Multicast Scheduling Algorithms in Mobile Networks

Márton Nagy and Suresh Singh

Department of Computer Science
University of South Carolina
Columbia, SC 29208
e-mail: {nagy,singh}@cs.sc.edu
phone: (803) 777-4322, 777-2596

July 15, 1997

Contents

Abstract

We examine ways in which data can be efficiently *multicasted* to mobile users in a *cellular network* infrastructure.

Using a theoretical approach, a mathematical description of the model is given and a reasonable measure of efficiency is defined. Then an upper bound is obtained on the worst case efficiency of any algorithm, and an example is presented with a constant efficiency that achieves that bound.

As part of a more practical approach we present an algorithm and examine its performance in view of the theoretical bounds, then briefly look at how some of these algorithms can be applied in a more general scenario.

Keywords: Wireless Networks, Multicast Algorithms

1 Introduction

Mobile Computing refers to a new paradigm in computing where users, equipped with cheap portable computers (such as Personal Digital Assistants), will be able to access remote data and services while on the move. Thus, a user driving in a car will be able to browse email, order groceries from a Internet store or place videophone calls while on the road! How can we provide continuous connectivity for users who are mobile?

A *cellular network* infrastructure is typically used to connect mobile users to the Internet. A geographical region, such as a highway or campus, is divided into *cells* each of which contains a *base station* that provides a connection end-point for roaming mobiles. The base stations are connected to the wired infrastructure to provide access to the Internet. A cell may be as small as tens of meters in diameter (in the case of in-building picocellular networks [8]) or as large as a mile in diameter (macrocellular network). In order to maintain connections for mobile users, it is necessary to keep track of their *location*, ensure that the network can *route* packets to the user's current location and, in addition, use new data transfer protocols that are optimized for the mobile environment. Several authors have developed solutions to some of these problems. Thus, [7] presents a solution for location management, [4, 6, 9] discuss the problem of routing packets in a mobile environment and [2, 10, 5] present transport layer protocols optimized for the mobile environment.

We believe that a large number of services requested by mobile users will be multicasted to them (for efficiency reasons) from the various service-providers located around the world. Some examples of such services include traffic updates transmitted by a police helicopter in a busy downtown area, TV programming, telephone (videophone) conference calls between coworkers, etc. Thus, in this paper we study the problem of optimally scheduling multicasts in mobile cellular networks. It is important to note, however, that our focus is on efficient bandwidth management (link layer issues) rather than on higher layer protocol support for multicasting. That problem has been addressed by other authors in [3].

1.1 Cellular Network Model

We assume that a region is divided into *cells*, each of which is serviced by a base station or Mobile Support Station. This MSS provides a connection end-point for the roaming mobile. The MSSs are all connected to the Internet. Thus, a user in a cell receives data via the base station in the current cell. As the user roams, the data is forwarded to the new base station and gets transmitted to the user in that new cell. These terms are illustrated in Figure 1.

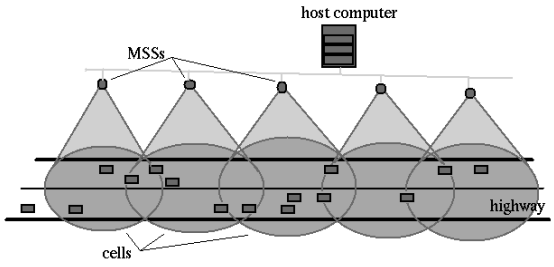

Figure 1: A portion of a highway covered by cells.

Wireless networks are subjected to the twin constraints of high bit error (of the order of 10^{-5}) and low available bandwidth (10-100 Kbps). While retransmissions and FEC (Forward Error Correcting codes) can be used to reduce the impact of high bit error, these techniques further reduce the available bandwidth. One method that may be used to increase the bandwidth available to each user is to reduce the size of the cells thereby reducing the

number of users per cell. This method works well but suffers from one drawback – as a user roams, she will move from one cell to another frequently requiring data to be forwarded from one base station to another and possibly triggering retransmissions of the same data (a user may have moved while receiving a transmission in the old cell but the old base station does not know that the user has moved away and will, thus, continue transmitting). Thus, we have a need to develop data transmission protocols that maximize the throughput over the wireless links.

This paper examines ways that increase the effective bandwidth by cleverly selecting what to transmit and when, in the case when lots of mobiles request the same information.

1.2 Description of the Problem

There is not much that can be done to increase the effective bandwidth in the case when mobiles request different data. The well-known approaches are frequency or time division multiplexing. The first approach transmits to all computers at the same time using a different frequency range, but the bandwidth of these connections is just a fraction of the bandwidth of the wireless link. The second approach transmits at the maximum bandwidth, but only to one computer and for a shorter period of time, then it switches over to another computer, cycling through all of them in turn. Both of these methods have cumulative bandwidth equal to the bandwidth of the wireless link.

As opposed to the behavior described in the previous paragraph, it may happen that different computers want to receive the same information. The effective throughput can be greatly increased in these cases by *multicasting* the data to all the intended receivers. Unfortunately, the problem of scheduling multicasts is complicated due to the fact that the receivers may roam from one cell to another. Thus, it is possible that a receiver may arrive at a new cell and notice that the multicast has proceeded up to byte b while it had only received bytes up to $b' < b$ in its previous cell. Similarly, a receiver may enter a cell where there are no other multicast group members. In this case, the new base station will need to join the multicast group (to receive data) and then begin transmitting data to the receiver. In all these cases, we see that individual retransmissions are necessary to recover the data lost during the periods while a mobile crosses through cell boundaries. Since the number of mobiles and hence the time spent on retransmissions vs. trans-

missions varies in each cell, the number of bytes the mobiles received up to a certain time is different in each cell; consequently the starting byte number of the multicast and its duration must be cleverly selected in each cell.

