# DVB Application Layer FEC Evaluations

DVB Document A115

May 2007

# Contents

# DVB Application Layer FEC Evaluations

## 1 Introduction

This DVB Blue Book is intended to provide background on the evaluation process behind the choice of the hybrid approach in the DVB application layer forward error correction (AL-FEC) specification which is included in the DVB-IP Phase 1 Handbook [1]

It consists of two documents that were created by DVB as part of the AL-FEC evaluation process. Document A presents the evaluation criterion that were agreed before the selection process started and Document B is the report on the evaluation process. Document B also gives the rationale for the choice of the hybrid approach used in the DVB AL-FEC specification.

Please note that the content of both documents should be considered as informative only.

## References

1. ETSI  TS 102 034 v1.3.1 (2007), Digital Video Broadcasting (DVB); Transport of MPEG-2 TS based DVB Services over IP based Networks

2. 3GPP Technical Specification 26.346 – Multimedia Broadcast Multicast Service – Protocols and Codecs

3. SMPTE Specification 2022-1: Forward Error Correction for Real-time Video/Audio Transport Over IP Networks.

# Document A

# Criterion for selection of Forward Error Correction for the protection of audiovisual streams delivered over IP Network Infrastructure

## A1 Requirements

Audiovisual services delivered over networks are subjected to the inherent properties of those networks including latency and errors. DVB commercial requirement is quoted as:

*"Inclusion of suitable error protection strategies such as an FEC mechanism to enable DVB services to be carried over typical IP access networks with an acceptable quality of service (maximum 1 visible artefact/hour).*

- *The selected solution shall be in line with work of other standards bodies such as DSL-Forum. If necessary, DVB should liaise with relevant other bodies.*
- *The selected solution shall provide flexibility so that it covers a reasonable range of networks and a variety of business models (trade-off versus payload). Furthermore, the selected solution shall be extensible to cover likely future streaming requirements.*
- *The selected solution shall be implementable on a range of HNEDs without significantly increasing product cost."*

The DVB Technical Module has agreed that the IP Infrastructure group should recommend an (optional) application layer FEC. It is agreed that it should work end to end including the core and home network where required

The FEC scheme selection process should take into account:

1. Packet loss characteristics of practical IP access network implementations e.g. DSL. These might include the use of interleaving at the physical layer to improve transport performance.
2. Further packet losses that could occur in the core network due to congestion and/or the home environment e.g. wireless technologies.
3. Sensitivity of A/V coding to errors.
4. Practical viability and flexibility of FEC scheme (encoding and decoding) to meet the min and max correction at minimal cost (processing, memory) for large numbers of simultaneous streams.
5. Ongoing cost of bandwidth inefficiency inherent in the code – i.e. difference between the bandwidth required by the code and the theoretical minimum bandwidth needed for service in the given loss conditions."
6. Pre-computation of the FEC to enable later usage when the content is streamed.
7. Carriage directly over RTP in the future i.e. without an MPEG2 transport stream
8. Dynamically varying length of IP packets carrying A/V content.

## A2 System description

Figure 1 is an example of video service delivery over a DSL network from source (top left) to set top box (top right). It highlights the components through which the service is delivered and the logical position of the Application Layer FEC. Key points brought out by this diagram are:

a) There are other possible mechanisms that affect the delivery of acceptable quality of service (maximum 1 visible artefact/hour). These are DSL layer FEC/interleave, video/audio coding type and any error concealment at the decoder. The application layer FEC performance should provide adequate protection from errors with and without these mechanisms present (shown as min and max correction in fig 1).

b) When these other mechanisms are present, the application layer FEC should take into account the effect of failure of these other mechanisms under severe error conditions.

c) When these other mechanisms are present, the 'load' on the application layer FEC is reduced under normal error conditions, leading to possible 'cost' reductions in terms of latency, memory, processor etc.

d) Gaps in the core network domain and home network domain highlight the possible presence of other network types that could introduce service affecting packet loss. These networks should ideally be taken into account in the specification of application layer FEC performance, though will vary between implementations.

