Providing Seamless Communication in Mobile Wireless Networks [*]

Bikram S. Bakshi P. Krishna D. K. Pradhan N. H. Vaidya

Department of Computer Science
Texas A&M University
College Station, TX 77843-3112
E-mail: {bbakshi,pkrishna,pradhan,vaidya}@cs.tamu.edu
Phone: (409) 862-3411

April 15, 1996

Abstract

This paper presents a technique to provide seamless communication in mobile wireless networks. The goal of seamless communication is to make the motion of a mobile host transparent to the applications executing on it. Active handoffs (handoffs during an active connection) cause pauses in communication between the mobile host and the fixed network, resulting in applications seeing a substantial performance degradation. The motivation behind this study is to find a cost effective solution for minimizing impact of active handoffs on connection throughput. Existing solutions either provide total guarantee for seamless communication, incurring heavy network bandwidth usage (multicast based approach), or do not provide any guarantee for seamless communication (unicast based approach). Some other solutions are tuned to give good performance for specific protocols (e.g., fast-retransmit approach). This paper proposes a novel *staggered* multicast approach which provides a probabilistic guarantee for seamless communication independent of the communication protocol used. The main advantage of the staggered multicast approach is that it exploits the performance guarantees provided by the multicast approach and also provides the much required savings in the wired network bandwidth.

We present experimental results of performance improvements achieved by our scheme, for data transfer using TCP over a wireless network in the presence of active handoffs. The conclusions however, are generic in that they apply to protocols other than TCP.

[*]Research reported is supported in part by Texas Advanced Technology Program under grants 999903-029, and 009741-052-C

1 Introduction

Mobility has opened up new vistas of research in networking. With the availability of wireless interface cards, mobile users are no longer required to remain confined within a premises to get network access. Users of portable computers would like to carry their laptops with them whenever they move from one place to another and yet maintain transparent network access through the wireless link. Integrated voice, data and image applications are going to be used by millions of people often moving in very heavy urban traffic conditions.

A typical wireless network with mobile users is implemented using a wired network of hosts, some of which are augmented with wireless interfaces [9, 10, 11]. Such hosts are called *base stations* (*BS*)[1]. The base stations provide a gateway for communication between the wireless and wired network. Due to the limited range of wireless transreceivers, a mobile user can communicate with a *BS* only within a limited geographical region around it. This region is referred to as a base station's *cell*. Each *BS* is responsible for forwarding data between the mobile user in it's cell, and the wired network.

When a mobile host is engaged in a call or data transfer, it will frequently move out of the coverage area of the base station it is communicating with, and unless the call is passed on to another cell, it will be lost. Thus, the task of forwarding data between the wired network and the mobile user must be transferred to the new cell's base station. This process, known as *handoff*, is transparent to the mobile user. Handoff helps to maintain an end-to-end connectivity in the dynamically reconfigurable network topology.

Providing connection-oriented communication [12, 13, 14, 15, 16] to mobile users, requires that the user be always connected to the rest of the network in the presence of user mobility. Providing seamless communication [1, 8] is a stronger requirement than mere connection-oriented communication; in addition to maintaining the connection, the network needs to ensure that the performance does not degrade due to handoffs. Performance degradation happens because the base station previously serving the mobile host, drops all packets in its queue destined for the mobile host, and these packets have to be retransmitted from the source to the new base station. The delay tolerable can be quantified by a *quality of service* (QOS) parameter specified by the user. Note that simply forwarding data packets to the new base station does not provide any guarantees for seamless communication.

Transmission Control Protocol (TCP) [4] is the most popular reliable connection-oriented protocol in use in the internet today. Commonly used network applications such as *ftp, telnet, NFS, www-access* etc., rely on this protocol for their network communication. Fixed wired links typically provide fairly lossless data transfers, leaving TCP to worry about congestion control and avoidance in the network.

[1]Base stations are sometimes called *mobile support stations.*

Without going into details of the congestion control measures in TCP, we briefly introduce the general ideas behind these schemes. Two parameters of interest in this discussion are *congestion window* (*cwnd*), and slow-start-threshold (*ssthresh*) maintained by each TCP connection for use in flow-control. The value of *cwnd* fluctuates as new acknowledgements of previously sent data packets stream in. The maximum amount of unacknowledged data that TCP can have on the network at any time, is the minimum of the receiver's advertised window and *cwnd*. The parameter *ssthresh* is used to control the rate of growth of *cwnd* depending on the state of network congestion perceived by the source. If the TCP source percepts congestion on the network, it invokes congestion control measures [5] which result in the following series of events:

- The congestion window decreases thus limiting the amount of unacknowledged data on the network
- The connection goes into slow-start which throttles the rate at which the window can grow to previous levels
- The backoff interval of the retransmission timer is set to double with each consecutive timeout

While fixed wired links offer a virtually error free transmission medium (*Bit Error Rates* (BER) of the order of 10^{-8} to 10^{-12}), wireless links are much more unreliable. BER in such links is of the order of 10^{-2} to 10^{-6}, and they are highly sensitive to direction of propagation, multipath fading, and other interference [21]. Communication in such environments is much slower as compared to wired networks, as it requires more extensive error-correction mechanisms for reliable data transfer, and is also limited by device power requirements.