This selection process is the main topic of the paper [1]. The authors measure the effectiveness of a multicasting algorithm in terms of the cumulative bandwidth of the mobile computers in a cell. They show what is the most efficient way to multicast and give a linear algorithm that determines how the transmission should proceed at each step, but only in the case when the mobile network consists of a single cell. In this paper, we present algorithms that maximize the efficiency of the multicast for the multicell case where mobiles are allowed to roam from one cell to another.

2 The Theoretical Approach

2.1 The Model

Throughout the paper we will focus our attention on the case that seems the simplest to understand, but is still complex enough so that the answers uncovered can be applied to other situations. Consider a cellular network providing wireless connection to a portion of a highway. Two of the most significant factors that help simplify the analysis of this problem are the following. On a highway the movements of the mobiles are much more uniform and predictable than in the case where the mobiles roam within buildings or in downtown areas. In addition, while in the latter examples certain two-dimensional areas are covered by cells, the case of the highway is "linear": every cell has exactly two neighbors, one to the left and one to the right.

The characteristics of such a model can be described as follows.

- cells are assumed to have identical parameters – same diameter and the same bandwidth
- every mobile has a direction (goes either to the left or to the right, and never turns around)
- the cell crossing times of the mobiles has small variance
- the only fade-periods are those introduced by the hand-offs as the mobile moves between consecutive cells

- the hand-off time between cells is assumed to have small variance and is bounded by some maximum value

- the cell crossing times of the mobiles are assumed to be at least twice as large as the maximum duration of the hand-off times

It is clear that the complications are caused by the fade-periods. If the hand-off time was zero, then the bandwidth of such a network would be equal to the raw bandwidth of the mobile support stations. Since this is not the case, the data transmitted while a mobile is in fade must be retransmitted to it later. Such a lost data portion is referred to as a **hole** in the stream of the data the mobile has received.

Algorithms that do not place any restrictions on when the cells should retransmit these holes raise further problems. They are obviously not suitable for real time connections where data should be delivered as soon as possible and preferably in-order. Without knowing when a retransmission will finally occur mobiles must be able to deal with more than just a single hole: if a request for a retransmission of a hole is not satisfied while crossing through a cell, the next hand-off may introduce new ones. In this scenario mobile support stations need to have big buffers to be able to deal with retransmission requests referring to very old bytes. Finally the overhead in control messages increases since a mobile's retransmission request may be sent to a number of consecutive cells till the last one finally takes care of it.

Because of these negative effects we restrict our attention to algorithms that honor retransmission requests of the mobiles before they leave the cell. Furthermore, the algorithms we consider should never introduce new holes by transmitting data segments further ahead, out-of-order. These algorithms will actually ensure the following:

- Mobiles leaving a cell have received a continuous stream of data up to that point without any holes.

- Holes are only introduced when a mobile enters a cell whose transmission byte-count is higher than the byte-count the mobile was expecting next.

- At any time a cell either transmits new data in-order, or suspending this, it retransmits a portion of the data transmitted earlier, then resumes with transmission of the new data.

Assume that the cells follow an algorithm that takes as input the arrival times of the mobiles and their byte-count, then specifies what to transmit.

Fix the duration of the experiment to start at time 0 and conclude at time T, but think about T as a variable, rather than a constant. To be more precise, the algorithm is considered to run forever, but our experiment only looks at a finite portion of it. Doing this will eliminate the problems that arise while examining the algorithm close to the starting and ending times, and letting $T \to \infty$ will give us a good estimate for the long-term behavior.

Let the highway be covered by c cells (indexed from 1 to c). Let w stand for the bandwidth of the MSSs covering the highway. Consider "slotted time" and select the time unit in such a way, so that we may assume a continuous in-order transmission of bytes during any time segment $[t, t+1)$, for any integer $t < T$. Let m denote the number of the mobiles that are present on the highway during some part of $[0, T]$.

Definition 2.1 *Let $b_k(t)$ denote the starting byte-count of the data segment transmitted by cell k during the time interval $[t, t+1)$ for $t < T$ and $1 \leq k \leq c$. Let $B_k(t)$ denote the next highest in-order byte-count that cell k could transmit at time t.*

Here are some obvious facts: The inequality $b_k(t) \leq B_k(t)$ always holds. All the $B_k(t)$ functions are non-decreasing. $B_k(t+1)$ equals either $B_k(t) + w$ or $B_k(t)$ for every $t < T$, depending on whether new data was transmitted or a retransmission took place. Furthermore $b_k(t) = B_k(t)$ also means that new data is transmitted by cell k during $[t, t+1)$, whereas $b_k(t) < B_k(t)$ refers to a retransmission.

The functions $b_k(t)$ and $B_k(t)$ may depend on the parameters of the mobiles (their number, their arrival and departure times to/from each cell, etc...) The efficiency of the algorithms will be measured as follows.

