A Feedback Implosion Suppression Algorithm for Satellite Reliable Multicast

Gun Akkor, John S. Baras, and Michael Hadjitheodosiou
Center for Satellite and Hybrid Communication Networks,
Institute for Systems Research, University of Maryland,
College Park, MD 20742, USA.
e-mail: {akkor, baras, michalis}@isr.umd.edu

***Abstract*— In this paper, we propose a knapsack-based feedback suppression algorithm for reliable multicast transport protocols operating over a satellite network. A reliable transport protocol needs to identify the packets which failed to reach a given destination. This is achieved through feedback packets returned to the source. For multicast services, receiver feedback has been shown to lead to the *feedback implosion* problem. Feedback implosion is a well-studied problem and various solutions exist in the literature. However, these solutions mainly focus on terrestrial networks and do not take into account the inherent characteristics of the satellite channel and the architecture of the deployed network. Therefore, we need to revisit the problem and provide a new set of solutions for efficient integration to next generation satellite systems. In this paper, we introduce a feedback implosion suppression algorithm, which effectively suppresses the amount of feedback relayed through the satellite channel, while ensuring that the critical information is conveyed in a timely fashion. The performance of the algorithm is evaluated through simulations.**

I. Introduction

Broadband satellite systems are quickly becoming an integral component of broadband communication networks. They have several attractive characteristics, such as breadth of broadcast "reach", ubiquitous access, global coverage, and as they move to the next generation, higher frequency band systems, they plan to offer large and flexible capacity, which would make them a natural technology option for carrying a variety of multicast services. However, despite the potential of satellite multicast, and the current market for various satellite-based broadcast services, there exists little support for reliable multicast via satellite. This is largely because most of the multicast protocols have been designed primarily for wireline terrestrial networks, without taking into account the inherent characteristics of the satellite channel and the deployed network topologies [1], [2].

Many of the problems in wireline terrestrial multicast would have either different roots or different solution spaces in satellite multicast. Therefore, it is necessary to revisit some of these problems and provide a new set of solutions for future integration of satellite systems with multicast services. For reliable multicast protocols, *feedback implosion* is one such problem that has drawn considerable attention from the research community [2]–[5]. In terrestrial networks, feedback implosion occurs for two reasons. Firstly, the flow of feedback packets from multicast receivers, which are located typically at the leaves of the multicast distribution tree, to the multicast sender, which is the root of the tree, causes network traffic concentration around the links close to the sender. Secondly, a sender is required to process a large number of feedback packets, which may be prohibitive in terms of memory, storage and computational load. However, in satellite multicast, there are other aspects. Receivers are a single hop away from the satellite and there is no physical hierarchy between the satellite and the receivers. Therefore, the multicast schemes that are based on physical or logical hierarchy for aggregation of receiver feedback or local recovery cannot be applied to this topology [6]–[9]. As a result, the *feedback implosion* problem becomes very challenging [10].

The fact that the return channel (uplink) is a shared medium and the satellite spectrum is limited and expensive, makes it necessary to try to minimize the amount of bandwidth required for feedback, which can then be used by real traffic, thus making the system more competitive in price and performance. Assigning a separate return channel to every receiver would result in the waste of resources and would not scale. Considering that (i) feedback information may contain redundant information (due to correlations among the loss pattern of receivers) and, (ii) most multicast algorithms only need to track the behavior of a subset of receivers with the worst case channel conditions, the challenge is to design efficient algorithms to select and filter-out information from multiple receivers to allow only the most relevant feedback information to be conveyed to the source using as little bandwidth as possible.

In this paper, we address this issue by introducing a feedback implosion suppression algorithm that allows the use of a fixed number ($\ll$ the number of receivers) of return channels by the multicast group while ensuring that the critical information is conveyed to the satellite in a timely fashion. In [10], authors propose a feedback suppression algorithm for a similar satellite system, however, our work significantly differs from their work in the way (i) priority feedback is determined and, (ii) feedback is transmitted over the return channels.

