# A Logical Ring Reliable Multicast Protocol for Mobile Nodes

Ioanis Nikolaidis* Janelle J. Harms
Department of Computing Science
University of Alberta
Edmonton, Alberta
Canada T6G 2H1
{yannis,harms}@cs.ualberta.ca

## Abstract

*A protocol for the reliable multicast of data to mobile hosts is described. A logical ring is maintained between all the base stations that handle the multicast traffic of the same multicast group. A token passing protocol enforces a consistent view between all base stations with respect to the frames that are considered delivered to all mobiles. The interaction of the reliable multicasting and the handoff events of the mobiles is controlled by a special handoff protocol. A description of the management protocol is also given. We use simulation results to illustrate the feasibility of the approach and reveal the performance interplay between the base station buffer size and the token rotation frequency.*

## 1. Introduction

Multicasting is a well–known complicated problem for which no single universally suitable solution is known (for a recent extensive survey see [1]). Reliable multicasting is also a complex problem. The actual degree of required reliability varies with the application depending on the real–time nature of the traffic and the extent to which data loss can be tolerated. The complexity for reliable multicasting is mainly due to the simultaneous requirement for efficiency and scalability. To provide reliability, a solution to the ACK (and feedback in general) implosion is necessary. Additional complexity stems from the join and leave dynamics of the receivers.

The addition of mobility compounds the complexity of the reliable multicasting problem. Wireless links, supporting mobile hosts, tend to be of low bandwidth and high error rate. The mobile hosts are also limited in terms of power and memory. Locating and routing towards mobile hosts is in fact a separate problem in its own right. Mobile-IP [2] has been proposed for routing support of transparent best effort delivery to mobile hosts. The combination of unreliable multicast datagram delivery to mobile nodes has been addressed in the past as well [3, 4].

It is reasonable to assume that the receivers of a given multicast group can be a combination of fixed and mobile hosts. The problem considered in this paper is that of reliable multicast for the mobile nodes alone. The proposed protocol provides *exactly–once* semantics for streamed data. Mobile hosts that join a group late receive the frames from the time that they are added to the multicast group (not from the beginning of the session). This form of multicasting is valuable for applications where the receivers must, by definition, be able to receive continuously and reliably a stream of data, as in the case of stock quotes delivery in stock exchange trading floors, or for delivery of highly compressed video traffic (where data losses can create perceptible artifacts) or for any other streamed *push* application.

Few researchers have looked at providing reliable multicast service in networks with mobile hosts [5, 6]. In [5], emphasis is given to the power and memory limitations of mobiles. ACKs are collected by the base stations and sent back to the source. In [6], a three–tiered scheme is proposed where "supervisory" hosts collect ACKs and forward them to the source. Explicit "delete" messages from the source trigger the deallocation of frames at the supervisory hosts or the base stations. Clearly, the involvement of the source comes at the cost of state information storage, processing costs and state information exchanges. A form of feedback aggregation is a necessity for reliable multicasting. However, it is not essential that the mobility of nodes be exposed, and accounted for through specialized protocols at the source or, for that matter, at any other node within the *fixed* part of the network. Instead, it is advantageous for the protocol to allow the integration of fixed and mobile recipients in a manner which is transparent to the source and any other node that belongs to the fixed part of

*This works was supported in part by NSERC under grant OGP0194424. The simulations were developed by Jian Xu.

the network. That is, handling the reliable mobile multicast (with its special needs for consistency during handoffs and location–dependent channel outages) ought to be dealt with by the mobile–aware components of the network, i.e., by the mobiles and base stations alone.

We propose a protocol which is different from the ones presented in [5, 6] in that it imposes no restrictions on the particular mechanisms used for reliable multicasting within the fixed part of the network. The base stations coordinate their actions regarding the mobile part of the network through a logical ring token passing protocol. The decoupling of the dynamics of the mobile part from those of the fixed part is enhanced with the inclusion of buffer space provided by the base stations. Finally, as it will become clear, the necessary state information and control message complexity is small.

The remainder of this paper is structured as follows: Section 2 presents the protocol in detail. In particular, 2.1 presents the token rotation protocol, 2.2 presents the handoff protocol and 2.3 the management protocol (which handles the join/leave requests of mobile and/or base stations). Section 3 presents a simulation study of the protocol. After an extensive presentation of the simulation setup, section 3.1 introduces the throughput results under several study configurations. The interplay of the protocol parameters is also explained. Section 4 summarizes the properties of the protocol and indicates areas of future research.