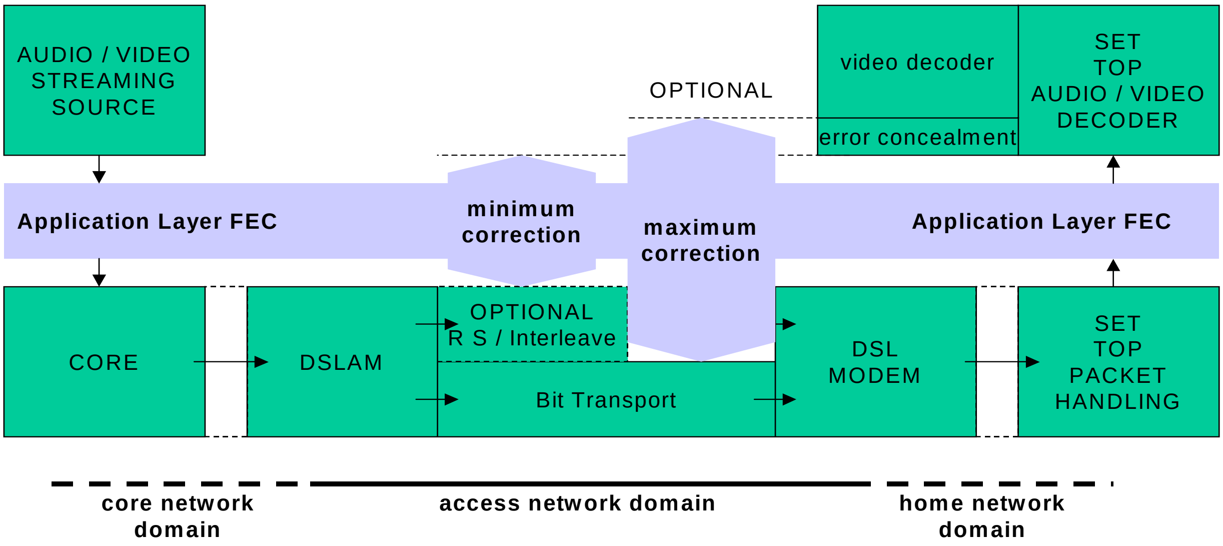


Figure 1 - minimum and maximum correction requirement for DSL access network domain

# A3 Packet loss characteristics

The packet loss characteristics should be provided by network operators and DSL chip vendors, ideally in the form of data collected from implementations or (if this is too commercially sensitive) in the form of a statement on what level of errors should be corrected by the application layer i.e. the requirements.

Worst case end-to-end packet loss metrics can be provided in terms of average loss rate, and loss distribution (independent random vs bursty) for the IP packets, independent of bit rate. Note: methods for characterisation of the loss distribution need further discussion.

Results for impulsive noise in DSL networks are available from the ITU and (until other information becomes available) they will be used as the basis of the evaluations. Although DSL is clearly an important case (where the results may vary widely), it is desirable to allow for other core, access and home networks also.

# A4 FEC Scheme Evaluation Criteria

Assume the following criteria:

1. Consider 3 error distributions: A. random losses (PLR 1e-3 to 1e-5), B. burst losses (PLR 1e-3 to 1e-5 with distributions based on ITU DSL results) C. better than 1e-5
2. Additional latency due to FEC depending on applications (VOD = 100 ms, Broadcast = 400 ms)
3. Bit-rate for VOD = 2Mbit/s, Broadcast = 2 and 6Mbits (both based on H264/AVC
4. Target mean time between FEC blocks that contain uncorrectable errors = 4 hours

Data should be provided for each FEC proposal, specifying the performance for each set of parameters employed to illustrate range of performance available in terms of:

- Overhead required by the FEC to achieve the target performance in each of the given loss conditions (FEC data)/( protected data) (%)
- Flexibility:
    - Changing the overhead or/and the block size dynamically (within or between FEC blocks)
    - Range of protection periods
    - Suitability for use with a wide variety of FEC sending strategies
- STB memory requirement for buffering / processing (bytes)
- STB processing requirement measured as:
    - maximum and average number of XOR operations
    - maximum and average number of conditional statements (IF..THEN)
    - maximum and average number of context switching
    - maximum and average size of additional temporary memory needed
    - maximum and average number of threads (if threaded)
- Headend memory requirements for buffering (bytes)
- Headend processing requirement measured as:
    - maximum and average number of XOR operations
    - maximum and average number of conditional statements (IF..THEN)
    - maximum and average number of context switching

