Adaptive Resource Management Algorithms for Indoor Mobile Computing Environments

Songwu Lu and Vaduvur Bharghavan
Coordinated Sciences Laboratory
University of Illinois at Urbana-Champaign
Urbana, IL - 61801
{slu, bharghav}@crhc.uiuc.edu

Abstract

Emerging indoor mobile computing environments seek to provide a user with an advanced set of communication-intensive applications, which require sustained quality of service in the presence of wireless channel error, user mobility, and scarce available resources. In this paper, we investigate two related approaches for the management of critical networking resources in indoor mobile computing environments:

- adaptively re-adjusting the quality of service within pre-negotiated bounds in order to accommodate network dynamics and user mobility.
- classifying cells based on location and handoff profiles, and designing advance resource reservation algorithms specific to individual cell characteristics.

Preliminary simulation results are presented in order to validate the approaches for algorithmic design. A combination of the above approaches provide the framework for resource management in an ongoing indoor mobile computing environment project at the University of Illinois.

1 Introduction

Recent years have witnessed explosive research in the field of mobile computing, resulting in the development of indoor mobile computing environments which seek to provide a mobile user with not only a basic toolchest of applications – editors, local and disconnected file systems, compilers, email, ftp, telnet, etc. – but also more advanced applications such as multimedia (video/audio/teleconferencing), WWW browsers, distributed file systems, distributed databases, etc. The latter class of applications is communication-intensive, and will stress the wireless networking component of the mobile computing environments severely. State-of-the-art solutions are inadequate to cope with the anticipated load of such applications in any but the smallest research prototype-sized mobile computing environments. A fundamental problem is that the wireless medium is a scarce shared resource, and needs to be managed efficiently to provide an acceptable quality of service to communication-intensive applications.

This paper investigates two related approaches for resource management in indoor mobile computing environments:

- providing loose quality of service (QoS) bounds, and adaptively changing the QoS within these bounds depending on dynamic network conditions.
- classifying cells based on location and behavior profiles, and designing advance resource reservation algorithms for each class of cells.

The overall approach is to establish QoS bounds for a connection, advance reserve the minimum required resources (for handoff without QoS re-negotiation) in the next-predicted cell, and adaptively re-adjust QoS levels (within the pre-specified bounds) of ongoing connections if required, in order to accommodate new connections and connection handoffs.

The organization of the paper is as follows. Section 2 motivates the issues addressed in this paper and Section 3 describes the system model. Section 4 provides an overview of our algorithms for resource management. Section 5 describes the adaptive resource reservation algorithm and Section 6 describes the advanced reservation algorithm. Section 7 presents preliminary performance results and Section 8 concludes the paper.

2 Issues in Resource Management

In order to provide effective resource management in mobile computing environments, the following issues which are unique to such environments need to be addressed:

- wireless channel error and inter-cellular mobility: the case for *loose QoS bounds* and *QoS adaptation*.
- 'seamless' inter-cellular mobility and providing QoS guarantees: the case for *advance resource reservation*.
- location dependent behavior of cells: the case for *classification of cells* and location dependent resource reservation algorithms.

2.1 Loose QoS Bounds and QoS Adaptation

Since the wireless medium is the critical shared resource in an indoor mobile computing environment, an equitable distribution of the scarce wireless bandwidth among the contending users involves the negotiation of QoS between applications and the network. While QoS negotiations in wired networks typically result in the network providing fixed levels of deterministic or statistical service guarantees (via resource reservation) to the application [2, 3], two factors motivate a modification of the standard QoS concept in wireless networks: (a) physical characteristics of the wireless media, and (b) user mobility. Wireless media are prone to error; thus standard assumptions such as negligible channel error are not true in the wireless scenario. User mobility implies that the QoS negotiated between the application and the network in one cell may not be honored if the user crosses cell boundaries. In order to provide 'seamless mobility' across cells, and to accommodate for wireless channel error, our end-to-end QoS negotiation process results in the network providing QoS within an agreed-upon bounded range. Either the network or the application may dynamically initiate QoS re-negotiation using the runtime support provided in the environment [14]. By specifying the QoS bound, the guaranteed service and the best-effort service can be unified in a single framework: the service is guaranteed in the sense that each connection is guaranteed its minimum QoS requirement; the service is best effort in the sense that the network provides best-effort service beyond the minimum QoS support. Besides, it enables the network to conduct transparent resource adaptation and vary the QoS level dynamically, which is especially meaningful for the time-varying effective capacity of the wireless link.

2.2 Advance Resource Reservation

For a user to be provided the illusion of seamless mobility, the user should not notice perceptible change in QoS upon mobility between cells. This motivates advance reservation of resources for a user who is expected to enter a cell from a neighboring cell. A brute-force approach thus reserves resources for an application in all the neighboring cells of its current cell [7]. In an environment where wireless networking resources are scarce, such a conservative advance reservation scheme is wasteful. A more efficient alternative is to predict the next cell of the user, and reserve resources for the user's applications only in the next predicted cell(s) [1]. In case of erroneous prediction, the resource management scheme may (a) provide a pool of resources at each cell to account for such an eventuality, (b) resort to dynamic QoS re-negotiation, or (c) use a combination of the above approaches. In this paper, we provide algorithms to predict the next cell of a mobile user, and advance reserve resources in the next cell. We handle wrong predictions by using both the buffer pool approach, and adaptively readjusting the QoS of ongoing connections within pre-specified bounds in order to handle new connections.

2.3 Classification of Cells

In an office environment with small cells, the behavior of cells is spatially dependent – thus the resource reservation algorithm at a cell needs to be tailored to its behavior profile. For example, an office has regular occupants and needs to advance reserve resources only for its occupants. A corridor typically experiences linear movement through it and can predictively reserve resources in the next cell, given the previous cell of a mobile user. Meeting rooms are typically reserved in advance and it may be possible to estimate the resources which need to be reserved in advance of the meeting. We attempt to make a classification of the cells into offices, corridors and lounges, and propose intuitive advance reservation algorithms for each class of cells.

3 System Model

3.1 Network Model

The network has a cellular architecture, consisting of a wired backbone component and a wireless cellular component. Base stations are connected to the backbone network, and provide wireless networking access to portable computers within a geographical region around them called *cells*. Two cells are called *neighbors* if it is possible for a handoff to occur between them. Neighboring cells overlap with each other, thus ensuring continuity of network access when a user moves between cells. All wireless network traffic is constrained to be either uplink (portable to base-station) or downlink (base-station to portable).

The backbone wired network and the wireless network are assumed to have low level mechanisms to provide QoS support to connection-oriented real-time traffic, as well as best effort support to connectionless data traffic.

3.2 Application Model

This work is prompted by emerging communication-intensive applications which will stress the wireless network severely. The applications of interest generate periodic multimedia traffic as in teleconferencing applications, or bursty data traffic as in WWW browsers. While the focus of our research is to devise adaptive resource management strategies for multimedia applications, bursty bulk data traffic may also specify QoS requirements, or may use the available resources in a best-effort manner.

Many applications are adaptive in nature and can therefore generate variable QoS requirements. For example, most video compression standards, like MPEG, JPEG and JBIG, have a notion of 'progressive mode' or 'hierarchical mode', e.g. a lossy compression using MPEG encoding can yield a data rate from 1.5Mbps to 6.0Mbps for a digital NTSC signal, a lossy compression to a CD quality audio can yield data rate from 384kps to 1.41Mbps depending on the quality desired [10]. In the wireless communication environment, recently developed hardware for video coding can adaptively deliver digital video at rates between 60K bps and 600K bps [11]. In fact, we believe that with adequate runtime support [14], future mobile computing applications can use QoS bounds in order to adapt effectively to dynamic network conditions.

3.3 Reservation Model

A cell manages its resources by two instruments: (a) reservation of resources for each ongoing connection or predicted connection handoff, and (b) maintaining a pool of resources in order to handle new connection requests, unforeseen events, and best-effort traffic.

