RICE UNIVERSITY

Architecture and Algorithms for Scalable Mobile QoS

by

Bahareh Sadeghi

A Thesis Submitted
in Partial Fulfillment of the
Requirements for the Degree

Master of Science

Approved, Thesis Committee:

Dr. Edward Knightly, Chair
Assistant Professor
Electrical and Computer Engineering

Dr. Behnaam Aazhang
Professor
Electrical and Computer Engineering

Dr. Robert Nowak
Assistant Professor
Electrical and Computer Engineering

Houston, Texas

April, 2000

Architecture and Algorithms for Scalable Mobile QoS

Bahareh Sadeghi

Abstract

Supporting Quality of Service is an important objective for future mobile systems, and requires resource reservation and admission control to achieve. In this thesis, we introduce a scalable scheme to admission control termed Virtual Bottleneck Cell. Our approach is designed to scale to many users and hand-offs, while simultaneously controlling "hot spots". The key technique is to hierarchically control the virtual system, ensuring QoS objectives are satisfied without requiring accurate predictions of the users' future locations. We develop a simple analytical model to study the system and illustrate several key components of the approach. We formulate the problem of how to group the cells to form the virtual system as an optimization problem and propose a heuristic adaptive clustering algorithm as its solution. Finally, we perform simulations in a two-dimensional network to compare the performance obtained with VBC and adaptive clustering with alternate schemes, including the optimal offline algorithm.

Acknowledgments

I would like to express my deep gratitude to my thesis advisor Dr. Edward W. Knightly for guiding me and teaching me how to do research. I also appreciate the kindness of Dr. Behnaam Aazhang and Dr. Robert Nowak, who were on my committee and enriched the thesis with their enlightening discussion.

I am grateful to all my friends, and particularly to Rice Networks Group members, for their friendly assistance and insightful discussion.

My special thanks goes to my wonderful husband Ahmad Khoshnevis, whose great help, patience and support accompanied me.

Finally, My heartfelt thanks to my father Ghorbanali Sadeghi and my mother Zahra Dastkar, for their great vision and support through all these years.

Contents

Illustrations

Chapter 1

Introduction

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

To satisfy the performance demands of such future mobile users, the network must limit the severity, frequency, and duration of overload due to hand-offs and user mobility. Consequently, networks must employ *admission control* as the key mechanism for ensuring mobile Quality of Service (QoS) measures.

In [JK99], the authors have devised a taxonomy of admission control algorithms which illustrates several key tradeoffs in terms of granularity of resource control, mathematical tractability, and efficacy of accurately controlling the admissible region while also provisioning the desired quality of service.

At the finest granularity, an algorithm dynamically reserves system capacity on a per-user basis [NS96b, RNT96, LAN97, CS98]. For example, capacity would be reserved for a particular user at future times in nearby cells as dictated by the mobile's current location and velocity, past mobility behavior, and/or other model-based prediction techniques. While such fine-grained control indeed has the potential to accurately manage QoS, there are fundamental limitations to such an approach in addition to communication and computational overheads [JK99], and therefore corresponding limits on its scalability to many users and hand-offs.

At issue are the time scales of system control. As illustrated in Figure 1.1, 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: extending location estimations at the hand-off time scale to session QoS measures at the session time scale. Moreover, this gap will likely widen in the future as the former time scale is expected to be reduced in future pico-cellular environments, while the latter time scale may remain the same. Finally, as we will show in Chapter 5, even if a user's future locations are precisely known *a priori*, admission control can still be conservative if the hand-off *times* corresponding with those locations are not also known.

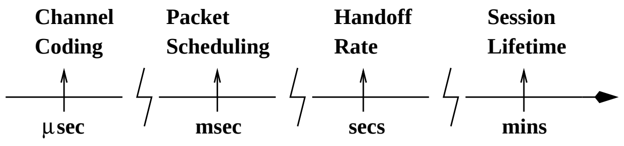


Figure 1.1 Time Scales of System Control

In this thesis, we develop a new admission control algorithm which achieves scalable control of mobile quality of service. 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 VBCs and 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 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 hand-offs. In this way, we ensure that our solution is scalable to a large number of users, even in micro- and pico-cellular environments with a potentially large number of hand-offs 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 coarse grained approach does not preclude management of “hot spots” and system bottlenecks.

To analyze the performance of the VBC algorithm and illustrate several key design issues, 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.

While the aggregation ensures scalability of VBC approach, the clustering policy addresses the question of its accuracy. The performance of VBC is dependent on how the cells are grouped in the network, i.e. clusters configuration. In order to answer the key question of “How to cluster the cells?”, we formulate the problem of clustering the cells in the network as an optimization problem. However due to the fact that there is not any stochastic model available for the mobility model of the users and also the high complexity of the problem there is not an analytical way to find the optimal clustering algorithm. Hence, we propose a heuristic approach as an adaptive clustering algorithm, which is both accurate and fast enough to capture the changes of users' mobility pattern in the network. The key point in developing this algorithm is its ability to discover the correlations among occupancies of neighboring cells, and form clusters based on these correlations. Where the traditional clustering algorithms [Spa80] may be used to find the optimal clustering configuration for a stationary system off-line, because of their high computation overhead they can not be applied to the network as an online algorithm.

We then perform an extensive set of simulation 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 the Perfect Knowledge Algorithm (PKA), a globally optimal off-line algorithm devised in [JK99], 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 successfully control the admissible region and can outperform an alternate scheme in which user hand-off locations are known *a priori* [JK99].

Thus, we devise a VBC admission control algorithm as a novel way to characterize and control system QoS via a scalable approach. In this way, our approach differs from previous work which focused on developing sophisticated mobility models and prediction techniques for allocating system resources [AZ99, CC97, CS98, LAN97, NS96a, Sin96, TBA99]. Consequently, our technique subsumes per-user mobility via aggregate control and is significantly simpler in terms of communication and computational overhead.

The remaining part of the thesis is organized as follows. In Chapter 2 we describe the system model and outline the VBC approach. Next, in Chapter 3, we develop an analytical model to study the problem. In Chapter 4, we define the clustering problem and introduce our adaptive clustering algorithm, and in Chapter 5 we provide the results of our simulation experiments. Finally, in Chapter 6, we conclude.

