The problem of multicast in mobile networks

K. Brown

Department of Computer Science
University of South Carolina
Columbia, SC 29208

S. Singh

Department of Computer Science
University of South Carolina
Columbia, SC 29208

Abstract

We consider the problem of multicasting optimally to mobile users in a cellular mobile network. In the absence of mobility, a single channel can be used to multicast to all mobile users within a cell. However, mobility combined with the effects of fading necessitate a more complex channel allocation policy. We present an optimal channel allocation algorithm for multicasting within a cell and indicate how it can be extended to the multicell case.

1 Introduction

In this paper we consider the problem of **multicasting** in a **cellular mobile network**. The **user model** to keep in mind is one where a user equipped with a PDA (sophisticated form of a Newton or a Sony Magiclink) roams about in unpredictable ways but always needs to remain connected to the rest of the world. We believe that a large number of services requested by such mobile users will be multicast to them (for efficiency reasons) from the various service-providers located around the world (note that this is true of users who are not mobile, as well). Thus, users may subscribe to a service that multicasts the latest stock market quotes from Wall Street, for example.

The **network model** we consider is a cellular network. We assume that populated areas such as buildings, roadways, etc. are covered by a cellular radio network. Figure 1 illustrates a possible coverage of a campus (such as a University campus) using a microcellular and possibly a macrocellular network. Every cell (macrocell, microcell or picocell) has a *base station* that is responsible for providing users with a connection endpoint. 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. Our implementation of the networking software responsible for forwarding data and handling QoS requirements is discussed in [2].

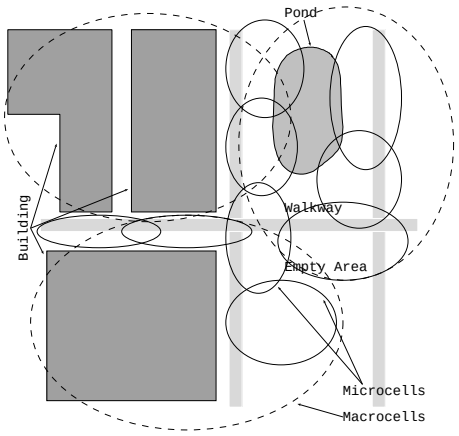

Figure 1: Cellular Coverage of a campus.

2 The Problem of Efficient Multicast

2.1 Assumptions and Notation

Let us assume that the set of Mobile Hosts (MH) belonging to a particular multicast group is denoted by $\mathcal{M}$. It is likely that these MHs will be located in different cells. Let $c_1, c_2, \ldots, c_k$ denote the cells that contain $n_1, n_2, \ldots, n_k$ (where $n_i > 0$ and $\sum n_i = |\mathcal{M}|$) of these mobile hosts respectively. This set of cells is called a *cell group* and denoted by $\mathcal{C}$. Let us further assume that the *bandwidth* allocated to this multicast group in each of the cells is denoted by $B_1, B_2, \ldots, B_k$ bytes/sec. That is, data is transmitted on a specified channel at B_i bytes/sec and is received by all MHs of the multicast group located in cell c_i. We assume that the transmissions on the uplink as well as on the downlink are managed by the MSS thus avoiding collisions and we assume that the channel allocation policy used is *deterministic* (a policy such as TDMA[1] could be used, for example).

[1] The reason for this choice is that most proposed standards for communication within a cell such as, the European standard (GSM), the North American standard (IS-54) and another Pan-European standard (DECT) use TDMA (see [3]).

2.2 Some Examples of the Effects of Mobility

Figure 2 illustrates a system in which cells c_1, c_2, c_3 and c_4 contain mobiles that belong to $\mathcal{M}$. Thus $\mathcal{C} = \{c_1, c_2, c_3, c_4\}$. Three types of moves are illustrated in the figure. In the first type of move, a MH moves from cell c_1 into cell H. Cell H does not initially contain any MHs from $\mathcal{M}$ and therefore does not belong to $\mathcal{C}$. In this case H joins $\mathcal{C}$ and the mobile support station in H fetches multicast data for the MH and commences transmission. In the second type of move illustrated in the figure, the MH moves from c_2 into G and then into c_4. In this case, G initially joins $\mathcal{C}$ but leaves the cell group after the MH enters c_4. The third type of move is one where the MH moves from c_3 into c_2, both of which belong to $\mathcal{C}$.