Resources are reserved at a cell for a connection if (a) the connection is ongoing in the cell, or (b) the connection is in a neighboring cell, and the resource management algorithm predicts that the user of the connection will move into the cell.

Within the above framework, the resource management algorithms seek to do the following: (a) perform end-to-end resource reservation for connections, (b) predict the next cell for a user based on user/cell profiles, and reserve resources in advance based on the next-cell prediction, (c) maintain a dynamically adjustable portion of reserved resources in order to handle unforeseen events, and (d) provide smooth connection handoff when a user moves between cells. The focus of this paper is on the first two functions.

3.4 Cell Classification Model

3.4.1 Cells and Zones

The cell where a portable is located is called its *current cell*. Neighbors of this cell are called the portable's *neighboring cells*. The current cell and the set of neighboring cells form the *neighborhood* for a portable at any time. The set of cells in the mobile computing environment is called the *universe* of cells. The universe is divided into distinct geographical regions called *zones*. Each zone has a *profile server*, whose role is defined in Section 3.4.3. The locational hierarchy for a portable is thus the current cell, neighborhood, zone, and universe.

For an indoor mobile computing environment, cells are divided into three classes based on location: *office*, *corridor* and *lounge*. Lounges are further classified into three subclasses based on their activity: *meeting room*, *cafeteria* and *default*. Section 6 provides a detailed description of each class and its impact on resource management. While the classification of cells is prompted by the modeling of an indoor office environment, the resource management algorithm designed for lounges may also be applicable to outdoor mobile computing environments [12].

3.4.2 Static and Mobile Portables

A portable[1] is defined to be *static* if it has remained in the same cell for a threshold period T_{th} of time; otherwise, it is defined to be *mobile*. The motivation for this classification is that we expect static portables remain stationary and mobile to continue moving.

Therefore, for a static portable, (a) resources are not reserved for its connections in its neighboring cells, and (b) the QoS for its connections is upgraded to the maximum level that the network can provide, within the pre-negotiated QoS bounds.

Likewise, for a mobile portable, (a) resources are reserved for its connections in its next-predicted cell, and (b) the QoS for its connections are kept at the pre-negotiated minimum level (minimizes QoS adaptation during inter-cell mobility).

3.4.3 Profiles and Profile Servers

Each cell and portable in the mobile computing environment has a *profile*. The profile of a portable contains its identification, authentication information, and an aggregated history of its previous handoffs, which is used to predict its next cell given its current cell. The profile of a cell contains its identification, authentication information, cell class, the set of its neighbors and their cell class, and an aggregated history of its previous handoffs, which is used to predict the next cell for a portable (in absence of information from the portable's profile). Table 1 specifies the contents of the portable and cell profiles.

Each zone has a profile server. The profile server maintains the cell-profiles for all the cells in its zone and the portable-profiles for all the portables currently in its zone, and updates the cell/portable-profile upon each handoff. The profile server keeps track of each handoff for each portable

Table 1: Cell and Portable Profiles

type	handoff activity	profile[2] contents
office (c)	predictable	$\omega(c)$, $\eta(c)$, $\forall i \in \eta(c)$, $< i, \forall j \in \eta(c), \{j, p_j\} >$
corridor (c)	predictable linear movement	$\eta(c)$, $\forall i \in \eta(c)$, $< i, \forall j \in \eta(c), \{j, p_j\} >$
lounge (c) meeting room	spikes	$\eta(c)$, booking calendar, $\forall i \in \eta(c)$, $< i, \forall j \in \eta(c), \{j, p_j\} >$
lounge (c) cafeteria	slow time-varying	$\eta(c)$, $\forall i \in \eta(c)$, $< i, \forall j \in \eta(c), \{j, p_j\} >$
lounge (c) default	uniformly distributed	$\eta(c)$, $\forall i \in \eta(c)$, $< i, \forall j \in \eta(c), \{j, p_j\} >$
portable		$\forall i, \forall j \in \eta(i)$, $< j, i, next\text{-}prd\text{-}cell >$
functions	$\omega(c)$	occupants of office c
	$\eta(c)$	neighbors of c

in each cell, and computes the next-predicted cell by a brute force method by aggregating the past history for each portable and each cell. The actions of the profile server are fairly straightforward. We present a summary below.

- For each portable in its zone, the profile server maintains the following information about the last N_{pP} handoffs from each cell in the zone, for that portable: <portable id, current cell id, previous cell id, next cell id>. Based on the above information, the profile server can predict the next cell for a portable, given its previous and current cell. The aggregate history in the portable-profile consists of the set of <previous cell, current cell, next-predicted-cell> triplets for the zone (see Table 1).

- For each cell in its zone, the profile server maintains the following information about the last N_{pC} handoffs of the cell: <previous cell of handing off portable, next cell of handing off portable>. Based on the above information, the profile server can predict the next cell for a portable in a cell, given its previous cell. The aggregate history in the cell-profile consists of the set of <previous cell, probability of handing off to each neighboring cell> for its neighborhood (see Table 1).

- The difference between the next-cell prediction based on portable-profile and cell-profile is that the former is specific to the portable, while the latter aggregates the handoff information of all portables in its cell.

- A base station caches its cell-profile, and portable-profiles of all the portables currently in its cell. During handoff, it sends an update message (about the handoff) to the profile server, and passes on the cached portable-profile to the next cell. Once a portable becomes static in a cell, the base station refreshes the portable-profile from the profile server.

4 Overview of Resource Management Algorithms

Figure 1 provides a list of resource management algorithms in mobile computing environments and their interaction. The

[1] By a portable, we mean the user of a portable.

[2] Every profile contains the identification and authentication information of the entity.

overall operation is as follows:

When a portable requests a new connection with QoS bounds (if no QoS parameters are specified, the network will provide best-effort service), the network will conduct an *admission control* test (and tentatively reserve resources) during the forward pass to the destination via an appropriate route found by a *routing* algorithm. During the reverse pass, the network will allocate (i.e. firm reservation) resources, i.e bandwidth, buffer space and schedulability[3] for it. The network may implicitly invoke the conflict resolution algorithm when resource conflicts arise (see Table 2). Besides, the base station performs a *static/mobile test* for a portable. If the portable is *static*, resources are not reserved in advance in its neighboring cells. If it is *mobile*, the network will make a *predictive advance reservation* using either a prediction algorithm based on cell and portable profiles, or a *default advance reservation* algorithm (in the absence of useful information from the profiles).

To reduce transient behavior of connections to a *mobile* upon handoff, the backbone network will also set up multicast routes for the connection in all neighboring cells so that the network can *multicast* the packets to the pre-allocated buffer space in these neighbors. To set up these multicast routes on the *wired* network, end-to-end admission control test and associated resource reservation are also performed for them. However, the failure of the end-to-end test along any route will not cause the forced termination of the connection. When network conditions change or upon handoff, the network will (automatically) initiate *resource adaptation*, again invoking the *conflict resolution* algorithm to satisfy the QoS bounds for all the ongoing connections by re-allocating resources among connections within their pre-specified QoS bounds. Furthermore, the application can also initiate *adaptation* which causes the network again to initiate procedures for admission control, and resource reservation (including the predictive advance reservation).

4.1 Admission Control and Resolution of Resource Conflict

Admission control converts end-to-end QoS requirements into per-hop requirements and tests for the availability of resources at intermediate nodes. In the context of a mixed wireless/wireline network, admission control is conducted with respect to two types of connections: a *newly arriving* connection and a *handoff* connection. Admission control tests are also performed for the multicast routes (that enable multicasting to the neighbors of the current cell) on the wired network. Note that the possible failure of these tests will not cause the rejection of the connection in the current cell.