Chapter 2

Virtual Bottleneck Cell (VBC)

In this chapter, we first describe the system model and role of admission control. Then we overview our design goals for coarse-grained system control and sketch the 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 on-line admission control algorithm.

2.1 System Model

The system model that we consider is depicted in Figure 2.1. 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 [B$^+$98] and extended to the wireless network via a wireless/mobile QoS architecture (e.g., [Sin96, MS98]). We focus on traffic classes requiring higher 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

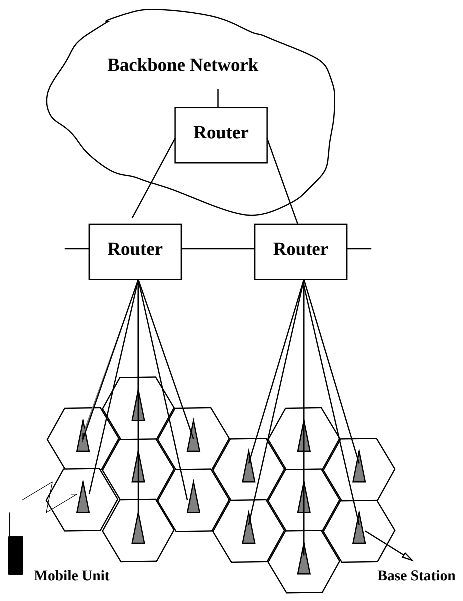


Figure 2.1 System Model

QoS measures are *statistical* in nature; consequently, demanded bandwidth will at times exceed the available capacity due to over-booking of system resources. The goal of the admission control algorithm is to limit the frequency, severity, and durations of such overload periods to within pre-specified 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. In following section 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 [GCM98, LB96, NWL97] for example, and are beyond the scope of this thesis.

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. We assume that while individual users transmit traffic at variable rate, their required capacity can be summarized by a single "effective bandwidth" using techniques such as in [KK99]. We then denote the aggregate bandwidth demanded at time t by the users in cell j by $\Omega_j(t)$, which we also refer to as the *occupancy* of cell j at time t, as it is also the (scaled) number of users occupying the cell when users have identical effective bandwidths.

2.2 Coarse-Grained Control

Our goal is to control QoS in mobile networks via a coarse-grained and scalable approach. Towards this end, we introduce a novel approach termed Virtual Bottleneck Cell. With VBC, we manage a network of virtual cells which subsume not only the behavior of individual cells within a cluster, but also of individual users within the cells. We will show that we can effectively control the system's QoS by managing resources in the virtual system, rather than by allocating capacity on a per-user or per-cell basis.

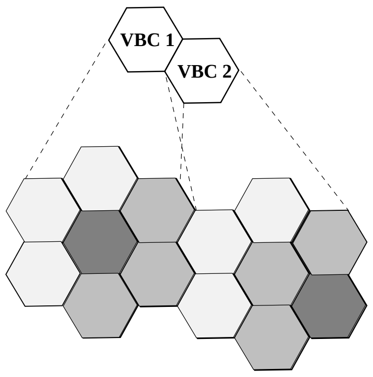


Figure 2.2 VBC Illustration

As illustrated in Figure 2.2, 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 hand-offs 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.

2.3 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 $\mathcal{C}_1, \cdots \mathcal{C}_K$, 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

$$\gamma_k \stackrel{\triangle}{=} \max_{j \in \mathcal{C}_k} \frac{E(\Omega_j - C_j)^+}{C_j} \tag{2.1}$$

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 thesis, we will refer to γ as simply "overload".

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

$$T_k \stackrel{\triangle}{=} \max_{j \in \mathcal{C}_k} \{E\tau_j : \Omega_j(u) > C_j \text{ for } u = [s, s + \tau]\} \tag{2.2}$$

which denotes the maximum mean duration of overflow of any cell in the VBC. (See also [CBN98] 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 (2.1) and (2.2) 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 hand-off 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 [ZK97] or even incur temporary service outages.

To maintain quality of service to within pre-specified 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 [KK99] for example). Next, the router managing the cluster for which the new user is requesting access (see Figure 2.1) 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

$$\hat{\gamma}_k = \frac{1}{W} \max_{j \in \mathcal{C}_k} \frac{1}{C_j} \sum_{s=t-W}^{t} \max(\hat{\Omega}_j(s) + \Omega_{new} - C_j, 0) \tag{2.3}$$

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

Similarly, denote

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

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

$$\hat{T}_k = \max_{j \in \mathcal{C}_k} \frac{\sum_{s=t-W}^{t} O_j(s)}{\sum_{s=t-W}^{t} 1(O_j(s) > O_j(s-1))}. \tag{2.4}$$

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 (2.3) and (2.4) 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 (2.3) and (2.4) 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 of 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 hand-offs; moreover, setting W larger than the session lifetime will skew the QoS predictions by including the effects of sessions that no longer exist. Further guidelines for setting such windows in a related system can be found in [QK98].

Chapter 3

Analytical and Numerical Investigation

In this chapter, we introduce a simple analytical model to study several aspects of admission control using virtual bottleneck cells. Our model consists of a one-dimensional cellular array similar to one which might be used in modeling highways.

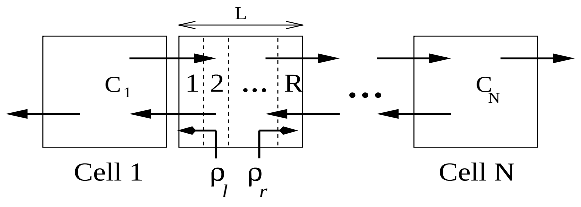


Figure 3.1 Network Model

As shown in Figure 3.1, the array consists of N cells with the same length L, and cell j having capacity C_j, $1 \leq j \leq N$. 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.