Definition 2.2 *Let the* **effective bandwidth of a cell** *denote the limit inferior of the number of* new *bytes transmitted over the total time of the experiment, i.e. the effective bandwidth of cell k equals*

$$\liminf_{T \to \infty} \frac{B_k(T) - B_k(0)}{T - 0}$$

The **effective bandwidth of an algorithm** *is defined to be the minimum of the effective bandwidths of all the cells. The* **efficiency of the algorithm** *is the ratio of the effective bandwidth of an algorithm and the raw bandwidth of the MSSs.*

Letting T grow without bound and taking the limit is the way to define the notion of the effective bandwidth if we want a precise mathematical formulation, but still be able to disregard minor variations in the algorithm that are localized close to the starting and the ending times. The reason that we use the limit inferior instead of the limit is that the latter may not be defined. Let us recall an important property of the limit inferior that will be used in the arguments of the next section. If the effective bandwidth of cell k equals to a constant C then for every positive ϵ there must be a time T_0 so that

$$\frac{B_k(T) - B_k(0)}{T - 0} > C - \epsilon \qquad \text{for all } T > T_0$$

Notice that the effective bandwidth is always bounded above by the raw bandwidth of the MSSs, hence the efficiency is a number between 0 and 1.

In the next two sections we examine the worst case scenario. It will be shown that no algorithm can do better than a 2/3 efficiency for certain mobile parameters. On the other hand we present an algorithm that can in fact achieve the 2/3 bound.

2.2 Upper Bound on the Worst Case Efficiency

This section takes the theoretical approach to show that there are cases when no algorithms can improve too much on the efficiency.

Theorem 2.1 *Given any algorithm it is possible to encounter such mobile parameters that reduce the efficiency of the algorithm to* 2/3 *or less.*

Proof: Let an algorithm be given that tells each MSS what to transmit in each time slot. Assume that the efficiency of the algorithm is greater than or equal to 2/3, i.e. the effective bandwidth of any cell is greater than or equal to $2/3 \cdot w$ (where w denotes the bandwidth of the MSSs). Our goal is to show that there is a cell whose effective bandwidth is actually equal to $2/3 \cdot w$, and hence the overall efficiency must be equal to 2/3.

Fix a small $\epsilon > 0$. Select a time T_0 (by the remark following Definition 2.2) that satisfies

$$\frac{B_k(T) - B_k(0)}{T - 0} > 2/3 \cdot w - \epsilon \qquad \text{for all } T > T_0,$$

and for every cell index k, then fix any $T > T_0$.

Consider the following parameters for the mobiles: Suppose that all of them have the same speed, but half of them move to the right and the other half to the left. Furthermore cars traveling to the same direction form a long line with the *consecutive cars being separated by the same distance which is equal to the product of the hand-off delay and the speed.* These mobile parameters ensure that for every time $t < T$ and every cell k there is a mobile moving to the right, that will arrive to cell k exactly at t (and there is a mobile moving to the left satisfying the same condition).

Let $b_k(t)$ and $B_k(t)$ be defined as in Definition 2.1. Let d denote the hand-off delay. Consider two neighboring cells k and k'. Let us partition the interval $[0, T]$ as $t_0 = 0, t_1, t_2, \ldots, t_n = T$ in such a way so that $B_k(t_i) = B_{k'}(t_i)$ for all $i \leq n$ and inside every subinterval $[t_{i-1}, t_i]$ either $B_k(t) \leq B_{k'}(t)$ holds for all t or $B_k(t) \geq B_{k'}(t)$ holds for all t. Now we can group these subintervals according to which one of the functions $B_k(t)$ and $B_{k'}(t)$ dominates the other. Let

$$\begin{aligned} I &= \{\, i : 1 \leq i \leq n \text{ so that } B_k(t) \geq B_{k'}(t) \text{ for all } t \in [t_{i-1}, t_i] \,\}, \quad \text{and} \\ I' &= \{\, i : 1 \leq i \leq n \text{ so that } B_k(t) \leq B_{k'}(t) \text{ for all } t \in [t_{i-1}, t_i] \,\} - I. \end{aligned}$$

The union of the disjoint sets I and I' will now include all indices $1, \ldots, n$. To illustrate the process of partitioning the time interval and grouping the subintervals consider the situation described on Figure 2. One possible way to partition the time segment $[0, 14]$ is $0, 8, 12, 14$. Then if the continuous line denotes B_k and the dotted one $B_{k'}$, then $I = \{1, 3\}$ and $I' = \{2\}$. (Notice that partitioning in the required way may not be unique: In our example another possibility would be for instance $0, 2, 7, 12, 14$.)