The introduction of QoS bounds induces a new problem for admission control, i.e. resource conflict. The problem of resource conflict arises when the network cannot accept a new connection without reducing currently allocated resources (within pre-negotiated QoS bounds). In this paper, we provide an algorithm for the resolution of resource conflicts in order to accommodate new connections (note that the resolution of resource conflicts readjusts the resources allocated to each connection but does not violate the pre-negotiated bounds of existing connections).

[3]By schedulability, we mean resources at switches that provide delay guarantees.

4.2 Adaptation

Adaptation may be triggered by changes in the measured QoS over the wireless link, upon resource availability on the wired network, upon portable handoff, or when the application initiates a QoS re-negotiation.

In our algorithm, adaptation is conducted only for a connection from a *static* portable. Besides, adaptation can be initiated by either the network, or the application running on the portable.

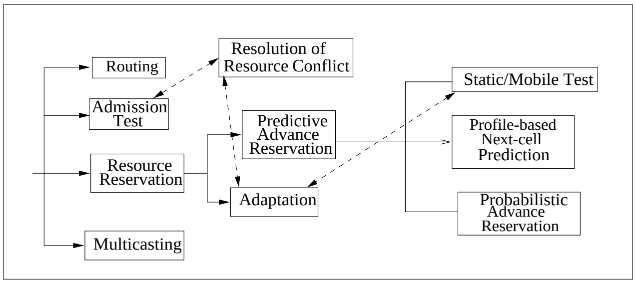

Figure 1: **Overview of Resource Management Algorithms**

4.3 Predictive Advance Resource Reservation

The advance resource reservation algorithm[4] consists of three related components: bandwidth reservation, buffer space reservation, and schedulability reservation.

Resource reservation is achieved by the predictive advance reservation algorithm based on portable/cell profiles, and the default algorithm in absence of profile information. We distinguish *static* portables from *mobile* portables. For a *static* portable, bandwidth (as well as schedulability and buffer space) is not reserved in advance in its *neighboring* cells on a portable-specific basis; instead, each base station in the *neighboring* cells sets aside a dynamically adjustable fraction of resources (i.e. bandwidth, buffer space, and schedulability) ($5\% - 20\%$) to accommodate "unforeseen" events (e.g. sudden mobility of static portables)[5]. For a *mobile* portable, the minimum acceptable resources are reserved in its next-predicted cell(s) (the algorithm for predicting the next cell is described in Section 6).

5 Adaptive Resource Reservation

5.1 Admission Test

The admission control algorithm is designed for both new connections and connection handoffs. It is also conducted for the multicast routes (in the neighboring cells) that are set up for a mobile [6].

To request a *new* connection, the application specifies the following QoS parameters: lower and upper bounds on bandwidth, $[b_{\min}, b_{\max}]$; upper bound on end-to-end delay,

[4]Our focus of discussion here is on the wireless link, but note that the wired links need to change accordingly.

[5]Note that no multicast routes in neighboring cells will be set up corresponding to this fraction of resources. Therefore, in case of sudden movement of static portable, an end-to-end admission test and associated resource reservation have to be conducted; this might cause some handoff delay, but it reduces the handoff dropping due to the adjustable fraction of reserved resources for the wireless link

[6]This is for WFQ.

[7]This is for RCSP with $b^*(\cdot)$ RJ regulators [13].

Table 2. Admission Test for a New Connection Request

	forward pass test (at link l)	destination node	reverse pass reservation (same link l)
bandwidth	$b_{\min,j} \leq C_l - b_{resv,l} - \sum_{i=1}^{n} b_{\min,i}$		*static*: $b_j := b_{\min,j} + b_{stamp}$; *mobile*: $b_j := b_{\min,j}$
delay	$d_{l,j} := L_{\max}/b_{\min,j} + L_{\max}/C_l$	$d_{\min,j} := (\sigma_j + nL_{\max})/b_{\min,j} + \sum_{i=1}^{n}(L_{\max}/C_i) \leq d_j$	$d'_{l,j} := d_{l,j} + (d_j - d_{\min,j})/n + \sigma_j/(nb_{\min,j})$
jitter	$(\sigma_j + lL_{\max})/b_{\min,j} \leq \bar{\sigma}$	$(\sigma_j + nL_{\max})/b_{\min,j} \leq \bar{\sigma}$	$(\sigma_j + nL_{\max})/b_{\min,j}$
buffer	$\sigma_j + lL_{\max}$ [6] $\sigma_j + L_{\max} + b_{\max,j}d_{1,j}$, $l = 1$. [7] $\sigma_j + L_{\max} + b_{\max,j}(d_{l-1,j} + d_{l,j})$, $l \neq 1$. [7]		$\sigma_j + lL_{\max}$ [6] $\sigma_j + L_{\max} + b_j d'_{1,j}$ [7] $\sigma_j + b_j(d'_{l-1,j} + d'_{l,j})$ [7]
packet loss	$p_{e,l}$	$1 - \prod_{i=1}^{n}(1 - p_{e,i}) \leq p_e$	

d; upper bound on end-to-end delay jitter $\bar{\sigma}$; and maximum packet loss probability p_e.

The admission test and associated resource reservation for new connections are conducted during a round-trip process [3]. In the forward pass, tests on bandwidth, delay and jitter, buffer, and packet loss are conducted, and resources of bandwidth, buffer space and schedulability are reserved to the greatest level of local QoS support. At the destination, the end-to-end QoS requirements are compared with the network's end-to-end availability. During the reverse pass, the network reclaims the over-reserved resources.

Table 2 presents the admission control test conducted at each node[8]. As we describe in the next section, these tests also incorporate the resource conflict resolution algorithm. The notations are as follows: the traffic model (σ_j, ρ) is used for connection j, $L_{\max}$ is the largest packet size, C_l is the link speed, and b_{stamp} is defined in subsection 5.3.1. Besides, we use a "uniform" relaxation policy for delay resource reclamation, and assume the inter-link independence for packet loss probability. We also use two representative scheduling disciplines is used at intermediate network nodes[13]: the working-conserving weighed fair queueing (WFQ) and the non-working-conserving rate controlled static priority (RCSP) to illustrate the admission test. We also assume there exists a maximum delay bound for the wireless link. propagation delay is omitted for simplicity of presentation.

The admission test for a *handoff* connection is the same as that for a new connection except that connection handoff is able to use the (advance) reserved resources (e.g. $b_{resv,l}$ for bandwidth) and the network treats the connection as from a *mobile*.

It should be noted that the end-to-end admission test and associated resource reservation (with minimum pre-negotiated QoS bound) are also performed for the multicast route (to be used by the *multicasting* mechanism) on the wired network; however, the acceptance of a new or handoff connection is not contingent upon the success of these admission tests.

5.2 Resolution of Resource Conflicts

A problem related to admission control is resource conflict resolution[9]. Since we provide for QoS bounds, resource conflicts arise under two conditions: (a) excess resources need to be distributed among competing connections, and (b) a new connection arrives and can be admitted without violating the pre-negotiated lower QoS bounds of ongoing connections, though the currently available excess resources as insufficient to admit the connection. The former case involves increasing the resources for connections originating or terminating at static portables, while the latter case involves reducing the resources for ongoing connections in order to accommodate a new connection. In this paper, we consider both cases.

There are two important issues in resource conflict resolution: (a) the policy for allocation of excess resources among competing connections; (b) the mechanism to achieve the policy in the proposed network architecture.

Our policy for allocation of excess bandwidth is based on the *maxmin* optimality criterion[8], which is is both fair and efficient – it is fair in the sense that all connections constrained by a bottleneck link get an equal share of this bottleneck capacity; it is efficient in the sense that the bottleneck resource is utilized up to its capacity.