3.1 Overload

To calculate the overload γ defined in Equation 2.1, 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

$$E(\Omega_j - C_j)^+ = \sum_{i=1}^{\infty} i \ \Pr(\Omega_j = i + C_j). \tag{3.1}$$

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

$$\begin{aligned} \Pr(\Omega_j > C_j) &= \Pr(n_{hj} + n_{gj} > C_j) \\ &= \sum_{i=0}^{\max(C_j, n_{gj}^{\max})} \Pr(n_{gj} = i) \Pr(n_{hj} > C_j - i) \end{aligned} \tag{3.2}$$

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

$$F(\tau) = \Pr(T_h \leq \tau) = 1 - e^{\mu\tau}$$

Then, to compute Equation (3.2), we define the function

$$\Theta(x, y, \tau) = \begin{pmatrix} x \\ y \end{pmatrix} (1 - F(\tau))^y F(\tau)^{x-y},$$

and the vectors

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

and

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

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 $(n_{j0}, ..., n_{j(R-1)})$ 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 $l = 0, ..., R-1$. 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 $\Pr(n_{gj} \leq i)$, 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

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

$$0 \le i \le \frac{\lambda(R+1)}{2}. \tag{3.3}$$

Similarly, n_{hj} is the sum of all active users that initiated their calls in cell i, $j < i \le N$, 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, $1 \le k < j$, and are moving to the right. Therefore, defining the vectors

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

with elements

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

and

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

with elements

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

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

$$\Pr(n_{hj} \le i) = \sum_{\overline{\Lambda}_j(n,(j-1)R)=\overline{0}}^{\overline{\Phi}_j} \sum_{\overline{\Lambda}_j(m,(N-j)R)=\overline{0}}^{\overline{\Psi}_j} \mathcal{X}_j \mathcal{Y}_j \tag{3.4}$$

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

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

and

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

Thus, combining Equations (3.1)–(3.4), we have an expression for γ_k, cluster k's overload measure.

3.2 Overflow Timescale

We next turn to the overflow timescale of the virtual bottleneck cell defined in Equation 2.2. 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 hand-off 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

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

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

3.3 Numerical Examples

We now perform numerical investigations applying the analysis above. In Figure 3.2, we show the results for $N = 5$, $R = 1$, $\rho_r = \frac{2}{3}$, $\rho_l = \frac{1}{3}$, $\lambda = 9$, and $C_j = 10$ for $1 \leq j \leq 5$. The figure depicts the measure of overload for each of the five cells, i.e., $\frac{E(\Omega_j - C_j)^+}{C_j}$ 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 the overload-measure and the probability of

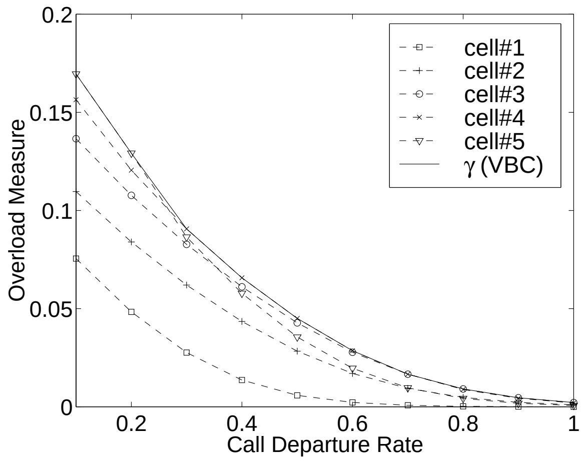


Figure 3.2 Overload Measure vs. Call Departure Rate

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 3.3 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 2.2, 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

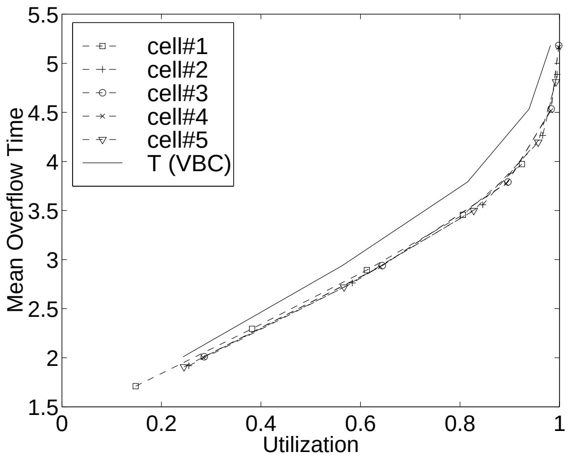


Figure 3.3 Mean Overflow Time vs. Utilization

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.

Chapter 4

Clustering

In the previous chapters we showed that given a cluster configuration, VBC admission control is a scalable algorithm for resource provisioning based on the aggregated information of the users in the network. But accuracy of such a method depends on the clustering configuration. In this chapter our goal is to find the answer to the key question of "How to cluster the cells?".

There are many parameters one should consider in clustering the network:

Users Mobility Model: The most important parameter in clustering is mobility model of the users. Users mobility model is partly dependent on the underlying physical structure, i.e., the pattern of highways and roads along which the movements of the users happen, which enforces a specific mobility pattern as part of users mobility model. The cell residence time and call holding time distributions are two other factors involved in users mobility model.

Physical Architecture: Underlying physical architecture of the network is another parameter which affects the efficiency of the clustering policy. Having cells which are connected to different sub-networks or routers in one cluster, adds communication overhead to the network. Hence we focus on the clustering problem within the set of cells connected to the same router.

Cluster Size: There is a trade-off between gained utilization and probability of crossing clusters (inter-cluster Hand-off) in choosing the size of the clusters. Since the admission is based on the aggregated information of a group of cells, choosing a very big cluster, results in an unnecessary decrease in the utilization. The cells in a cluster should be chosen in a way that there exist some correlation among their occupancies. For example, if the probability that users of a cell in a given cluster move to any congested cell in that cluster is very low, the users of this cell will suffer from admission decisions which are too conservative (which results in lower utilization) since the admission decision is based on the worst case measures.