The remainder of this paper is organized as follows. In Section II, we describe the generic reliable transport protocol

 0-7803-7974-8/03/$17.00

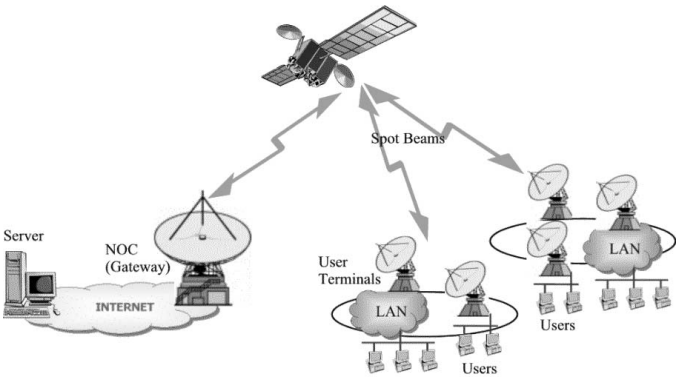

Fig. 1. Network Architecture

which operates between the source and the receivers. In Section III, we present our feedback suppression algorithm. Section IV describes the simulation environment, discusses the performance metrics of interest, and presents results. Section V concludes the paper.

II. A Reliable Multicast Transport Protocol

We start by describing the underlying reliable transport protocol, which operates between the source (NOC) and the receivers (user terminals), and uses the receiver feedback to detect packet losses and to initiate retransmissions. We present only the protocol behavior relevant to the operation of our policy. Several variations of this protocol exist in the literature and it has been shown to perform favorably in the context of satellite multicast services [3], [11], [12].

The envisioned system is based on a Ka-band satellite with DVB/MPEG-2 compliant forward and return links. The satellite system supports a star topology network with on-board processing (OBP) option, which is capable of supporting multiple spot-beams. The system provides point-to-multipoint IP multicast service to a large number of satellite users located inside its footprint through a central *network operations center* (NOC) (Figure 1).

We assume the transport protocol performs as follows between the NOC and user terminals:

- IP packets are encapsulated into a DVB/MPEG-2 transport stream of fixed length (188 Bytes) packets at the NOC.
- NOC organizes the transport stream into source blocks (SB) of k packets and encodes them using a suitable *expandable FEC code*. An expandable FEC *encoder* takes k source packets as input and generates as many unique encoding packets as requested on demand. An expandable FEC *decoder* has the property that any set of $k \cdot (1+\epsilon)$ unique encoding packets is sufficient to reconstruct the original k source packets, where ϵ is a small decoding overhead. Construction of an expandable FEC code with small decoding overhead is described in [13]. The NOC initially generates $n \geq k$ unique encoding packets corresponding to the packets of the SB and transmits the encoding block (EB) to the multicast group over the satellite. The NOC does not wait for the receiver feedback on the EB, and proceeds with the transmission of the next block.
- A shortened Reed-Solomon RS(204, 188, t=8) code is applied to each forwarded packet. The 204 octets packet is then forwarded to a convolutional interleaver. The interleaved symbols are encoded with a rate punctured convolutional inner code with optional code rates of $1/2, 2/3, 3/4, 5/6, 7/8$. The bit stream is transmitted over the channel using QPSK modulation.
- We assume that the channel decoder at the receiving end can detect corrupted packets and discard them, i.e the receiver's error detection process is assumed to be perfect. Therefore, if the receiver transport layer receives at least $k \cdot (1+\epsilon)$ unique encoding packets for a particular EB, it can decode the block and pass the source block to upper layers. Otherwise, it buffers the uncorrupted packets of the EB and creates a *request* for additional encoding packets. This request is the feedback transmitted to the NOC and it identifies the EB and the number of additional encoding packets required to decode the EB. At any point in time, a receiver may have buffered several EB(s) which require additional encoding packets to complete. In this case, receivers sort their requests in a predetermined order and transmit them one-by-one.
- An encoding packet transmitted for a particular encoding block benefits all receivers which await additional packets to complete the block. Therefore, NOC will be interested in transmitting $l_{\max} = \max\{l_1, \ldots, l_R\}$ additional encoding packets for the EB, where l_r is the number of additional encoding packets requested by receiver r for the EB, and R is the number of participating receivers (i.e. the number of receivers that have transmitted a request for the particular block). NOC collects requests from receivers and generates $l_{\max}$ additional encoding packets for the corresponding SB. These packets are transmitted to the multicast group either using a separate channel, or are piggy-backed with the next transmitted EB.

In general, additional encoding packets may also get corrupted during the subsequent transmissions and some receivers may still need extra packets after the completion of the first request process. Therefore, receivers continue this repeat-request process until they accumulate enough encoding packets for decoding of the EB.