Maximizing throughput for bulk data transfer over **lossy** wireless links **without handoffs** is a separately studied issue. Various approaches have been proposed in [17, 18, 19, 22]. While each solution has been shown to improve performance, little investigation has gone into finding efficient solutions for seamless handoffs.

If applications using TCP were run in a mobile environment, losses during to active handoffs could cause these applications to perform poorly. This performance degradation is brought about because TCP misinterprets losses during handoffs as congestion. As a result, TCP invokes congestion control measures as listed above and throughput decreases. Our goal is to find a cost effective solution for minimizing impact of active handoffs on the performance of a connection. For this study then, we assume a lossless wireless medium, so that errors on wireless do not affect results for performance degradation due to handoffs.

The results presented in this paper are for bulk data transfer from a fixed host to a mobile host (*forward direction*). This is a more realistic scenario than transfer of data from the mobile host to the fixed host(*reverse direction*). The authors in [3] observed very little difference in the results obtained for data transfer in either direction for their scheme. We will thus use their results for comparing the effectiveness of our scheme for TCP even though the direction of data transfer in our case is opposite to theirs.

The remainder of this paper is organized as follows. Section 2 discusses related work, and Section 3 presents our proposed solution. Section 4 presents the implementation outline and the underlying assumptions used in our study. And finally Sections 5 and 6 present the results and conclusions of our study.

2 Related Literature

Keeton et.al. in [2] proposed a set of algorithms to provide connection oriented network services to mobile hosts for real time applications like multimedia. Their solutions lay excellent groundwork for research in this area but did not guarantee seamless communication. In fact their scheme was shown to suffer from extended intervals of time when service to the mobile host was disrupted. A study done in [1] shows that if the handoff protocol required forwarding data between the BSs connected by physical links, then a high bandwidth (between 48Mbps and 96Mbps) is required just to forward these data packets. Moreover, loops can be formed in the connection path if forwarding is employed. This will lead to inefficient network utilization.

Total Multicast Based Approach: A total multicast based solution was proposed in [1]. In this approach, the data packets for a mobile host are multicast to the BSs of the neighboring cells so that when the host moves to a new cell, there are data packets already waiting for it and thus, there is no break in service. It is evident, however, that this scheme is not cost effective. As the number of users in the network increases, the amount of network bandwidth used up by the multicast connections is going be prohibitively high. In [1], the cost of such a multicast scheme was determined to be the buffer overhead at the BSs. Our view of the problem is that the major component of cost incurred in a multicast based approach will be the *amount of extra bandwidth used*, and not the buffer overhead at each BS[2] This argument is supported by the availability of cheap memory but expensive network bandwidth.

Fast-Retransmit Approach: Caceres and Iftode in [3] present a *fast-retransmit* approach to reduce the effect of active handoffs on throughput for TCP connections. During handoff a mobile host sends a certain number of *duplicate* acknowledgments to the sender. This causes the source to immediately retransmit the lost datagrams and invoke congestion control measures. The net effect then, is to hasten TCP's response to a handoff. While this approach shows improvements

[2]Note that for TCP, the buffer requirement is anyway limited by the Maximum Window Size of the connection.

in throughput during active handoffs, it requires modification of the TCP protocol at the mobile host. Moreover, the scope of this approach is limited as it works only for TCP and may not be used for other protocols.

3 Proposed Approach

We now present the proposed approach for providing seamless communication to mobile users. Our work differs from existing protocols in that the network load incurred by the proposed approach is significantly lower as compared to others, while retaining the generality of it's application domain.

We define *cell latency* as the period for which a mobile host remains in a cell without handing off to another base station. Total multicast-based schemes result in wastage of network bandwidth (during cell latency periods) and the communication links get unnecessarily loaded. As the number of mobile hosts in a cell increases, the total network usage due to a total multicast connection for each host will become enormous. Due to this extra network usage, new connections might be blocked because the network capacity is exceeded.

On the other extreme, solutions using unicasting simply drop packets during handoff, expecting the source to retransmit them to the new base station. While forwarding approaches do not drop packets, they do not guarantee seamless communication either. A multicast based approach takes a vey conservative view of user mobility, essentially assuming a *zero* cell latency value, the unicast approach takes a completely opposite view - having the *source* retransmit all packets dropped during handoff. It would seem logical to choose a solution that exploits the advantages of both the multicast and unicast approaches.

Keeping the above observations in mind, we propose to:

- Use multicast only when necessary
- "Stagger" the multicast for a substantial part of the *cell latency*

3.1 Staggered Multicast

If it can be ascertained with some degree of confidence, that the mobile user will remain in the same cell for a certain period of time, multicast could be avoided till just before the mobile host handoffs to another base station. The mobile host will then get correctly sequenced packets as soon as it establishes contact with the new base station. The performance degradation brought about when the mobile host has to wait for the source host to retransmit packets (dropped at the previous base station) can thus be minimized.