Our conflict resolution algorithm presented here is based on a distributed rate allocation algorithm which was originally proposed in the congestion control context [8]. The mechanism is based on exchanging resource availability information between network switches via some signaling channels. In next subsection, we will describe how to resolve resource conflict in the context of resource adaptation. The same algorithm applies to the case of admission tests. Before describing the resolution algorithm in details. we first define the notion of "bottleneck link".

The notion of excess bandwidth available to a connection at a link is critical to our definition of a bottleneck. In the following, the *excess*[10] available bandwidth for connection j at link l is denoted by $b'_{(av,l),j}$. For the network, the *excess* available network bandwidth at link l is denoted by $b'_{av,l}$, defined as $b'_{av,l} := C_l - b_{resv,l} - \sum_{i \in L(l)} b_{\min,i}$ where $L(l)$ denotes the set of all on-going connections at link l.

A network link l is a "connection bottleneck link" for an *unsatisfied* connection j (i.e. $b'_{(av,j),i} < b_{\max,j} - b_{\min,j}$ at some link i) if $b'_{(av,j),l} = \min_{i \in L(path,j)} b'_{(av,j),i}$ where $L(path,j)$ is the set of all links that connection j traverses from end to end.

Accordingly, the "network bottleneck link" is defined as follows: If all connections traversing a link l have infinite demand, then link l is a network bottleneck if the following

[8] Note that for a connection from a static portable, the delay and jitter tests are conducted using $b_{\min,j}$ resulting maximum possible end-to-end delay, which is necessary for adaptation since we do not adapt delay resources in the adaptation algorithm (see 5.3).

[9] We assume here that the routing algorithm is unable to find a path which can satisfy the requirements of the new connection.

[10] By excess, we mean the amount beyond the minimum required resource. This explanation is used hereafter.

is true:

$$b'_{av,l}/N_l = \min_{i \in L(net-links)} (b'_{av,i}/N_i) \quad (1)$$

where N_i denotes the number of all connections traversing link i.

In the presence of any connections that have finite demand, the above definition is applied in a recursive manner such that in each iteration, the satisfied connections with $b_{\max,j} \le b_{\min,j} + b'_{(av,i),j}$ are removed and their resources are deducted from the total, and the definition is re-applied to the rest of unsatisfied connections at link i with $b_{\max,j} > b_{\min,j} + b'_{(av,i),j}$ until the unsatisfied connection set remains unchanged.

A network bottleneck link is necessarily a connection bottleneck for all connections passing through it, but the converse may not be true in general.

5.3 Resource Adaptation

By resource adaptation, we mean adaptation with respect to bandwidth as well as buffer space. However, no delay or jitter is adapted in this paper, and tests for delay and jitter are conducted with respect to the worst-case scenario during the admission test (see Table 2).

For application initiated adaptation, the network essentially treats it as a new connection request.

For network-initiated adaptations, the network performs bandwidth adaptation only for connections from a *static* portable (for a frequently handing-off *mobile* portable, the control and processing overhead might completely compromise the performance improvements due to adaptation). Our focus here is the bandwidth adaptation; the adaptation of reserved buffer space changes accordingly similar to that in the admission test (see Table 2). Adaptation is initiated for connection from a *static* portable when the following bandwidth change at link l is detected:

$$(b'_{av,l}(t) < b'_{av,l}(t^-)) \quad \text{OR}$$
$$(b'_{av,l}(t) \ge \sum_{i \in L(l)} b'_{(av,l),i}(t^-) + \delta \quad \text{and} \quad \mathcal{M}(l) \ne \Phi) \quad (2)$$

where δ is a threshold value that is introduced to control the frequency of bandwidth adaptation, and $\mathcal{M}(l)$ is a set (maintained by link l) of all connections that consider l as a connection bottleneck link. If $b'_{av,l} < 0$, then some connections are notified to do re-negotiation.

Besides the adaptation that is performed in the current cell for a static portable, reservation for bandwidth and buffer space might also be adapted accordingly in its neighboring cells. Specifically, the dynamically adjustable fraction of bandwidth B_{dyn} (reserved to accommodate "unforeseen events", e.g. sudden movement of static portables) in the neighboring cells is adjusted to the following policy: B_{dyn} has to be adapted to accommodate at least a connection (with the maximum allocated bandwidth) from a *static* portable that is residing in its neighboring cells. has to be adjusted, as does the buffer reservation in the neighbors.

5.3.1 A Bandwidth Adaptation Algorithm

The bandwidth adaptation algorithm is based on a distributed algorithm for optimal rate allocation which satisfies the maxmin optimality criterion.

In order to have a meaningful notion of optimal allocation, we consider a period of instability (when connections are initiated, adapted and terminated) delimited by periods of stability. We prove that our algorithms converge to optimal allocation of bandwidth as defined by the maxmin criterion during a period of stability following a period of instability.

We now describe the algorithm in [8], followed by a successive refinement to the algorithm.

Each source node (for the wireless link, the base station will be the 'source') of a connection maintains an estimate of its optimal bandwidth share, and it updates this estimate via the *periodic* sending of control packets (to other network nodes of the connection). In the control packets, the next estimate for optimal bandwidth for the connection is contained, and the source node updates the bandwidth for the connection based on the bandwidth value in the returned control packet.

In order to adapt the above algorithm to a mobile computing environment, we propose an event-driven approach which initiates adaptation upon handoffs and dynamically changing network capacities.

A preliminary approach is the following: The network switches exchange information on the current network resources by exchanging ADVERTISE control packets. Every network switch maintains a list of all its connections for each of its links, monitors its traffic and calculates the fair share of its excess available capacity on a per connection basis, referred to as "advertised rate".

When a switch has detected changes in bandwidth availability for a link, it initiates two ADVERTISE packets for every connection traversing this link along the two upstream and downstream directions for the connection. The initiating switch will put its calculated "advertised rate" into a field (in the ADVERTISE packet) called "stamped rate", denoted by b_{stamp}. The stamped rate represents the switch's desired bandwidth for the connection. Upon receiving an ADVERTISE packet for a specific connection, every intermediate switch compares its own advertised rate with the stamped rate included in the control packet. If the "stamped rate" is higher or equal to the "advertised rate", the stamped rate is reduced to the "advertised rate"; otherwise, the stamped rate remains unchanged. Besides, the intermediate switch will initiate ADVERTISE packets for every other connection traversing the same link. At the source and the destination (for a connection) ADVERTISE packets will be forwarded back to the initiating switch. After receiving both control packets, the initiating switch will repeat the above process. It has been shown in [8] that a total of four round trips are required to ensure convergence. The initiating switch then sends out UPDATE messages for its connections to adjust bandwidth according to the minimum of the two latest received stamped rates contained in the control packets. If any switch receives both UPDATE and ADVERTISE packets for a connection simultaneously, it processes the UPDATE packet first, and then proceed to handle the ADVERTISE packet. A global ID and a sequence number are included in an ADVERTISE packet to avoid possible infinite loop due to the flooding mechanism for the ADVERTISE packets.

The computation of the "advertised rate" is as follows[11]. A switch maintains the last seen stamped rates for all its ongoing connections, referred to as "recorded rates". The set

[11]In the following calculation, we assume that every connection has infinite bandwidth demand; for a connection with finite bandwidth requirement $b_{\max}$, we can create an artificial link that has capacity $b_{\max}$ at the entry of the connection.

of connections with recorded rates below or equal to the advertised rate are called "restricted" connections, denoted by $\mathcal{R}$. The connections in $\mathcal{R}$ are unsatisfied connections. Every switch, upon receiving an ADVERTISE control packet for a connection which is currently unrestricted at this switch, will compute a new stamped rate for this connection, under the assumption that this switch is a bottleneck for this connection.