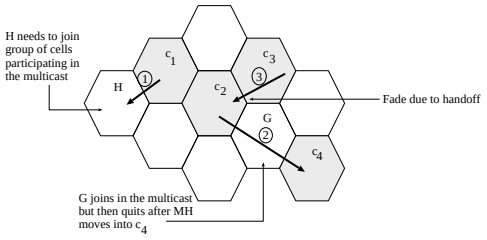

Figure 2: Effect of mobility on multicasts.

To better illustrate the focus of this paper, let us consider the third type of move illustrated in Figure 2. Let B_2 and B_3 indicate the bandwidth allocated to the multicast in cells c_2 and c_3 respectively. Assume that the multicast begins at time 0 and the mobile moves from c_3 into c_2 at time T (for simplicity assume that the move is instantaneous). If $B_2 > B_3$ then, when the MH enters cell c_2, the next byte to be transmitted to the multicast group in c_2 is $TB_2 + 1$ whereas the last byte received by the MH in c_3 is TB_3. In this case we have a choice of policies regarding the allocation of B_2 – continue with the multicast in c_2 starting from byte $TB_3 + 1$ (as suggested in [1]) so that the new MH can 'catch up' with the rest of the MHs in c_2, or ignore the new MH and continue with the multicast (in this case the MH would treat bytes TB_3 to TB_2 as being lost), or split the multicast channel in two where one channel begins the multicast from byte $TB_3 + 1$ while the other continues from $TB_2 + 1$. A similar problem arises if $B_2 < B_3$. In this case the MH will already have received data that is currently being transmitted in cell c_2 and one policy might require that the new MH wait for the multicast to 'catch up' before it can begin receiving. Another policy may require that the multicast channel be split to ensure that the new MH can continue receiving data for the multicast and not be idle.

Another problem associated with the third type of move illustrated in Figure 2 is that the MH may experience a period disconnection or **fade** when it moves from c_3 into c_2. In this case, even if $B_2 = B_3$, the MH, upon entering c_2, will see that it has not received all the bytes from the multicast (those transmitted during the fade). Here, again, we need to make a decision about how to reallocate (or whether to reallocate) the multicast bandwidth available in c_2. It is noteworthy that a MH may also encounter periods of fade while in the same cell (e.g., while moving around corners of buildings).

In a typical system, we can expect hundreds of MHs to belong to $\mathcal{M}$. If most of them roam we will have a situation where MHs randomly and frequently enter and leave cells. This movement needs to be considered in any bandwidth allocation policy. For example, consider the case when a MH moves from cell c_1 to c_2 to c_3 in Figure 2. If $B_1 < B_2 < B_3$ then, when the MH enters c_2, it is behind in the multicast (i.e., MHs in c_2 have received bytes not received by the new MH). Let us assume that the bandwidth allocation policy in each cell determines that the new MH will have to regard bytes $T_1B_2 - T_1B_1$ (where T_1 is the MHs latency in c_1) as being lost – i.e., the multicast in c_2 continues without change. Say the MH stays in cell c_2 for T_2 time and then roams into c_3. If the bandwidth allocation policy in c_3 is the same as in c_2, bytes $B_3(T_1 + T_2) - B_2(T_1 + T_2)$ will be lost to the MH. Thus, in time $T_1 + T_2$, the MH has lost a total of $(B_3 - B_1)T_1 + (B_3 - B_2)T_2$ bytes. While some loss may be acceptable for some applications, it is easy to construct scenarios where the cumulative loss can be very large calling into question the bandwidth allocation policy used.