## 2. The Protocol

The protocol provides the mechanism for reliable delivery of frames of a multicast group from a set of base stations (BS nodes) to a set of mobile nodes (M nodes). At any point in time, an M node is associated with a single BS node. M nodes can freely move between BS nodes, thus causing handoffs. The description of the protocol includes neither a description of a reliable multicasting protocol for the fixed network nor a particular reliable data link protocol for the wireless/mobile part. Several proposals already exist for both fixed network reliable multicasting and reliable data link protocols. We therefore consider it redundant to propose new protocols for them. Instead, we assume that each BS node operates as a normal fixed network reliable multicast node and uses an arbitrary reliable data link protocol to communicate with the M nodes. In this sense, it is assumed that a BS node will convey acknowledgements and flow control information to any nodes (intermediate or the root of the multicast tree) subject to the specific reliable multicast protocol used over the fixed network.

A BS node buffers multicast frames intended for delivery to its M nodes. M nodes maintain a structure describing the unsuccessfully received frames, as gaps in the sequence of correctly received frames. BS nodes maintain a similar structure for the unsuccessfully forwarded frames to *each* of the M nodes. Figure 1 illustrates an example data structure for a BS node handling five M nodes. All frames from 351 to 356 have been delivered to all its M nodes but are not discarded yet, in anticipation of other M nodes handing–off to this BS node from other BS nodes. An M node entering a BS through the handoff process may request the (re–)transmission of "fossilized" frames.

Clearly, the view of BS nodes may differ from that of M nodes in terms of delivered frames, because of corrupt data and ACK transmissions. At all times, the frames delivered at an M node (from M's point of view) is a superset of the frames perceived as delivered by the corresponding BS. When handing off to a new BS, an M node informs the new BS node about the undelivered frames. Subsequently, the new BS starts maintaining the structure for the undelivered frames to this mobile. Eventually, as we will see in the following, the new BS will inform the old BS that it should dissolve the entry that it maintained regarding the undelivered frames to the M node.

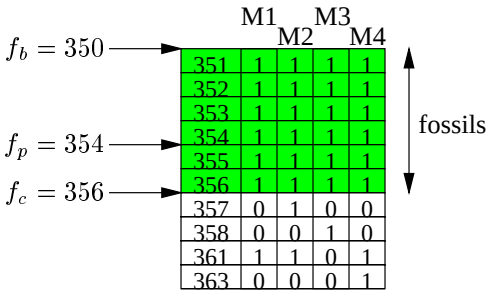


**Figure 1. Example BS node structure.**

The reliable delivery of streamed data to M nodes must ensure that a frame not received by an M node due to error or to a handoff will be eventually received, unless the M node shuts down or opts out of the multicast group. While link error losses can be handled through the reliable data link protocols, the solution is not sufficient to deal with handoffs. For example, assume that $M$ hands off from BS $B1$ to BS $B2$. At the time of the handoff, $B1$ was still in the process of trying to send reliably to $M$ the frames with sequence numbers $N, N+1, \cdots, N+k$ for a certain $k$. That is, $M$ has received all frames with sequence number up to $N-1$. Let $B2$, at the same point in time, be trying to send frames $N+2, N+3, \cdots, N+m$ to its own set of M nodes. To ensure that $M$ can receive $N$ and $N+1$ from $B2$, $B2$ must not have released the buffered frames $N$ and $N+1$ even if they were successfully delivered to its own M nodes. That is, a BS node must anticipate that an M node handing off from elsewhere may request frames that the BS node considers successfully delivered to its own set of attached M nodes just prior to the handoff. Thus, even if delivered reliably to its own set of M nodes, a BS must

not deallocate the frames. Therefore, it is necessary to introduce a protocol that allows the eventual release of the buffer space. This protocol will continuously inform BS nodes as to which successfully delivered frames should no longer be stored.

The problem described in the previous paragraph is not uncommon in wireless environments where location–dependent fading phenomena could cause $B1$ to be "left behind" in forwarding reliably the frames to its mobiles. A nearby base station, not having experienced similar losses due to fading is "ahead" in the sequence number range of the frames it forwards. Clearly $B2$ alone cannot anticipate how far in the past it should store the sequence of frames that have been reliably received by its own set of M nodes. To this end, it needs to communicate with other BS nodes to confirm what is the lowest sequence number below which all frames are to be considered as globally delivered to *all* M nodes for the specific multicast group. In the next section, we describe how this can be accomplished by a token passing protocol that traverses a logical ring consisting of the BS nodes and conveys information used to enforce a consistent global view as to which is the maximum sequence number below which all frames can be considered as delivered to all M nodes (regardless of the BS node they are attached to).