Let us fix an $i \in I$ and a $t \in [t_{i-1}, t_i]$ so that during $[t, t+1)$ transmission of new bytes takes place in cell k (i.e. $B_k(t) < B_k(t+1)$). Consider the mobile moving from cell k' to k, that leaves cell k' exactly at time t and hence arrives at cell k' at $t+d$. When mobile j leaves cell k' it has received a continuous transmission with no holes and it expects to receive bytes starting from $B_{k'}(t)$. But instead it goes to fade during a hand-off (and thus does not receive any bytes for time d). By the time it comes out of fade (at $t+d$), cell k already passed that byte-count as we can see it from the following sequence of inequalities:

$$B_{k'}(t) \leq B_k(t) < B_k(t+1) \leq B_k(t+d)$$

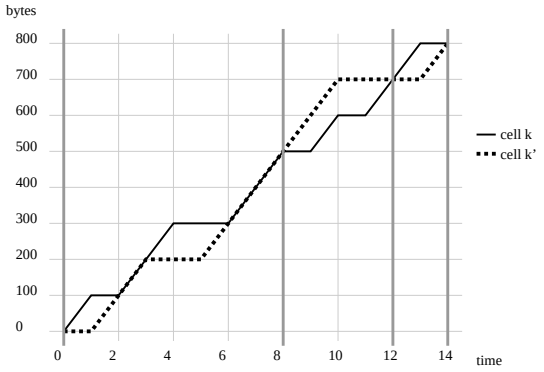

Figure 2: An example of the functions $B_k(t)$ and $B_{k'}(t)$

Consequently the byte segment $[B_k(t), B_k(t+1))$ must be retransmitted sometime while the mobile is in cell k. *This statement holds for any t at which transmission of new bytes starts, hence any transmission of new byte segments induces a retransmission of the exact same bytes.* The total number of these bytes is $B_k(t_i) - B_k(t_{i-1})$ during $[t_{i-1}, t_i]$. As i was selected arbitrarily from the set I, cell k must retransmit at least

$$\sum_{i \in I} B_k(t_i) - B_k(t_{i-1})$$

bytes during $[0, T]$. Recall that T was selected so that the inequality $B_k(T) - B_k(0) > (2/3 \cdot w - \epsilon) \cdot T$ is valid, i.e. cell k must transmit more than $(2/3 \cdot w - \epsilon) \cdot T$ new bytes during $[0, T]$. Since w is the raw bandwidth of the MSS, the total number of bytes that can be transmitted during that time is $w \cdot T$. Consequently the number of bytes that can be used for retransmission is less than

$$w \cdot T - (2/3 \cdot w - \epsilon) \cdot T = (1/3 \cdot w + \epsilon) \cdot T.$$

Comparing this to the number of bytes that must be retransmitted results the inequality

$$\sum_{i \in I} B_k(t_i) - B_k(t_{i-1}) < (1/3 \cdot w + \epsilon) \cdot T.$$

Notice that the role of k and k' is interchangeable, so an argument similar to the above justifies,

$$\sum_{i \in I'} B_{k'}(t_i) - B_{k'}(t_{i-1}) < (1/3 \cdot w + \epsilon) \cdot T$$

Let the terms on the left side of the last two inequalities denoted by β and β' and consider their sum. Recall that $B_k(t_i) = B_{k'}(t_i)$ for all $0 < i < n$, but not necessarily for $i = 0$ or $i = n$. Therefore $\beta + \beta'$ is a telescopic sum: every term $B_k(t_i)$ in the middle (for $0 < i < n$) is added once then subtracted once, so in affect they cancel out. The only question is what happens at $t_0 = 0$ and $t_n = T$. Let us therefore check four cases depending on whether 1 or n are in the set I or I'.

If $n \in I$ and $1 \in I$, then

$$(2/3 \cdot w + 2\epsilon) \cdot T \geq \beta + \beta' = B_k(T) - B_k(0)$$

If $n \in I'$ and $1 \in I'$, then

$$(2/3 \cdot w + 2\epsilon) \cdot T \geq \beta + \beta' = B_{k'}(T) - B_{k'}(0)$$

If $n \in I$ and $1 \in I'$, then $B_k(T) \geq B_{k'}(T)$, so

$$(2/3 \cdot w + 2\epsilon) \cdot T \geq \beta + \beta' = B_k(T) - B_{k'}(0) \geq B_{k'}(T) - B_{k'}(0)$$

If $n \in I'$ and $1 \in I$, then $B_{k'}(T) \geq B_k(T)$, so

$$(2/3 \cdot w + 2\epsilon) \cdot T \geq \beta + \beta' = B_{k'}(T) - B_k(0) \geq B_k(T) - B_k(0)$$

Notice that in all four cases the expression on the right side of the inequalities is the total number of bytes transmitted during the experiment in either cell k or in cell k'. Dividing both sides of the inequality by T and then taking the $\liminf$ as $T \to \infty$ shows that at least one of those cells have an effective bandwidth less than $2/3 \cdot w + 2\epsilon$ As ϵ can be selected arbitrarily small at the beginning, the effective bandwidth must be less than or equal to $2/3 \cdot w$, and because of the starting hypothesis, it actually must be equal to $2/3 \cdot w$. Hence the efficiency of the algorithm with the given mobile parameters is $2/3$. This finishes the proof.

2.3 Two Algorithms With Constant Efficiencies.

The previous section proved that no algorithm can do better than a $2/3$ efficiency for certain mobile parameters. Here we describe an algorithm that can in fact achieve that upper bound.

Theorem 2.2 *There exists an algorithm that can always achieve the* $2/3$ *efficiency independently of the mobile parameters encountered.*

Proof: For a real number x let the $\{x\}$ denote its fractional part, that is $\{x\} := x - \lfloor x \rfloor$. We specify the algorithm by defining the function $b_k(t)$.