If we used a different bandwidth allocation policy, one where all bandwidth is allocated to the incoming MH to allow it to catch up, it is easy to see that the multicast is unduly delayed. In the example from the previous paragraph, when the MH enters c_2, transmission starts from byte $B_1T_1 + 1$ since the MH has received bytes upto B_1T_1 in c_1. The MH catches up with the other MHs in c_2 after time $T_1\frac{B_2-B_1}{B_2}$, at which point the multicast proceeds with all MHs in c_2 receiving new data. When the MH leaves c_2, it has received bytes upto $B_1T_1 + B_2T_2$ (as have all the other MHs in c_2) while the MHs in c_3 have received bytes upto $B_3(T_1 + T_2)$. The multicast in c_3 is now held up for time $t = \frac{T_1(B_3-B_1)+T_2(B_3-B_2)}{B_3}$ waiting for the new MH to catch up. Thus the MHs in

c_3 receive bytes upto $B_3(T_1+T_2)$ in time T_1+T_2+t whereas, if we use the greedy bandwidth allocation method from the previous paragraph, they would have received bytes upto $2B_3(T_1+T_2)-T_1B_1-T_2B_2$ or $T_1(B_3-B_1)-T_2(B_3-B_2)$ bytes more. It is easy to see that this second bandwidth allocation scheme increases delays seen by all MHs. While this may be justified in some instances, it is easy to construct scenarios involving multiple roaming MHs that will cause the delay to become inordinately large.

In this paper we look at the problem of multicast within a single cell only. The problem of optimal multicast in a multicell environment is much harder and we currently only have partial solutions to that problem.

3 Single Cell Problem

Let us consider the problem of multicast in a *single cell*. MHs arrive into the cell and depart from it randomly. At any time t, we can group MHs within a cell according to the *next byte* they expect to receive. Let these groups be denoted as, $g_1^t, g_2^t, \ldots, g_k^t$. The next byte that MHs in group g_i^t expect to receive is b_i^t. Let $\mathcal{B}$ bytes/sec denote the total amount of multicast bandwidth available that is divided into k channels. Transmissions along channel C_i^t begin at byte b_i^t and proceed at rate B_i^t. If $B_i^t > B_{i+1}^t$, it is possible for the transmissions along channel C_i^t to catch up with the transmission along channel C_{i+1}^t. In this case the groups g_i^t and g_{i+1}^t *merge* together and transmission for this new group proceeds at a rate of $B_i^t + B_{i+1}^t$.

3.1 A Greedy Optimal Case

Let us assume that, from the point of view of a service provider (who provides the service delivered by the multicast), a policy that maximizes throughput is the optimal policy where the throughput may be defined as:

Definition 1 Throughput *is the* total number of new bytes transmitted per second *to MHs in* $\mathcal{M}$.

To illustrate our definition, suppose the bandwidth allocated to the multicast in the cell under consideration is B bytes/sec. Say the cell contains n MHs who are receiving the multicast. In the case where no MH is mobile, all the n MHs will begin hearing the multicast together. The throughput is thus, nB bytes/sec (if we assume that the base station always has data to transmit).

How does the bandwidth allocation policy affect throughput? Consider a situation where a multicast begins at time 0 and proceeds at a rate of B bytes/second (let us continue to assume that the MSS always has data to transmit) in cell c. The throughput is nB bytes/sec in c if c contains n MHs, as above. Now assume that at time T a roaming MH, who has received bytes upto TB' ($B' > B$), enters cell c. If the multicast proceeds without any change, the throughput will continue to remain nB bytes/sec for time $T\frac{B'-B}{B}$ after which it will become $(n+1)B$ (i.e., the roaming MH will begin to receive *new* data). If we use a different bandwidth allocation policy that requires, for example, that the channel be split in two equal halves when a new MH arrives, the throughput is $n\frac{B}{2}+\frac{B}{2}$ bytes/sec. It is easy to see that other bandwidth allocation policies will have different throughputs as well.