If a handoff without staggered multicast causes packets to be dropped at the previous base station, a mobile receiver using TCP will generate duplicate acknowledgements for each out of sequence packet it receives from the next base station. These duplicate acknowledgements will in turn cause TCP at the source to invoke congestion control measures in addition to retransmitting the lost packets (as TCP attributes all dropped packets to congestion). This process is called fast-retransmit. Such an event may be viewed as a disruption in smooth operation of TCP. In the context of TCP, thus, we define a *disruption* in service as a invocation of congestion control measures at the source. We have assumed no packet losses, errors, or congestion on either the wired or wireless links, hence source timeouts/fast-retransmits can be attributed solely to packet losses during handoff.

Parameters: Let l_i be the cell latency of the mobile host before the *i-th* handoff. The value of $t_i \leq l_i$ gives us a measure of the stagger time than can be safely introduced before initiating a multicast. This way, we will save on the network usage, and still guarantee seamless communication with a high probability. We now present analysis for the parameters of interest our study.

Let P_i be the probability of disruption during the *i-th* handoff. If t_i denotes the time when multicast is initiated before the *i-th* handoff, then the time spent in multicast mode t_{mi} before the *i-th* handoff is given by

$$t_{mi} = l_i - t_i$$

A disruption will occur when a mobile host initiates a handoff before multicast has been initiated. The probability of disruption during the *i-th* handoff can be given as,

$$P_i = Pr[t_i > l_i]$$

Let the number of handoffs occurring over the length of the connection time T_c be N_h. Let $P_{disrupt}$ be the average probability of disruption during a handoff. $P_{disrupt}$ is determined as,

$$P_{disrupt} = \frac{1}{N_h} \sum_{i=1}^{N_h} P_i$$

The value of $P_{disrupt}$ may now be used as a measure of the Quality of Service (QOS) of this connection.

Figure 1 presents an example for total guarantee of seamless communication. The times B, D, F, and H represent the time at which handoff takes place. The times A, C, E, and G represent the time at which multicast is initiated. The cell latencies for Figure 1 are $l_1 = t_1 + t_{m1}$, $l_2 = t_2 + t_{m2}$, and so on. In this example, no disruptions occur and applications on the mobile host will not see a perceptible degradation in performance. For total guarantee of seamless communication, the following should hold.

$$\forall i, 1 \leq i \leq N_h, t_i < l_i$$

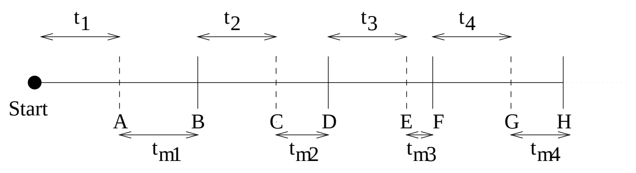

Figure 1: Total Guarantee

i.e., for all handoffs a multicast is initiated within the associated *cell latency* interval.

Applications like *ftp, web-browsing, etc.* do not have a strict requirement of disruption free service during every handoff. A probabilistic guarantee is sufficient for such applications, i.e., a non-zero (but small) $P_{disrupt}$ is acceptable. To illustrate this probabilistic scheme we present an example as shown in Figure 2. The times B, D, E, and G represent the time at which handoff takes place. The times A, C, and F represent the time at which the multicast is initiated. As noticed in the figure, there is a disruption in service during handoff at time E, because, there was no multicast initiated before the handoff. Thus, a disruption occurs during the *i-th* handoff when t_i is greater than the cell latency time l_i.

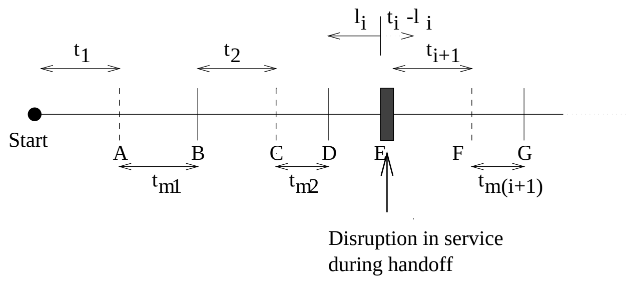

Figure 2: Probabilistic Guarantee

4 Implementation

Multicast Group Mapper: We define a multicast group g_i for mobile host m_i as the set of base stations that are included in the multicast operation for m_i. The members of g_i for m_i are determined by a *multicast group mapper (mgm)*, a persistent server process running on the wired network (Figure 3). It should be noted that the cell topologies of the wireless network are fixed and this information can be used by the *mgm* to decide membership of g_i. In the simplest case, g_i for m_i in any cell, contains the base stations serving all cells neighboring the current cell of m_i. More intelligent choices of g_i may be made based on user mobility patterns as well as its current location, speed, direction of motion, topological constraints [1] etc.