On the other hand, having very small clusters increases the probability of inter-cluster hand-offs. When a user leaves a cluster and enters a cell in a new cluster, no matter how the hand-in call is treated by the new cluster, there is no more any guarantee on the QoS metrics the network provides him. As an example, if entering a new cluster, the users be treated as new calls and do not get admitted to the new cluster, their calls will be dropped. In another policy, inter-cluster hand-offs are treated as intra-cluster hand-off calls and totally admitted in the new cluster. In such a scenario, not only the QoS requirements of hand-in user in the destination cluster have not been taken into account, but also it may cause degradation in service for other users who are already in the destination cluster. In this thesis, regardless of the choice of the inter-cluster hand-off treaty policy, we consider the probability of inter-cluster hand-off as a general measure of QoS in the network.

In the remainder of this chapter we first define the clustering problem in our system and then formulate it as an optimization problem. Then we will discuss the issues regarding its solvability. In section 4.2 of this chapter we will design a heuristic adaptive clustering algorithm to find the optimal clustering configuration at each time in the network.

4.1 Problem Definition

In a *traditional* clustering problem the goal is to find *similarities* in a set of objects and group them such that the objects in a cluster are similar to one another and dissimilar from the objects of any other cluster. It is usually done based on a given distance measure, defined to measure the distance between the objects in a given space. Then the objects are grouped in different clusters so that the distance between the objects in each cluster is minimized [Mir96, Rom84]. While, the traditional clustering algorithms [Spa80] can be used to find the optimal solution to a stationary clustering policy off-line, they are not appropriate for a dynamic system due to their high computation overhead.

The main motivation behind grouping the cells in the network, and making the admission decision based on the aggregated information of the underlying cells of the cluster, is the fact that a congested cell may dominate a region around it. It means that if a user requests admission to the network in a cell in vicinity of a highly loaded cell, with high probability, that user may end up in the congested cell. In such a situation, it is preferred that the call be blocked rather than experience some degradation in service, due to moving into the congested cell. Hence the key insight to the clustering problem is capturing the similarities which arise because of a shared, varying parameter among the cells in the network, which is the flow of the user's movements.

Figure 4.1 shows 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 hand-offs between cell A and each of its neighbors. If for example, there are too many hand-offs from cell B to cell A, then admission of any new call in cell B affects the QoS metrics in cell A, since the users of cell B hand-off

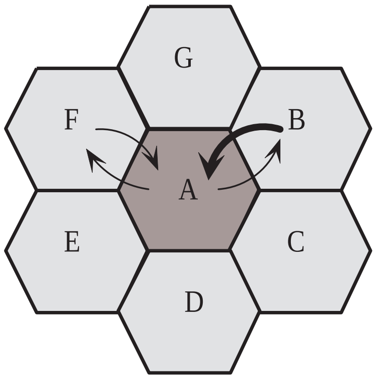


Figure 4.1 Clustering in Neighborhood of a hot spot

to cell A with high probability. Hence, we need to annex these two cells to form a cluster. On the other hand, if we annex cell F and cell A to form a cluster, where the aggregated amount of hand-offs 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 QoS metrics of cell A. But, given that cell A is overloaded and there are not many hand-offs from cell F to cell A, it results in an unnecessary high probability of call block in cell F, which reduces the utilization of our system.

We now define our clustering problem as an optimization problem and show that a clustering algorithm based on the users' mobility pattern in the neighboring cells of the network, groups the cells successfully into clusters, in a way that the objective of the defined optimization problem is maximized.

Clustering Problem: Given a sub-network consisting of M cells, all connected to the same router, along with the overload and 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 satisfied), minimizes probability of inter-cluster hand-off (which as defined below is a measure of the service certainty) and at the same time maximizes the network utilization.

Note that in Chapter 2 we have already shown that given the assumption that the users do not leave the clusters, the VBC admission control policy satisfies QoS requirements. In a real network, where the users leave the clusters, the goal is to maximally utilize the network resources and at the same time to minimize the number of calls experiencing degradation in service due to hand-offs between different clusters, so that the QoS requirements of the network are satisfied.

Let $\mathcal{K}$ denote any possible clustering policy of a sub-network of size M during the observation period of T. Then, $\mathcal{K}$ can be written as a T by M matrix, where row t, $t \leq T$, is the cluster configuration of the M cells at time t. Also let $\mathcal{D}$ denote the set of all possible clustering policies $\mathcal{K}$. 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 [Jai99], such that

$$A^x_{j,t} = 1(L(x,t) = j) \tag{4.1}$$

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, $j = 1, \cdots, M$ 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

$$U_T^{\mathcal{K},\mathcal{A}} = \frac{\sum_{j=1}^{M} \sum_{t=1}^{T} \Omega^{\mathcal{A}}(j,t)}{T \sum_{j=1}^{M} C_j} \tag{4.2}$$

with the empirical probability of inter-cluster hand-off is given by

$$\hat{P}_{HO}^{\mathcal{K},\mathcal{A}}(T) = \frac{\sum_{x\in\mathcal{A}} \sum_{s\leq T} 1(F^{\mathcal{K}}(x,s) \neq F^{\mathcal{K}}(x,s-1)]}{\sum_{x\in\mathcal{A}} \sum_{s\leq T} 1(L(x,s) \neq L(x,s-1))} \tag{4.3}$$

which is the ratio of inter-cluster hand-offs to the total hand-offs attempts.

Let $P_{\overline{\mathcal{K}}}(HO)$ denote the probability of inter-cluster hand-off for clustering policy $\overline{\mathcal{K}}$, optimized in sense of minimizing the probability of inter-cluster hand-off in the network. An example of $\overline{\mathcal{K}}$ for a stationary off-line clustering is having all cells in the sub-network in one cluster. Note that even in that case, the probability of inter-cluster hand-off is greater than zero, since the users still can leave the sub-network. We also denote the probability of inter-cluster hand-off for any given clustering algorithm $\mathcal{K}$, by $P_{\mathcal{K}}(HO)$. Then we define *Cluster Isolation Factor* (CIF) as:

$$CIF = \frac{1 - P_{\mathcal{K}}(HO)}{1 - P_{\overline{\mathcal{K}}}(HO)} \tag{4.4}$$

Note that with no clustering (i.e. when each cell forms a cluster by itself) the probability of inter cluster hand-off in the network is 1 and hence CIF equals 0; on the other hand, having the clustering configuration which minimizes the inter-cluster hand-offs 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 4.2, subject to the empirical QoS requirement $\hat{P}_{HO}$ or equivalently $\widehat{CIF}$. Note that $\mathcal{A}$ is a function of both admission control and clustering algorithms.