Given the set $\mathcal{R}$, the "advertised rate" μ_l at link l is calculated by

$$\mu_l = \begin{cases} b'_{av,l} & \text{if} \quad N_l = 0; \\ b'_{av,l} - b'_{\mathcal{R}} + \max_{i \in \mathcal{R}} b'_{\mathcal{R},i} & \text{if} \quad N_l = N_{\mathcal{R}}; \\ \frac{b'_{av,l} - b'_{\mathcal{R}}}{N_l - N_{\mathcal{R}}} & \text{otherwise} \end{cases}$$

where $b'_{\mathcal{R}}$ is the total excess capacity consumed by all restricted connections, $\max_{i \in \mathcal{R}} b'_{\mathcal{R},i}$ is the maximum excess capacity consumed by a single restricted connection, N_l is the total number of connections, and $N_{\mathcal{R}}$ is the number of restricted connections.

It turns out that after this first calculation of μ_l, some connections that were previously "restricted" with respect to the old "advertised rate", can become "unrestricted" with respect to the new advertised rate. In this case, these connections are re-marked as unrestricted and the advertised rate is re-calculated once more. It can be shown that the second re-calculation is sufficient to ensure that any connection marked as restricted before the second re-calculation remains un-restricted with respect to the newly calculated advertised rate.

It should be noted that for a new connection, admission control is still a single round trip process: in the forward pass of admission test, the source will carry the requested bandwidth $[b_{\min}, b_{\max}]$ of the new connection. As the packet travels through the network, besides the test performed for $b_{\min}$, delay, jitter and buffer space, the stamped rate is also reset to the smallest of the connection's $b_{\max} - b_{\min}$ and the advertised rates of all links on the packet's forward route. Moreover, every switch also adds this connection to its connection list. The reverse pass is the standard reservation (relaxation) process (see Table 2).

Since the above algorithm essentially floods the network with ADVERTISE packets it may generate a lot of unnecessary traffic. We now present a refinement to the above algorithm which significantly reduces the number of overhead messages.

Every switch maintains a set $\mathcal{M}(l)$ (for its link l) consisting of all (unsatisfied) connections that consider link l as a "connection bottleneck link". In case of a switch detecting "new" available bandwidth for link l at time t, it initiates ADVERTISE packets *only* for those connections belonging to the set $\mathcal{M}(l)$. During the round trip traveled by the two ADVERTISE packets for a connection j, every switch along the route updates its $\mathcal{M}(k)$ set for link k upon receiving ADVERTISE packets: it adds j to $\mathcal{M}(k)$ if $\mu_k < b_{stamp}$, and removes j if $\mu_k > b_{stamp}$. The initiating switch also updates its $\mathcal{M}(l)$ but only after it completes the current adaptation process. In the case when a switch detects resource unavailability characterized by $(b'_{av,l}(t) < b'_{av,l}(t^-))$, it initiates ADVERTISE packets only for those connections that have higher record rate than the advertised rate. Upon receiving an ADVERTISE packet for a specific connection traversing a link l, besides the operations for this connection described above. Every switch (other than the switch that initiates the ADVERTISE packet) will perform the following operations for *other* connections traversing the same link: If the received stamped rate is smaller than its record rate for the connection, after recalculating its advertised rate (in this case, some connections can be up-gradated), the switch initiates ADVERTISE packets only for those connections within the set $\mathcal{M}(l)$. If the received stamped rate is larger than its record rate for the connection (another switch has detected resource increase), after recalculating its advertised rate, it initiates ADVERTISE packets only for those connections that have higher record rate than the advertised rate. In either case, the switch will perform a four-round trip adaptation process by repeating the process of sending out the ADVERTISE packet four times. This is needed to ensure convergence as described in the following theorems:

Theorem 1 Algorithm Convergence *Given a set of connections with stable routes and bandwidth requirements after a period of instability, for an arbitrary set of network links with changes in excess bandwidth resources and arbitrary initial conditions on the state of all links, sources, and destinations, and any number of control packets in transit, the event-driven adaptation algorithm described above will converge to the maxmin optimality criterion within a finite number of steps. Besides, after reaching steady state, for any ongoing connection, then the maximum optimal-rate difference between its current steady state and its previous steady state is bounded by the interval* $[0, \delta]$.

Proof: For brevity, we only outline the proof below:

Step 1: convergence of the simplistic event-driven algorithm can be proven based on arguments in [8] with minor modifications.

Step 2: initiation of ADVERTISE packets only along connection in the set $\mathcal{M}(l)$ will not affect the convergence property. The argument used here is that when bandwidth upgrading is performed, the initiation of ADVERTISE packets for connections other than those in $\mathcal{M}(l)$ is unnecessary since the link l is not a bottleneck link for those connections and fact that the success of upgrading bandwidth only depends on the bottleneck link.

Step 3: the initiation of ADVERTISE packets for connections that have smaller record rate than the advertised rate will not affect the convergence .

Step 4: the maximum optimal-rate difference is obtained by observing the adaptation condition given by eqn. (2).

6 Advanced Resource Reservation

Advanced resource reservation is based two factors: (a) prediction of the next cell of a mobile user, and (b) aggregate handoff activity of cells.

The algorithm for prediction of the next cell of a mobile user given its current and previous cells uses a simple three-level approach.

- The first-level prediction uses the *portable* profile as follows: knowing the *previous cell id*, together with the *current cell id*, the base station checks the *next-predicted-cell* field (see Table 1) in the portable profile. If it is not empty, then the prediction is successful.
- The second-level prediction uses the *cell* profile as follows: if there is a neighboring *office* cell of which the user is a regular occupant, then the office cell is nominated as the next cell. Otherwise, the prediction is made on aggregate history of handoffs in the cell.

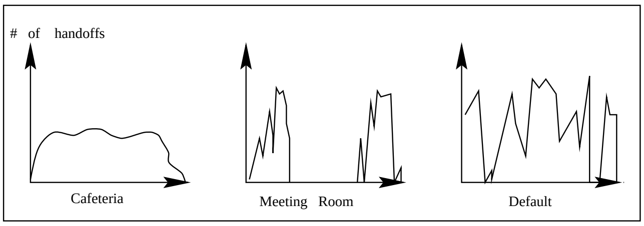

Figure 2: **Handoff Activity In a Lounge**

- In the event of there being no next-predicted cell from the first two levels, the default predictive algorithm will be used to conduct advance resource reservation (Section 6.3).

The rest of the section describes the classification of cells and prediction of aggregate cell behavior.

6.1 The Office and the Corridor Cases

An *office* is a cell with a small set of 'regular' occupants. An office cell makes advanced bandwidth reservations only when the mobile portable in its neighboring cells is a regular occupant of the office.

A *corridor* is a cell such that users typically move in the same direction across the cell, i.e. knowing the previous cell, the next cell can be predicted easily.

6.2 The Lounge Case

A *lounge* is a cell which has many 'non-regular' users. A lounge does not distinguish an individual user's behavior, but aggregates the behavior of all the users in its cell. Based on aggregate behavior, a lounge can be further classified into three categories: *meeting room*, characterized by bursts of handoffs at the start and conclusion of meetings; *cafeteria*, characterized by a slow time-varying profile; and *default*, characterized by a *random* time-varying profile.

6.2.1 The Meeting Room Case

In a meeting room, the majority of handoffs occur at and around the start and conclusion of meetings, with few handoffs in between. For the meeting room, the profile includes a *booking calendar* of meeting schedules. Each meeting specifies the start time T_s, stop time T_a, and the required resources N_m (currently, we specify N_m in terms of the number of users). The reservation policy for a meeting room is as follows:

(a) Starting from time $T_s - \Delta_s$ ($\Delta_s = 10$ minutes in our simulations), the base station in the meeting room will advance reserve resources for the total number of N_m attendees. The base station also maintains a counter $N_{arrived}(t)$ to record the number of attendees that have arrived by time t. At any time t, the base station advance reserves resources for $N_m - N_{arrived}(t)$ users. After time T_s, the base station starts a timer (5 minutes) and releases unused reserved resources for the meeting upon expire of the timer.