Algorithm 2.1 *Let d be selected to be a constant value that is at least as big as the maximum hand-off delay. Define $b_k(t)$ for cell k ($1 \leq k \leq c$) and $t \in [0, T]$ as follows:*

$$b_k(t) = \begin{cases} \frac{2}{3}wt - \frac{4}{3}wd + 2wd\left\{\frac{t-2d}{6d}\right\}, & \text{if } k \text{ is odd} \\ \frac{2}{3}wt - \frac{4}{3}wd + 2wd\left\{\frac{t+d}{6d}\right\}, & \text{if } k \text{ is even} \end{cases}$$

According to the algorithm all the cells with an odd index transmit exactly the same way and so are all the cells with an even index. Notice that the odd and even b_k functions are just translations of each other. Figure 3 shows their graph. Both types of functions are piece-wise linear and are built

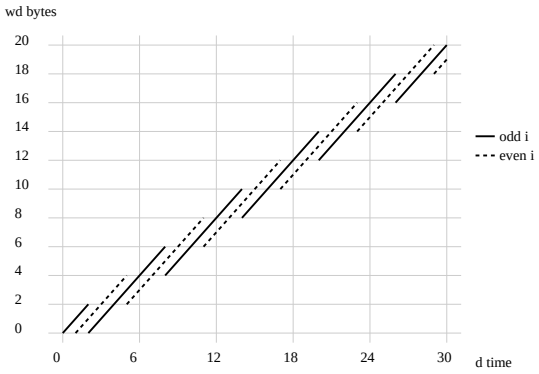

Figure 3: $b_k(t)$ of Algorithm 2.1

from line segments having slope w. They increase for a $6d$ time-period, then have a discontinuity point, where they "drop" $2wd$ bytes lower and continue increasing from that point on. Consequently the $B_k(t)$ functions differ from $b_k(t)$ only on the intervals $[t_0, t_0 + 2d)$, where t_0 is a discontinuity point. Furthermore the inequality $0 \leq B_k(t) - b_k(t) \leq 2wd$ is always true. Figure 4 graphs both of the functions $B_k(t)$ and $b_k(t)$ to illustrate the above facts.

Let us compute the efficiency of the algorithm. Because of the similarity of the odd and even case we will only perform the computations for the former one. Fix an odd cell index k.

Consider $b_k(t)$. Notice that the linear part of $b_k(t)$ is $2/3 \cdot wt$, and while the rest of the terms are not constants they are be bounded. More precisely

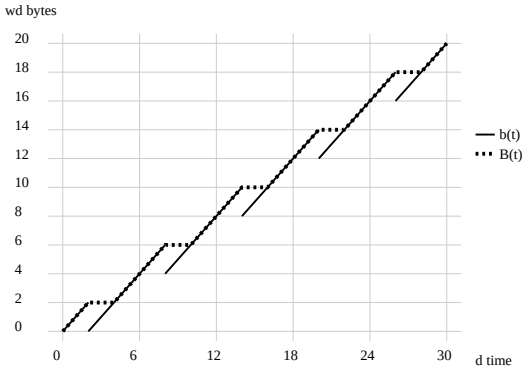

Figure 4: $b_k(t)$ and $B_k(t)$ of Algorithm 2.1 for odd k

$|b_k(t) - 2/3 \cdot wt| \leq 4/3 \cdot wd + 2wd$. Now let us write the function $B_k(t)$ as the sum of three quantities, the first two of which are bounded by a constant not depending on t.

$$B_k(t) = (B_k(t) - b_k(t)) + (b_k(t) - 2/3 \cdot wt) + 2/3 \cdot wt$$

Our plan is to divide both sides by t, then substitute $t = T$ and take the limit as $T \to \infty$, thus obtaining a formula for the effective bandwidth. Notice that we will be able to do some simplifications regarding the first two summands:

$$\liminf_{T\to\infty} \frac{B_k(T) - b_k(T)}{T} = \liminf_{T\to\infty} \frac{2wd}{T} = 0$$

and

$$\liminf_{T\to\infty} \frac{b_k(T) - 2/3 \cdot wT}{T} = \liminf_{T\to\infty} \frac{4/3 \cdot wd + 2wd}{T} = 0$$

The computation of the effective bandwidth can now be carried out as follows:

$$\liminf_{T\to\infty} \frac{B_k(T)}{T} = 0 + 0 + \liminf_{T\to\infty} \frac{2/3 \cdot wT}{T} = 2/3 \cdot w$$

This quantity is independent of the cell index k. Hence the effective bandwidth of the algorithm is $2/3 \cdot w$ as well, resulting in a 2/3 efficiency.

Next we show that if the cells transmit according to the algorithm, then they provide a continuous stream of data without any holes to all mobiles, and they successfully honor retransmission requests of those who lost some data during the fade periods. The following claim will be useful in establishing this.

Claim 2.3 *For any two time values $t_1 < t_2$ that are at least $2d$ apart and for any cell index k the following holds.*

(i) $b_k(t_1) \leq b_k(t_2)$*; equality holds if and only if* b_k *has a discontinuity in* $[t_1, t_2)$ *and* $t_1 + 2d = t_2$

(ii) $b_k([t_1, t_2))$ *is an interval that contains* $[b_k(t_1), b_k(t_2))$

Proof: Part (i) is obvious in view of the observations following the definition of the algorithm. For the proof of (ii) we may assume that t_1 and t_2 are exactly $2d$ apart. The reason is that any interval larger than $2d$ can be written as the union of intervals whose size is exactly $2d$ in such a way that consecutive ones overlap. The image of the large interval is the union of the images of the intervals of size $2d$. But they are again intervals (by what we are about to prove) and again satisfy the property that consecutive ones overlap. Hence the image of the large interval is a single interval.

