecture and Algorithms for Scalable Mobile QoS

BAHAREH SADEGHI * and EDWARD W. KNIGHTLY
Department of Electrical and Computer Engineering, Rice University, Houston, TX 77005, USA

Abstract. Supporting Quality of Service (QoS) is an important objective for future mobile systems, and requires resource reservation and admission control to achieve. In this paper, we introduce an admission control scheme termed Virtual Bottleneck Cell, an approach designed to scale to many users and handoffs, while simultaneously controlling "hot spots". The key technique is to hierarchically control an aggregated virtual system, ensuring QoS objectives are satisfied in the underlying system without per-user resource management such as advanced reservations of bandwidth in a user's predicted future locations. We develop a simple analytical model to study the system and illustrate several key components of the approach, such as balancing the conflicting design objectives of high utilization, scalability, and ensured QoS. We formulate the problem of clustering cells into virtual system as an optimization problem and propose a heuristic adaptive clustering algorithm as a practical solution. Finally, we evaluate the scheme by developing a simple analytical model, devising an optimal off-line algorithm, and performing simulations of a two-dimensional network.

Keywords: admission control, QoS, adaptive clustering, mobile networks

1. Introduction

Next generation wireless and mobile devices will support applications ranging from traditional cellular voice to web browsing and interactive multimedia applications. Concurrently, packet networks are evolving from the best effort model of the past to networks which support multiple service classes [26]. An important challenge is to incorporate user *mobility* into future network service models and resource management algorithms [24].

To satisfy the performance demands of such future mobile users, the network must limit the severity, frequency, and duration of overload due to handoffs and user mobility. While special-purpose scheduling algorithms can mitigate the effects of error prone wireless links [7,20,21], *admission control* and resource reservation must ultimately be employed to pro-actively ensure that mobile users' Quality-of-Service requirements can be satisfied.

In cellular *voice* systems, guard channels provide a simple but effective mechanism for controlling QoS by statistically allocating capacity in each cell exclusively for users handing off [23]. In contrast, the expected diversity of future applications, traffic types, QoS requirements, and mobility patterns has prompted a significant research effort for alternate solutions [1,5,6,8,12,14,18,22,24,25]. For example, capacity can be reserved for a particular user at future times in nearby cells as dictated by the mobile unit's current location and velocity, past mobility behavior, and/or other model-based prediction techniques.

While such reservation schemes have demonstrated significant performance advantages over even well-engineered guard channels, they incur two limitations when applied to future networks. First, per-user prediction and dynamic resource reservation place computation and communication burdens on the network's infrastructure which increase polynomially with the number of users and handoffs [10]. Hence, the scalability and applicability of such solutions to future micro- and pico-cellular networks is not well established. Second, as illustrated in figure 1, Quality of Service is assured only with the proper mechanisms at all time scales, ranging from channel access at the "bit" time scale, to admission control at the session time scale. A key difficulty encountered with location prediction is that it must bridge two fundamentally different time scales, and extend location estimations at the handoff time scale to session QoS measures at the session time scale. Unfortunately, this gap can widen in pico-cellular environments in which cell residence times decrease while session lifetimes do not. Finally, we have experimentally found that advance reservation schemes require accurate prediction of both location and *time*, namely even if a user's future locations are precisely known *a priori*, admission control can still be conservative if the handoff *times* corresponding with those locations are not also known.

In this paper, we develop a new admission control algorithm which achieves scalable QoS control of mobile users. Our key technique is to aggregate users and a cluster of cells into a Virtual Bottleneck Cell (VBC) in such a way that by controlling parameters of the virtual cells we ensure that QoS is satisfied in the underlying system. We develop an approach to characterize and control system QoS via two parameters. The first, which we refer to simply as "overload", is the mean fraction of capacity that is over-booked: it reflects the extent to which bandwidth demand exceeds available capacity, and

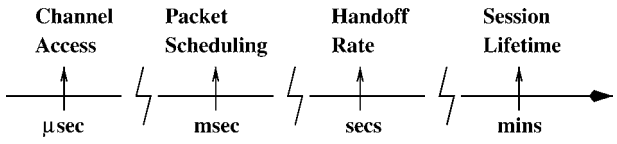


Figure 1. Time scales of system control.

* Corresponding author.

consequently, the severity and frequency that users must adapt to lower bandwidths. The second parameter is the "outage time scale": when a cell is overloaded, this refers to the mean time until the cell returns to a non-overloaded state.

Our approach is motivated by two key design objectives. First, by managing resources in an aggregated virtual system, we control system QoS without requiring accurate predictions of the times and locations of each user's future handoffs. In this way, we ensure that our solution is scalable to a large number of users and handoffs, such as in micro- and pico-cellular environments with a potentially large number of handoffs per user. Second, we ensure that when parameters of the aggregated VBC are properly controlled, QoS levels in cells of the actual systems are also guaranteed to be satisfied, even in environments with heterogeneous spatial demands. In other words, our approach manages "hot spots" and system bottlenecks to simultaneously achieve scalability and efficient and accurate resource control.

As VBC is an aggregate QoS scheme, an important issue is the mechanism for aggregation and deaggregation of cells, i.e., the cell clustering policy. To address this issue, we formulate the clustering policy as a constrained optimization which seeks to maximize system utilization subject to a limit on inter-cluster handoffs. This formulation has the effect of achieving a balance of scalability, strong QoS assurance, and efficient resource utilization. We show that a closed form solution for the optimal clustering policy cannot be obtained without strong assumptions on user mobility patterns, namely, a detailed and accurate stochastic model of user mobility. As such models are not available, we propose a heuristic adaptive clustering algorithm, with the goal of accurately approximating the ideal solution while responsively adapting to changes in user mobility behavior and system conditions. The algorithm's key technique is to discover the correlations among occupancies of neighboring cells, and form clusters based on these correlations only as resources become overloaded.

To analyze the performance of the VBC algorithm and illustrate several important design issues, we analyze the system in three ways: theoretical analysis, comparison with optimal off-line benchmarks, and simulation.

First, we develop a simple analytical model to study this system. We illustrate our approach's ability to control system bottlenecks, and explore the implications of heterogeneous user demands on system performance.

Second, we develop a technique based on [4,10], which we term *Perfect Knowledge Algorithm* (PKA). PKA serves as a benchmark for evaluating algorithms which manage mobile QoS. We show that it is the *optimal off-line* admission control algorithm in that it obtains the maximal admissible region subject to the empirical QoS constraints and system rules. In particular, PKA considers a set of users' admission requests in which each user has an associated bandwidth demand and mobility pattern (i.e., times and locations of handoffs over the duration of the session), and the goal is to select the optimal subset of users for admission which maximizes the system's utilization while satisfying the required QoS. We show that the general problem can be formulated as a non-linear constrained optimization problem. Moreover, for the special case of zero probability of handoff drop, we show that the solution can be expressed as a *linear* constrained optimization problem, and computed efficiently using standard tools.

Finally, we perform an extensive set of simulations and admission control experiments using a two-dimensional 64-cell network. We first study the performance and characteristics of the adaptive clustering algorithm. Then we utilize PKA to assess the performance of our approach in more realistic scenarios. We find that the VBC algorithm with the adaptive clustering policy is able to control the admissible region within a narrow region.

The remainder of this paper is organized as follows. In section 2 we describe the system model and role of admission control. In section 3 we develop the VBC approach and in section 4, we define the clustering problem and propose an adaptive clustering algorithm. To analyze the system, in section 5, we introduce an analytical model to study the problem. In section 6 we describe an optimal off-line benchmark which we apply to simulation experiments in section 7. Finally, in section 8, we conclude.

2. System model

The system model that we consider is depicted in figure 2. It consists of a collection of base stations connected to routers or switches which are in turn inter-connected over a backbone network. Multiple service classes are provided over the backbone network via a mechanism such as [3] and extended to the wireless network via a wireless/mobile QoS architecture (e.g., [16,24]). We focus on traffic classes requiring higher

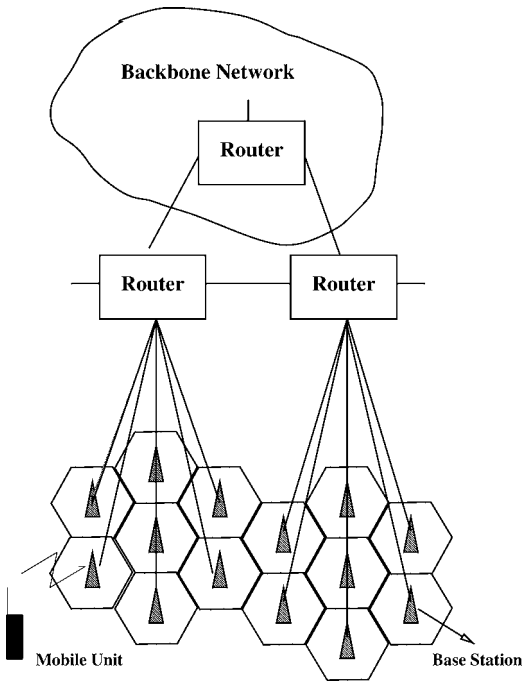


Figure 2. System model.

priority than "best-effort" service, including not only users of interactive multimedia applications, but also users of traditional applications such as web browsing that wish to subscribe to a premium service with bounded outage times.

In such a mobile-QoS network, *admission control* is employed to ensure that each traffic class is allocated sufficient system resources to meet its quality of service demands. Moreover, for efficient resource utilization, such reservations and hence all QoS measures are *statistical* in nature; consequently, demanded bandwidth will at times exceed the available capacity due to overbooking of system resources. The goal of the admission control algorithm is to limit the frequency, severity, and durations of such overload periods to within prespecified limits defined by the service. Indeed, the extent to which demand overloads the system capacity and the time scales of the overload will be the key Quality of Service metrics that we consider. Below, we formally define these QoS metrics and develop an approach to provision resources to meet these objectives in a coarse-grained manner.