The careful reader will notice the similarity to the Global Virtual Time (GVT) calculation (needed for the so-called "fossil collection") in distributed simulation, and in particular in the context of optimistic parallel simulation, as in Time Warp [7]. In Time Warp's case, we seek to compute the maximum local virtual time over all parallel processes below which no event is ever going to roll back. Similarly, in the presented reliable multicast protocol, a maximum sequence number below which no frame is going to ever be (re)transmitted is calculated. As the GVT advances, memory, that was used for snapshots of the simulation variables at previous time instants, is released. Similarly, in the proposed protocol, the buffer taken in BS nodes by delivered frames is released. The two key differences between the reliable multicast protocol and the GVT calculation are (a) the GVT calculation is typically performed between tightly coupled processors, where communication latencies are small and (b) in parallel simulation the re-generation of the events based on a snapshot of earlier simulation state is possible, allowing state saving to be selective, while data frames need to be all stored due to the inability to generate future frames based on the information of earlier ones. To indicate the similarity to GVT calculation algorithms we will call the acknowledged frames that are stored in anticipation of future handoffs the "fossil" frames (see Figure 1)

// $f_p, f_c$ and $f_b$ are per-group variables of the BS.
// $t_{hold}$ is per-group time parameter.
// $s_p$ and $s_c$ are fields of the rotating token.
// Initially: $f_p = 0, f_c = 0, s_p = 0, s_c = \infty$.

*upon_token_arrival*:

1. **wait($t_{hold}$);**
2. **if** $s_c = f_p$ **then**
3. $s_p \leftarrow s_c$;
4. $s_c \leftarrow f_c$;
5. **else**
6. $s_c \leftarrow \min\{s_c, f_c\}$;
7. **endif**
8. $f_p \leftarrow f_c$;
9. $f_b \leftarrow s_p$;
10. **discard_fossils($f_b$);**
11. **pass_token_to_successor();**

**Figure 2. The BS node token handler.**

## 2.1. The Token Protocol

A token is maintained for each multicast group and it includes two fields, $s_c$ and $s_p$. The token is transported through the fixed part of the network. Upon arrival to a BS node, the algorithm of Figure 2 is executed. The $f_c$ maintains the maximum sequence number acknowledged by all mobiles attached to the current BS node. Once the token is received, the BS node updates the $s_c$ field of the token (line 6, Figure 2) such that, once it rotates through all the BS nodes, the value of $s_c$ reflects the maximum sequence number that has been acknowledged by all mobiles at all the BS nodes. In the next rotation of the token, all stations can be informed about this maximum and will copy it in their $f_b$ variable (line 9, Figure 2, and also Figure 1). At the same time, they will discard all "fossil" frames with sequence numbers less than or equal to $f_b$.

During the rotation, the calculation of the new minumum globally delivered sequence number is overlapped with the announcement of the previous globally delivered sequence number (calculated in previous rotation(s)). The overlap is accomplished by using the second field, $s_p$, which is the previously calculated sequence number. Once a rotation completes, $s_c$ is moved to $s_p$. To allow a completely distributed token rotation protocol, the BS node that moves $s_c$ to $s_p$ is not a specially designated BS but the BS that had assigned the minimum value of $s_c$ in the previous round. Thus, each BS node, maintains also, $f_p$ which is the memory of the sequence number used by the BS node in the $s_c$ calclation of the token in the previous round. If the rotating

token is found with $s_c$ equal to $f_p$ (line 2, Figure 2), then the BS node identifies itself as the BS node that had the minimum of all $f_c$ in the previous round. Subsequently, it moves $s_c$ to $s_p$ (lines 3 & 4, Figure 2).

The $f_c$ of each BS advances as more frames get acknowledged by the mobiles. Thus, the maximum sequence number calculated over all BS nodes continuously advances. That is, assume that $s_p$ was a certain value in the most recent rotation. In the next rotation it is going to be larger than its previous value because it is the minimum of a set of numbers (the $f_c$ over all stations) which are larger than or equal to the numbers used to calculate their minimum in the previous rotation. Hence, as long as the $f_c$ values advance, so does $s_p$ and more fossilized frames can be discarded.

A property of the presented protocol which is used in the handoff processing (in the next section) is that the value of $f_c$ increases as frames are acknowledged from the M nodes, but the value of $f_c$ is only "sampled" on token visit instants. Thus, between visits of the token, it is conceivable that $f_c$ can decrease as long as it does not decrease to a value lower than $f_p$.