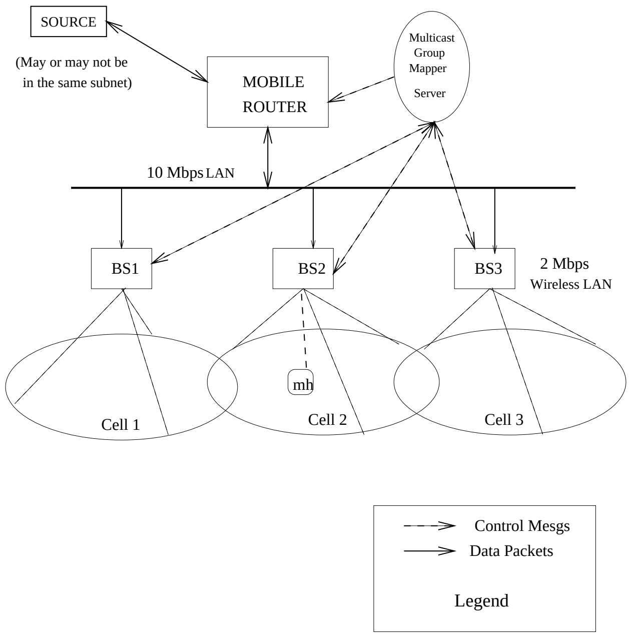

Figure 3: Network Configuration

User Profile: A *user-profile* is a record specific to and maintained individually by each mobile host m_i. It contains the following information.

- l_{avg}: The average value of cell latency over all handoffs (since connection start up) for m_i. Since, there is no prior knowledge of the user mobility, best we can do is estimate cell latency. The efficiency of the estimator will depend on the estimation algorithm used. We use a simple running average algorithm to estimate the average cell latency. If n_h is the number of handoffs of m_i since start of connection, then

$$l_{avg} = \frac{1}{n_h} \sum_{j=1}^{n_h} l_j$$

- $stagger$: The % of l_{avg} to be used for delaying multicast to all members of g_i. Depending on the value of $stagger$, we will get different values of $P_{disrupt}$. Figure 5 presents the variation in $P_{disrupt}$ with $stagger$ values. Let, t_i be the amount of stagger introduced before a multicast is initiated to all members of g_i in the *i-th* handoff interval. It is determined as,

$$t_i = stagger \times l_{avg}$$

4.1 Protocol and Message Flow

Please refer to Figure 4 for the following discussion on message flows in our scheme. The thick lines in Figure 4 represent the data packets being transferred over the wired network, and the thin lines represent the data packets being transferred over the wireless medium between the base station and the mobile host. The thick dashed lines represent the control messages being transferred over the wired network, and the thin dashed lines represent the control messages being transferred over the wireless medium. The *Mobile-Router* is responsible for routing all packets to and from the base stations attached to the fixed network (Figure 3).

At time t_0 let m_i be in the cell of $BS1$. At this time it initiates a connection to some other host via base station $BS1$ (step 1). During the connection set up phase, m_i transmits its *user-profile* to $BS1$. $BS1$ requests the *Mobile-Router* (step 2) to send it data packets destined for m_i, which are then transmitted to m_i over the wireless interface (step 4). $BS1$ also forwards the *user-profile* to the *mgm* (step 3) which decides a g_i for m_i in the cell of $BS1$ (consisting of $BS1$ and $BS2$), based on the wireless topology information it maintains. The *mgm* sets a timer to expire after t_i ($t_i = stagger \times l_{avg}$) [3]. On the expiration of this timer (time t_1), the *mgm* requests the *Mobile-Router* to initiate multicast of data packets to all members of g_i (step 5)[4]. The multicast continues till m_i completes it's handoff to $BS2$ (step 6 at time t_2). The handoff information passed on to $BS2$ includes the following.

- User-profile of m_i (which now includes a non-zero estimate of l_{avg}.)
- Highest sequence number n_i among data packets received by m_i

$BS2$ confirms this handoff request and starts transmitting data packets with sequence number greater than n_i to m_i over its wireless interface (step 7). $BS2$ then sends a *handoff-confirm* message to the *mgm* (step 8), which in turn sends *release* messages to all members of g_i (step 9) causing them to remove all packets for m_i from their queues. The *mgm* also asks the *Mobile-Router* to stop the multicast (time t_3) for m_i. The *mgm* again decides a new g_i for m_i (based on the id of the base station sending the *handoff-confirm* message), and sets a new timer to mark the beginning of multicast to members of the new g_i. Steps 5-9 are repeated till the connection is torn down.

Note that during multicast phases, no data is being transmitted over wireless links except from the base station currently serving the mobile host. All packets being received by the base stations in g_i (other than the current base station) are put in a FIFO queue of size equal to the maximum window size of the connection. It is possible that the mobile host stops in a cell for an extended period of time. This may cause multicast to never cease as no handoffs take place. This can easily be rectified by setting a different timer at the *mgm* to expire after the average cell-latency period l_{avg} (or any suitable multiple of it) if the multicast has not stopped.

[3]Note that at connection set up, in the absence of any previous cell latency information, $l_{avg} = 0$.

[4]An alternative could be to have a timer on each base station in g_i. Each of them may independently request a 'join' of the multicast group for m_i when their timer expires. In general, any reliable multicast protocol may be used.