Finally, we note that during the overload periods, some established sessions will obtain a reduced service, and be forced to temporarily adapt to a lower bandwidth. Mechanisms and policies for adapting to such overload situations are developed in [9,15,19], for example, and are beyond the scope of this paper.

Throughout, we focus on a single QoS-controlled class, and denote the available capacity or bandwidth of cell j to the users in the QoS class by C_j, and the demanded bandwidth or occupancy at time t of this same group of users by $\Omega_j(t)$.

3. Virtual Bottleneck Cell (VBC)

In this section, we first overview our design goals for scalable system control. We then introduce Virtual Bottleneck Cell (VBC) as our approach towards achieving these design goals and sketch a particular algorithm as an example of controlling QoS in the VBC, and hence in the system itself. We describe the key QoS metrics that we use to manage a cluster of cells and show how they can be empirically measured for an online admission control algorithm.

3.1. Scalable control

To control QoS in mobile networks in a scalable way, we propose a novel approach termed Virtual Bottleneck Cell. With VBC, a network of virtual cells is managed to aggregate not only the behavior of individual users within the cells, but also of individual cells within a cluster. We will show that system-wide QoS can be effectively controlled by managing resources in the virtual system, rather than by allocating capacity on a per-user or per-cell basis.

As illustrated in figure 3, we aggregate the state of clusters of cells into VBCs with the following objectives:

- *Scalable, low overhead QoS control.* For many mobile users with a potentially large number of handoffs in micro/pico-cellular environments, our approach manages QoS by controlling *aggregated* system parameters rather than tracking individual users through the system. We provide a concise representation of the virtual system to significantly reduce communication and computational overheads.
- *QoS assurance in "hot spots" and system bottlenecks.* We ensure that by controlling parameters in the virtual system, we guarantee certain QoS levels in cells of the actual systems, even in environments with heterogeneous spatial demands.

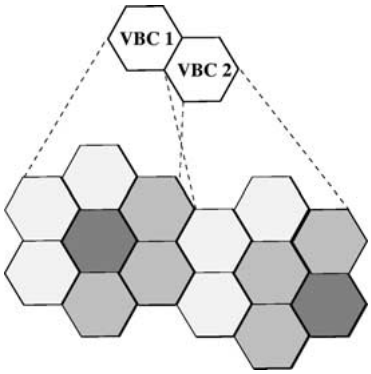


Figure 3. VBC illustration.

3.2. Sketch VBC algorithm

Here, we outline a particular algorithm towards achieving the objectives above. We consider two Quality of Service measures: overload, and the time scale of overload.

Consider a set of cells $\mathcal{C}$ which form a cluster, and a group of K clusters $\mathcal{C}_1, \ldots, \mathcal{C}_K$, for which QoS is to be provisioned. We construct a Virtual Bottleneck Cell for each cluster and characterize the overload of cluster $\mathcal{C}_k$ by

$$\gamma_k \stackrel{\triangle}{=} \max_{j \in \mathcal{C}_k} \frac{E(\Omega_j - C_j)^+}{C_j}, \tag{1}$$

where $(x)^+$ denotes $\max(x, 0)$. This measure reflects the frequency and severity of overflow, i.e., how *often* overload occurs, and the extent to which the system is overloaded. Throughout this paper, we will refer to γ as simply "overload".

Second, we define the overflow time scale of VBC k by

$$T_k \stackrel{\triangle}{=} \max_{j \in \mathcal{C}_k} \left\{ E\tau_j \colon \Omega_j(u) > C_j \text{ for } u = [s, s+\tau] \right\}, \tag{2}$$

which denotes the maximum mean duration of overflow of any cell in the VBC. (See also [2] for a related measure of a user's "degradation period ratio".)

Notice that aggregation of the cells' behavior into the virtual cells via the "max" in equations (1) and (2) ensures that by controlling parameters of the VBC, the QoS condition is also satisfied in each constituent cell of the cluster.

Thus, γ_k describes the severity of VBC k's overload, whereas T_k describes the durations over which demand exceeds capacity. We provision resources according to these

QoS measures rather than the more traditional probability of handoff drop in order to generalize our solution to systems in which users *adapt* to overflow situations rather than having their session dropped all together. For example, rather than dropping sessions upon overflow, users may prefer to temporarily renegotiate to lower reserved bandwidths or even incur temporary service outages.

To maintain Quality of Service to within prespecified levels set by the class requirements, we employ admission control and resource reservation as follows. First, a new user is assigned a bandwidth Ω_{new} according to its traffic characteristics and the underlying medium access scheme (see [13], for example). Next, the router managing the cluster for which the new user is requesting access (see figure 2) only admits the user to the requested class if the predictions of the two aforementioned QoS measures are within the class' requirements. Hence, for a particular cluster k, the empirical overload of the VBC, after incorporating the impact of the new user, is adaptively computed using measurements at the base stations constituting the cluster as

$$\widehat{\gamma}_k = \frac{1}{W} \max_{j \in \mathcal{C}_k} \frac{1}{C_j} \sum_{s=t-W}^{t} \max\big(\widehat{\Omega}_j(s) + \Omega_{\text{new}} - C_j, 0\big), \quad (3)$$

where W denotes the measurement window and $\widehat{\Omega}_j(s)$ denotes the measured occupancy of cell j at time s.

Similarly, denote

$$O_j(s) = 1\big(\widehat{\Omega}_j(s) + \Omega_{\text{new}} > C_j\big)$$

as an indicator function of overload in cell j at time s, including resources that would be demanded by the new user if it visits cell j. Then the VBC's mean outage time scale is given by

$$\hat{T}_k = \max_{j \in \mathcal{C}_k} \frac{\sum_{s=t-W}^{t} O_j(s)}{\sum_{s=t-W}^{t} 1(O_j(s) > O_j(s-1))}. \quad (4)$$

Thus, when a new user requests a QoS-controlled session in a particular cell, the network admits the session at the requested QoS level only if the predicted service levels as given by equations (3) and (4) are satisfied in the corresponding virtual cell. Consequently, the user will have limited durations and severity of outages while moving within the boundaries of the cluster.

Notice that the admission test ensures that if the new user had been active for the past W slots, the empirical QoS measures would have been satisfied in every cell of the VBC for that duration. If in the future, users move in such a way that the empirical QoS measures go above their target values, future sessions will be blocked based on the updated measurements of the network conditions. Similarly, as users exit the system, the measured parameters of equations (3) and (4) decrease over time allowing new users to be admitted to the system. This adaptiveness of the admission control algorithm reveals the importance of the measurement window: proper setting of W is required for any measurement-based algorithm, as it must strike a balance between system responsiveness and stability. In this case, it should be set to be larger than the mean cell residence time but smaller than the mean session lifetime. Specifically, setting it smaller than the residence time will not incorporate the key system feature that is being controlled, viz., outages due to handoffs; moreover, setting W larger than the session lifetime will skew the QoS predictions by including the effects of sessions that no longer exist.

4. VBC cell clustering

In the previous sections we showed that for a given cluster configuration, VBC admission control provisions resources based on aggregated information of users' and cells' empirical characteristics.

Here, we address the algorithm by which cells are clustered into virtual cells. An efficient clustering policy must achieve a balance in cluster size: networks divided into smaller clusters will have aggregated VBCs that accurately represent the conditions throughout the cluster. However, smaller clusters also increase the number of inter-cluster handoffs, which are undesirable since no resources are reserved outside of a user's initial cluster.[1] On the other hand, larger clusters will *reduce* the number of inter-cluster handoffs, yet will also reduce the correlation among loads in a cluster's cells. In particular, users will not be admitted if there is overload anywhere in the VBC. Thus, if the VBC is too large (consists of too many cells), users will be unnecessarily rejected.

4.1. Optimal static clustering

Standard clustering techniques seek to find similarities in a set of objects and group them such that objects within a cluster are similar to one another, and dissimilar from objects in other clusters. For example, a distance measure among objects can be defined so that objects are grouped in different clusters to minimize the total distance between objects in each cluster [17].

In contrast, the *cell* clustering problem must not only cluster cells with similar workloads, but must also account for user mobility among cells, i.e., the extent to which neighboring cells impact each other's overload behavior. More importantly, the ideal clustering policy is clearly dependent on the system inputs (user mobility behavior and demand), and hence, clustering must be *dynamic* in practice. Thus, we first devise the optimal static clustering policy, and then develop a heuristic adaptive algorithm to approximate this behavior.

We formulate the clustering problem as follows. Given a subnetwork of M cells along with their empirical overload and outage time scale measures, find the combination of groups of cells (clusters) such that first, each cluster is connected, i.e., it consists of neighboring cells, and second, applying the VBC admission control in the network (which ensures overload and time scale measures requirements are

[1] We therefore consider the probability of inter-cluster handoff as a general measure of system QoS.

satisfied), minimizes the probability of inter-cluster handoff (which as defined below is a measure of the service certainty) and at the same time maximizes the network utilization.