Let us now consider the problem of optimal bandwidth allocation for an interval $[0,T]$. For simplicity, let us assume that no arrivals or departures take place during this time. At time 0 there are k groups. Let $0 < T_1 < T$ be a time after which the number of groups is $k_1 < k$. That is, $k-k_1-1$ groups merged with other groups at time T_1. Assume that all $k-k_1+1$ groups merged *together* (the case when several independent merges take place simultaneously is a simple generalization). Say $l+1 = k-k_1+1$ and the groups that merge together are $g_i^0, g_{i+1}^0, \ldots, g_{i+l}^0$.

Observation 1 *Let,*

$$B' = \sum_{j=i}^{i+l} B_j^0$$

Then, the throughput, contributed by groups $g_i^0, g_{i+1}^0, \ldots, g_{i+l}^0$, for the interval $[0,T_1]$ is increased if the bandwidth allocation for channels $C_i^0, C_{i+1}^0, \ldots, C_{i+l}^0$ at time 0 is,

$$\tilde{B}_i^0 = B', \tilde{B}_{i+1}^0 = \ldots = \tilde{B}_{i+l}^0 = 0$$

Proof: The proof is an easy algebraic exercise and is omitted. □

Theorem 1 *Let $g_1^0, g_2^0, \ldots, g_p^0$ be the set of groups at time 0. Then, given any bandwidth allocation β, we can construct a different policy $\tilde{\beta}$ that yields a higher throughput. $\tilde{\beta} = \{\tilde{B}_1^0, \tilde{B}_2^0, \ldots, \tilde{B}_p^0\}$ at time 0 is such that,*

$$\tilde{B}_i^0 = \mathcal{B} \text{ for some } i \text{ and } \tilde{B}_j^0 = 0, \forall 1 \leq j \leq p, j \neq i$$

Proof: Let q denote the number of groups at time t if we use β. If $q = p$ then select i such that,

$$n_i = \max\{n_1, n_2, \ldots, n_p\}$$

Clearly this allocation gives a higher throughput and satisfies our condition.

If $q < p$, on the other hand, then two or more groups merged at various times prior to t. Let $0 < t' \leq t$ denote the latest time such that the number of groups before t' is greater than q and after t' is exactly q. In other words no merges take place between time t' and t.

Step 1: Let k denote the index of the group g_k^t such that,

$$|g_k^t| = \max\{|g_j^t|, 1 \leq j \leq q\}$$

Then we achieve a higher throughput if we allocate all of the bandwidth $\mathcal{B}$ to $g_k^{t'}$ $(= g_k^t)$ for the period $[t', t]$.

Step 2: At time t', let $b_1^{t'} < b_2^{t'} < \ldots < b_q^{t'}$ denote the bytes that groups $g_1^{t'}, \ldots, g_q^{t'}$ expect to receive next. Let,

$$g_j^{t'} = \bigcup_{l=a_j}^{b_j} g_l^0, a_j \geq j$$

Then, according to Observation 1, a better bandwidth allocation scheme, to improve throughput, is,

$$\gamma_{a_j}^0 = \sum_{l=a_j}^{b_j} B_l^0, \text{ and } \gamma_l^0 = 0, a_j < l \leq b_j \qquad (1)$$

This holds true for each of the groups $g_j^{t'}$.

Step 3: Consider the number of bytes transmitted during time 0 to t' to form group $g_j^{t'}$. This number is,

$$b_j^{t'} - b_{a_j}^0 \leq t'\gamma_{a_j}^0$$

The total number of bytes transmitted during the time t' to form *all* groups $g_j^{t'}$ is less than or equal to,

$$\sum_{j=1}^{q} t'\gamma_{a_j}^0 = t'\mathcal{B} \qquad (2)$$

Step 4: Consider a new bandwidth allocation δ constructed as follows:

$$\delta_{a_1}^0 = \mathcal{B}, \text{ for time } [0, t_1] \text{ where, } t_1 = \frac{b_1^{t'} - b_{a_1}^0}{\mathcal{B}}$$

for the period $[t_1, t_2]$,

$$\delta_{a_2}^{t_1} = \mathcal{B} \text{ where, } t_2 = \frac{b_2^{t'} - b_{a_2}^0}{\mathcal{B}}$$

and so on,

$$\delta_{a_q}^{t_{q-1}} = \mathcal{B} \text{ for the period } [t_{q-1}, t']$$

$$\text{where, } t_{q-1} = \frac{b_{q-1}^{t'} - b_{a_{q-1}}}{\mathcal{B}}$$

It is easy to see that the total number of new bytes transmitted during the time $[0, t']$ is $t'\mathcal{B}$ and, furthermore, this bandwidth allocation method yields a system configuration identical to the one at the end of Step 2.

Step 5: We can modify the order of transmissions in Step 4 so that at time 0,

$$\delta_{a_k}^0 = \mathcal{B} \text{ for period 0 to } \frac{b_k^{t'} - b_{a_k}^0}{\mathcal{B}}$$

(recall that $g_k^{t'}$ is the largest group at time t'). This does not affect the total throughput obtained in Step 4.

Step 6: At time $\bar{t} = \frac{b_k^{t'} - b_{a_k}^0}{\mathcal{B}}$ the group $g_k^{t'}$ has already been formed. Since $g_k^{t'}$ is the largest group at time t', it is also the largest group at $\bar{t}$. Therefore, if we allocate all of the bandwidth $\mathcal{B}$ to this group for the time $[\bar{t}, t']$, the total throughput achieved will be at least as large as the throughput obtained by the bandwidth allocation scheme in Step 4. If we now look at the bandwidth allocation for the entire period $[0, t]$, we note that,

$$\tilde{\beta} = \{0, 0, \ldots, \tilde{B}_{a_k}^0 = \mathcal{B}, 0, \ldots, 0\}$$

and the throughput obtained is greater than that obtained using β. □

Corollary 1 *The* optimal *bandwidth allocation is one in which all the bandwidth is allocated to one group.*

Proof: Follows from Theorem 1. □

We thus have a simple algorithm to find the optimal bandwidth allocation:

Algorithm 1 *1. Let* throughput(g_i^0, T) *be the throughput obtained using a bandwidth allocation scheme where all of the bandwidth $\mathcal{B}$ is allocated to group g_i^0 at time 0 for the entire period T.*

2. Let g_j^0 be such that,

$$\text{throughput}(g_j^0, T) \geq \text{throughput}(g_i^0, T), \forall 1 \leq i \leq p, i \neq j$$

then the optimal bandwidth allocation scheme allocates all the bandwidth to g_j^0.

Note that the *complexity* of this algorithm is linear in p, the number of groups at time 0. It is, however, noteworthy that the bandwidth allocation that results from the above algorithm is one where many MHs will not receive any data from the multicast for the period $[0, T]$.

3.2 Extensions

The above algorithm maximizes throughput for a *given* T under the assumption that no arrivals or departures take place during that time. It is easy to construct examples where the above algorithm is not optimal in the presence of arrivals or departures. How, then, can we use this algorithm in a real world scenario? Figure 3 illustrates how Algorithm 1 may now be applied in a real system. Say mobile users roam into and out of the cell at times $t_1, t_2, t_3, \ldots$. We apply Algorithm 1 to each interval $T_1 = [0, t_1], T_2 = [t_1, t_2], T_3 = [t_2, t_3], \ldots$ independently. It is important to observe that we cannot do better than this *unless we can predict future arrivals or departures.*

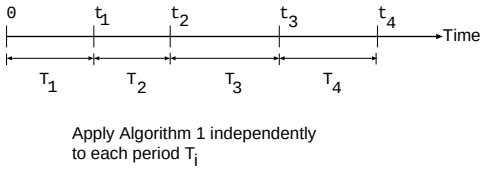

Figure 3: Application of Algorithm 1 in reality.