The BS nodes possess a maximum buffer space, for the sake of the frames forwarded to the mobiles subscribed to a multicast group. The buffer includes fossilized frames. In the event that the buffer space is exhausted, the BS node must use the flow control features of the fixed network reliable multicasting in order to slow down (or even stop) forwarding frames until buffer space becomes available. Thus, a crucial parameter is the buffer space used on a per-multicast-group basis for buffering frames (including the fossilized ones). The BS node buffer space interacts with the token rotation. The more often the token rotation, the more frequent the chance for discarding frames, the less likely that a flow control "slowdown" will be needed, and the less the demand for a large BS node buffer. At the same time, we note that there is no reason for the token rotation to be very frequent. In each rotation, a certain number of BS nodes are able to reclaim some space used by fossilized frames. A very frequent token rotation in a network with small propagation and queueing delays does not improve the performance, because the actual gains (discarded fossils) is small or non-existent per token rotation.

Finally, the number of frames stored in the BS node buffers depends on the traffic volume of the particular multicast group. Thus, overall the protocol performance is dependent on the traffic load pattern, the token rotation time, the buffer size of BS nodes and the rate of M node handoffs. To enable the control, to an extent, of the protocol performance, a token holding time parameter, $t_{hold}$, is introduced for each multicast group. It indicates the amount of time that a BS node is entitled to hold the token per token visit, before it updates its content and passes it to its successor.

## 2.2. The Handoff Protocol

The token protocol guarantees that a BS node does not deallocate frames that may be needed by mobiles that handoff from other BS nodes. That is, a "global" window lower edge is enforced and is captured in the $s_p$ field of the token (being the minimum of the $f_b$ values of all BS nodes). However, the token protocol alone is insufficient to ensure the correct operation of the reliable multicast protocol. The handoffs and the token rotation are asynchronous events that can lead to a particular race condition. In the next paragraphs, we describe the race condition and the handoff protocol that is necessary in order to avoid it.

Consider the example of a mobile, $M$, which expects the frame with sequence number 50 while the rest of the mobiles under the same BS, $B1$, are waiting for sequence number 55 (or higher). The $f_c$ of $B1$ is therefore 49. $M$ hands off to a new BS, $B2$. Assume that the $f_c$ of $B2$ was 60 just before the handoff (and let us assume that its $f_p$ is 42). What should be the $f_c$ of $B2$ now, and should the handoff complete (thus allowing $B1$ to release all information regarding $M$)? It turns out that the correct action depends on several conditions. First, note the following sequence of events: before the beginning of the handoff of $M$ to $B2$, the token has reached $M2$ and therefore $s_c \leq 60$. Assume that before the token reaches $B1$, the handoff is completed and no information regarding $M$ is now available at $B1$. Assuming all the other mobiles of $B1$ have not advanced, the arrival of the token to $B1$ will result in $s_c \leq 55$. When eventually the token reaches $B2$ again without being forced by any other BS to reduce its $s_c$ to 50 (or less) it is possible that $s_p = 55$. Thus, $B2$ will be informed to discard sequence numbers $55$ and less even though it may have to send frames in the range $50$ to $55$ (inclusive) to $M$ which now "belongs" to $B2$.

The root of the problem is that a BS announces its $f_c$ by updating the token and subsequently setting $f_p = f_c$. Before the token completes a rotation back to the BS, a handoff event for a mobile $M$ forces the BS node to roll back to a sequence number less than the $f_c$ it had announced. In hindsight, the BS should not have announced $f_c$ as its lower edge. But it is not possible to know in advance which mobile will handoff to the BS that will cause a rollback. On the other end, if $B1$ erases any local record of $M$ assuming the handoff completed, there exists no information in either the token or any of the BS nodes that captures the lower edge in the sequence number window that $M$ awaits. Study of this problem indicates that to fix it, one approach is to disturb the token rotation algorithm and inhibit token rotations (and possibly reinitialize the token) in the event of handoffs and a second is to use more fields in the token (conveying information about all mobiles). The first approach unnecessarilly stalls the BS stations from advancing their windows

while the second requires large and variable sized tokens.

The solution provided herein does not modify the token structure and algorithm. Instead, the handoff is not completed (thus, $B1$ is not allowed to release the information about $M$) before a mobile catches up with $f_p$ of the new BS node, $B2$. While $M$ is catching up with $f_p$, $f_c = f_p$, that is, $f_c$ remains frozen at $f_p$. Since $B1$ is maintaining $M$'s information, its $f_c$ cannot increase while the handoff is still in progress. For example, assume that a mobile which awaits delivery of frames with sequence number greater than or equal to $k$. If $k$ falls between $f_p$ and $f_c$ of $B2$, then $M$'s handoff can be accommodated by setting $f_c = k$ in $B2$ and considering the handoff completed (thus notifying $B1$ to delete its entry of $M$). The complication only appears when $k$ is less than or equal to $f_p$ of $B2$, since $f_p$ represents a "commitment" taken by $B2$ (by using it to update the token in the last rotation).