Let $\mathcal{K}$ denote any possible clustering policy of a system of size M during the observation period of T time units. Then $\mathcal{K}$ can be written as a T by M matrix, where row t, $t \leqslant T$, is the cluster configuration of the M cells at time t. Consider a set of users $\mathcal{S}$ requesting admission to the network. For each user $x \in \mathcal{S}$, let its mobility pattern be defined by the matrix A^x of indicator functions [10], such that

$$A^x_{j,t} = 1\big(L(x,t) = j\big), \tag{5}$$

where $L(x,t)$ is the number of the cell in which user x is located at time t. Moreover, let $F^{\mathcal{K}}(x,t)$ denote the cluster number, $L(x,t)$ belongs to at time t. Also C_j, $j = 1,\ldots,M$, and $\Omega^{\mathcal{A}}(j,t)$ respectively denote the capacity of cell j, and the occupancy of cell j at time t for a set of admitted users $\mathcal{A} \in \mathcal{S}$.

For the observation period T, given a clustering policy $\mathcal{K}$ and a set of admitted users $\mathcal{A}$ the system utilization can be expressed as

$$U_T^{\mathcal{K},\mathcal{A}} = \frac{\sum_{j=1}^{M}\sum_{t=1}^{T}\Omega^{\mathcal{A}}(j,t)}{T\sum_{j=1}^{M}C_j} \tag{6}$$

with the empirical probability of inter-cluster handoff given by

$$\hat{P}_{\mathrm{HO}}^{\mathcal{K},\mathcal{A}}(T) = \frac{\sum_{x\in\mathcal{A}}\sum_{s\leqslant T}1(F^{\mathcal{K}}(x,s) \neq F^{\mathcal{K}}(x,s-1))}{\sum_{x\in\mathcal{A}}\sum_{s\leqslant T}1(L(x,s) \neq L(x,s-1))} \tag{7}$$

which is the ratio of inter-cluster handoffs to the total handoff attempts.

Let $P_{\overline{\mathcal{K}}}(\mathrm{HO})$ denote the probability of inter-cluster handoff for clustering policy $\overline{\mathcal{K}}$, optimized in sense of minimizing the probability of inter-cluster handoff in the network. An example of $\overline{\mathcal{K}}$ for a stationary off-line clustering is having all cells in the subnetwork in one cluster. Note that even in that case, the probability of inter-cluster handoff is greater than zero, since users can still leave the subnetwork. We also denote the probability of inter-cluster handoff for any given clustering algorithm $\mathcal{K}$, by $P_{\mathcal{K}}(\mathrm{HO})$.

We then define the *Cluster Isolation Factor* (CIF) as

$$\mathrm{CIF} = \frac{1 - P_{\mathcal{K}}(\mathrm{HO})}{1 - P_{\overline{\mathcal{K}}}(\mathrm{HO})}, \tag{8}$$

so that with no clustering (i.e., when each cell forms a cluster by itself) the probability of inter-cluster handoff in the network is 1, and hence, CIF equals 0; on the other hand, having the cluster configuration which minimizes the inter-cluster handoffs in the network, we have the maximum possible isolation among the clusters and CIF equals 1.

An optimal clustering policy $\mathcal{K}^*$ is the one that applied to the network along with the VBC admission control algorithm, maximizes utilization U, defined in equation (6), subject to the empirical QoS requirement $\hat{P}_{\mathrm{HO}}$, or equivalently, $\widehat{\mathrm{CIF}}$. An analytical solution to this optimization problem would require a model of the cell occupancies as in equation (6), which is a function of behavior of all users' mobility characteristics as in equation (5). However, due to the complex nature of a group of users' natural behavior, there is currently no suitable model available for the users' mobility pattern and, hence, the cell occupancies. Moreover, even in the simple case of static clustering with fixed sized clusters, the complexity of the problem for a one-dimensional array of M cells is 2^{M-1}. In general, the clusters can have different sizes and shapes varying with time and the only constraint on the shape of the clusters is connectivity, i.e., starting from any cell in the cluster, one should be able to go to all the other cells of the cluster without leaving the cluster, thus further increasing the complexity of the solution.

4.2. Adaptive clustering algorithm

Motivated by the intractability and dynamic nature of an ideal clustering algorithm, we now develop a heuristic adaptive clustering algorithm as an approximation to the above optimization problem. In designing this algorithm, we exploit the mobility patterns of users' movements in order to form the clusters.

Figure 4 illustrates the design objectives by depicting the neighboring cells of a congested cell A. In order to make an accurate decision regarding admission of new calls in this group of cells, we measure the aggregated amount of handoffs between cell A and each of its neighbors. If for example, there are excessive handoffs from cell B to cell A, then admission of new users in cell B affects the QoS metrics in cell A, since the users of cell B handoff to cell A with high probability. Hence, these two cells must be annexed to form a cluster. On the other hand, if cell F is annexed with A to form a cluster, and the aggregated amount of handoffs from cell F to cell A is too low, then in making decision on admission of any user in cell F we are considering the load of cell A. But, given that cell A is overloaded and there are not many handoffs from cell F to cell A, it results in unnecessary rejection of admission requests in cell F, which reduces the system utilization.

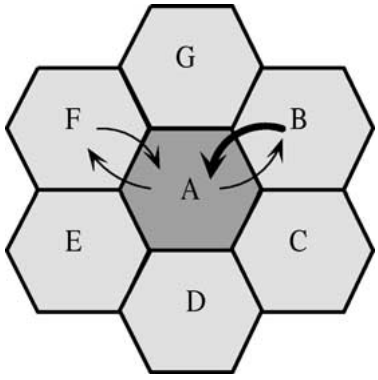


Figure 4. Clustering in neighborhood of a hot spot.

Adaptive Clustering Algorithm

```
1. Initial Clustering: M Clusters of Size 1;
2. for (j = 1, ..., M; t = 0, ..., T) {
3.   if (Ω(j, t) ⩾ βC_j) {
4.     for (all-neighbors-of-cell j) {
5.       if ([∑_W BW_in ⩾ α_h C_j]∩[neighbor-not-in-cluster])
6.         Add-Cluster-of-Neighbor-to-Cluster-of-Cell j;
7.     }
8.   }
8.   else if (Ω(j, t) < βC_j) {
9.     for (all-neighbors-of-cell j) {
10.      if ([neighbor-in-cluster] ∩ [separation-condition (cell j,
           neighbor) ]) {
11.        if (neighbor-not-connected-to-any-other-cell-in-cluster)
12.          Separate-Neighbor-of-Cluster-of-Cell j;
13.        else if (neighbor-connected-to-cluster) {
14.          separation-var = 1;
15.          for (all-cells-in-Cluster-connected-to-Neighbor) {
16.            separation-var = separation-var ∩ separation-condition
                 (Cell, Neighbor);
17.          }
18.          if (separation-var = 1)
19.            Separate-Neighbor-of-Cluster-of-Cell j;
20.        }
21.        else
22.          return;
23.      }
24.    }
25.  }
26. }

27. separation-condition (cell i, cell k) {
28.   if ([Ω(i, t) < βC_i] ∩ [∑_W BW_in < α_l C_i]){
29.     if ([Ω(k, t) < βC_k] ∩ [∑_W BW_in < α_l C_k]) {
30.       return 1;
31.     }
32.   }
33.   return 0;
34. }
```

Figure 5. Adaptive clustering algorithm.

More formally, the adaptive clustering algorithm is presented in pseudocode in figure 5 and described as follows.

Initial state. The algorithm starts from the initial state where each individual cell in the subnetwork forms a cluster of size one.

Clusters annexation. Consider cell j in figure 6(a) which belongs to cluster B. Whenever the occupancy of cell j exceeds some multiple of the capacity of cell βC_j, $\beta \geqslant 0$, the handed-in bandwidth from the neighboring cells of cell j, cells i and k in figure 6(a), during the past W time slots is measured, where W is a prespecified fixed window size. As soon as the measured value of handed-in bandwidth for any of the neighbors of cell j, cell i in our example, exceeds $\alpha_h C_j$, $\alpha_h > 0$, the original cluster of the neighboring cell will join the cluster of cell j to form a new cluster.

Cell separation. If the occupancy of cell j in figure 6(b) becomes lower than βC_j, then the handed-in bandwidth of

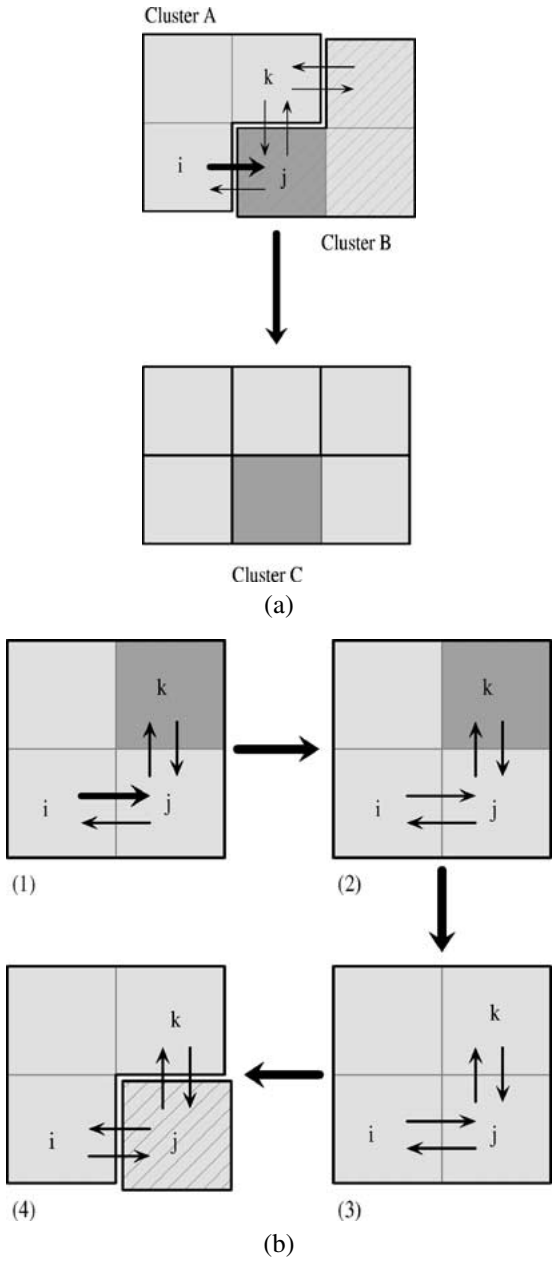


Figure 6. Process of (a) clusters annexation and (b) cell separation.

those neighbors of cell j which are in the same cluster with cell j, cells i and k in figure 6(b), will be measured for the past W time slots. If this value is less than $\alpha_l C_j$, $0 \leqslant \alpha_l \leqslant \alpha_h$, and the neighboring cell is neither connected to any other cells of the cluster nor overloaded, it will separate from the cluster to form a cluster by itself. In the case that the neighboring cell is connected to some other cell in the cluster, it will remain a part of the cluster until the condition for separation holds for all its neighboring cells which belong to the same cluster. In figure 6(b) we see that in state (2), considering only cell i, cell j can leave the cluster, but since cell k is in overload status, it does not let any of its neighboring cells separate from the cluster. In state (3), cell k is no more overloaded, and hence, cell j can leave the cluster and form a cluster by itself (state (4)).