    - maximum and average size of additional temporary memory needed
    - maximum and average number of threads (if threaded)
    - maximum memory bandwidth

- Scalabilty, e.g. suitability for hardware implementation and cost
- How much data is lost when the FEC fails? Visibility of artefacts when FEC fails
- Ability to discard the FEC flow and process only the original packets as normal
- Ability to add or remove FEC correction packets

Additionally, systems considerations should be addressed including:

- Continued functioning of existing STB products in presence of FEC data
- Option for new STB products to use or ignore FEC data
- Confirmation of FEC scheme IPR compliance with DVB rules
- Support of combined protection of audio and video packets

# Document B

This document contains the evaluation report of the DVB-IPI committee on the proposed AL-FEC codes. Note that the two codes originally proposed were the Pro-MPEG Code of Practice 3 code as now specified in SMPTE 2022-1 [3] and the Digital Fountain Raptor code essentially as specified in 3GPP TS 26.346 [2]. The eventually standardized code specified in TS102034 v1.3.1 [1] is a hybrid of these two original proposals.

# AL-FEC evaluation report for DVB-IPI

V0.5 – 24 August 2006

# B1 Introduction

The paper provides results of the DVB-IPI evaluation process for forward error correction for IP TV. Two candidate FEC codes have been considered, the Digital Fountain Raptor code, as proposed in [2], and the Pro-MPEG Code of Practice 3 based proposal, described in [3].

Document A provides the agreed evaluation criteria, with the exception that at subsequent conference calls it was agreed to consider “additional latency due to FEC” of 100ms and 400ms (rather than “protection periods”,) and “mean time between packet loss” (rather than “mean time between FEC blocks with errors”

During the evaluation process, it was realised that a key issue in determining the FEC performance is the sequencing and timing of the sending of source and FEC packets. This issue is discussed further in Section B2 and examples of sending arrangements are described in Annex A.

This paper also includes new simulation results for the following cases:

- “concurrent interleaved sending” – in which FEC packets are interleaved with the source packets they protect – these results are included in Annex B.
- “hybrid code” – in which a mixture of Pro-MPEG and Raptor packets are sent - these results are included in Appendix 1.

# B2 Sending arrangement considerations

An important issue in the evaluations was the way the different codes arrange data packets (source and FEC “repair” packets) for sending. Many different arrangements are possible for both codes. Since the arrangement can slightly impact the latency introduced by the FEC code with particular settings, and since these evaluations considered fixed latency budgets, the choice of sending arrangement affects the choice of parameters which are possible within the latency budget and therefore affects the bandwidth requirements of the codes.

An additional consideration with respect to sending arrangements is whether the resulting data stream has a constant bit-rate.

Annex A describes the sending arrangements proposed in more detail.

# B3 Bandwidth costs

A primary objective of the simulations performed as part of this evaluation exercise was to measure the bandwidth overhead required to achieve a target quality of service. Although not the only evaluation criteria for AL-FEC, bandwidth consumption represents an ongoing cost of the solution for the operator: excessive bandwidth consumption may translate into lower service quality, fewer services or a smaller target market.

In order to assess bandwidth requirements, simulations were performed according to the agreed cases. For each case, the simulated time was 96 hours and the *mean time between packet loss* was measured. The minimum bandwidth required was assessed by performing repeated simulations, gradually increasing the FEC overhead until the target mean time between packet loss was achieved. Note that in the case of the Pro-MPEG code, increasing the bandwidth required that a different code was used – i.e. change in the L and D parameters and possibly change in the type of parity packets sent: row, column or both.

## B3.1 Loss models

Two loss models were used in the simulations, independent random packet loss and a loss model based on DSL Repetitive Electrical Impulse Noise (REIN).

The REIN model results in fixed length (8ms) burst losses which are randomly placed in order to achieve an overall loss rate within the $10^{-6}$ to $10^{-3}$ loss range of interest. As such, the results below for the REIN case give a good indication of the code performance in the presence of burst losses.