(b) Starting at time $T_a - \Delta_a$ ($\Delta_a = 5$ minutes in our simulations), the base station at the meeting room ask its neighboring cells to reserve bandwidth for the number of leaving attendees $N_{arrived}(T_a - \Delta_a)$ according to its cell profile. The base station maintains a counter $N_{left}(t)$ to record the number of attendees that have left by time t, and notifies its neighbors to reserve resources according to the number $N_m - N_{left}(t)$ at time t. At T_a, the base station starts a timer (15 minutes), and asks its neighbors to release reserved resources upon expire of the timer.

6.2.2 The Cafeteria Case

If the current cell is a cafeteria, then the reservation policy predicts the number of handoffs $N_{handoff}(t+1)$ at next time instant $t+1$, and asks its neighboring cells to advance reserve bandwidth for $N_{handoff}(t+1)$ handoffs according to its cell profile. If at least one of its neighboring cells is a *default* cell, it will also predict the number of arriving portables (due to handoffs from its neighboring cells) $N_{arv}(t+1)$ at time $t+1$. Then it updates its current total reserved bandwidth to accommodate $N_{arv}(t+1)$ arriving portables at time $t+1$. The reason for doing this is that since the current cell has a *default* neighbor which provides poor quality of next-cell prediction, it should not totally "trust" that default cell, therefore, the current cell will also predict by itself the arriving calls (due to handoffs from its neighbors) at the next time instant.

The algorithm for prediction of the number of handoffs $N_{handoff}(t+1)$ at the next time instant $t+1$ is based on a linear model due to the slow time-varying nature of a cafeteria. Denote the linear model as $n = a \cdot t + m$, using the handoff data n_{t-2}, n_{t-1}, n_t during the last 3 time slots and applying the standard Least-square technique, a, m can be easily calculated by

$$a = \frac{n_t - n_{t-2}}{2}, \qquad m = \frac{(5+3t)n_{t-2} + 2n_{t-1} - (3t+1)n_t}{6}.$$

Then, the predicted number of handoffs from the cell at next time slot $t+1$ would be given by $N_{handoff}(t+1) = a \cdot (t+1) + m$.

The prediction for the number of arriving portables $N_{arv}(t+1)$ at time $t+1$ follows a similar procedure.

It should be noted that by knowing the total number of handoffs $N_{handoff}(t+1)$, the number of handoffs to a specific neighboring cell can be computed based on the aggregate history for the cell (which is provided by the profile server).

6.2.3 The Default Case

A lounge cell which does not conform to either the meeting room or cafeteria cases is labeled the *default* case. For the default cell, we adopt a one-step-memory policy for the prediction of the number of handoffs, denoted by $N_{handoff}(t+1)$, at next time-instant. That is, the number of handoffs at time $t+1$ is simply the number of handoffs at current time, denoted by $N_{handoff}(t)$, that is, $N_{handoff}(t+1) = N_{handoff}(t)$.

In the case when at least one of its neighboring cells is also a *default*, since a default neighbor provides poor quality of next-cell prediction, the current default should not totally "trust" its default neighbor. Therefore, the current cell will also try to predict the total amount of bandwidth to be reserved in it at the next time instant, to accommodate handoff portables from its default neighbor and others. To achieve this goal, we present a default prediction algorithm based on probabilistic arguments in next subsection.

6.3 Default Advance Reservation Algorithm

We now present a probabilistic reservation algorithm, which governs the reservation policy for the default cell.

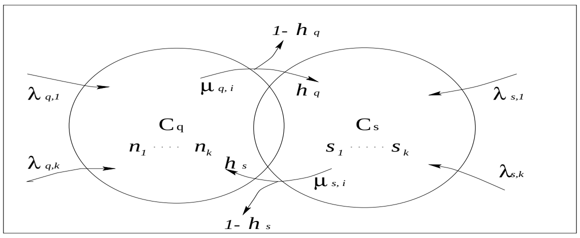

Figure 3: **Model for Reservation Analysis**

Assuming that the system information (the number of current ongoing connections in each cell, the arrival rate of a new connection request in a cell, the average duration of a connection in a cell, and the hand-off probability) is available, our objective is to keep the hand-off dropping probability below some pre-specified design threshold. As in [5], the idea here is to look ahead at a time window given by $[t, t+T]$, where t is the current time and T is the window size. During this time window, we keep the fraction of dropped handoff connections below some pre-specified level P_{QOS}.

Figure 3 shows the model for the algorithm. We consider two neighboring cells C_s and C_q[12]. We consider k connection types, labeled $i = 1, 2, \ldots, k$, each having different bandwidth requirements $[b_{\min,i}, b_{\max,i}]$. Moreover, we assume that the new connection request (of type i) arrival to cells C_q, C_s are with rates $\lambda_{q,i}, \lambda_{s,i}$, respectively; the mean connection duration (of connection type i) $1/\mu_{q,i}, 1/\mu_{s,i}$ in cells C_q and C_s is exponentially distributed. When a mobile leaves cell C_q, it is handing off to cell C_s with probability h_q, and it leaves the system entirely (i.e. terminating) with probability $1 - h_q$.

Consider any connection (of type i) in cell C_q, we denote the probability that a connection remains in the same cell C_q as $p_{s,i}$, and the probability that it hands-off to cell C_q during time T as $p_{m,i}$. Furthermore, we assume that the probability that a mobile hands-off more than once during time T is negligible. In addition, we ignore the hand-off effects due to connections newly admitted into cell C_s during $[t, t+T]$. This implies that whenever there is a space conflict between a handoff connections and an already existing connection in a cell, the connection with a later arrival time is dropped. Thus, if we interpret the handoff blocking probability as the fraction of connections that are interrupted while in progress, then we can ignore future arrivals of connection requests. Note that this model is slightly different from the one in [5] in the way the handoff probability is modeled. Further, we allow for multiple connection types.

Based on the above exponential distribution assumption, $p_{s,i}$ and p_m can be calculated as follows:

$$p_{s,i} = e^{-\mu_i T}, \qquad p_{m,i} = (1 - e^{-\mu_i T}) h_q \; .$$

In our model we consider a homogeneous system, where each cell has maximum bandwidth B_c, and denote P_{QOS} as the lowest tolerable non-blocking probability in a wireless system where all connections of different types require the same P_{QOS} throughout their connection. Therefore, the design objective is to reserve a minimum quantity of bandwidth, such that the desired quality of service (in terms of P_{QOS}) of the *existing* connections in cell C_q at (future) time $t + T$ is maintained.

[12] Conceptually, we can extend this model for the multiple neighbor scenario. However, for simplicity of notation, we only present the two cell scenario.

We assume that there are n_i connections (of type-i) in cell C_q at current time t and denote the *maximum* allowed number of (type-i) connections in cell C_q at t (by admitting some new connection requests) as N_i, the probability that j_i connections (out of N_i connections at time t) are in the same cell C_q at time $t + T$ has a binomial distribution, denoted $B(j_i; N_i, p_{s,i})$, which is defined as

$$B(j_i; N_i, p_{s,i}) = \binom{N_i}{j_i} p_{s,i}^{j_i} (1 - p_{s,i})^{N_i - j_i} \; . \tag{3}$$

Let us assume that there are s_i (type-i) connections in cell C_s at time t. Then the probability that l_i connections of type-i (out of s_i connections at time t) in cell C_s handoff to cell C_q by time $t + T$ is as follows:

$$B(l_i; s_i, p_m) = \binom{s_i}{l_i} p_m^{l_i} (1 - p_m)^{s_i - l_i} \; . \tag{4}$$

Therefore, the total nonblocking probability of existing connections at $t + 1$ is obtained by

$$P_{nb} = Prob\Big(\sum_{i=1}^{k} b_{\min,i}(l_i + j_i) \leq B_c\Big), \qquad l_i, j_i \geq 0 \; . \tag{5}$$

where we will provide only $b_{\min,i}$ to existing connections (via conflict resolution) to accommodate new connections in case of resource conflict. Then the following should be satisfied

$$P_{nb} \geq 1 - P_{QOS} \tag{6}$$

Then, the total reserved bandwidth for cell C_q is given by

$$b_{resv.q} \geq B_c - \sum_{i=1}^{k} b_{\min,i} N_i \; . \tag{7}$$

6.4 Summary of Profile-based Predictive Advance Reservation

We now summarize the advance reservation algorithms presented in Sections 6.1 - 6.3.