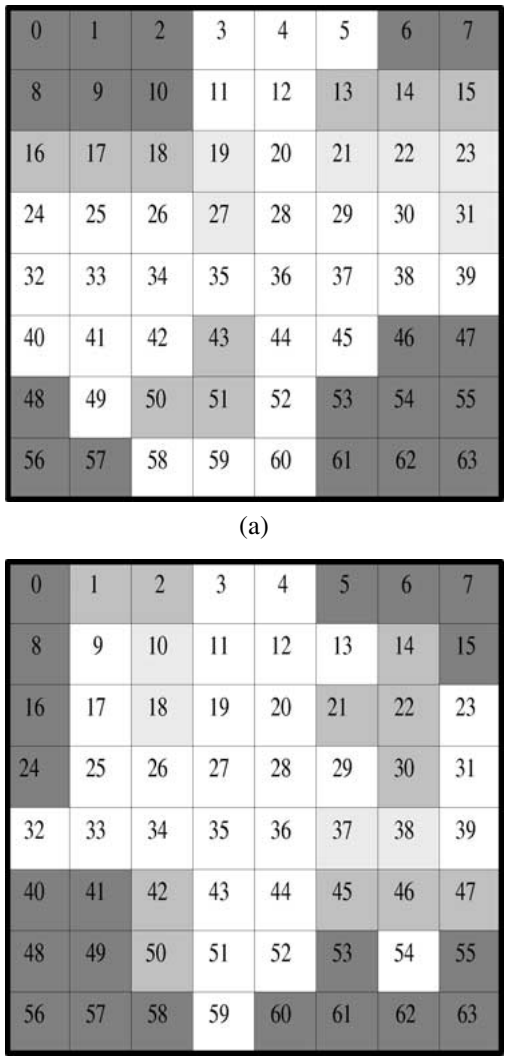


Figure 7. An example of adaptive clustering. (a) $t = 25$. (b) $t = 100$.

If when one cell leaves its original cluster, it results in separation of two or more parts of the original cluster, each separated part will form a new cluster as well as the separated cell itself.

By applying this algorithm to the network, the clusters will be formed around the hot spots and bottleneck cells in the network, and as the time passes and the congested areas change (e.g., from downtown in the morning to the suburb in the afternoon) then the clusters will also change the location and follow the area of congestion.

The clustering algorithm has several parameters. The first, β, controls the level of sensitivity to overload. The smaller β is, the sooner the clusters form. Hence, in average there will be larger clusters in the network, reducing utilization but increasing CIF. The remaining parameters, α_h and α_l, control the adaptivity of the algorithm. Larger values of α_h increase the time for two clusters to annex and smaller values of α_l increase the time for a cell to separate from a cluster. Thus, when α_h and α_l are larger, clusters are smaller in size so that utilization is higher and CIF is lower.

4.3. Example

Figure 7 depicts an example of the adaptive clustering algorithm. Considering the subnetwork of 64 cells as shown in the picture, the system starts at time $t = 0$, from the initial state where each cell forms a cluster by itself. The users are introduced to the network with Downtown Mobility Model as discussed in section 7.1, therefore, the four cells 0, 7, 56, and 63 are with high probability the destination of the users, assigned to each user upon origination of the call.

Figure 7(a) shows the configuration of the clusters in the network at time $t = 25$. The cells with white color are clusters of size one, where the ones with the same shade which are connected to each other, form clusters of higher sizes. For example, the cells 6 and 7 form a cluster and cells 13, 14, and 15 form another cluster.

As shown, the formation of clusters is concentrated around the hot spots (cells 0, 7, 56, and 63) of the system. The adaptive clustering algorithm implies that those two neighboring cells join each other to form a cluster, that the moving average of the amount of handed bandwidth between them exceeds a prespecified value. As time passes and the users' mobility patterns change, the moving average of the handed-off bandwidth between cells also varies. The adaptive clustering algorithm follows these variations and reforms the clusters. Comparing figures 7(a) and (b), we see the changes in clusters at time $t = 100$ compared to time $t = 25$. The changes in clustering configuration is due to the existing randomness in the movements of the users. For example consider cell number 49; the occupancy of this cell both at $t = 25$ and $t = 100$ is less than 80% of its capacity, which is the threshold for starting clustering process in this example. But at $t = 25$ it forms a cluster by itself, whereas at $t = 100$ it has annexed to its neighbors to form a bigger cluster. The reason is that the aggregated handed-in bandwidth from cell 49 to cell 48 (which is a congested cell), measured during the past 10 time slots, exceeds the threshold of 30% of the capacity of cell 48 (for this example). The figures show that although the clusters adaptively change in shape and size, their concentration is around those cells that are highly occupied and are considered the bottlenecks of the system. Hence, the adaptive clustering algorithm is successful in finding such cells and forming clusters around them.

Finally, we note that in practice, the underlying physical architecture of the network is another factor that affects the efficiency of the clustering policy. Having cells which are connected to different subnetworks or routers in one cluster increases communication overhead and may not be desirable. We have focused on the clustering problem within the set of cells connected to the same router.

5. Analytical and numerical investigation

In this section, we introduce a simple analytical model to study several aspects of admission control using virtual bottleneck cells. Our model consists of a one-dimensional cel-

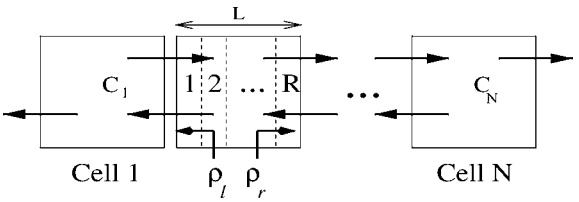


Figure 8. Network model.

lular array similar to one which might be used in modeling highways.

As shown in figure 8, the array consists of N cells with the same length L, and cell j having capacity C_j, $1 \leqslant j \leqslant N$. The arrival of users and their speed of movement is deterministic. We consider time to be slotted and denote the rate of new call arrivals by λ. Further, let ρ_r be the fraction of users that move to the right and ρ_l be the fraction of users that move to the left such that $\rho_r + \rho_l = 1$. Upon establishing a new session, a user immediately begins moving with constant velocity v. Thus, each cell can be viewed as being divided into $R = L/v$ regions. Moreover, new users arrive in a cell so that the number of new arrivals in any time slot is the same in all regions of the cell. Finally, we assume that the duration of a session's lifetime is exponentially distributed with mean $1/\mu$, so that μ is the rate at which users depart from the system.

5.1. Overload

To calculate the overload γ defined in equation (1), we first compute the severity of the overload in each cell j by calculating the expected value of the amount of resources demanded beyond the available capacity as

$$E(\Omega_j - C_j)^+ = \sum_{i=1}^{\infty} i \Pr(\Omega_j = i + C_j). \tag{9}$$

Note that since $\Pr(\Omega_j = i + C_j)$ is the fraction of time that $i + C_j$ users are active, $E(\Omega_j - C_j)^+$ is the sum of occupancies beyond the available capacity weighted by the fraction of time spent in that occupancy. Thus, to calculate γ, we first compute the probability that a cell is overloaded (i.e., its demanded capacity exceeds C_j) as

$$\begin{aligned}\Pr(\Omega_j > C_j) &= \Pr(n_{hj} + n_{gj} > C_j) \\ &= \sum_{i=0}^{\max(C_j, n_{gj}^{\max})} \Pr(n_{gj} = i) \Pr(n_{hj} > C_j - i),\end{aligned} \tag{10}$$

where n_{hj} denotes the total number of hand-in calls, and n_{gj} denotes the number of calls that originated in cell j. Note that there exists an $n_{gj}^{\max}$ which is the maximum possible number of calls originated in cell j, and is obtained when all sessions that originated in cell j have a call holding time long enough to leave the cell before being terminated. We observe that only the calls that originated in the last $(R-1)$ time units may still be in the same cell (due to the users' constant velocity), and that in each time unit, λ/R users leave the cell, so that

$$n_{gj}^{\max} = \frac{R+1}{2}\lambda.$$

Let T_{h} denote the call holding time for a specific session so that its distribution is given by

$$F(\tau) = \Pr(T_{\mathrm{h}} \leqslant \tau) = 1 - \mathrm{e}^{\mu\tau}.$$

Then, to compute equation (10), we define the function

$$\Theta(x, y, \tau) = \binom{x}{y} \big(1 - F(\tau)\big)^y F(\tau)^{x-y},$$

and the vectors

$$\overline{\Lambda}_j(n, l) = [n_{j0}, \ldots, n_{jl}],$$

and

$$\overline{\Delta}_j = [\delta_{j0}, \ldots, \delta_{j(R-1)}],$$

where

$$\delta_{jl} = \min\left(i - \sum_{q=1}^{l-1} n_{jq}, \frac{\lambda}{R}(R - l)\right),$$

and each element of $\overline{\Lambda}_j(n, R-1)$, n_{jl}, represents the number of active users in region l of cell j. The probability that $(n_{j0}, \ldots, n_{j(R-1)})$ sessions are still active in the R regions of cell j is calculated by multiplying the individual probabilities of n_{jl} users being active in region l of cell j, for $l = 0, \ldots, R-1$. The different combinations of the number of users in various regions such that the total number of users is less than or equal to i must then be considered. The summation over these different combinations yields $\Pr(n_{gj} \leqslant i)$, which is the probability that the number of users originally admitted in cell j is less than or equal to i, and is given by

$$\Pr(n_{gj} \leqslant i) = \sum_{\overline{\Lambda}_j(n,R-1)=\overline{0}}^{\overline{\Delta}_j} \prod_{r=1}^{R} \Theta\left(\frac{\lambda(R-r)}{R},\ n_{j(r-1)}, r\right),$$
$$0 \leqslant i \leqslant \frac{\lambda(R+1)}{2}. \tag{11}$$