To find an analytical solution to the optimization problem defined above requires a model of the cell occupancies as in Equation 4.2, which is a function of behavior of all users' mobility characteristics as in Equation 4.1. However due to the complex nature of a group of user's natural behaviors, there is not any suitable model available of the user's 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. These factors increase the complexity of the problem beyond 2^{M-1}.

Motivated by the intractability and dynamic nature of an ideal clustering algorithm, we next develop a heuristic adaptive clustering algorithm as an indirect solution to the defined optimization problem. In designing this algorithm, we exploit the mobility patterns of users' movements in order to form the clusters. In Chapter 5 we show that such an approach successfully forms the clusters in the network in a way that both the QoS metrics of the system are satisfied and the utilization of the network is close to the optimal solution.

4.2 Adaptive Clustering Algorithm

It is rather clear without exact stochastic models of system variables, it is not possible to find the accurate solution of the optimization problem introduced in Section 4.1. Moreover due to its complexity, it is too complicated a numerical problem to be solved on the fly. On the other hand the varying nature of users mobility, requires that clusters follow the changes in occupancy correlations among the cells, in order for the admission policy to be useful and effective.

In this section we discuss a heuristic adaptive clustering policy which effectively captures the variations of mobility patterns of the users and hence the occupancy correlations among the neighboring cells and forms the clusters based on this information. The adaptive clustering algorithm is presented in pseudo-code in Figure 4.2. The design principles of this algorithm are:

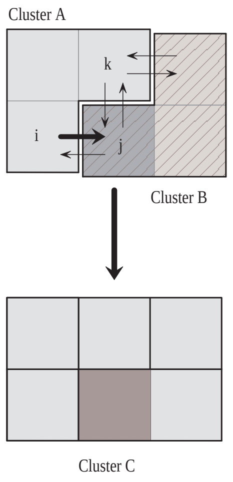
Initial state: The algorithm starts from the initial state where each individual cell in the sub-network forms a cluster of size one.

Clusters annexation: Consider cell j in Figure 4.3 (a) which belongs to cluster B. Whenever the occupancy of cell j exceeds some multiple of the capacity of cell βC_j, $\beta \geq 0$, the handed-in bandwidth from the neighboring cells of cell j, cells i and k in Figure 4.3 (a), during the past W time-slots is measured, where W is a pre-specified 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 $\alpha_h C_j$, $\alpha > 0$, the original cluster of the neighboring cell will join the cluster of cell j to form a new cluster.

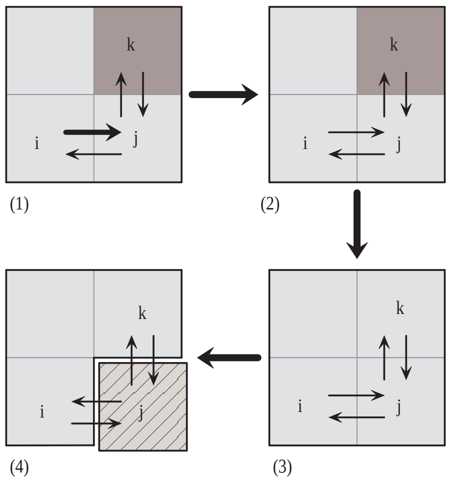
```
Adaptive Clustering Algorithm
1.   Initial Clustering: $M$ Clusters of Size 1;
2.   for ($j = 1, \cdots, M$ ;  $t = 0, \cdots, T$)  {
3.       if ($\Omega(j,t) \geq \beta C_j$)  {
4.          for (all-neighbors-of-cell $j$)  {
5.             if ([$\sum_W BW_{in} \geq \alpha_h C_j$] $\cap$ [neighbor-not-in-cluster])
6.                Add-Cluster-of-Neighbor-to-Cluster-of-Cell $j$;
7.          }
8.       }
8.       else if ($\Omega(j,t) < \beta C_j$)  {
9.             for (all-neighbors-of-cell $j$)  {
10.                if ( [$\sum_W BW_{in} < \alpha_l C_j$] $\cap$ [neighbor-in-cluster])  {
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.                      return;
15.                }
16.             }
17.       }
18.  }
```

Figure 4.2 Adaptive Clustering Algorithm

Cell separation: If the occupancy of cell j in Figure 4.3 (b) becomes lower than βC_j, then the handed-in bandwidth of those neighbors of cell j which are in the same cluster with cell j, cells i and k in Figure 4.3 (b), will be measured for the past W time-slots. If this value is less than $\alpha_l C_j$, $0 \leq \alpha_l \leq \alpha_h$, and the neighboring cell is not connected to any other cells of the cluster, 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 4.3 (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)).



(a) Clusters Annexation



(b) Cell Separation

Figure 4.3 Process of Clusters Annexation and Cell Separation

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 parameters involved in the clustering algorithm can be divided into two main categories:

Controlling the sensitivity to overload: The parameter β controls the level of sensitivity to overload in the algorithm. The smaller β is, the sooner the clusters form. Hence, in average there will be larger clusters in the network which reduces utilization but increases CIF.

Controlling the adaptivity: The parameters α_h and α_l control the adaptivity of the algorithm. For bigger values of α_h it takes a longer time for two clusters to annex. For small values of α_h the separation of a cell from its cluster happens slower. Hence, the bigger α_h and α_l, the smaller the clusters formed in the network. Small clusters result in higher utilization at the cost of a low CIF.