## B3.2 Multicast case

For the multicast case, a maximum additional latency of 400ms was used. The graphs below show the FEC overhead required to achieve a mean time between packet loss of four hours, plotted against packet loss for both independent random packet loss and Repetitive Electrical Impulse Noise simulated. The overhead calculation is based on the actual number of bytes sent, including IP and other headers, not just the ratio of repair packets to source packets.

The figures also include a plot for an “Ideal Block Code” – this represents the theoretical lowest overhead which could achieve the target quality within the maximum latency using a block FEC code and gives a useful guide as to how much of the bandwidth dedicated to FEC is actually needed to provide the required FEC protection and how much is overhead due to inefficiency in the FEC code itself.

Note that the overhead scale in each graph may be different, to show the range of interest.

### B3.2.1 Results with constant sending arrangement

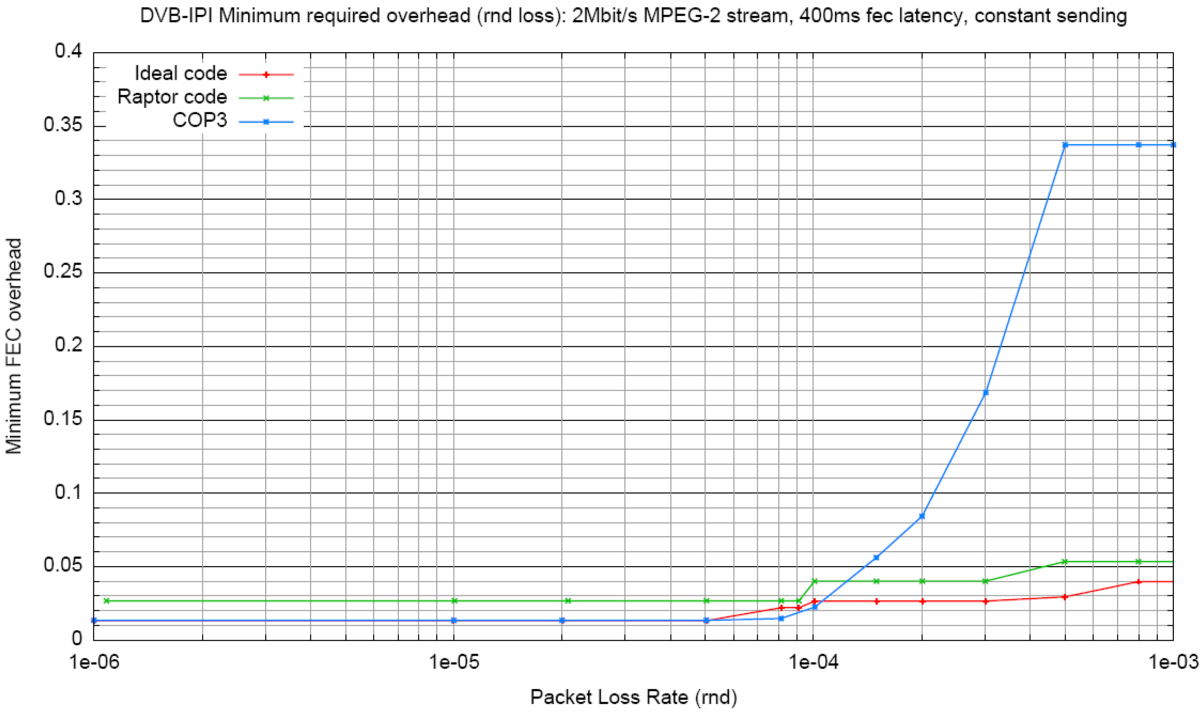


Figure 6: 2Mbit/s MPEG-2 Transport Stream, 400ms latency, Random Loss, constant sending