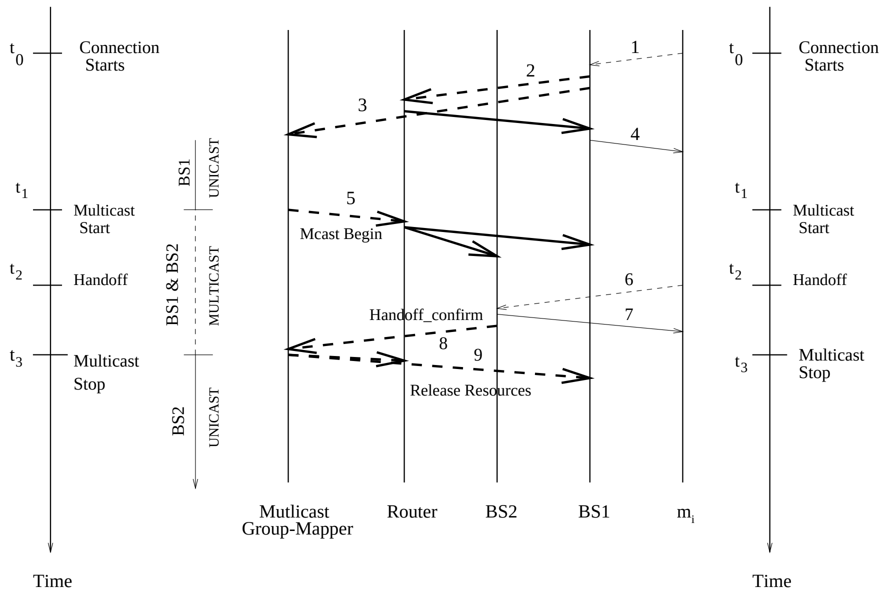

Figure 4: Protocol and Message Flow

4.2 Performance Comparison

The overhead of the staggered multicast scheme can be characterized by the total time T_m spent in the multicast mode as compared to the length of connection T_c. T_m is determined as

$$T_m = \sum_{i=1}^{N_h} t_{mi}$$

where, t_{mi} is the time spent in multicast mode before the *i-th* handoff, and N_h is the number of handoffs occurring over the length of connection. The total time spent in the unicast mode, T_u, is then given by the difference, $T_c - T_m$. We determine the overhead of the staggered multicast scheme as the fraction of the total connection time spent in the multicast mode,

$$Overhead = \frac{T_m}{T_c}$$

Note that for a total multicast solution [1],

$$P_{disrupt} = 0 \quad \& \quad Overhead = 1$$

While for a unicast solution,

$$P_{disrupt} = 1 \quad \& \quad Overhead = 0$$

It is clear that the performance of our approach is always lower bounded by the performance of the unicast approach, and upper bounded by the performance of the multicast approach. The cost incurred, however, is always a small fraction of the multicast approach. In addition, the solution may be used for any protocol (TCP or otherwise), and the protocol itself need not be modified for correct operation of this scheme. Another major advantage of this approach is that we do not require any assumptions about the overlap between adjacent cells in a wireless network. See Section 4.3 and Section 5 for more on cell overlap and its effect on results.

4.3 Simulation Environment

The performance of our scheme was evaluated using the Network Simulator (NS) from Lawrence Berkeley Labs with extensions incorporated to simulate handoffs and staggered multicast events. NS is an extensible simulation engine built using C++ and Tcl/Tk that can simulate various flavors of TCP available today for wired networks. TCP-Tahoe was used for the purposes of our simulation. For more details on NS refer to [23].

The network configuration used is shown in Figure 3. The wired segment is a 10Mbps LAN, connected to which are base stations equipped with 2Mbps wireless interface cards. 1536 bytes is chosen as the packet size over the wired as well as the wireless links. Maximum window Size was fixed at 64 Kbytes, and an end-to-end propagation delay of 10 ms was used. We assume that there are no losses over the wireless link (as we are only interested in studying the impact of mobility on performance).

Cell latencies are modeled as exponentially distributed variables with a mean of 10 seconds. One set of simulations consists of 250 runs, each executing for a 3000 second period. Values of *stagger* ranged from 0.25 to 0.98 for each 3000 second execution. The value 3000 seconds for execution of a single run allows a substantial number of handoff events to take place during each execution, while a mean of 10 seconds for cell latency allows the TCP connection to reach it's maximum throughput between handoffs (on average).

Cell Topologies: Each set of simulations was carried out for three possible cell topologies. In the first case, there is sufficient overlap between adjacent cells to allow the mobile host to remain in contact with both base stations during handoff. As a result there is no loss in communication between the mobile host and the fixed network. Such a cell topology is said to have a *blackout period* of 0 seconds. In the rest of the paper, we will refer to such a topology as *Cell Topology I.* In the second case, there is minimal (almost none) cell overlap between adjacent cells, such that the mobile host has no prior warning about which base station it is going to handoff to next. In this case the mobile host will not be accessible for data transfer for a short time during handoffs (assumed 25ms in this paper) We will refer to such a topology as *Cell Topology II.* Lastly, we consider scenarios where there is no overlap between adjacent cells and the mobile host completely loses contact with base stations on the fixed network for *blackout periods* of 1 second. This topology will be referred to as *Cell Topology III.*

While these topologies are by no means exhaustive, they do cover a wide spectrum of cell topologies. Our intent is to show that even though the staggered multicast approach is generic, it performs quite well for TCP when compared to specific approaches fine tuned just for TCP [3].