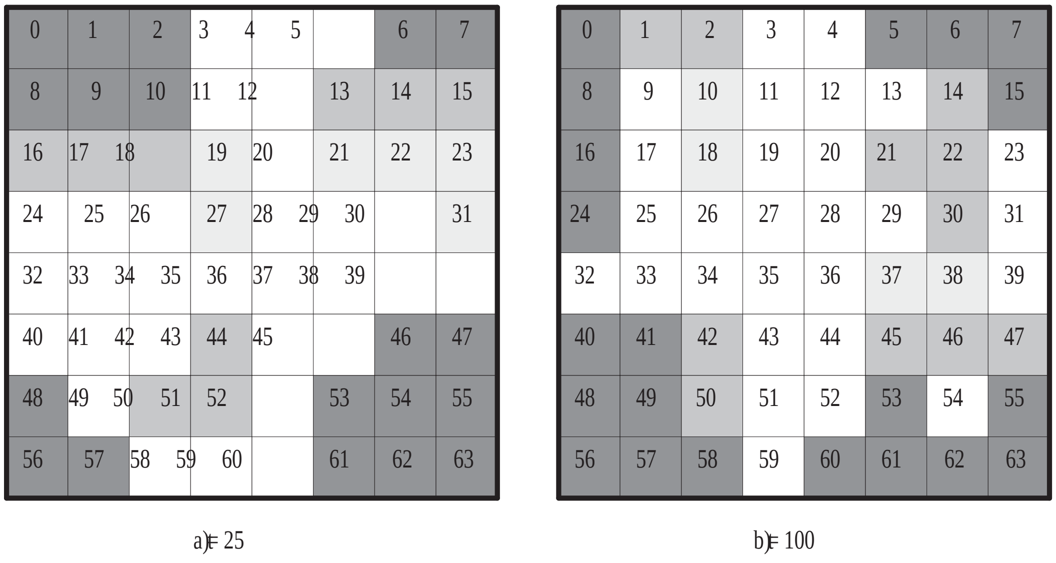


Figure 4.4 An Example of Adaptive Clustering

In Figure 4.4 we show an example of the adaptive clustering. Considering the sub-network 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 5.1, therefore the four cells 1, 7, 56, and 63 are with high probability the destination of the users, assigned to each user upon origination of the call.

Figure 4.4 (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 we see, the formation of clusters is concentrated around the hot spots (the cells 1, 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 pre-specified value. As the time passes and the users' mobility pattern changes, the value of moving average of the handed bandwidth between the cells also varies. The adaptive clustering algorithm follows these variations and reforms the clusters. Comparing Figure 4.4 (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 $t = 25$ an $t = 100$ 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.

In the next chapter we will study the characteristics and dependencies of the clustering parameters and will discuss the simulation results showing the performance of the VBC admission control algorithm with adaptive clustering policy.

Chapter 5

Experimental Results

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

In this chapter we first introduce the simulation environment and then present the results.

5.1 Simulation Environment

The simulation environment we use in our simulations is identical to the one introduced in [Jai99, JSK99], consisting of a two dimensional 64 cell network as shown in Figure 5.1. Hand-offs 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 hand-off between cell 63 and cell 7 will be considered an inter-router hand-off.

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.

The time is slotted to 1 minute intervals and both the call holding time and the cell residence time have geometric distribution with means 10 and 7, respectively, if

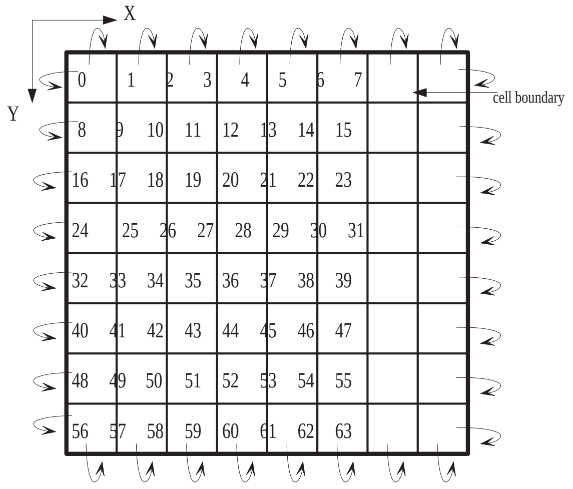


Figure 5.1 Cellular topology

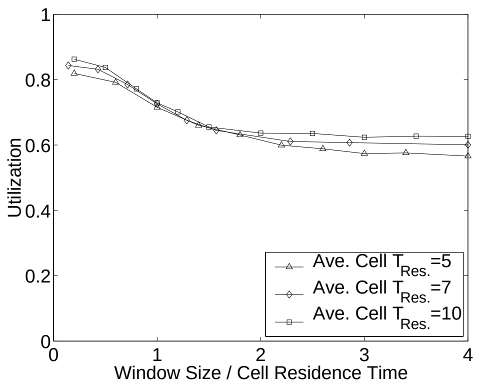
not mentioned otherwise. Simulation time for all the results presented is 6 hours; during which, a large number of users were introduced to the network. The capacity of each cell is 10 Bandwidth Unit (BU) and each user requires 1 BUs.

The traces of the users' movements were mostly produced using the simulator of [Jai99], but to implement the adaptive clustering and the VBC admission control algorithms, C++ code was written.

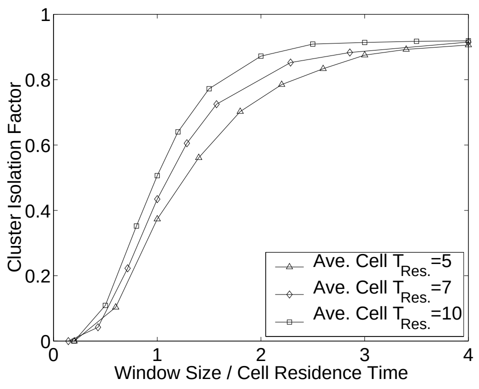
5.2 Design Issues of Clustering

In Figure 5.2 network utilization and CIF have been plotted v.s. 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 chapter, has been chosen to be the adaptive clustering policy with the goal of minimizing the number of inter-cluster hand-offs in the network. Such clustering policy is achieved by setting the three parameters β, α_h, and α_l equal to zero, since $\beta = 0$



(a) Utilization



(b) Cluster Isolation Factor

Figure 5.2 Impact of Measurement Window Size on Performance of the System.

means that the network is continuously measuring the amount of transferred bandwidth among the cells looking for the new clusters to make; also $\alpha_h = 0$ results two clusters to annex as soon as any hand-off happens between them, and finally $\alpha_l = 0$ means that no cell in the network separates from any cluster.