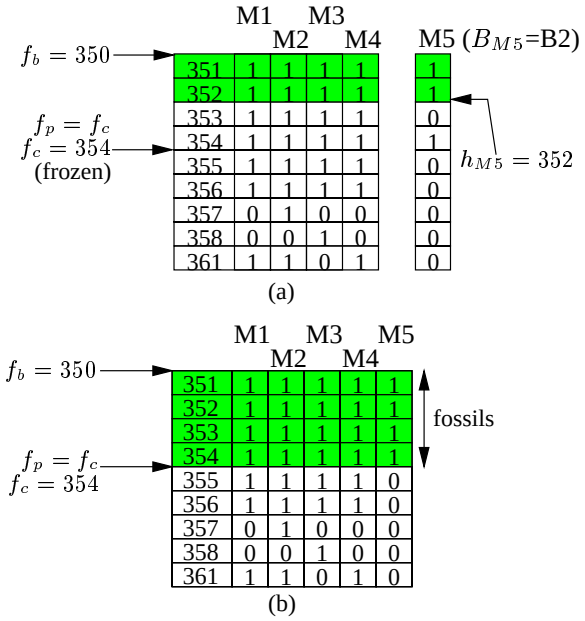


**Figure 3. BS node state during, (a), and upon completion, (b), of a handoff.**

To generalize the protocol, a BS node needs to maintain two variables (apart from the bitmap for the undelivered frames) for each M node which is in the process (but has not completed) a handoff to this particular BS node. The first, $h_i$, (where $i$ identifies the M node) is the M node's maximum sequence number for already received frames. The second, $B_i$, is the previous BS node where the M node was located and which has to be informed for the handoff to complete. While any of the $h_i$'s is less than $f_p$, $f_c$ is frozen at $f_p$. When a $h_i$ first exceeds $f_p$, it informs its old BS, $B_i$ to erase the record it maintains for this particular mobile, and, hence, to complete the handoff. From this moment on, mobile $i$ is operating the same way as all the other mobiles attached to this BS node. For example, if the original state of the BS node was that of Figure 1, and a mobile, $M5$, has just started its handoff process to this BS node, then the state of the BS node will now be that of Figure 3.a. Figure 3.b illustrates what happens immediately after the successful transmission of the frame with sequence number 353 to $M5$, at which point the handoff of $M5$ can be completed.

In the event of a handoff of an M node, $M$, from a BS node $B2$ to a BS node $B3$ occurring while a handoff from $B1$ to $B2$ is not yet completed (in the sense of $h_i < f_p$ at $B2$), all that is necessary is the transfer of the information about the mobile from $B2$ to $B3$. $B3$ is informed by $B2$ which base ($B1$) it should notify when the handoff is completed. In this way, $B1$ still acts as the base station that inhibits advancing the sequence numbers of the frames received by $M$ while the mobile is in the process of handing off to new BS nodes.

The delayed completion of a handoff at the old BS node until the mobile catches up with the frames, that are declared as fossils in its new BS node, results in a small reduction of the throughput. However. as we will see, its impact is very minor given that handoffs that are not completed quickly (and where $k \leq f_p$) are rare and even less likely to occur when the wireless channel exhibits small frame error probability.

### 2.3. The Management Protocol

The management operations are separated into M node and BS node management. M nodes declare their interest to join a multicast group, to "connect" to the group, assuming that the BS nodes are already members of the group (otherwise, the BS node must first become a member of the multicast group, which is handled by the BS node management). The BS starts sending to the mobile all frames starting with sequence number $f_c$, thus implicitly assuming that all frames starting from $f_c$ should be delivered to the new mobile and, similarly, that all frames with sequence number less than $f_c$ can be considered as non-essential since the connection of the mobile started at a later point in time.

The connection termination is more complicated, because, apart from a termination explicitly requested by the M node, an M node may shut down unexpectedly or be in the process of a hand–off and thus appear unreachable. To solve this problem, an inactivity timer is introduced which forces the M node to emit *keep–alive* messages. If the keep–alive message is not received, the connection is dropped unilaterally from the BS node, erasing the information it maintained about the M node. The keep–alive is set such that it is larger than the longest anticipated latency before the indication of a handoff in progress is received from another BS node.

The BS node management includes the group membership protocol to the fixed network reliable multicasting (which is outside the scope of the paper) and the management with respect to the token and handoff protocol. When a BS node wishes to join a reliable multicast group for handling traffic towards M nodes, apart from becoming a leaf in the multicast of the fixed part of the network, it also joins the token rotation protocol and starts buffering the multicast traffic frames. Initially, the BS node participates passively in the token rotation, that is, it does not update the token fields but just inspects them. When the token is received and all the frames in the range from $s_p$ to $s_c$ are present in the BS node buffer, then the BS node becomes an active participant of the token-protocol. This ensures that the BS node will not be asked to retransmit old frames that it never received anyway. Finally, to terminate its participation to the multicast group, a BS node must first wait until no M nodes are attached to it and to subsequently request its predecessor to forward the token in the future to the BS node's successor.