Similarly, n_{hj} is the sum of all active users that initiated their calls in cell i, $j < i \leqslant N$, in the last $R(N - j)$ time units and are moving to the left, and also those that initiated their calls in the last $R(j-1)$ time units in cell k, $1 \leqslant k < j$, and are moving to the right. Therefore, defining the vectors

$$\overline{\Phi}_j = [\phi_{j0}, \ldots, \phi_{j((j-1)R)}]$$

with elements

$$\phi_{jl} = \min\left(j - \sum_{q=1}^{l-1} n_{jq}, \frac{\lambda}{R}\rho_r\right),$$

and

$$\overline{\Psi}_j = [\phi_{j0}, \ldots, \phi_{j((j-1)R)}]$$

with elements

$$\psi_{jl} = \min\left(j - \sum_{q=1}^{l-1} m_{jq} - \sum_{q=1}^{(j-1)R} n_{jq}, \frac{\lambda}{R}\rho_l \right),$$

we can then express $\Pr(n_{hj} \leqslant i)$ as

$$\Pr(n_{hj} \leqslant i) = \sum_{\overline{\Lambda}_j(n,(j-1)R)=\overline{0}}^{\overline{\Phi}_j} \sum_{\overline{\Lambda}_j(m,(N-j)R)=\overline{0}}^{\overline{\Psi}_j} \mathcal{X}_j \mathcal{Y}_j, \quad (12)$$

where $\mathcal{X}_j$ and $\mathcal{Y}_j$ are expressed as functions of Θ as

$$\mathcal{X}_j = \prod_{r=1}^{(j-1)R} \Theta\left(\frac{\lambda}{R}\rho_r, n_{jr}, r\right),$$

$$\mathcal{Y}_j = \prod_{v=1}^{(N-j)R} \Theta\left(\frac{\lambda}{R}\rho_l, n_{jv}, v\right).$$

Thus, combining equations (9)–(12), we have an expression for γ_k, cluster k's overload measure.

5.2. Overflow timescale

We next turn to the overflow time scale of the virtual bottleneck cell defined in equation (2). We begin by computing the distribution of the overflow time in a constituent cell under the same assumptions of the model above.

Let h denote the call handoff rate. The probability that the overflow time in cell j with capacity C_j is greater than s time units, $\Pr(\tau_j > s)$, is the probability that *more* than C_j users remain in cell j for at least s time units given that the cell is overloaded. Hence,

$$\Pr(\tau_j > s) = \sum_{m=1}^{\infty} \Pr(\Omega_j = C_j + m) \sum_{n=1}^{m} \mathcal{Z}_{js},$$

where $\mathcal{Z}_{js}$ is defined as

$$\mathcal{Z}_{js} = \binom{C_j + m}{C_j + n} \left(\mathrm{e}^{-s(\mu+h)}\right)^{C_j+n} \left(1 - \mathrm{e}^{-s(\mu+h)}\right)^{(m-n)}.$$

Thus, the overflow time scale of the VBC can be easily computed as the maximum $E\tau_j$ of all cells in the cluster.

5.3. Numerical examples

We now perform numerical investigations applying the analysis above. In figure 9, we show the results for $N = 5$, $R = 1$, $\rho_r = 2/3$, $\rho_l = 1/3$, $\lambda = 9$, and $C_j = 10$ for $1 \leqslant j \leqslant 5$. The figure depicts the measure of overload for each of the five cells, i.e., $E(\Omega_j - C_j)^+/C_j$ for $j = 1, \ldots, 5$, for different call departure rates and, hence, different mean call holding times. The plot indicates that as $1/\mu$ increases, the overload measure increases since users stay longer in the network, and hence, hand off a larger number of times.

Since the number of users who move to the right is twice the number of those who move to the left, we observe that

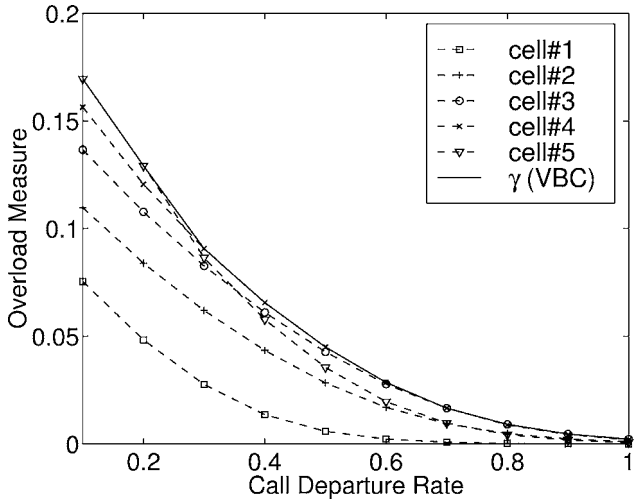


Figure 9. Overload measure versus call departure rate.

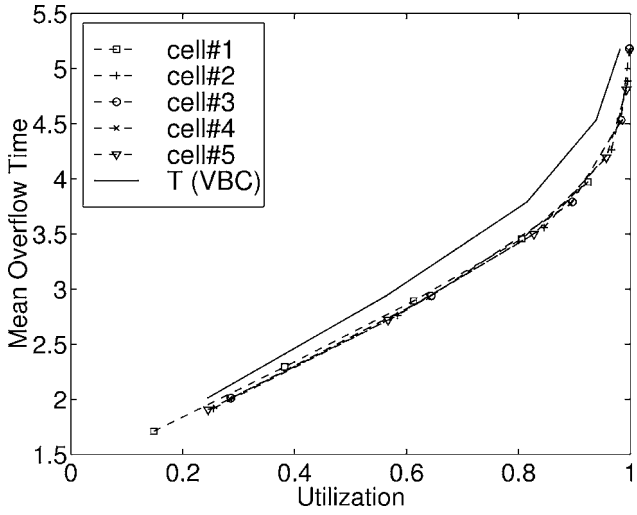


Figure 10. Mean overflow time versus utilization.

the overload measure and the probability of overload in various cells is different. Across a wide range of call departure rates μ, cells 5, 4, and 3 have the highest overload measure, whereas cell 1 has the lowest. It is clear that most of the users that originate their session in cell 1 will eventually end up in cells 3, 4 or 5, which form bottlenecks in this case. However, observe that no single cell is the bottleneck in all cases, and thus, performing admission control according to overload in the VBC ensures that the underlying QoS requirement is satisfied in all cells of the cluster even in the *worst case*.

In figure 10 we show the mean overflow time (in time units) for the virtual bottleneck cell as well as all five cells of the underlying system. This overflow time is plotted versus the utilization of the system with μ set to 0.8 in all cases. We define the VBC's mean overflow time as the maximum mean overflow times of all underlying cells as given by equation (2), whereas utilization is the successfully utilized system capacity *averaged* over all cells of the network.

We observe that as the utilization increases, the mean overflow time also increases, and hence, admission control must

be employed to limit its value. The plot also shows that there are not significant differences among the mean overflow times of the five cells for a given utilization. In addition, the mean overflow time of the VBC closely follows those of the underlying cells in the network, staying less than 0.5 time units above the mean overflow time of any cell. This illustrates that an admission decision based on the behavior of the VBC ensures that the QoS requirement is satisfied in all underlying cells without resulting in a significant decrease in the system's utilization.

In summary, we presented an analysis of a simple system in which user mobility patterns result in spatially heterogeneous resource demands. We showed that the Quality of Service parameters in the virtual bottleneck cell closely envelop those in the underlying system, demonstrating VBC's potential to accurately control system bottlenecks in a coarse-grained way, with little cost in system utilization.

6. Perfect knowledge algorithm

Admission control algorithms make a sequence of admission/rejection decisions in which resources are reserved for each admitted user. The performance of a particular algorithm can be assessed by evaluating the accuracy of its admission decisions, that is, whether the algorithm properly limits the handoff dropping probability to below the target P_{drop}, (and more generally, whether it limits the overload and outage time scale) and whether it does so while maximally utilizing the system's resources, admitting as many users as possible subject to the QoS constraint.