- 1. *next-predicted-cell-ID*(portable profile, current state) $\neq$ *empty*:

  $\Rightarrow$ Resv(next-predicted-cell)
- 2. Otherwise, predict based on cell profile:
  - If *type* (current cell) = *office* :
    - 1. neighboring-cell=*office* & mobile portable = regular occupant of neighboring office $\Rightarrow$ Resv (neighboring office)
    - 2. mobile portable=regular occupant of current office $\Rightarrow$ No_Resv(neighboring cells)
    - 3. otherwise $\Rightarrow$ predict based on aggregate history
  - If *type* (current cell) = *corridor* :
    - 1. *type* (neighboring cell) = *office* & mobile portable = neighboring office regular occupant $\Rightarrow$ Resv (neighboring office)
    - 2. predict based on aggregate history
  - If *type* (current cell) = *meeting room* :
    - 1. at the *start* of meeting $\Rightarrow$ Resv(current cell) to accommodate $N_m - N_{arrived}(t)$.

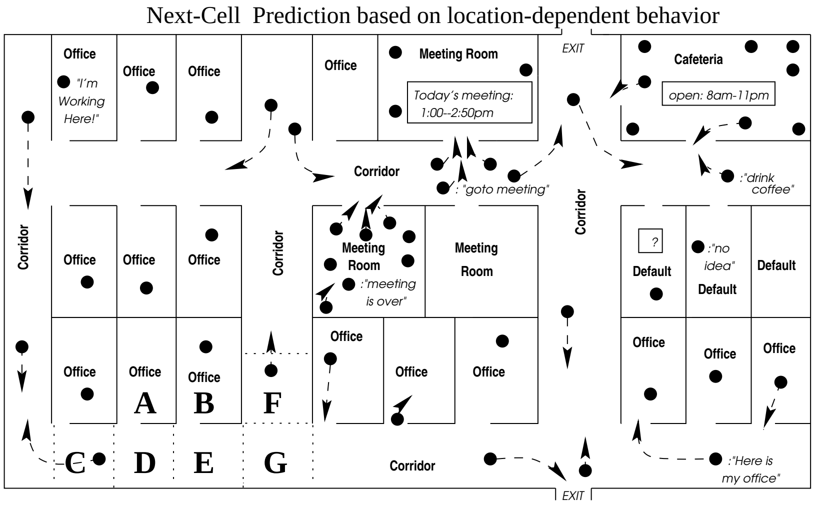

Figure 4: **An In-door Environment**

- ∗ 2. at the *conclusion* of meeting ⇒ Resv (neighbors) to accommodate $N_m - N_{left}(t)$;
- If *type* (current cell)=*cafeteria*:
  - ∗ 1. Resv(neighbors) to accommodate $N_{handoff}(t+1)$ portables
  - ∗ 2. If at least one neighbor is *default*: ⇒ Resv(current cell) to accommodate $N_{arrv}(t+1)$ portables
- If *type* (current cell)=*default*:
  - ∗ 1. Resv(neighbors) to accommodate $N_{handoff}(t+1)$ portables;
  - ∗ 2. If at least one neighbor is *default*: ⇒ Apply the probabilistic reservation algorithm, i.e. Resv(current cell) with reserved bandwidth no less than that specified by ($eqn.7$).

In the case that a cell does not have its cell profile, the base station has to execute the default reservation algorithm initially; meanwhile, the base station will try to set up a cell type based on the following learning process: the profile server aggregates the handoff information for the cell, executes the different categories of prediction algorithms and tries to categorize the cell on basis of its profile behavior.

7 Simulation Results

In this section, we provide preliminary simulation results for advance resource reservation using profile-based prediction and the default resource reservation algorithm. Our results in Section 7.1 partially validate the approach of cell classification in the indoor mobile computing environments where we perform the simulations. In Section 7.2, we show how to choose some design parameters for the default algorithm via a simulation example.

7.1 Simulations for the Prediction Based on Profiles

We performed measurements to track user mobility and validate the cell classification scheme. In particular, we sought to validate the mobility model for the office and meeting room (lounge) cases, since we propose deterministic advance resource reservation schemes in these cases. These measurements were made in the ECE Department of the University of Illinois over the course of the Spring 1996 semester. Note, that in these measurements, we could only measure the mobility of users across artificially demarcated 'cells', but not actual user workload, since we do not yet have a large-scale indoor mobile computing environment in place. The purpose of these simulations is thus more as a means to validate the cell classification scheme than a performance measurement of the advance reservation algorithm.

For the office case, we tracked the user mobility of the occupants of two adjacent offices (one faculty office with one 'regular' occupant, and one student office with four 'regular' occupants - three students and the faculty member). In Figure 4, the faculty office is labeled **A** while the student office is labeled **B**. Adjacent corridor cells are marked **C** through **G**. For a total of 127 handoffs for the faculty member from cell **C** to cell **D** over one workweek, we observed 94 handoffs into cell **A** (from **D**), 20 handoffs into cell **B** (**D** to **E** to **B**), and 13 handoffs to either **F** or **G**. For a total of 218 handoffs for the three students from cell **C** to cell **D** over the workweek, we observed 12 handoffs into cell **A**, 173 handoffs into cell **B**, and **31** handoffs into either **F** or **G**. During the same workweek, a total of 1384 handoffs were measured from cell **C** to cell **D**. 39 handoffs occurred into cell **A** from other users, and 17 handoffs occurred into cell **B** from other users. While such handoff patterns may not be representative in other office environments or other users, they do indicate two points: (a) deterministic reservation for only the occupants of an office cell is valid, and (b) brute force advance reservation in all neighboring cells of a current cell is extremely wasteful.

For the meeting room case, we measured user mobility into and out of classrooms for various sizes of classes, and times of the semester. The class sizes varied from 14 to 125, and the location of the classes varied from corner classes to large auditoriums. As expected, the handoffs into the classes were mostly aggregated in a 10 minute period around the start of the class, while the handoffs out of the classes were mostly aggregated in a 5 minute period after the class.

We simulated the following three advanced reservation algorithms for the measured handoffs shown in Figure 5: (a) brute force reservation in the neighborhood of a user, (b)

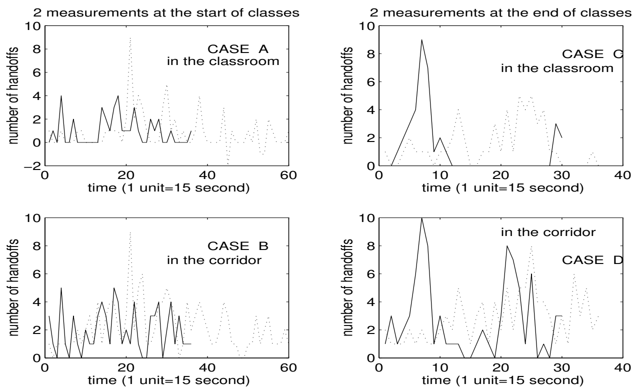

Figure 5: Measured data from two classrooms
Total number of handoffs (from the *start* to the *end* of the class) in the corridor:
Measurement 1: 204; Measurement 2: 312

advance reservation based on aggregation of previous handoffs from a cell to its neighbors, and (c) the meeting room algorithm described in Section 6. In Figure 5, the solid lines show the handoffs corresponding to a lecture class of 35 students, while the dotted lines show the handoffs corresponding to a laboratory class of 55 students. Figure 5.a plots the handoffs into the classroom at the start of the class, while Figure 5.b plots the total number of handoffs occurring just outside the class at the same time (a fraction of the students who walk by the class actually enter into the class). Figure 5.c plots the handoffs out of the classroom at the end of the class while Figure 5.d plots the total handoff activity at the same time. Since we could not make real measurements of a user workload, we simulated using the following parameters: cell throughput 1.6Mbps, each user opens one connection of either 16Kbps (75%) or 64Kbps (25%).