Now if $b_k(t)$ has no discontinuity in $[t_1, t_2)$, then the image of the interval is obviously an interval, namely $[b_k(t_1), b_k(t_1) + 2dw)$. If on the other hand there is a discontinuity at $t_0 \in [t_1, t_2)$, then

$$b_k([t_1, t_2)) = b_k([t_1, t_0) \cup [t_0, t_2)) = [b_k(t_1), b_k(t_0) + 2wd) \cup [b_k(t_0), b_k(t_2)),$$

but by part (i) $b_k(t_1) = b_k(t_2)$ holds, so the union on the right is actually the single interval $[b_k(t_0), b_k(t_0) + 2wd)$. This finishes the proof of the Claim.

To finish the proof of Theorem 2.2 we just need to establish that the algorithm works as promised: it provides a continuous stream of data to all mobiles, it does not introduce new holes (except maybe when mobiles arrive at a cell), and retransmits holes that occurred while mobiles crossed cell boundaries.

Consider a mobile leaving cell k and arriving to cell k' at time t. Because of the definition of d, (which is selected to be at least as large as the maximum hand-off delay), the mobile in question must have been in cell k at time $t-d$, so its transmission byte-count must be at least as large as $B_k(t-d)$. Let t_0 be selected to be a discontinuity of $b_{k'}(t)$ satisfying $t_0 - 2d \leq t \leq t_0 + 4d$. We check two cases.

If $t \in [t_0, t_0 + 4d)$, then $b_k(t-d) = b_{k'}(t)$ holds. In particular $B_k(t-d) \geq b_{k'}(t)$ is true. This equation basically says that no hole is introduced when the mobile enters cell k'.

Suppose that $t \in [t_0 - 2d, t_0)$, thus $t_0 - 3d \leq t - d < t_0 - d$. It is easy to check from the graphs of the functions, that $t_0 - 3d$ is a discontinuity of $b_k(t)$. Therefore $B_k(t)$ is constant on the interval $[t_0 - 3d, t_0 - d)$ and its value is equal to $b_{k'}(t_0)$. In particular $B_k(t-d) = b_{k'}(t_0)$ holds. Consequently, even if a hole is introduced when the mobile enters cell k', at time t_0 a retransmission of the missing bytes will start, thus ensuring continuous transmission from that point on.

(A closer inspection of the algorithm can even give a bound on how soon the retransmission will happen. It will start at most $3d$ times later than the mobile leaves cell k.) This concludes the proof of Theorem 2.2.

In the rest of this section we present another algorithm for the highway problem. The first has a constant $1/2$ efficiency. In this sense it is inferior to the algorithm presented above, nevertheless it is easy to implement in practice and it will play an important role during the discussion of the practical approach.

Algorithm 2.2 *Let d be selected to be a constant value that is at least as large as the maximum hand-off delay. Define $b_k(t)$ for cell k ($1 \leq k \leq c$) and $t \in [0, T]$ as follows:*

$$b_k(t) = \frac{1}{2}wt - \frac{1}{2}wd + wd\left\{\frac{t-d}{2d}\right\}$$

Figure 5 shows the graph of this function with d set to 2. It must be clear

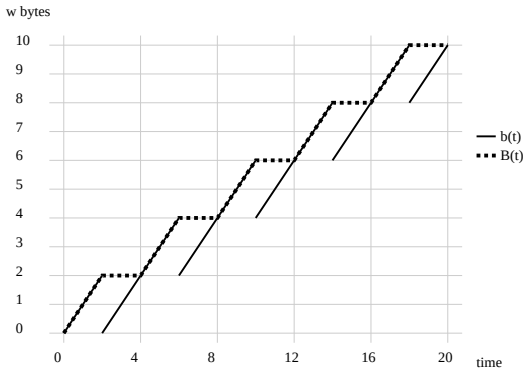

Figure 5: $b_k(t)$ and $B_k(t)$ of Algorithm 2.2 with $d = 2$

from the figure that cells working in accordance with this algorithm transmit new data for a certain period of time, then retransmit the exact same data portion again, and continue in this fashion. The moral of this method is that mobiles who received the transmission for the first time can simply disregard the retransmission. On the other hand mobiles crossing through cell boundaries will miss some data portion in one cell, but they will arrive in the next cell in time to receive the retransmission of the bytes they just missed. This is guaranteed to work out, because the time period of the transmissions/retransmissions is chosen to be at least as large as the maximum hand-off delay. (It is 2 time units in the picture.)

The effective bandwidth of each cell is clearly $1/2 \cdot w$, and so is the effective bandwidth of the algorithm. Consequently the efficiency is 1/2.

3 The Practical Approach

In the previous chapter we addressed the issue of multicasting from the theoretical viewpoint. The algorithms described there can be applied successfully in all cases, but they have an important drawback. Since they do not depend on the actual mobile parameters, their efficiency is the same constant value, even if there are only a few mobiles present and hence much higher efficiencies could be achieved.

In this chapter we offer solutions for this problem. We devise algorithms whose worst case behavior coincides with those mentioned in the previous chapter, but which does take mobile parameters into account and work more efficiently if fewer are present.