The *user-profile* transmitted by m_i to the base station during handoff, is used by the mgm to calculate t_i ($t_i = stagger \times l_{avg}$).

5 Results

Figures 5, 6 and 7 summarize the results of our experiments for the three cell topologies mentioned. The standard deviation in our results varies from 0.3% to 1.8%.

Figure 5 presents variation in $P_{disrupt}$ on the vertical axis with variation in $stagger$ shown on the horizontal axis. It is clear that as $stagger$ increases, the probability of disruption also increases. As an example for *Cell Topology II*, corresponding to a $stagger$ of 0.50, 31.6% of handoffs result in disruption. The value drops to 17.2% for $stagger = 0.25$.

Figure 6 illustrates the variation of average connection throughput with $stagger$. As can be observed in the figure, there is a 16%, 19% and 24% degradation in throughput if unicast scheme is used for cell topologies I, II and III respectively. The degradation is expected to be larger if the end-to-end propagation delay is reduced from the current value of 10ms. This is because in a high bandwidth environment, propagation delay is a major component in the end-to-end delay. This delay governs the round-trip time estimates as maintained by the TCP agent at the source. If the propagation delay is reduced, the performance of a TCP connection becomes more sensitive to delays during handoffs. The vertical axis in Figure 6 denotes the fraction of the maximum throughput achievable when staggered multicast is used as compared to when total multicast is used. As is expected, average achievable throughput decreases with increase in $stagger$. The improvements in throughput for TCP in [3], are similar to the improvements in average throughput obtained by our scheme for similar cell topologies.

One interesting observation in this result is that even though *Cell Topology I* has 0 seconds blackout period, the throughput achievable is not equal to the throughput of the total multicast scheme. This is because multicast might not have been initiated soon enough before a handoff, causing packets dropped at the old base station to be retransmitted from the source.

Figure 7 illustrates the variation of *average staggered multicast overhead* with $stagger$. Overhead is expressed as a fraction of the time when staggered multicast is used as compared to total multicast. For example, for *Cell Topology II* and for $stagger = 0.50$, the average multicast overhead is 60.5%.