For the 35 student class, the offered load was 59%. The brute force reservation algorithm registered 2 connection drops, while the other two algorithms did not drop any calls. For the 55 student class, the offered load was 94%. The brute force reservation algorithm registered 7 connection drops, the aggregation algorithm registered 4 connection drops, while the meeting room algorithm did not drop any connection. The reason for connections being dropped in the former two cases is that as load increases, reservations for users who walk along the corridor but do not enter the classroom causes wasteful reservations of scarce resources. In the meeting room algorithm, resources are advance reserved for the expected number of occupants, thereby eliminating any wasteful reservation.

7.2 Performance the Default Reservation Algorithm

In this subsection, practical design issues of the advance resource reservation algorithm provided in Section 6 are discussed via a simulation example. Two design parameters are of significance in the algorithm: the time window T and the target dropping probability P_{QOS}. The right choice of these two parameters will balance the tradeoff between the handoff dropping probability (P_d) and the overall blocking probability (P_b). Ideally, a good choice of P_{QOS} will keep the overall blocking probability as small as possible while the handoff dropping probability is no larger than P_{QOS}, and a good choice of T will make the prediction accurate.

We show how the design is performed using the following example. We consider a situation where there are two types of connections going on in the two cells that have identical characteristics. The capacity of each cell is 40. The bandwidth requirement for connection type 1 is 1, and the arrival rate is 30, mean holding time is 0.2 and the handoff probability is 0.7; the bandwidth requirement for connection type 2 is 4, and the arrival rate is 1, mean holding time is 0.25 and the handoff probability is 0.7. The performance of the default advance resource reservation algorithm is shown in Figure 6.

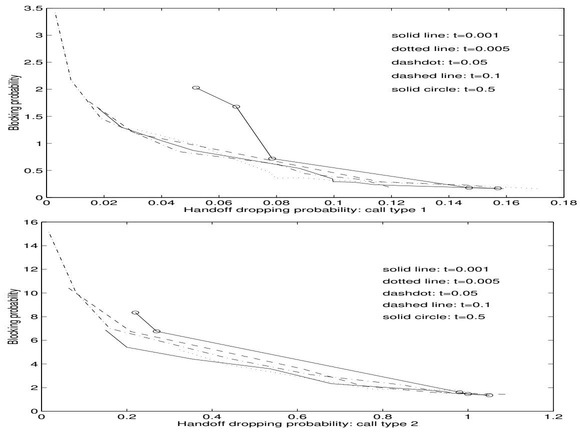

Figure 6: Performance of the default algorithm

A family of curves have been obtained for different values of the time window T. Each curve is a plot of P_d versus P_b. As is to be expected, P_b decreases with increasing P_d. All curves lie on top of each other for large P_d. (Note that, large P_d denotes that all connections are admitted irrespective of whether the handoff connections will be dropped or not, i.e., a new connection is admitted if there is sufficient bandwidth independent of its effect on handoff dropping.) Thus, a value at t_1 is better than another value at t_2 if the curve for t_1 lies below the curve for t_2. For this example, it seems beneficial to choose t small. However, there is very little difference for $t < 0.05$. Once a particular value of t is chosen, the network designer must choose an operating point. Then the value of P_{QOS} should be available once an operating point is chosen. Our simulation results (not shown here due to space limitations) show that our reservation algorithm outperforms the static reservation algorithm in all scenarios we have simulated[12].

8 Conclusion

As advanced communication-intensive applications become available in mobile computing environments, the necessity to provide QoS to applications and to efficiently manage wireless networking resources will become more pronounced. It is clear that the wireless network characteristics and user mobility will motivate a modification of the traditional notion of QoS, in order to accommodate flexible, dynamically re-negotiable bounds. This work attempts to provide algorithms for resource management with such bounds in mind.

Four factors motivate the work in this paper: (a) user mobility and wireless channel error motivate the use of loose QoS bounds, (b) the goal of providing seamless mobility and QoS guarantees motivates the use of advance resource reservation, (c) the location dependent behavior of users and cells motivates cell classification and location-dependent reservation algorithms, and (d) the use of QoS bounds and adaptation to dynamic network conditions introduces the problem of resource conflict.

The focus of this paper has been on proposing the algorithms for adaptive resource management, and predictive advance reservation. In the former case, we adapt previous work in [8], while in the latter case, we propose an intuitively obvious classification of cells and simple algorithms based on this classification. Our preliminary simulations have partially validate our approach, though the lack of a large-scale indoor mobile computing testbed, which is a part of ongoing work, precluded insightful performance measurements of our algorithms in a real-world scenario.

Acknowledgments

The authors would like to thank D. Dwyer, V. Gupta, R. Srikant and anonymous reviewers for constructive suggestions to improve the paper.

References

[1] D. A. Levine, I. F. Akyildiz and M. Naghshineh, "The Shadow Cluster Concept for Resource Allocation and Call Admission in ATM-based Wireless Networks," *Proc. MOBICOM'95*, Berkeley, California, November 1995.

[2] L. Zhang, S. Deering, D. Estrin, S. Shenker and D. Zappala, "RSVP: A New Resource ReServation Protocol," *IEEE Network Magazine*, September 1993.

[3] C. Parris, H. Zhang, and D. Ferrari, "Dynamic Management of Guaranteed Performance Multimedia Connections", *Multimedia Systems Journal*, vol. 1, 1994.

[4] S. Floyd and V. Jacobson, "Link-sharing and resource management models for packet networks," *IEEE/ACM Trans. on Networking*, Vol. 3, No. 4, August 1995.

[5] M. Naghshineh and M. Schwartz, "Distributed Call Admission Control in Mobile/Wireless Networks," *IBM Research Report*, RC 20036, February 1995.

[6] S. Seshan, H. Balakrishnan and R. H. Katz, "Handoffs in Cellular Wireless Networks: The Daedalus Implementation and Experience," *homepage: http://daedalus.cs.Berkeley.edu/*.

[7] K. Lee, "Adaptive Network Support for Multimedia," *Proc. MOBICOM'95*, Berkeley, California, November 1995.

[8] A. Charny, D. Clark and R. Jain, "Congestion Control with Explicit Rate Indication," *Proc. ICC'95*, June 1995.

[9] D. D. Clark, S. Shenker and L. Zhang, "Supporting real-time applications in an integrated services packet network: architecture and mechanism," *Proc. SIGCOMM'92*, August 1992.

[10] R. Schafer and T. Sikora, "Digital video coding standards and their role in video communications," *Proc. of the IEEE*, June 1995.

[11] B. Belzer, J. Liao, J. D. Villasenor, "Adaptive Video Coding for Mobile Wireless Networks," *Proc. IEEE ICIP-94*, Austin, Texas, 1994.

[12] S. Lu, V. Bharghavan and R. Srikant, "Adaptive resource reservation for indoor wireless LANs," Coordinated Science Laboratory, Univ. of Illinois at Urbana-Champaign, May 1996.

[13] H. Zhang, "Service disciplines for guaranteed performance service in packet-switching networks," *Proc. of IEEE*, October 1995.

[14] V. Gupta, and V. Bharghavan, "A methodology for adaptive computing," *Technical Report*, Coordinated Science Laboratory, Univ. of Illinois at Urbana-Champaign, May 1996.