The algorithm that will be introduced in the second section has a worst case efficiency of 1/2. Because of a technical reason it cannot stay above the 2/3 bound, hence further solutions will be investigated, that either modify the algorithm slightly, or combine it with the algorithm that produces the constant 2/3 efficiency.

3.1 The Simulation

The difference between the algorithms applicable to the highway problem is the way they deal with retransmission requests of the mobiles. The algorithm presented below takes care of retransmission requests as soon as the neighboring cells transmission byte-counts are higher than the byte-count

requested by the mobile. This simple condition immediately ensures that all data portions are transmitted at most twice, resulting an efficiency greater than or equal to 1/2.

Algorithm 3.1 Let $b_k(t)$ and $B_k(t)$ be defined as in Chapter 2, and let d denote the maximum hand-off delay. Initially holeSet is empty and $B_k(0)$ is 0. The algorithm determines $b_k(t)$ on the base of the current holeSet, $B_k(t)$ and $B_{k\pm1}(t-d+1)$:

```
for each newly arrived mobile do
   if ( mobile.byteCount < B_k(t) ) then
      holeSet.Add( Hole(mobile.byteCount,B_k(t)) )
   endif
end
lowestHoleBytecount := holeSet.MinimumStartBytecount()
lowestNeighborBytecount := min( B_{k-1}(t-d+1) , B_{k+1}(t-d+1) )
if ( lowestHoleBytecount < lowestNeighborBytecount ) then
   b_k(t) := lowestHoleBytecount ;  B_k(t+1) := B_k(t)
   holeSet.UpdateStartBytecounts()
else
   b_k(t) := B_k(t) ;  B_k(t+1) := B_k(t) + w
endif
```

The performance of the algorithm is illustrated in the following figures. The first one represents the results of a simulation in which 6 consecutive cells were considered. The number of mobiles on the highway is specified as a rate: their average number over a time period of length "maximum hand-off delay". This is is convenient, because if cars moving in the same direction are about that much apart, then the highway can be considered full for all practical purposes. Notice though, that this rate can be higher than 1.

We varied the mobile rate from 0 to 1.1, and measured the efficiency. Each experiment was repeated 10 times, then the average and the standard deviation calculated to verify that the data is statistically significant and of small variance. Finally the experiment was also repeated with three different maximum hand-off delays: 2, 3 and 4 time units. The three functions have almost the same graph, showing that the "rate of the mobiles" is a good choice for the independent variable (as opposed to just the total number of mobiles).

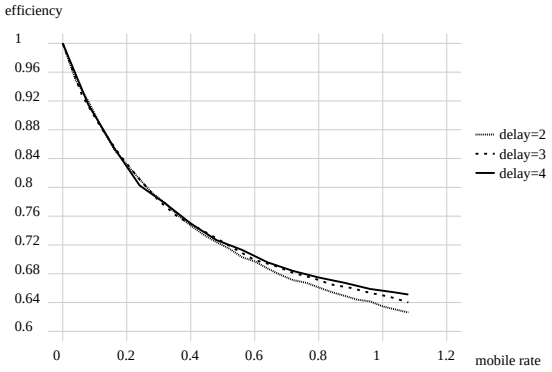

Figure 6: The efficiency vs. the mobile rate for different delays

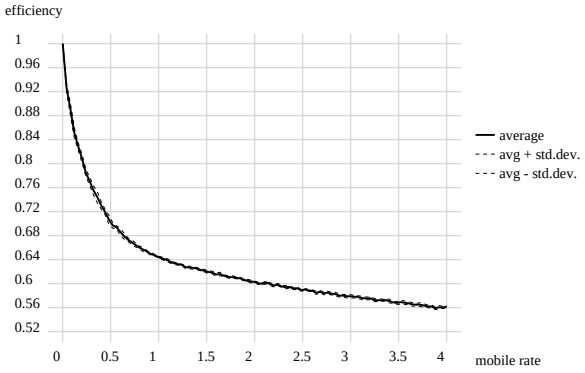

Figure 7: The efficiency vs. the mobile rate — 20 cells

Figure 7 illustrates that the consecutive repetitions of the experiment does in fact show small variance. This measurement was taken while simulating 20 cells, and each experiment repeated 8 times. On the graph not only the average efficiency, but plus/minus the standard deviation is plotted to show the variance.

It can be seen from either one of the the graphs that the efficiency does depend on the mobile parameters, and its value is high for small mobile rates. While the efficiency stays above 1/2 at all times, at around 0.8 mobile rate the efficiency drops below the 2/3 value. This behavior may be seen as a "technical problem". What happens is that when the highway is close to being full of mobiles, and two cells happen to be at the same byte-count for a short period, the algorithm will suggest the transmission of the same bytes

again for both cells. During this a hole will be introduced for every mobile crossing between the two cells *in either direction*, which induces a retransmission in *both cells*. If enough mobiles are present, every byte transmitted needs to be retransmitted a little bit later in both cells, resulting an effective bandwidth of $1/2 \cdot w$. Locally in these two cells the transmission will proceed just as Algorithm 2.2 describes it, and cells further away will not be able to change this characteristics unless the rate of mobiles drops.

We offer two different ways to increase the efficiency in these cases. As stated above the problem starts because the algorithm suggests the transmission of the same bytes to both MSS.