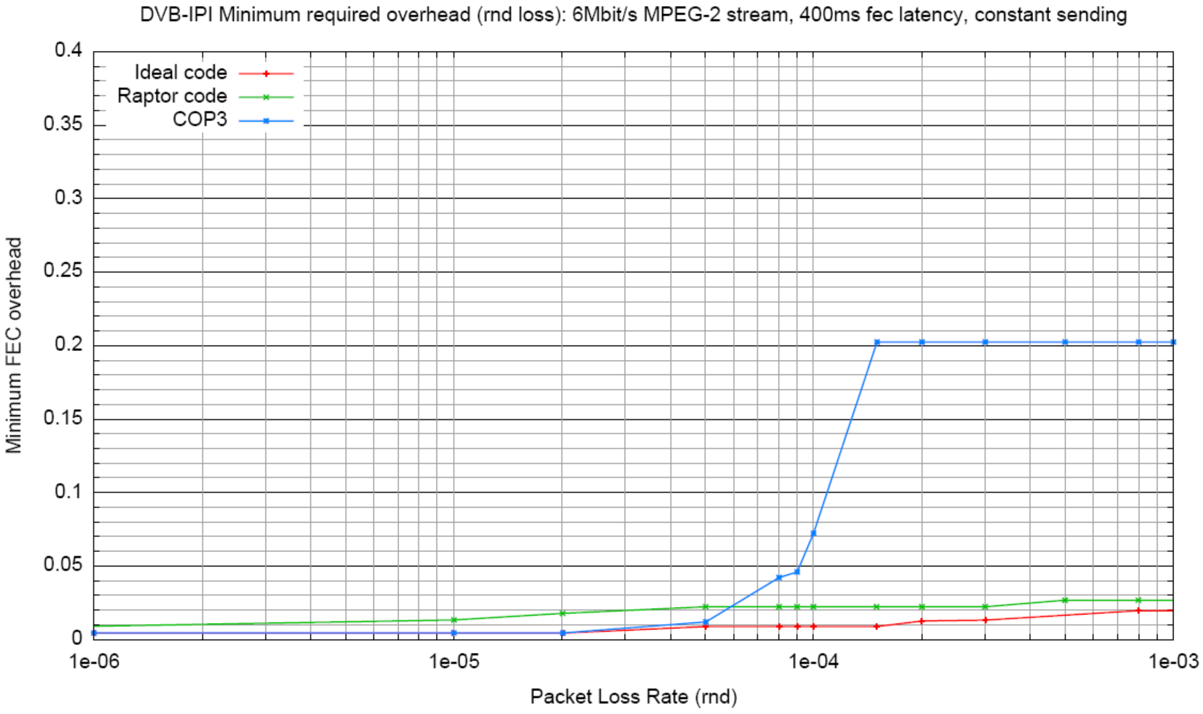


Figure 7: 6Mbit/s MPEG-2 Transport Stream, 400ms latency, Random Loss, constant sending