In this section, we utilize the framework of [4,10] to develop a benchmarking algorithm for evaluating admission control schemes in mobile multi-service networks. We term our approach *Perfect Knowledge Algorithm* (PKA) as it exploits knowledge of future handoff events to ensure that the maximal admissible region is obtained while satisfying the empirical P_{drop} constraint. Consequently, PKA, while unrealizable in practice, serves its benchmarking purpose by enabling us to evaluate the performance and effectiveness of a practical on-line admission control algorithm by comparing utilization and QoS values obtained by a certain algorithm with those obtained using the idealized PKA.

6.1. General P_{drop}

PKA considers a collection of call requests and target QoS values and outputs the set of accept/reject decisions that results in the highest mean utilization of resources subject to the required P_{drop}. We first consider the general case of $0 \leqslant P_{\text{drop}} \leqslant 1$. In particular, consider a set of users $\mathcal{S}$ requesting admission to the network, and let user $x \in \mathcal{S}$ be described by its required capacity c^x, and let its mobility pattern be defined by the matrix A^x of indicator functions such that

$$A^x_{h,j,t} = \begin{cases} 1 & \text{if } L(x,t) = j \text{ after the } h\text{th handoff,} \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

For a set of admitted users $\mathcal{A} \in \mathcal{S}$, the system utilization can be expressed as

$$U_T^{\mathcal{A}} = \frac{\sum_{j=1}^{M} \sum_{t=1}^{T} \Omega^{\mathcal{A}}(j,t)}{T \sum_{j=1}^{M} C_j} \tag{14}$$

with the empirical dropping probability through time T given by

$$\widehat{P}^{\mathcal{A}}_{\text{drop}}(T) = \frac{\sum_{x\in\mathcal{A}} \sum_{s\leqslant T} 1(L(x,s)\neq L(x,s-1)\cap \Omega^{\mathcal{A}}(L(x,s),s) \geqslant C_{L(x,s)})}{\sum_x \sum_{s\leqslant t} 1(L(x,s)\neq L(x,s-1))}, \tag{15}$$

which is the ratio of failed handoffs to handoff attempts, with $1(\cdot)$ denoting an indicator function.

Our goal is to find the set of users $\mathcal{A}^*$ to admit which maximizes U subject to the empirical QoS requirement $\widehat{P}_{\text{drop}}$. We formulate the problem as a nonlinear constrained optimization problem as follows.

We describe user x's success in utilizing the system via a vector defined as $\alpha_0^x = 1(x \in \mathcal{A})$ and $\alpha_h^x = 1(\text{handoff } h \text{ is successful})$. For example, if user x hands off to cell j at time t, $\alpha_h^x = 1(\Omega(j,t) + c^x < C_j)$. Further, if user x is admitted and successfully hands off three times, $\vec{\alpha}^x = \{111100000\ldots\}$.

PKA selects the optimum set $\mathcal{A}^*$ by maximizing the utilization, expressed (without normalizing) as

$$U_T^{\mathcal{A}^*} = \max_{\substack{\alpha_0^1,\ldots,\alpha^1_{H(1)},\ldots,\\ \alpha_0^N,\ldots,\alpha^N_{H(N)}}} \sum_{x\in\mathcal{S}} \sum_{t=1}^{T} \sum_{j=1}^{M} \sum_{h=0}^{H(x)} c^x \alpha_0^x \cdots \alpha_h^x A^x_{h,j,t}, \tag{16}$$

where $H(x)$ denotes the number of handoffs made by user x. Equation (16) must be maximized subject to both the system constraints

$$\sum_{x\in\mathcal{S}} \sum_{h=0}^{H(x)} c^x \alpha_0^x \cdots \alpha_h^x A^x_{h,j,t} \leqslant C_j, \tag{17}$$

for all $1 \leqslant t \leqslant T$ and all $1 \leqslant j \leqslant M$, and satisfaction of the empirical dropping probability

$$\frac{\sum_{x\in\mathcal{S}} \sum_{h=1}^{H(x)} \alpha_0^x \cdots \alpha_{h-1}^x (1-\alpha_h^x)}{\sum_{x\in\mathcal{S}} \sum_{h=1}^{H(x)} \alpha_0^x \cdots \alpha_{h-1}^x (\alpha_h^x \alpha^x_{H(x)} + h(1-\alpha_h^x)(1-\alpha^x_{H(x)}))} \leqslant P_{\text{drop}}. \tag{18}$$

Thus, describing each user by a mobility matrix of indicator functions and a vector of handoff indicator functions, allows us to determine the optimal set of admissible users $\mathcal{A}^* = \{\alpha_0^1, \alpha_0^2, \ldots, \alpha_0^N\}$ using standard methods for solving non-linear constrained optimization problems such as multi-start gradient-search. However, as the number of state variables is quite large (the total number of handoff attempts over all time and all users), we now turn to the special case of $P_{\text{drop}} = 0$ which we show has a more manageable solution.

6.2. $P_{\text{drop}} = 0$

For the special case of $P_{\text{drop}} = 0$, we can formulate the optimal solution with a simpler mobility matrix and a per-user admittance indicator, rather than the above handoff vector. Consequently, the optimal admissible region will be solvable via a constrained linear optimization problem.

In particular, let user $x \in \mathcal{S}$ be described by its required capacity c^x, and let its mobility pattern be defined by the matrix A^x of indicator functions such that

$$A^x_{j,t} = 1\big(L(x,t) = j\big). \tag{19}$$

Moreover, we reduce α^x to a (scalar) indicator function of admittance, i.e., $\alpha^x = \alpha^x_0$. With this formulation, PKA for $P_{\text{drop}} = 0$ can be expressed as a linear constrained optimization problem, maximizing utilization

$$\max_{\alpha^1,\ldots,\alpha^N} \sum_{x\in\mathcal{S}} \sum_{t=1}^{T} \sum_{j=1}^{M} c^x \alpha^x A^x_{j,t},$$

subject to the system and QoS rules, which are concisely described as

$$\sum_{x\in\mathcal{S}} c^x \alpha^x A^x_{j,t} \leqslant C_j$$

for all $1 \leqslant t \leqslant T$ and all $1 \leqslant j \leqslant M$.

In practice, the optimal solution $\mathcal{A}^* = \{\alpha^1, \alpha^2, \ldots, a^N\}$ can be computed quite efficiently, due to the reduction in the number of state variables, the linear nature of the problem, and the fact that matrices A^x are extremely sparse. We show experimental results for our implementation of PKA with $P_{\text{drop}} = 0$ in section 7.

7. Experimental results

Here, we use an extensive set of simulation experiments to investigate the performance of the VBC admission control algorithm and the adaptive clustering policy and to study the characteristics of different parameters involved.

7.1. Simulation environment

The simulation environment we use in our simulations is identical to the one introduced in [11], consisting of a two-dimensional 64 cell network as shown in figure 11. Handoffs occur between each cell and its four neighbors which share an edge with the cell. The network wraps around so that, for example, any user leaving the bottom edge of cell number 63 will enter the upper edge of cell 7. The 64 cell area represents a set of cells connected to the same router; so a handoff between cell 63 and cell 7 will be considered an inter-router handoff.

Users follow the Downtown Mobility Model, and the four cells 0, 7, 56, and 63 are considered as downtown areas; the users are highly likely to choose one of these cells as their destination as they are initiated. The movement is through a random path toward the destination with a probability distribution in favor of the shortest path.

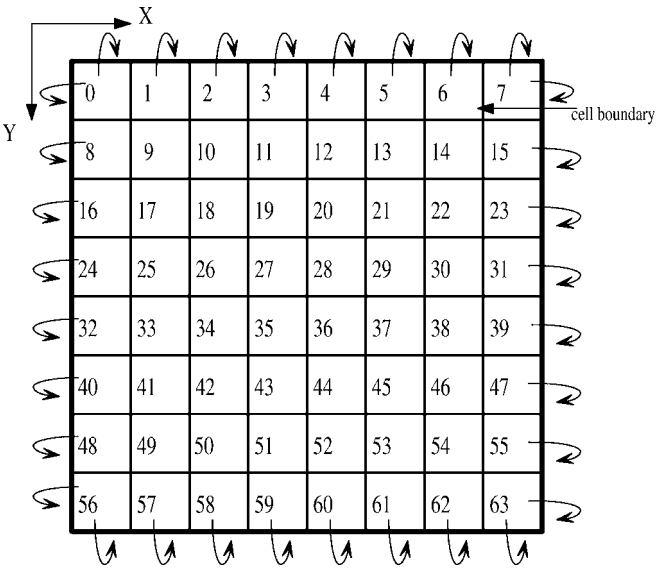


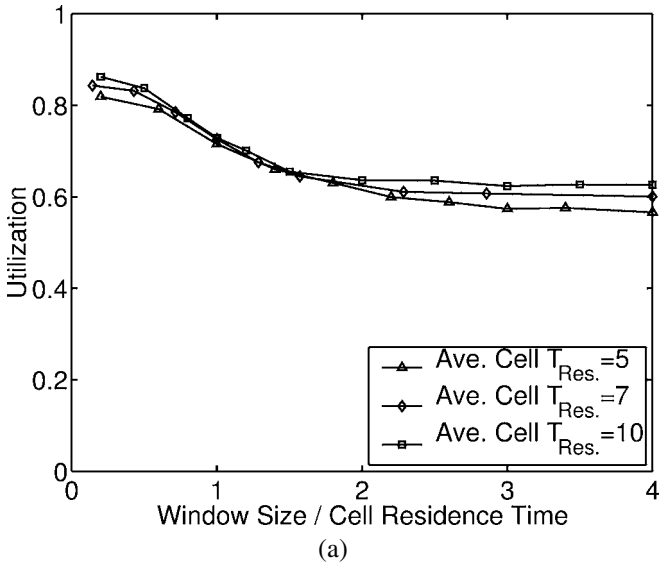
Figure 11. Cellular topology.