## 3. Simulation Study

The objective of the simulation study is to illustrate the detrimental effects that the token protocol and the handoff protocol have on the throughput of the system, subject to particular mobility and channel outage characteristics. We consider a random network topology on which a multicast tree is built as shown on Figure 4. The propagation delay is set to two thirds of the speed of light. A total of 9 BS nodes and 45 M nodes are simulated. Upon initialization, 5 nodes are assigned to each BS. Subsequently, the M nodes hand off between BS nodes by picking the next BS node in a uniformly random manner. The time between handoffs is exponentially distributed with a certain inter-handoff mean parameter. The simulations continue until we observe that the equilibrium has been reached.

We use a simplified outage model to capture the effects of mobility patterns that include the impairment due to location–dependent fading phenomena. Outage periods alternate with clear periods. Any transmission during a channel outage is corrupted and considered lost. The forward and backward channels (to and from an M node) are independent. The channels for each mobile are also independent. Outage and clear periods are distributed exponentially. The outage model is parameterized by the mean outage period and the fraction of time that the channel is in an outage state.

The reliable multicasting over the fixed part of the network is accomplished by a hop–by–hop version of selective repeat which has appropriately selected window size to maximize the throughput. Because no errors are assumed in the fixed part of the network, its most important influence is in the flow control. As the buffers of the BS nodes gradually fill up, the intermediate nodes in the multicast tree use their flow control to slow further the upstream nodes, up to, possibly, the root of the tree.

Due to the complex interaction of the topology, propagation delays and protocols (fixed part multicasting, fixed part datalink, mobile part multicasting, mobile part datalink) the study calibrates all other protocols to reach a 100% throughput under the assumption of no errors in the wireless channel. Subsequently, the proposed protocol is introduced as well as the outage periods of the wireless channel, and hence any difference from the 100% throughput is purely because of the reliable mobile multicasting under the chanel state model. Note that the simulations are performed using a greedy traffic source. All the fixed network links are 10 Mbps, and all frames are 1 kilobyte. The wireless link speed is 10 Mbps in the forward direction (from BS node to M node). In the backward direction (M node to BS node) the channel is assumed to be of sufficient capacity and multiplexed in a time-division manner such that all the M nodes can send ACKs to the BS node without loss of efficiency due to collisions.

### 3.1. Performance Results

The metric of interest is the efficiency $\eta$ expressed as percentage of the achieved throughput over the link rate. All links have the same speed and no interfering traffic is simulated. All BS nodes use the same $t_{hold}$. In the plots, the total $\sum t_{hold}$ is displayed. The performance was found to be dependent on $\sum t_{hold}$ and not the individual values of $t_{hold}$.

First, we determine the efficiency subject to the channel state model since it differs from classical bit/frame-error models. The BS buffer size is 1000 frames and $\sum t_{hold} = 0$. Figure 5.a provides the corresponding results. The inter-handoff was one second. As can be seen, a channel which is in outage 5 % of the time, results in a maximum anticipated throughput of 60 %, similarly a 10 and 20 % of the time outage corresponds to maximum throughput of less than 40 % and less than 20 % respectively. The actual duration of the outage makes essentially no difference when we assume greedy traffic, as the flat lines suggest.

In the second set of experiments (Figure 5.b), the circulation of the token is slowed down ($\sum t_{hold}$ = 0.2 and 0.4 seconds). The efficiency deteriorates as expected but a recovery of the efficiency to its maximum 60 % (as observed in Figure 5.a for outage 5 %) is still possible by increasing the buffer size. The result is a clear demonstration of the performance interplay between $t_{hold}$ and the BS node buffer size. What is worth pointing out is that for different $t_{hold}$, the buffer size above which the efficiency is maximized depends to a certain extend on $t_{hold}$. Note that the required