III. Feedback Suppression Policy

Looking at the return information, we observe that, it is sufficient for the NOC to track only the maximum number of encoding packets requested by any receiver per encoding block. The volume of feedback would be minimized if only the receiver with the maximum packet requirement responds to the NOC per encoding block. We can consider two extreme scenarios for this situation: (i) all receivers communicate among each other through a secondary network (possibly a terrestrial connection) before deciding on whether to transmit a

request, and suppress the feedback of all receivers but the one with the maximum additional encoding packet requirement; (ii) every receiver is assigned a separate return channel, which is used to communicate the request information to the NOC, which then computes the maximum. The former scenario requires additional infrastructure and collaboration between the receivers which are contradictory to reasons for deployment of a satellite network. The latter situation gives rise to the feedback implosion problem as well as the waste of uplink resources.

Our solution strategy overcomes this problem by minimizing the number of transmitted request messages by the help of a knapsack-based algorithm that runs at the NOC without relying on any collaboration among receivers. We assume that only a fixed number $m \ll R$ of return channels are allocated to the multicast session for transmitting request packets. The problem is, then, to determine at every uplink transmission time, which of the R multicast receivers will be given a chance to transmit a request in one of the m available return channels. This problem fits nicely to a multiple knapsack problem formulation [14] when one considers every available return channel as a knapsack that needs to be filled with items (receiver requests) of weight equal to the number of additional packet requirement such that total weight is maximized, i.e. receivers with maximum packet requirements are assigned a channel for feedback transmission.

We first make the following modifications to the receiver operation:

- Instead of transmitting a request immediately in the next uplink transmission time, receivers buffer their requests in a queue and wait until they are assigned a feedback channel for transmission. In the meantime, if additional encoding packets are received, the pending requests are updated to reflect the new numbers. A request for additional encoding packets can not be for more than k (taking $\epsilon = 0$ for simplicity) encoding packets, at any time, since k unique encoding packets suffice to complete the decoding of an EB.
- Every receiver r maintains a value w_r, which is the average number of additional encoding packets required to complete all the pending encoding blocks. This information is readily available at each receiver and can be calculated simply by summing over the number of additional encoding packets required to complete all pending encoding blocks and dividing it by the number of encoding blocks in the buffer. Clearly, $0 \leq w_r \leq k$. This value is called the weight of that particular receiver and is sent to the NOC along with a request packet, whenever the receiver is assigned a return channel.
- Receivers transmit the first request in their queue to the NOC, whenever they are assigned a return channel.

In general, receivers with higher packet request averages should be given priority over other receivers since their requests would probably account for the requests of receiver with lower averages. This is due to the fact that an additional encoding packet transmitted for an encoding block would benefit all receivers which have pending packets to complete that block. However, this would decrease the robustness of the algorithm against instantaneous changes and variations in the reception quality of receivers. Because, a receiver with a low packet request average is suppressed by its peers, it can not transmit a feedback packet when the condition changes and it starts to build up a high average. Therefore, we must also allow receivers with previously low request averages to transmit their requests in order to update their status at the NOC.

Using these observations as the basis, the NOC solves the problem of deciding which receiver will be allowed to transmit a request in future uplink transmission times (because of the propagation delay, satellite can only calculate this channel allocation layout for future times). We view each of the available uplink channel as a knapsack with capacity $c_i = \max\{w_1^*, \ldots, w_R^*\} = c$ for $i = 1, 2, \ldots, m$ and each receiver as an item with weight w_r^* and profit $p_r = w_r^*$ for $r = 1, \ldots, R$, where w_r^* is the last update the satellite has for the weight of receiver r and R is the number of receivers. We, then, solve the following multiple knapsack problem:

$$\begin{aligned} \max \quad & \sum_{i=1}^{m} \sum_{r=1}^{R} p_r x_{ir} \\ \text{subject to} \quad & \sum_{r=1}^{R} w_r^* x_{ir} \leq c_i = c \quad \text{for } i = 1 \ldots m \\ & \sum_{i=1}^{m} \sum_{r=1}^{R} x_{ir} \leq 1 \end{aligned} \tag{1}$$

In the solution of this problem, $x_{ir} = 1$, if receiver r is assigned to return channel i. Once the channel allocation is determined, NOC broadcasts $\mathbf{x}$ and the value of c to all receivers either using a separate control channel or by piggybacking it in the next EB. Upon receiving a new channel allocation information, receivers apply it in the next uplink transmission time.