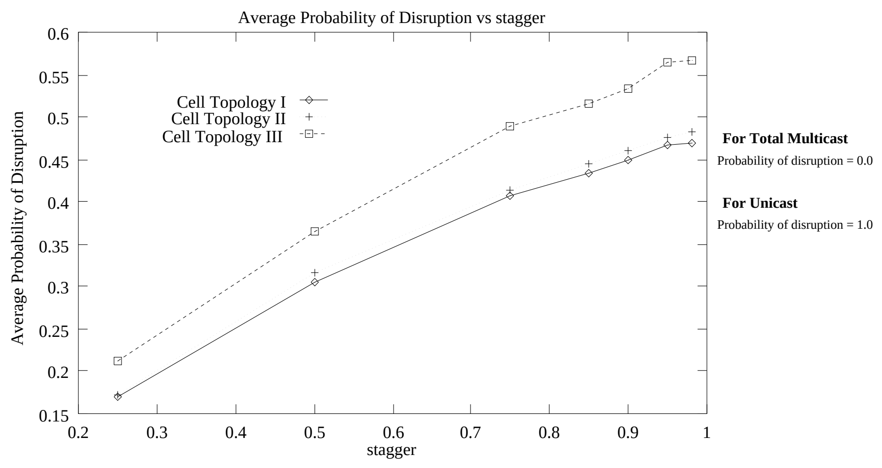

Figure 5: *Average Probability of Disruption vs stagger*

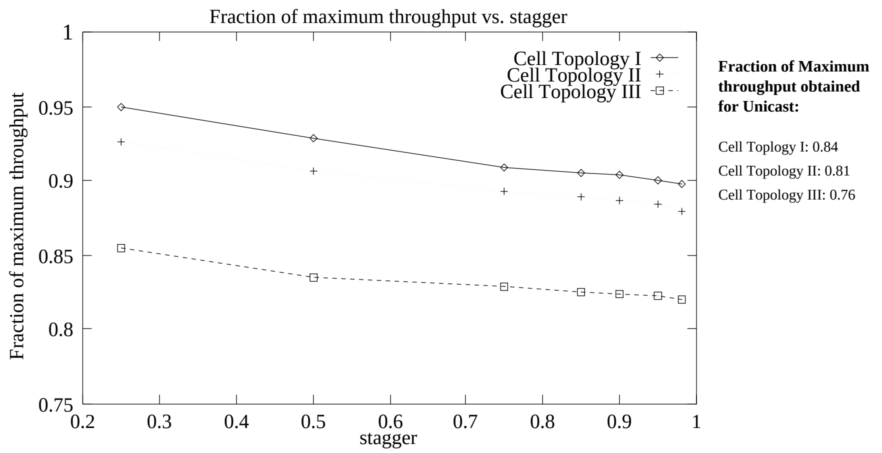

Figure 6: *Average Throughput vs stagger*

We note from Figure 6 that the average throughput obtained for $stagger = 0.25$ for *Cell Topology II*, is 92.6% of the maximum. Figure 7 shows that the average multicast overhead (for the same cell topology) for $stagger = 0.25$ is 76.4%. For $stagger = 0.50$ on the other hand, average throughput obtainable is 90.6% of the maximum, while the average overhead is only 60.5%. The curves for average throughput as well as average multicast overhead flatten as $stagger$ increases (for all cell topologies considered). Note that the average throughput obtained for any $stagger$ value is substantially higher than that obtained for the unicast scheme.

From Figure 7, we see that even for a $stagger = 0.98$, for *Cell Topology II*, the average overhead for staggered multicast is as high as 40%. This appears counter intuitive as one would relate a $stagger$ of 0.98 with an average multicasting overhead of around 2%. We would like to

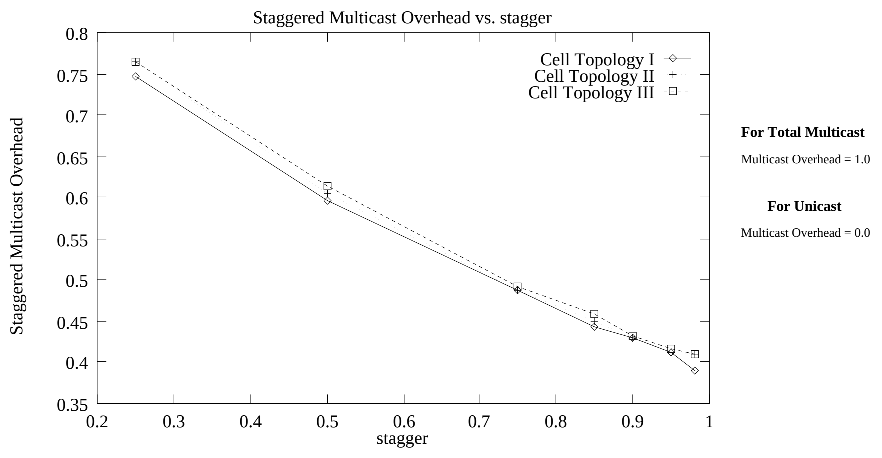

Figure 7: *Average Multicast Overhead vs stagger*

remind the reader, that the value of t_i ($t_i = stagger \times l_{avg}$) is calculated from the *running average* l_{avg} of the cell latency, and not the actual value of l_i (which would not be known a priori). Thus if l_{avg} is sufficiently different from from l_i for the *i-th* handoff, the % savings in multicast may not be as substantial as expected. Many ways for estimating l_i are possible. The choice of using a running average l_{avg}, was made only to keep the implementation simple. Even with such a straightforward method of calculating l_{avg}, we have been able to show the advantages of the staggered multicast approach. Indeed, there exists scope for improvement especially in the multicast overhead incurred. We are currently looking at ways to effectively model user mobility so that more accurate estimates of cell-latency. In the absence of empirical mobility data, however, our solution is a cost-effective way of providing seamless communication.

Lastly, we would like to point out that while larger values of *stagger* give lower obtainable throughput, they lower the multicast overhead too. The choice of an appropriate *stagger* value will have a direct impact on the load incurred by the network. Implementations of our scheme may also leave the decision of *stagger* value to the network itself depending on its load at the time of handoff. In this way, if the network load is very high, the network might decide on a higher *stagger* value even if the user QOS requirements demands a lower *stagger* value.

6 Conclusions

There are many user applications that do not require a "total" guarantee for seamless communication but would also not tolerate very poor performance on every handoff. A user will not want to pay a high cost for such applications. If a multicast based approach is used, the data packets will be multicast to the neighboring cells throughout the connection. This will be prohibitively expensive. On the other hand, if forwarding or unicasting is used, the user will see performance degradation during every handoff. Proposed in this paper is a novel staggered multicast approach which provides *probabilistic* guarantee for seamless communication. The staggered multicast approach partially provides the benefits of the multicast approach and also provides the much required savings in the wired network bandwidth.

The main **advantages** of the staggered multicast approach are summarized as below:

- The strategy is generic, in that it may be used for any protocol.
- The underlying protocol for reliable connection oriented data transfer (TCP in this paper) need not be modified
- No assumption about cell overlap is required. This approach will be very useful in environments where cell topology is not regular because of a large number of obstructions (e.g. walls and pillars within buildings, and larger objects outdoors). As a result, cell overlap cannot be guaranteed in such environments, and areas where signals cannot reach (called *dead zones*) may cause even longer blackout periods.
- The network bandwidth usage is significantly reduced when compared to a total multicast approach.
- A probabilistic guarantee for seamless communication is provided.