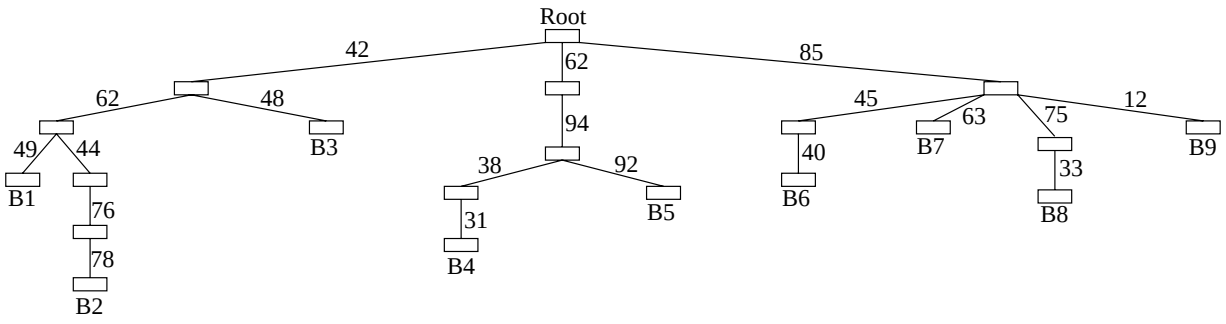


**Figure 4. The multicast tree of the simulation topology annotated with link distances (in km).**

buffer is a worst case example, because of the greedy traffic source. The necessary buffer size to maximize the efficiency is lower for smaller traffic demands. Finally, we note that the increase of $t_{hold}$ can also emulate the impact on the efficiency of larger network topologies and larger latencies between BS nodes.

The outage model attempts to capture the disruptions of the communication between mobile and base due to location–dependent channel fading and the mobility patterns. The model is only an educated guess and its particular parameters should not be taken literally. It is clear however, that a small outage percentage is sufficient to result in significant reduced efficiency (Figure 5.a). In the next set of experiments we consider as "reasonable" outage between 0 and 5 %, with any larger value being overly high for real efficient use. The question we wish to answer is, given a buffer dimensioning at the BS nodes that is known to maximize the throughput under no channel impairment (1000 in the case of the experiments) and uncertainty about the outage model parameters (between 0 and 5 %) what is the range of throughput that should be expected and does it depend on any other protocol parameter? The answer is given by Figure 5.c which indicates that for increasing token holding time, the anticipated difference in the throughput is decreasing. Another way to describe the same results is to state that for small token holding times, the outage model dominates the performance. At high token holding times, the increased token holding time still reduces the performance but the uncertainty about the parameters of the outage model has a lesser impact.

Finally, it is intuitively appealing to consider the inter-handoff times as having no impact on throughput. The reason for this lack of impact is the fact that the handoffs occur less frequently compared to frame transmissions, channel errors and even token rotation (depending on $t_{hold}$). Moreover, only a subset of the BS nodes is influenced by each handoff and, the influence disappears with the completion of the handoff. It is therefore not surprising to see in Figure 5.d that the efficiency remains essentially constant for variable inter–handoff time. In addition, it is once again verified that $t_{hold}$ has a much greater impact to efficiency than the handoff frequency.

## 4. Conclusions

In this paper we presented a reliable multicasting protocol for mobile stations that is based on a logical ring token–passing protocol between the BS nodes of the network. We note that the algorithm (using the rotating token) can be applied to reliable multicasting in the fixed network as well. However, the benefit of the token algorithm is that it allows the incorporation of the dynamic nature of mobile nodes and, in particular the handling of their handoffs without any undue large communication or processing overheads. The simplicity of the scheme is in our view the most important feature.

In the performance study, we attempted to isolate all external influence from other protocols or from the particular topology in order to correctly appreciate the impact of the proposed protocol parameters. We have established that, for given BS node resources (buffers) and given the uncertainty of the outage periods experienced by the mobile nodes (that depend, among other things, on their mobility patterns) the most important parameter is the token holding time. We are currently examining control mechanisms that would allow us to determine and dynamically adapt the token holding time depending on the traffic load, the round-trip-times between successive BS nodes of the logical ring and the available buffer at each BS node.

## References

[1] K. Obraczka, "Multicast Transport Protocols: A Survey and Taxonomy", *IEEE Communications Magazine*, January 1998, pp. 94-102.

[2] C. Perkins, "IP Mobility Support," *Internet Engineering Task Force, INTERNET DRAFT,* `draft-`