The time is slotted to 1 min intervals and both the call holding time and the cell residence time have geometric distribution with means 10 and 7, respectively, if not mentioned otherwise. Simulation time for all the results presented is 6 hours; during which, a large number of users were introduced to the network. The capacity of each cell is 10 Bandwidth Units (BU) and each user requires 1 BU. The traces of the users' movements were produced using the simulator of [10] with extensions for the adaptive clustering and VBC admission control algorithms.

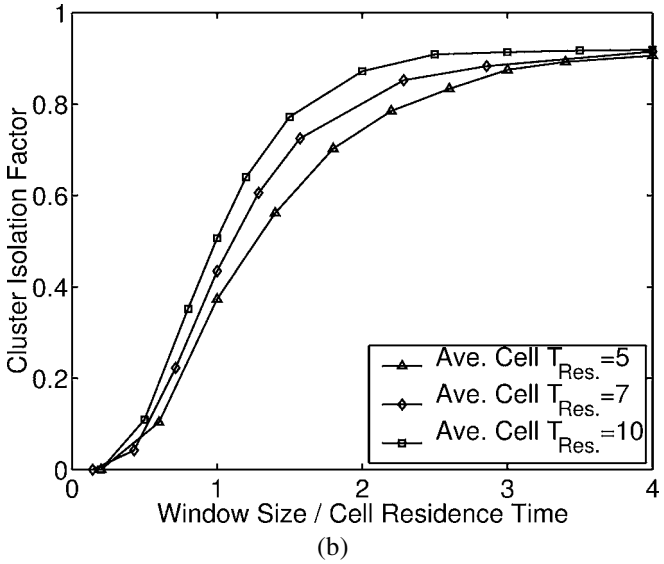
7.2. Design issues for clustering

Figure 12 depicts network utilization and CIF versus the ratio of window size over average cell residence time for $\beta = 0.8$, $\alpha_h = 0.4$ and $\alpha_l = 0.1$. The CIF reference clustering policy $\overline{\mathcal{K}}$, for the simulation results shown in this section, has been chosen to be the adaptive policy which minimizes the number of inter-cluster handoffs, achieved by setting the three parameters β, α_h, and α_l equal to zero, since $\beta = 0$ means that the network is continuously measuring the amount of transferred bandwidth among cells for new clusters to form, $\alpha_h = 0$ forces two clusters to annex as soon as any handoff happens between them, and $\alpha_l = 0$ indicates that no cell in the network separates from any cluster.

As shown in figure 12(a) for different values of average cell residence time, utilization decreases as window size over mean residence time increases. Also illustrated in figure 12(b), CIF increases as window size increases. The reason for this is that larger measurement windows tend to increase cluster size, and as VBC ensures QoS over the entire cluster, this correspondingly increases CIF and decreases utilization. Thus, we conclude that an ideal value of the window size is 1 to 2 times larger than the average cell residence time, as larger values decrease utilization and smaller window sizes than the



(a)



(b)

Figure 12. Impact of measurement window size on system performance. (a) Utilization. (b) Cluster Isolation Factor.

average cell residence time would not capture the true amount of capacity transfered among the cells.

Figure 13 illustrates the impact of the clustering threshold β on system performance for $\alpha_l = 0$ and $\alpha_h = 0$. The figure indicates that choosing β to be less than 1 has the best effect on the isolation of the clusters, as clusters will form preemptively before overload occurs. As an example, with $\beta = 0.8$, the system will have a utilization of approximately 60% where the clusters are 80% isolated compared to the reference clustering algorithm $\overline{\mathcal{K}}$.

Figure 14(a) shows the impact of the clustering annexation threshold α_h on utilization and CIF for $\beta = 0.8, \alpha_l = 0.1$, and window size equal to the average cell residence time (7 time units). Observe that CIF rapidly decreases for lower values of α_h indicating that smaller values for α_h (and lower-load, or preemptive cluster annexation) are preferable. For α_l, which denotes the threshold for separating cells from clusters, observe from figure 14(b) that changes of α_l in the valid

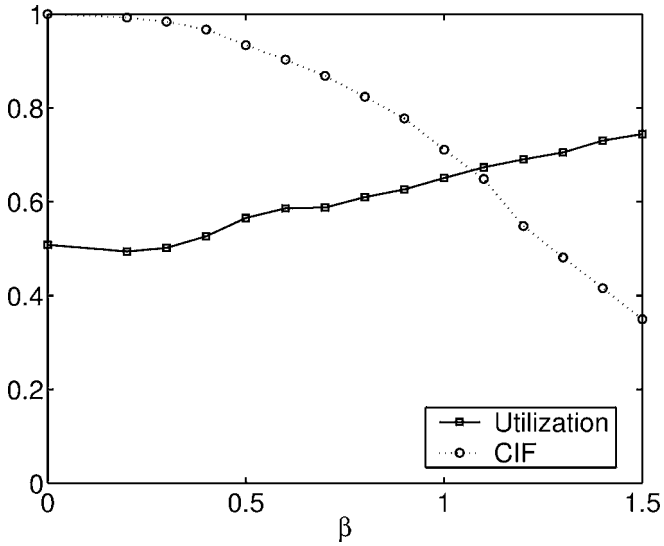


Figure 13. Impact of β on system performance.

range of $[0, \alpha_h]$ result in moderate changes in utilization and CIF, with lower values (and lower-load cluster separation) being slightly preferable. Thus, it is clear from the above experiments that the parameters are best set so that clusters form rapidly as the system approaches overload, and are not quickly separated as the load reduces. While ideal parameters are clearly dependent on the system workload, suggested initial settings based on our experiments are $\beta = 0.8$, $\alpha_h = 0.4$, and $\alpha_l = 0.1$.

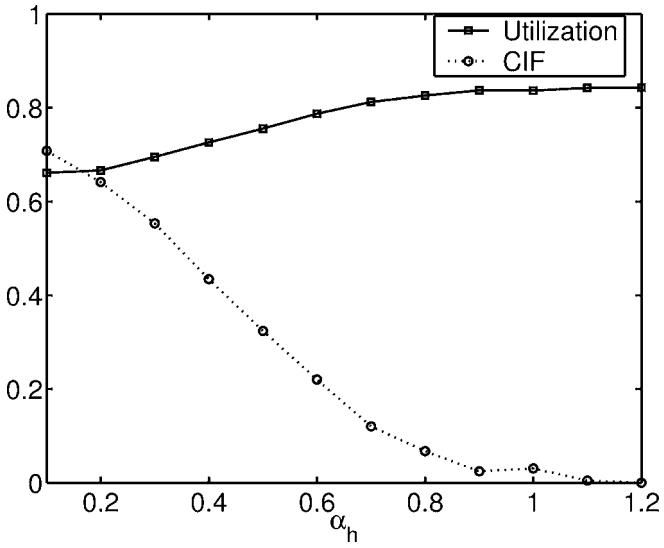
7.3. Performance of VBC admission control and adaptive clustering

In figure 15 we present the results of the simulation experiments showing the performance of the VBC admission control algorithm with adaptive clustering policy along with comparisons with two different benchmarks as well as performance of the VBC admission control with a semi-optimal clustering.

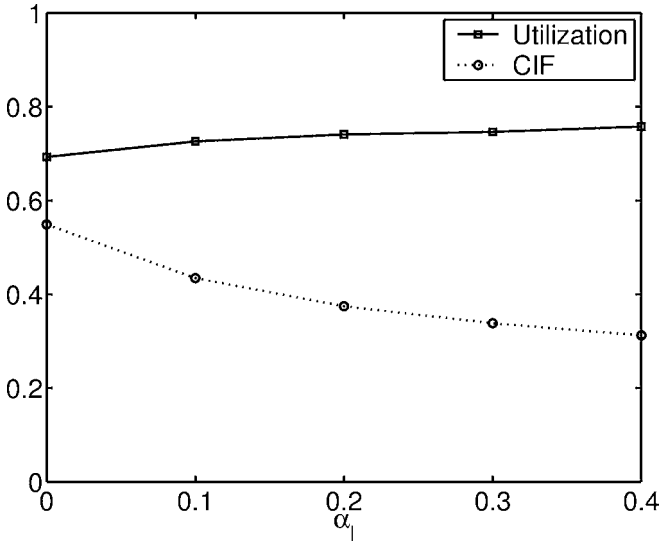
The first benchmark is PKA, the optimal off-line admission control algorithm described in section 6. The PKA curve depicts the average system utilization achieved for the optimal off-line admissible region for the case of no overload, i.e. $\gamma = 0$ (hence, the curve is flat). As shown, PKA obtains a utilization of approximately 91.54%.

As a second baseline case, we compare VBC admission control to a "location specification" algorithm [10] in which users prespecify the set of cells that they will visit during the duration of their session, and resources are reserved in each of the corresponding cells for the entire lifetime of the call. The network admits a new user only if overload will not occur at any time in any cell. Note that this algorithm is considerably more conservative than PKA as the times are not prespecified: hence, the capacity in each cell is reserved for the entire session duration.

The middle curve labeled VBC represents the admissible region obtained by our implementation of the VBC algorithm with the adaptive clustering policy. While no on-line algo-



(a)



(b)

Figure 14. Impact of α_h and α_l on system performance. (a) Clusters annexation threshold. (b) Cell separation threshold.

rithm can obtain utilization greater than PKA while satisfying the QoS constraints, we observe that the VBC algorithm performs quite well. In particular, over the entire range of overload values, VBC admission control is able to outperform the location specification approach. Moreover, despite our use of scalable coarse-grained system control and assurance that QoS is satisfied even in bottleneck cells, the VBC algorithm along with adaptive clustering is able to efficiently utilize system resources, obtaining average utilization in the range of 48% to 84% for the range of overload shown.