It is important to note that the solution of this problem may assign more than one receiver to a channel, while some receivers are not assigned a channel in the next uplink transmission time. If a receiver has weight, w_r^* that is closer to c, it is more likely to be assigned to a channel alone, since it fills the capacity of the knapsack (channel). A higher weight indicates that the receiver on the average needs a large number of additional encoding packets to complete all the pending encoding blocks and hence its feedback will probably benefit the group the most. Therefore it is assigned to a channel alone. Receivers with smaller weights will be sharing the channel with other receivers. A smaller weight indicates that the receiver needs, on the average, only a few encoding packets to complete all the pending encoding blocks and hence its feedback will only partially benefit the group. One may not consider the receivers with small weights, but this is not a good strategy since the algorithm uses the last updated value of the receiver weights rather than the instantaneous value. A

receiver with a small weight might build up a high average since its last update, and if we choose to assign only the first m receivers with the highest weight values to the return channels, it may not get a chance to transmit a request and update its weight information.

In order to avoid collisions, however, receivers do not transmit a repair request with probability equal to one in the channel they are assigned to. They employ a probabilistic back-off algorithm and transmit a request with probability proportional to their current weights. We discuss the possible choices on the functional relation between the transmission probability and receiver weights in Section IV-B. In the next section, we describe our simulation environment and present simulation results on the performance of our algorithm.

IV. Simulation Results

A. Channel Model

We model the satellite channel by a threshold-based 2-state Markov Chain. In this model the channel is either in "GOOD" state, if the transmitted signal experiences less than Γ dB. attenuation, or it is in "BAD" state, if the signal fade is more than Γ dB., where Γ is the fade attenuation threshold [15], [16]. We assume that if the channel is in "GOOD" state, channel coding is capable of correcting all bit errors and for simulation purposes the probability of bit error at the output of the channel decoder is zero. In the "BAD" state, the channel behaves as a Binary Symmetric Channel (BSC) with bit error probability equal to p_b at the output of the channel decoder. The channel state is characterized by the transition probability matrix given by

$$M = \begin{bmatrix} r & 1-r \\ 1-s & s \end{bmatrix}, \tag{2}$$

where r and $1-s$ are the probabilities that the channel will be in the "GOOD" state at the k^{th} symbol duration, given that the channel was in "GOOD" and "BAD" states, respectively at the $(k-1)^{th}$ symbol duration. Using the results of the ACTS Propagation Experiments, published in [17], for a fade attenuation threshold of $\Gamma = 10$ dB., we have $r = 0.9999813$, and $1 - s = 0.00172$.

In order to evaluate the performance of the transport protocol, we need to calculate the bit error probability at the output of the channel decoder. In order to estimate the value of p_b, we use the link budget calculations of a commercial satellite system proposed in [18]. The calculations confirm that for signal attenuation of less than 10 dB., link budget margins and channel coding are capable of keeping the bit error probability around 10^{-9} at the output of the channel decoder. For signal attenuation of more than 10 dB., on the other hand, bit errors occur with probability $p_b \geq 0.1$ at the output of the channel decoder [19], [20]. Using this result, in our simulations, we evaluate the time progress of the channel state for every bit transmission and assume that a bit is in error with probability $p_b = 0.1$ when the channel is in "BAD" state and no errors occur when the channel is in "GOOD" state. Since these errors are at the output of the channel decoder, we further assume that a packet is corrupted if at least one bit is in error. We use this packet loss pattern in our numerical results.

B. Simulation Environment

In our simulations, we assume that the NOC has a fixed number $B = 1000$ of source blocks, each with $k = 188$ packets. Each source packet is 188 Bytes. For each source block, the NOC constructs an encoding block of $n = 196$ encoding packets. There exists additional bandwidth of $a = 8$ packets for piggy-backing additional encoding packets in response to receiver requests. The NOC and the receivers perform the previously described reliable transport protocol over the satellite channel. The choice of these parameters does not affect the performance of our algorithm since we assume that they are constant whether the transport protocol performs feedback implosion suppression or not. Therefore, they are fixed throughout our simulations.