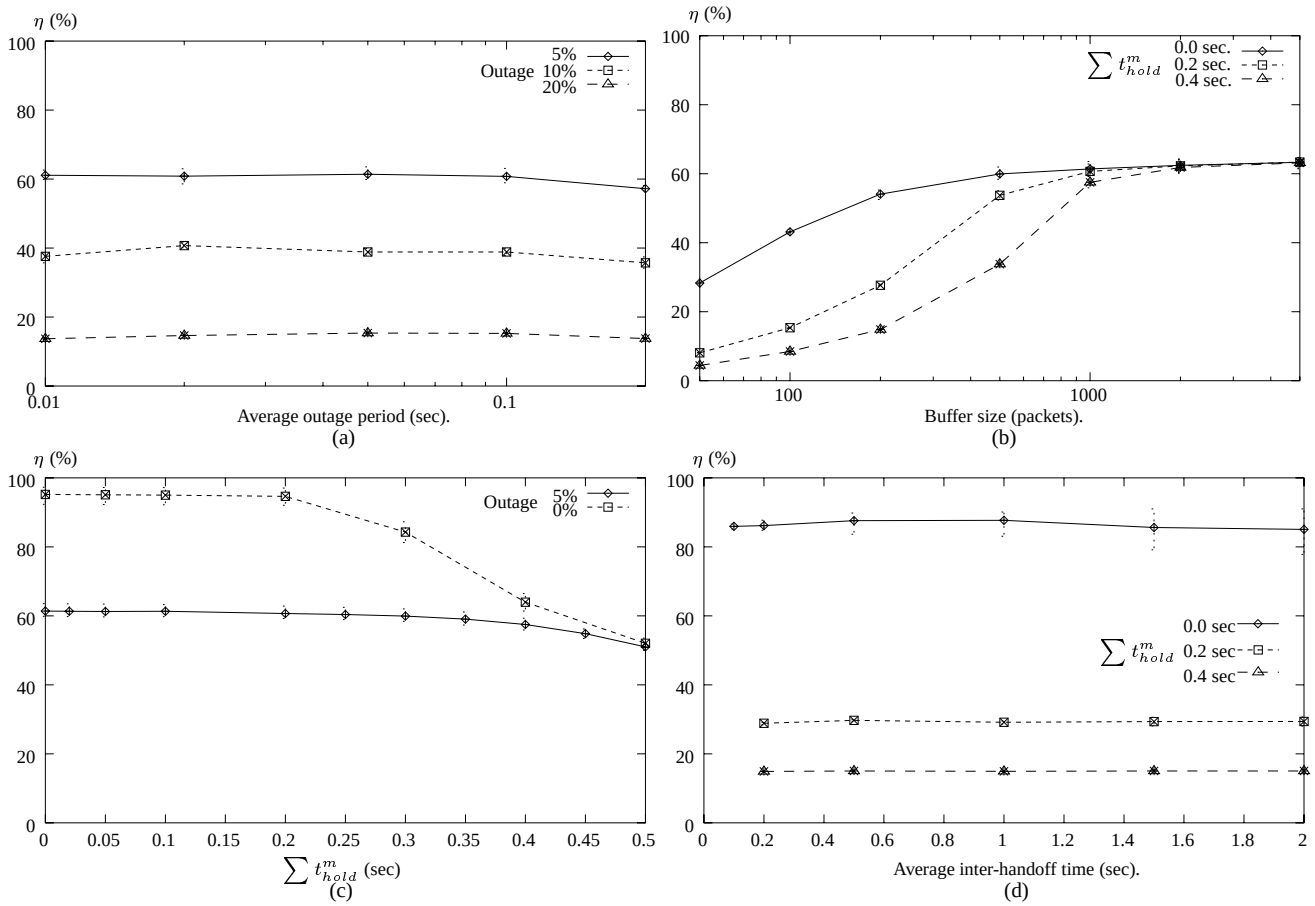


**Figure 5. Efficiency for (a) variable length of the outage period and outage percentage, (b) variable BS buffer size and $\sum t_{hold}$, (c) variable $\sum t_{hold}$ and outage percentage, and, (d) variable inter–handoff time and $\sum t_{hold}$.**


`ietf-mobileip-protocol-14.txt`, 21 December 1995.

[3] C. L. Williamson, T. G. Harrison, W. L. Mackrell, and R. B. Bunt, "Performance evaluation of the MoM mobile multicast protocol," *Mobile Networks and Applications*, vol. 3, no. 2, August 1998.

[4] V. Chikarmane, C. L. Williamson, R.B. Bunt and W. Mackrell, "Multicast Support for Mobile Hosts Using Mobile IP: Design Issues and Proposed Architecture", *Mobile Networks and Applications*, vol. 3, no. 4, 1998, pp. 365-379.

[5] A. Acharya and B. R. Badrinath, "A framework for delivering multicast messages in networks with mobile hosts", *Mobile Networks and Applications*, Vol. 1, No. 2, 1996, pp. 199-219.

[6] K. Brown and S. Singh, "RelM: Reliable Multicast for Mobile Networks", *Journal of Computer Communications*, Vol. 21, No. 16, 1998, pp. 1379-1400.

[7] D. Jefferson, "Virtual Time," *ACM Transaction on Programming Languages and Systems*, vol. 7, no. 3, pp. 404–425, July 1985.