Finally we compare the performance of our adaptive clustering algorithm with another method, which we call optimal static clustering. For the particular simulation scenario discussed in section 7.1, the optimal static clustering policy would form fixed clusters around the hot spots (cells 1, 7, 56, and 63). To obtain the utilization of the network, for a given overload value, we found the optimal clustering size for the

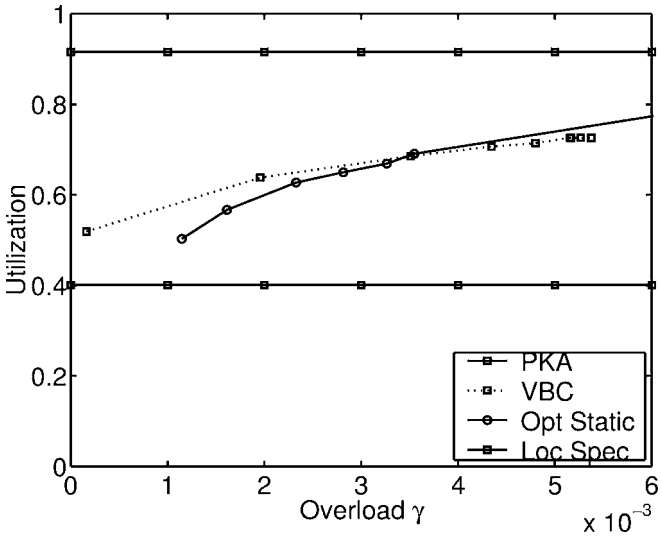


Figure 15. Performance of the VBC admission control.

mentioned policy, which maximizes the utilized bandwidth while satisfying the QoS requirements. As the figure shows, the adaptive clustering outperforms the optimal static scheme over a large range of overload and slightly underestimates the admissible region in high overload.

Thus, these results indicate that the VBC admission control algorithm with adaptive clustering algorithm is a scalable scheme for wireless networks that can effectively and accurately control the system's admissible region.

8. Conclusions

As mobile and wireless communication becomes increasingly ubiquitous, techniques for Quality of Service provisioning will encounter fundamental challenges in scaling to many users and many handoffs, especially in future micro- and pico-cellular systems. In this paper, we proposed new techniques with the ability of performing scalable and coarse-grained QoS control in future systems and introduced Virtual Bottleneck Admission Control as a particular algorithm based on this design philosophy. VBC provides a mechanism to characterize and control an aggregate virtual system while closely enveloping the behavior of the underlying cells, enabling efficient provisioning of system resources, even under heterogeneous spatial demands and "hot spots". A fundamental problem for spatial resource aggregation is cell clustering. We formulated the clustering problem as an optimization problem and designed a heuristic adaptive clustering algorithm as a practical approximate solution. To evaluate the performance of the scheme, we developed a simple one-dimensional analytical model, an optimal off-line algorithm for benchmarking, and performed extensive simulation experiments. We showed that our clustering algorithm is successful in adaptively capturing the variations in users' mobility patterns in the network and in forming the clusters around the network bottlenecks. Using the results of our analytical studies we showed that the characteristics of the virtual sys-

tem closely envelope the behavior of the underlying cells, enabling efficient provisioning of system resources. Through our simulation experiments, we first studied the characteristics of the parameters involved in adaptive clustering policy and then, applying optimal off-line admission control algorithm as a benchmark, found that the coarse-grained approach can effectively control the system's admissible region. Our findings indicate that scalability needs not to be achieved at the expense of efficient resource utilization and strong Quality of Service guarantees.

Acknowledgements

The authors are grateful to Rahul Jain and members of the Rice Networks Group for their insightful comments and discussions.

References

[1] A. Aljadhai and T. Znati, A framework for call admission control and QoS support in wireless environments, in: *Proceedings of IEEE INFOCOM'99*, New York (March 1999).

[2] C. Bisdikian, T. Kwon, Y. Choi and M. Naghshineh, Call admission control for adaptive multimedia in wireless/mobile networks, in: *Proceedings of the First ACM International Workshop on Wireless and Mobile Multimedia*, Dallas, TX (1998).

[3] S. Blake et al., An architecture for differentiated services, Internet RFC 2475 (1998).

[4] A. Borodin and R. El-Yaniv, *Online Computation and Competitive Analysis* (Cambridge University Press, 1998).

[5] C. Chao and W. Chen, Connection admission control for mobile multiple-class personal communications networks, IEEE Journal on Selected Areas in Communications 15(8) (1997) 1618–1626.

[6] S. Choi and K. Shin, Predictive and adaptive bandwidth reservation for handoffs in QoS-sensitive cellular networks, in: *Proceedings of ACM SIGCOMM'98*, Vancouver, BC (August 1998).

[7] D. Eckhardt and P. Steenkiste, Effort limited fair scheduling for wireless networks, in: *Proceedings of IEEE INFOCOM 2000*, Tel Aviv (March 2000).

[8] B. Epstein and M. Schwartz, Predictive QoS-based admission control for multiclass traffic in cellular wireless networks, IEEE Journal on Selected Areas in Communications 18(3) (March 2000) 523–534.

[9] J. Gomez, A. Campbell and H. Morikawa, A systems approach to prediction, compensation, and adaptation in wireless networks, in: *Proceedings of First ACM International Workshop on Wireless Mobile Multimedia*, Dallas, TX (October 1998).

[10] R. Jain and E. Knightly, A framework for design and evaluation of admission control algorithms in multi-service mobile networks, in: *Proceedings of IEEE INFOCOM'99*, New York (March 1999).

[11] R. Jain, B. Sadeghi and E. Knightly, Towards coarse-grained mobile QoS, in: *Proceedings of the 1999 ACM International Workshop on Wireless Mobile Multimedia*, Seattle, WA (August 1999).

[12] S. Jiang, B. Li, X. Luo and D. Tsang, A modified distributed call admission control scheme and its performance, Wireless Networks 7 (2001) 127–138.

[13] J. Kim and M. Krunz, Quality of Service over wireless ATM links, in: *Proceedings of IEEE INFOCOM'99*, New York (March 1999).

[14] D. Levine, I. Akyildiz and M. Naghshineh, A resource estimation and call admission algorithm for wireless multimedia networks using the shadow cluster concept, IEEE/ACM Transactions on Networking 5(1) (February 1997) 1–12.

[15] S. Lu and V. Bharghavan, Adaptive resource management for indoor mobile computing environments, in: *Proceedings of ACM SIGCOMM'96*, Stanford, CA (1996).

[16] I. Mahadevan and K. Sivalingam, An architecture for QoS guarantees and routing in wireless/mobile networks, in: *Proceedings of First ACM International Workshop on Wireless Mobile Multimedia*, Dallas, TX (October 1998).

[17] B. Mirkin, *Mathematical Classification and Clustering* (Kluwer Academic, 1996).

[18] M. Naghshineh and M. Schwartz, Distributed call admission control in mobile/wireless networks, IEEE Journal on Selected Areas in Communications 14(4) (1996) 711–717.

[19] M. Naghshineh and M. Willebeek-LeMair, End-to-end QoS provisioning multimedia wireless/mobile networks using an adaptive framework, IEEE Communications 35(11) (November 1997) 72–81.

[20] T. Nandagopal, T. Kim, X. Gao and V. Bharghavan, Achieving MAC layer fairness in wireless packet networks, in: *ACM MOBICOM 2000*, Boston, MA (August 2000).

[21] T. Ng, I. Stoica and H. Zhang, Packet fair queueing algorithms for wireless networks with location-dependent errors, in: *Proceedings of IEEE INFOCOM'98*, San Francisco, CA (March 1998).

[22] R. Ramjee, R. Nagarajan and D. Towsley, On optimal call admission control in cellular networks, in: *Proceedings of IEEE INFOCOM'96*, San Francisco, CA (March 1996) pp. 43–50.

[23] T. Rappaport, *Wireless Communications: Principles and Practice* (Prentice Hall, 1996).

[24] S. Singh, Quality of Service guarantees in mobile computing, Computer Communications 19(4) (April 1996) 359–371.

[25] A. Talukdar, B. Badrinath and A. Acharya, Integrated services packet networks with mobile hosts: Architecture and performance, Wireless Networks 5(2) (1999) 111–124.

[26] B. Teitelbaum et al., Internet2 QBone: Building a testbed for differentiated services, IEEE Network 13(5) (September 1999) 8–17.

Bahareh Sadeghi received the B.Sc. degree in electrical engineering from Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran, in 1996 and the M.S. degree in electrical and computer engineering from Rice University, Houston, Texas, in 2000. She is currently a Ph.D. candidate in Electrical and Computer Engineering Department at Rice University. Her major interest is Quality of Service in wireless networks.
E-mail: bahar@rice.edu

Edward Knightly received the B.S. degree from Auburn University in 1991, the M.S. degree from the University of California at Berkeley in 1992, and the Ph.D. degree from the University of California at Berkeley in 1996, all in electrical engineering. Since 1996, he has been an Assistant Professor in the Department of Electrical and Computer Engineering at Rice University. He currently serves on the editorial board of the Computer Networks Journal, IEEE/ACM Transactions on Networking, and IEEE Transactions on Multimedia. He served as a co-chair for the 1998 International Workshop on Quality of Service and is the finance chair for ACM MOBICOM 2002. He received the National Science Foundation CAREER Award in 1997 and the Sloan Fellowship in 2001. His research interests are in the area of theory, algorithms, and architectures for Quality of Service in wired and wireless networks.
E-mail: knightly@rice.edu