We need to define the relationship between the request transmit probability $p_r(w_r)$ of a receiver r and its weight w_r. Intuitively, $p_r(w_r)$ should satisfy $p_r(0) = a$, $p_r(c) = 1$ and be a non-decreasing function of the receiver weight, where $0 \leq a \leq 1$. In this paper, we adopt:

$$p_r(w_r) = \frac{\exp(\mu \cdot w_r/c)}{\exp(\mu)}, \tag{3}$$

where w_r is the current average of additional encoding packets receiver r wants to request and μ is an aggressiveness parameter. This function may not be the most suitable function for this purpose and performance of the algorithm for different types of functions is currently under investigation. In our simulations we set

$$\mu = 1 - \frac{m}{R}, \tag{4}$$

where m is the number of available return channels and R is the number of receivers. Therefore, receivers transmit their requests more aggressively when more channels are available, but back-off from transmitting as their average weight decreases for a fixed number of return channels.

C. Performance Metrics

There are several performance metrics which may be of interest in this scenario. In this paper, we consider the following two metrics: (i) *average number of rounds to complete a block*, and (ii) *bandwidth efficiency*. If an encoding block can not be completed at the end of the first round, additional encoding packets need to be transmitted in the subsequent rounds. The first metric measures the performance of the algorithm in terms of the average number of rounds it takes to complete a block. The second metric is the bandwidth efficiency in transmitting the original source packets and is given by the ratio of source block size to average number of packets transmitted to complete transmission of the block. Other possible metrics, which we would like to investigate in the future include encoding/decoding complexity and buffering requirements at the NOC and the receivers. In the next section, we present simulation results on the performance of our algorithm as a function of the number of available return channels.

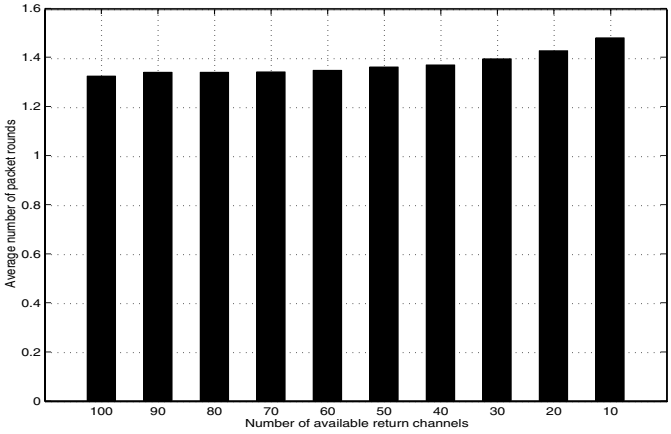

Fig. 2. Number of packet rounds versus number of available return channels for a set of $R = 100$ receivers.

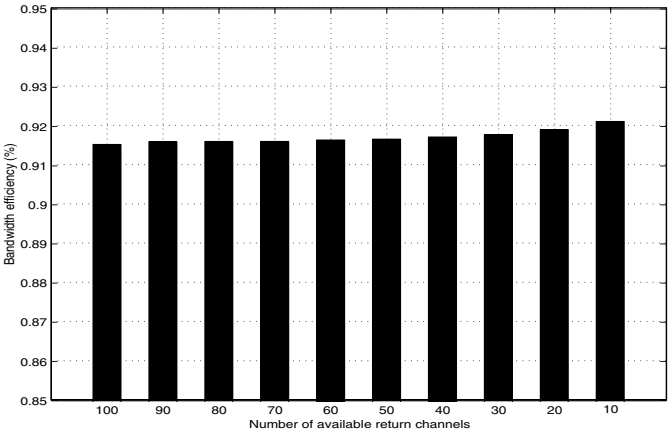

Fig. 3. Bandwidth efficiency as a function of number of available return channels for a set of $R = 100$ receivers.

D. Results

In Figure 2, we plot the average number rounds it takes to complete a block versus the number of available return channels for a receiver set of $R = 100$. We observe that average number of rounds increases slightly as the number of return channels decreases. However, the performance degradation is minimal compared to the savings in terms of the number of return channels. In Figure 3, we plot the bandwidth efficiency versus the number of available return channels for the same set. We observe that the bandwidth efficiency slightly increases as the number of return channels decreases. This is because the feedback suppression algorithm does not always calculate the maximum, as it is the case when there is no suppression, and on the average transmits at most as many packets as the case without the suppression. This results in the more conservative use of the bandwidth, but also increases the number of rounds it takes to complete a block as evident from Figure 2.