A related problem in applying Algorithm 1 is that if we fail to predict T accurately, the bandwidth allocation will be suboptimal. Figure 4 illustrates two cases when the predicted time T is either longer than or shorter than the actual time between events. The figure illustrates the worst case scenario for both cases. Figure 4(a) considers the case when $T < \tau$ where T is the predicted value and τ is the actual value between events. Assume that there are three groups at time 0 with $n-1$, $n-1$ and n MHs respectively. The optimal allocation **A**, computed using Algorithm 1 over the period $[0, T]$, has a throughput of $t_A = n\mathcal{B}$ over the interval $[0, \tau]$. The *actual* optimal allocation over period $[0, \tau]$, **B**, has a throughput of, $t_B = 2(n-1)\mathcal{B} - \frac{(n-1)T\mathcal{B}}{\tau}$. Observe that $t_B > t_A$ iff, $\tau > \frac{n-1}{n-2}T$

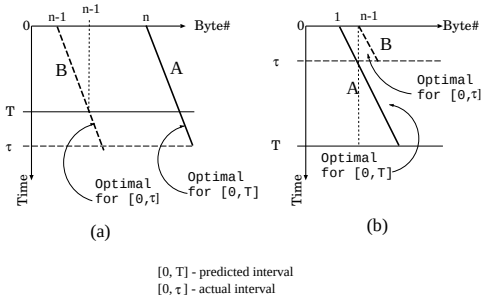

Figure 4: The problem with poor estimates for T.

Figure 4(b) illustrates the worst case when $T > \tau$. At time 0 we have two groups, one with 1 member and the other with $n-1$ members. The optimal allocation **A** over the period $[0, T]$ has a calculated throughput of $n\mathcal{B} - \frac{(n-1)\mathcal{B}\tau}{T}$. The *actual* throughput attained over period $[0, \tau]$, however, is only $t_A = \mathcal{B}$. The allocation **B** is optimal over $[0, \tau]$ and has a throughput of $t_B = (n-1)\mathcal{B}$. Thus, $t_B > t_A$ iff $n > 2$ for all $\tau < T$. It is easy to see why this is the worst case scenario for $\tau < T$.

It is noteworthy that we can use the above calculations to determine the *expected worst case error* and the *probability of a sub-optimal schedule* when the estimate for T is incorrect. If we assume that T is an exponential random variable with a mean of $1/\lambda$, the probability for a sub-optimal schedule is,

$$e^{-\frac{n-1}{n-2}} + e^{-\frac{1}{n-1}} - e^{-1}$$

when the estimate used for $T = 1/\lambda$. Similar calculations can be performed for other distributions as well and will be omitted from this paper.

4 Conclusions

The above discussion brings up an important question: under what conditions is Algorithm 1 optimal? We need a good estimate for T (this is feasible in cellular networks covering, for example, highways) such that few arrivals or departures take place during these intervals. Further, if we can anticipate arrivals and departures, we could modify algorithm 1 to take that into consideration. Finally, there is the issue of *fairness.* Algorithm 1 penalizes mobile users in order to maximize throughput. A simple solution to this problem is to divide $\mathcal{B}$ into two parts $\mathcal{B}_1$ and $\mathcal{B}_2$. $calB_1$ is divided equally among all groups while $\mathcal{B}_2$ is allocated using Algorithm 1.

Acknowledgments

Funding for this work was provided by the NSF under grant NCR-9410357.

References

[1] A. Acharya and B. Badrinath. A framework for delivering multicast messages in networks with mobile hosts. *Manuscript, DATAMAN Project, Rutgers University*, 1995.

[2] K. Brown and S. Singh. A Network Architecture for Mobile Computing. *IEEE INFOCOM, March 24-28*, pages 1388–1396, 1996.

[3] David J. Goodman. Trends in cellular and cordless communications. *IEEE Communications Magazine*, pages 31–40, June 1991.