Choosing the clustering measurement window size to be large, results in the fast formation of the clusters since the system measures the amount of transferred bandwidth through the hand-offs in a larger window of time in the past and the probability that the measured value exceeds the threshold is higher. Fast formation of the clusters, in its turn, results in having bigger clusters in average in the network, and reduces utilization of the network. On the other hand the bigger the cluster, the less the probability of inter-cluster hand-off and the more clusters will be isolated.

As shown in Figure 5.2 (a), for different values of average cell residence time, utilization decreases as window size over mean residence time increases. Also as one can see in Figure 5.2 (b) CIF increases as window size increases. System, by its measurements, tries to capture the mobility of the users among the cells; hence, it is clear that choosing the window size to be less than average cell residence time, the measurement would not reveal the true amount of bandwidth transfered among the cells. The graphs suggest that a proper value for the window size is 1 to 2 times bigger than average cell residence time.

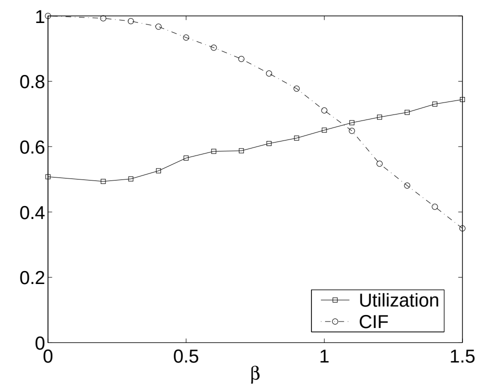
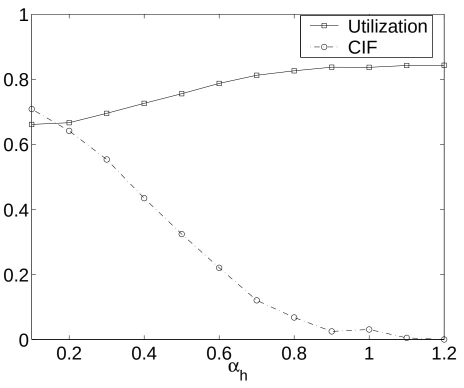


Figure 5.3 Impact of β on Performance of the System

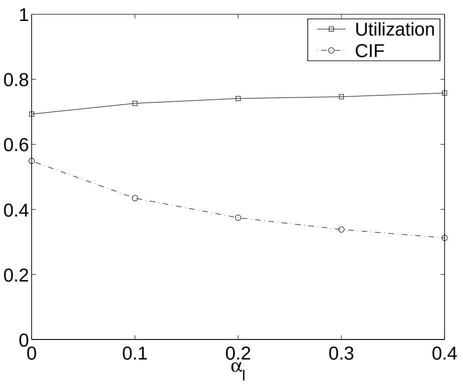
Figure 5.3 shows the impact of changes of the clustering process startup threshold β on the performance of the system for $\alpha_l = 0$ and $\alpha_h = 0$. The graph implies that choosing β to be less than 1 has the best effect on the isolation of the clusters. As an example, setting the parameter β to be equal to 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 5.4 (a) shows the impact of changes of the clustering formation threshold α_h, on utilization of the network as well as CIF for $\beta = .8$, $\alpha_l = 0.1$, and window size equal to the average cell residence time (7 time units). For lower values of α_h, there is a fast decrease in CIF, which suggests a smaller value for α_h.

The last parameter that can be used to adjust the clustering pace in the network is α_l which denotes the cell separation threshold. As shown in Figure 5.4 (b) having set the appropriate values for β and α_h, the changes of α_l in the valid range of $[0, \alpha_h]$



(a) Clusters Annexation Threshold



(b) Cell Separation Threshold

Figure 5.4 Impact of α_h and α_l on Performance of the System.

do not cause any significant changes in either utilization or CIF. The graph plotted in Figure 5.4 (b) is for $\beta = 0.8$ and $\alpha_h = 0.4$.

Knowing the effect of changes in these parameters on the clustering process, one can adjust them in a way to fulfill network requirements. Higher gains at the cost of lower QoS can be achieved by slowing down the process of clustering and setting higher values for the parameters. If, on the other hand, a better QoS is required, a fast and accurate clustering process will be needed which again can be easily achieved by making the network sensitive to the users' mobility by decreasing the values of the parameters. It is important to know that because of the variations in mobility patterns of the users in different environments, and the high dependency between the performance and efficiency of the network and the algorithm on one hand and the network structure and users' mobility and behavior patterns on the other hand, there is not a general solution for setting the parameters; but after a short term observation of the network's performance it would be rather easy to control the clustering pace such that the expectations of the network providers are satisfied. Regardless, based on our experiments, suggested initial settings are $\beta = 0.8$, $\alpha_h = 0.4$, and $\alpha_l = 0.1$.

5.3 Performance of the VBC Admission Control and Adaptive Clustering

In Figure 5.5 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.

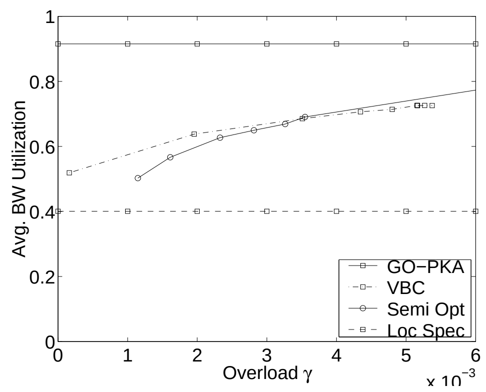


Figure 5.5 Performance of the VBC Admission Control

The first benchmark is *Globally Optimal Perfect Knowledge Algorithm* (GO-PKA) developed in [JK99]. GO-PKA is an optimal off-line admission control algorithm: given a set of users' admission requests in which each user has an associated bandwidth demand and mobility pattern (i.e., times and locations of hand-offs over the duration of the session), selects the optimal subset of users which maximizes the system's utilization while satisfying the required QoS (in this case, overload and overload time scale).