We can change the algorithm slightly as follows: the MSS looks at its neighbor's byte-count to determine if they are about to start retransmitting the same bytes. If so, one of them (say the one with the odd index) deviates from the algorithm: instead of retransmitting it transmits new data. This modification ensures that the behavior resulting the constant 1/2 efficiency will not happen, but — as simulation results made it clear — does not guarantee that a similar phenomenon is avoided with an efficiency that is just a little bit above 1/2.

A better solution is to combine Algorithms 3.1 and 2.1 as follows: For smaller mobile rates the algorithm of this chapter would take precedence, but if the mobile rate becomes larger than a preset value (say 0.75) the algorithm would dynamically switch itself to follow the behavior of the constant algorithm with the 2/3 efficiency.

4 Conclusion

The previous chapters examine ways in which data can be transmitted efficiently to mobile users on a highway. We looked at the problem from both the theoretical and the practical viewpoint. The theoretical approach started with a mathematical description of the model. After defining a reasonable measure of efficiency we obtained an upper bound on the worst case efficiency of any algorithm. We also showed an algorithm with a constant efficiency that achieves that bound. These efforts mainly targeted the extreme cases, hoping that the results show us some constraints that a practical algorithm should satisfy. Using the practical approach we looked at the performance of the algorithms in the case when the number of the mobiles is neither very small nor too large. The efficiency of the (modified) algorithm is between the

theoretical bounds, 2/3 and 1, but is not a constant. The smaller number of mobiles are present, the better the efficiency becomes; but even in the worst case it does not fall below 2/3.

It should be noted that the 2/3 bound is not dependent on the number of mobiles in the cell. It is therefore in sharp contrast with the case where the mobile computers request different data. Then multicasting is not possible — individual transmissions are necessary — and thus the efficiency drops below $1/m$, where m denotes the average number of mobiles in a cell. Compared to this scenario, the possibility of multicasting is a great advantage. When it is done according to the algorithm of Chapter 3, mobiles will receive data at a rate $2/3 \cdot w$ or higher, whereas transmitting individually even to two mobiles in a cell immediately reduces the data rate of the receivers to less than $1/2 \cdot w$.

Let us now briefly examine the possible ways these results can be generalized. Consider the general multicasting case, where mobiles roam around in a city block or a building. The behavior of the mobiles in these cases is not as uniform as in the case of the highway, but some of the algorithms we considered did not have strict assumptions about them either. So let Algorithm 2.2 be applied in the general case. Cells following this algorithm have the same transmission byte-counts. Because of this fact, it does not cause any complications that in this general case a cell can have more than two neighbors: mobiles coming from any direction still have the same byte-counts. Therefore the only requirement that is needed to ensure that the algorithm will work is the following: mobiles must spend at least as much time in each cell as the maximum hand-off delay. This property looks easy to satisfy in practice, by designing how the cells cover that portion of the city block. On the other hand we cannot control the mobiles: They can deliberately behave in ways that cannot be predicted very well (say they can enter a cell, then immediately take a U-turn and go back to the previous cell). Since we do not want to optimize the algorithm to handle these exceptional cases such mobiles may loose small data portions.

To be able to extend Algorithm 2.1 we must make stronger assumptions. If a 2-dimensional cell architecture ensures that mobiles spend at least twice as much time in each cell as the maximum hand-off delay, and in addition it is possible to label the cells "odd" and "even" in such a way that mobiles will always cross cell boundaries between cells of opposite parity, then Algorithm 2.1 may be successfully applied to implement an efficient multicasting scheme.

References

[1] Kevin Brown and Suresh Singh, "The problem of multicast in mobile networks", *Proceedings IC3N conference*, October, 1996, pp. 278-282.

[2] K. Brown and S. Singh, "M-UDP: UDP for Mobile Networks", *ACM Computer Communication Review*, Vol. 26(5), Pct. 1996, pp. 60-78.

[3] K. Brown and S. Singh,"RelM: Reliable Multicast in Mobile Networks", *J. Computer Communications*, (accepted, to appear).

[4] S. Cheshire and M. Baker,"Internet Mobility 4×4", *Proceedings ACM SIGCOMM'96*, (Oct. 1996), pp. 318-329.

[5] A. Bakre and B. R. Badrinath,"I-TCP: Indirect TCP for Mobile Hosts", *Proceedings of the 15th International Conference on Distributed Computing Systems, Vancouver, Canada,* June 1995, pp. 136–143.

[6] P. Bhagwat, S. Tripathi and C. Perkins,"Network Layer Mobility: an Architecture and Survey", *Technical Report*, CS-TR-3570, University of Maryland (September 13, 1995).

[7] B.R. Badrinath and T. Imielinski,"Location Management for Networks with Mobile Users", in Mobile Computing, eds. T. Imielinski and H.F. Korth, Kluwer Academic Publishers, (1996), pp. 129-152.

[8] R. Ghai and S. Singh, "An Architecture and Communication Protocol for Picocellular Networks", *IEEE Personal Communications Magazine*, Third Quarter 1994, pp. 36-46.

[9] F. Teraoka and M. Tokoro, "Host Migration Transparency in IP Networks: The VIP Approach", *SIGCOMM CCR*, Vol. 23, No. 1, Jan 1993, pp. 45-65.

[10] R. Yavatkar and N. Bhagawat,"Improving End-to-End Performance of TCP over Mobile Internetworks", *IEEE 1994 Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA*, (1994).