The main **disadvantages** of our approach are as follows.

- If the average cell-latency l_{avg} does not give a good estimate of cell latency in the current cell, then multicast overhead may be higher than expected, and throughput may be lower than achievable.
- The *Multicast Group Mapper Server* could become a bottleneck if the number of mobile hosts served by it is very large. Each mobile host handoff involves interaction with the server, and could result in larger handoff processing times. Having distributed servers, would be a way around this problem. We are currently looking at both efficient implementations, as well as techniques to prevent the Multicast Group Mapper from becoming a bottleneck in the system.
- Our solution can be used only when data transfer is taking place *to* a mobile host (and not *from* a mobile host).

References

[1] R. Ghai and S. Singh, "An Architecture and Communication Protocol for Picocellular Networks," *IEEE Personal Communications Magazine,* pp. 36-46, Vol.1(3), 1994.

[2] K. Keeton et.al., "Providing connection-oriented network services to mobile hosts," *Proc. of the USENIX Symposium on Mobile and Location-Independent Computing,* Cambridge, Massachusetts, August 1993.

[3] R. Caceres and L. Iftode, "Improving the Performance of Reliable Transport Protocols in Mobile Computing Environments," *IEEE JSAC Special Issue on Mobile Computing Networks,* 1994.

[4] RFC 793, "Transmission Control Protocol," September, 1981.

[5] V. Jacobson, "Congestion Avoidance and Control," *Proc. SIGCOMM,* pp. 314-329, Aug., 1988. (An updated version of this paper is available by anonymous FTP from ftp.ee.lbl.gov:congavoid.ps.Z.)

[6] P. Krishna, N. H. Vaidya and D. K. Pradhan, "Location Management in Distributed Mobile Environments," *Proc. of the Third Intl. Conf. on Parallel and Distributed Information Systems,* pp. 81-89, Sep. 1994.

[7] P. Krishna, N. H. Vaidya and D. K. Pradhan, "Efficient Location Management in Mobile Wireless Networks," Technical Report, Dept. of Computer Science, Texas A&M University, Feb., 1995.

[8] Bikram S. Bakshi, P. Krishna, N. H. Vaidya and D. K. Pradhan, "Providing Seamless Communication in Mobile Wireless Networks," Technical report TR-95-046, Dept. of Computer Science, Texas A&M University, Dec., 1995.

[9] W. C. Y. Lee, *Mobile Cellular Communications Systems,* McGraw Hill, 1989.

[10] D. M. Balston and R. C. V. Macario, *Cellular Radio Systems,* Artech House, 1994.

[11] S. Mohan and R. Jain, "Two User Location Strategies for Personal Communication Services," *IEEE Personal Communications,* Vol. 1, No. 1, 1994.

[12] Pravin Bhagwat and Charles. E. Perkins, "A Mobile Networking System based on Internet Protocol (IP)," *Proc. of the USENIX Symposium on Mobile and Location-Independent Computing,* Cambridge, Massachusetts, August 1993.

[13] J. Ioannidis et. al., "IP-based Protocols for Mobile Internetworking," *Proc. of ACM SIGCOMM,* 1991.

[14] J. Ioannidis and G. Q. Maguire Jr., "The Design and Implementation of a Mobile Internetworking," *Proc. of Winter USENIX,* Jan. 1993.

[15] Charles Perkins, "Providing Continuous Network Access to Mobile Hosts Using TCP/IP," *Joint European Networking Conference,* May 1993.

[16] F. Teraoka, Y. Yokote and M. Tokoro, "A Network Architecture Providing Host Migration Transparency," *Proc. ACM SIGCOMM Symposium on Communication, Architectures and Protocols,* 1991.

[17] Bikram S. Bakshi, P. Krishna, N. H. Vaidya and D. K. Pradhan, "Performance of TCP over Wireless Networks," Technical report TR-95-049, Dept. of Computer Science, Texas A&M University, Dec., 1995.

[18] H. Balakrishnan, S. Seshan, E. Amir, R. H. Katz, “Improving TCP/IP Performance over Wireless Networks,” *Proc. 1st ACM Conf. on Mobile Computing and Networking,* November 1995.

[19] A.Bakre and B.R. Badrinath, “I-TCP: Indirect TCP for Mobile Hosts,” *ICDCS*, 1995.

[20] A.Bakre and B.R. Badrinath, “Handoff and System Support for Indirect TCP/IP,” *Proc. Second Usenix Symp. on Mobile and Location-Independent Computing,* April, 1995.

[21] K Pahlavan, A. H. Levesque, “Wireless Information Networks,” Wiley-Interscience, 1995.

[22] P. Bhagwat et. al., “Enhancing Throughput over Wireless LANs Using Channel State Dependent Packet Scheduling,” *INFOCOM*, April, 1996.

[23] Sally Floyd, Steve McCanne, ”Network Simulator.” LBNL public domain software. Available via from ftp://ftp.ee.lbl.gov.