To obtain the curve labeled "GO-PKA", we used the simulator of [Jai99]. The figure depicts the average system utilization achieved for the optimal off-line admissible region for the case of no over-load, i.e. $\gamma = 0$ (hence the curve is flat). As shown, GO-PKA obtains a utilization of approximately 91.54%.

As a second baseline case, we compare VBC admission control to a "location specification" algorithm [Jai99] in which users pre-specify 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 GO-PKA as the times are not pre-specified: 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 adaptive clustering policy for the same traffic load. While no on-line algorithm can obtain utilization greater than GO-PKA, simultaneously 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 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 and its corresponding curve is labeled "Semi-Opt". For the particular simulation scenario discussed in Section 5.1, semi-optimal clustering policy (an intuitive optimal solution) would be forming 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 mentioned policy, which maximizes the utilized bandwidth while satisfying the QoS requirements. As the figure shows, the adaptive clustering outperforms the semi-optimal clustering even over a large range of overload and over the range that it's utilization is less than semi-optimal clustering, the difference is not significant.

These results show that the VBC admission control algorithm with adaptive clustering algorithm is a scalable scheme for wireless networks and can effectively and accurately control the system admissible region.

Chapter 6

Conclusions and Future Work

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 hand-offs. In this thesis, we proposed new techniques with the ability of performing scalable and coarse-grained QoS control in future systems, and introduced Virtual Bottleneck Cell Admission Control as a particular algorithm based on this design philosophy. We then developed a simple one-dimensional analytical model to investigate the performance of the proposed admission control algorithm. Using the results of our analytical studies we showed that the characteristics of the virtual system closely envelope the behavior of the underlying cells, enabling efficient provisioning of system resources, even under heterogeneous spatial demands and "hot spots". We then discussed the problem of clustering, which arises in any coarse-grained approach towards maintaining QoS in the networks and formulated it as an optimization problem. Defining the optimal clustering policy we then designed a heuristic adaptive clustering algorithm as a solution to the defined optimization problem. We showed that our clustering algorithm is successful in adaptively capturing the variations in users' mobility pattern in the network and hence forming the clusters around the network bottlenecks. Finally we performed simulation experiments in a two-dimensional system. Here we first studied the characteristics of the parameters involved in adaptive clustering policy and then applying two different optimal off-line admission control algorithms as benchmarks, found out that the coarse-grained approach can effectively control the system's admissible region. We finally compared our adaptive clustering algorithm with an intuitive optimal clustering policy and showed that our heuristic approach outperforms such a solution.

We believe that using the proposed design philosophy we have a powerful tool to maintain quality of service in multi-tier networks, where the users have the chance to hand-off not only to different cells in the network they are connected to, but also to different networks providing service in the area. In such a network scalability is an important issue in design of any algorithm. In addition to the issue of multi-tier networks, in our future work we will deal with expanding our admission control algorithm and the adaptive clustering for the more general cases of having more than one QoS class in the network.

Bibliography

[AZ99] 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, NY, March 1999.

[B+98] S. Blake et al. An architecture for differentiated services, 1998. Internet RFC 2475.

[CBN98] Y. Choi C. Bisdikian, T.Kwon 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.

[CC97] C. Chao and W. Chen. Connection admission control for mobile multiple-class personal communications networks. *IEEE Journal on Selected Areas in Communications*, 15(8):1618–1626, 1997.

[CS98] S. Choi and K. Shin. Predictive and adaptive bandwidth reservation for hand-offs in qos-sensitive cellular networks. In *Proceedings of ACM SIGCOMM'98*, Vancouver, British Columbia, August 1998.

[GCM98] 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.

[Jai99] R. Jain. *Admission Control in Multi-service Wireless.* M.S. Thesis, Rice University, May 1999.

[JK99] 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, NY, March 1999.

[JSK99] R. Jain, B. Sadeghi, and E. Knighyly. Towards corse-grained mobile qos. In *Proceedings of the Second ACM International Workshop on Wireless Mobile Multimedia*, Seattle, WA, 1999.

[KK99] J. Kim and M. Krunz. Quality of service over wireless ATM links. In *Proceedings of IEEE INFOCOM '99*, New York, NY, March 1999.

[LAN97] 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):1–12, February 1997.

[LB96] S. Lu and V. Bharghavan. Adaptive resource management for indoor mobile computing environments. In *Proceedings of ACM SIGCOMM '96*, Stanford, CA, 1996.

[Mir96] Boris Mirkin. *Mathematical Classification and clustering.* 1996.

[MS98] 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.

[NS96a] M. Naghshine and M. Schwartz. Distributed call admission control in mobike/wireess networks. *IEEE Journal for Selected Areas in Communications*, 14(4):711–717, 1996.

[NS96b] M. Naghshine and M. Schwartz. Distributed call admission control in mobile/wireess networks. *IEEE Journal for Selected Areas in Communications*, 14(4):711–717, 1996.

[NWL97] M. Naghshineh and M. Willebeek-LeMair. End-to-end QoS provisioning multimedia wireless/mobile networks using an adaptive framework. *IEEE Communications*, 35(11):72–81, November 1997.

[QK98] J. Qiu and E. Knightly. QoS control via robust envelope-based MBAC. In *IEEE/IFIP IWQoS '98*, Napa, CA, May 1998.

[RNT96] R. Ramjee, R. Nagarajan, and D. Towsley. On optimal call admission control in cellular networks. In *Proceedings of IEEE INFOCOM '96*, pages 43–50, San Francisco, CA, March 1996.

[Rom84] H. Charles Romesburg. *Cluster Analysis for Researchers.* 1984.

[Sin96] S. Singh. Quality of service guarantees in mobile computing. *Computer Communications*, 19(4):359–371, April 1996.

[Spa80] Helmuth Spath. *Cluster Analysis Algorithms.* 1980.

[TBA99] A. Talukdar, B. Badrinath, and A. Acharya. Integrated services packet networks with mobile hosts: Architecture and performance. *Wireless Networks*, 5(2):111–124, 1999.

[ZK97] H. Zhang and E. Knightly. RED-VBR: A renegotiation-based approach to support delay-sensitive VBR video. *ACM Multimedia Systems Journal*, 5(3):164–176, May 1997.