V. Conclusion

In this paper, we introduced a knapsack-based feedback implosion suppression algorithm for reliable multicast transport protocols operating over a satellite network. We showed that the suppression algorithm causes minimal degradation to the performance of the underlying transport protocol while effectively minimizing the volume of receiver feedback. We also showed that it is possible to improve the efficiency of the transport protocol in terms of the overall bandwidth by more conservative transmission of the additional encoding packets. The majority of the multicast group may benefit from this approach, while only a few receivers with maximum additional packet requirements are penalized. This paper presents a preliminary set of results, and various other performance related issues, such as the effect of variations in the loss pattern of receivers, size of the receiver set, choice of parameters for the transport protocol, and the computational load, are still under investigation.

References

[1] F. Filali and W. Dabbous, "Issues on IP multicast service behaviour over the next generation satellite-terrestrial hybrid networks," in *Proc. of Computer and Communications*, 2001, pp. 417–424.

[2] M. W. Koyabe and G. Fairhurst, "Reliable multicast via satellite: A comparison survey and taxonomy," *Int. J. Sat. Comm.*, vol. 19, no. 1, pp. 3–23, 2001.

[3] J. Nonnenmacher, E. W. Biersack, and D. Towsley, "Parity-based loss recovery for reliable multicast transmission," *IEEE/ACM Trans. Networking*, vol. 6, no. 4, pp. 349–361, August 1998.

[4] C. Diot, W. Dabbous, and J. Crowcroft, "Multipoint communication: A survey of protocols, functions and mechanisms," *IEEE J. Select. Areas Commun.*, vol. 15, no. 3, pp. 277–290, 1997.

[5] K. Obraczka, "Multicast transport protocols: A survey and taxonomy," *IEEE Commun. Mag.*, January 1998.

[6] J. P. Macker and W. K. Dang, "The multicast dissemination protocol," Naval Research Laboratory, Tech. Rep., 1996.

[7] S. Floyd, V. Jacobson, C. Liu, S. McCanne, and L. Zhang, "A reliable multicast framework for light-weight sessions and application level framing," *IEEE J. Select. Areas Commun.*, vol. 15, pp. 407–421, 1997.

[8] S. Paul, K. K. Sabnani, and D. M. Kristol, "Reliable multicast transport protocol (RMTP)," *IEEE J. Select. Areas Commun.*, vol. 15, no. 3, pp. 407–421, April 1997.

[9] T. Speakman, D. Farinacci, S. Lin, and A. Tweedly, "Pragmatic general multicast (PGM) transport protocol specification," Internet Draft, June 1999, work in progress, draft-speakman-pgm-spec-03.txt.

[10] S. Cho and I. F. Akyildiz, "A poker-game-based feedback suppression algorithm for satellite reliable multicast," in *Proc. of IEEE GLOBECOM 2002*, November 2002.

[11] S. Payne, "Reliable multicast via satellite," Master's thesis, University of Maryland, College Park, MD, 1999.

[12] K. Manousakis, "Reliable multicasting based on air caching for flat heirarchy networks," Master's thesis, University of Maryland, College Park, MD, 2002.

[13] M. Luby, "Information additive code generator and decoder for communication systems," U. S. Patnent No. 6,307,487, October 2001.

[14] D. Pisinger, "An exact algorithm for large multiple knapsack problems," *European Journal of Operational Research*, 1999.

[15] E. O. Elliott, "Estimates of error rates for codes on burst-noise channels," The Bell Systems, Tech. Rep., 1963.

[16] E. N. Gilbert, "Capacity of a burst-noise channel," The Bell Systems, Tech. Rep., 1960.

[17] J. Slack and M. Rice, "Finite state markov models for error bursts on the ACTS land mobile satellite channel," NASA ACTS Experiment Paper ACTS-96-046, 1996.

[18] E. J. Fitzpatrick, "SPACEWAY system summary," *Space Communications*, vol. 13, pp. 7–23, 1995.

[19] "HUGHES DIRECWAY System," www.direcway.com.

[20] J. Foerster and J. Liebetreu, "FEC performance of concatenated reed-solomon and convolutional coding with interleaving," IEEE 802.16 Broadband Wireless Access Working Group Report, June 2000.