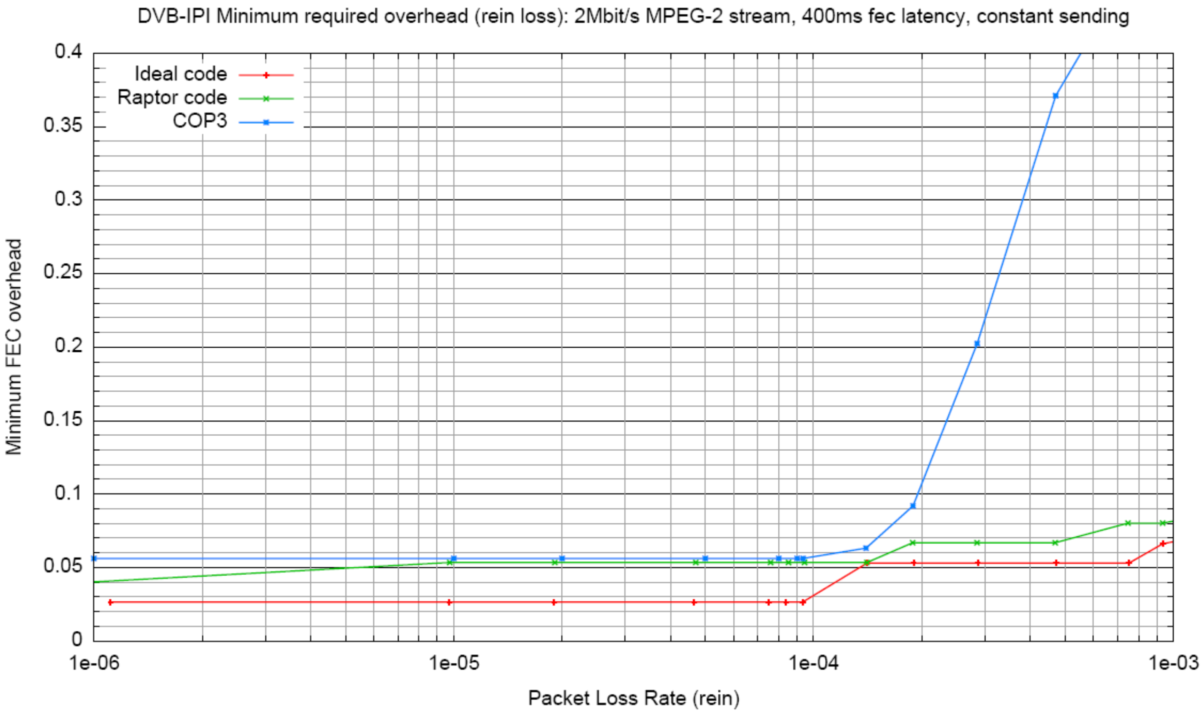


Figure 8: 2Mbit/s MPEG-2 Transport Stream, 400ms latency, REIN, constant sending

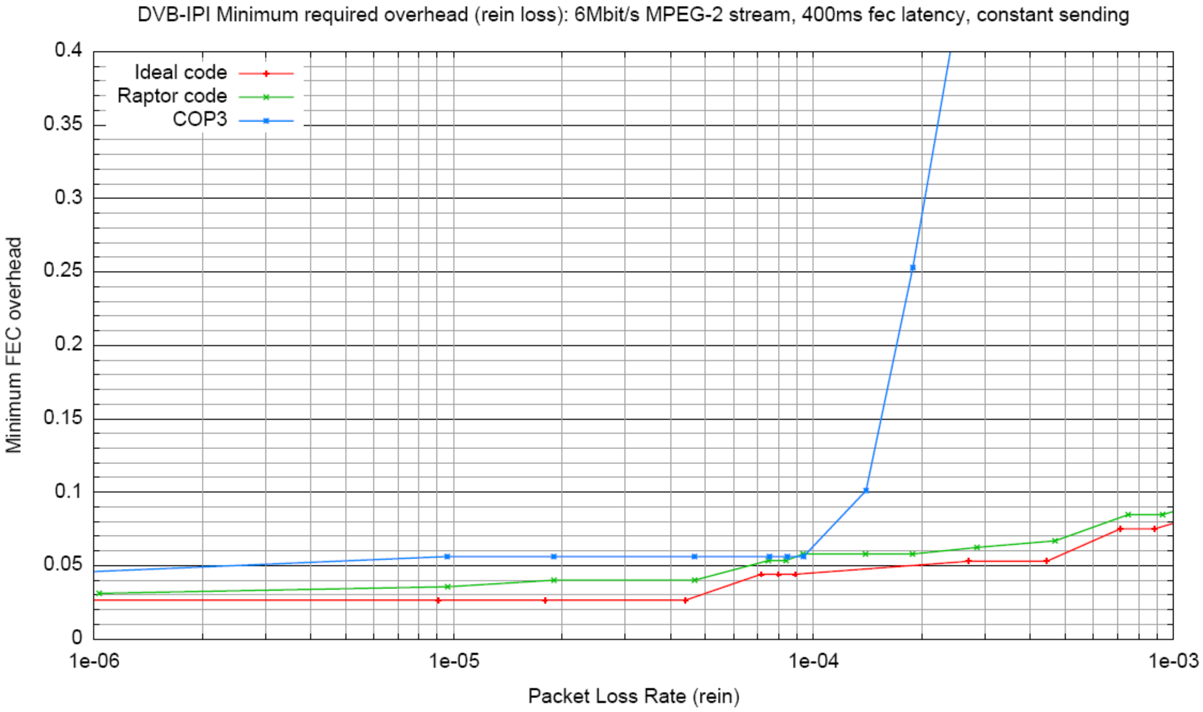


Figure 9: 6Mbit/s MPEG-2 Transport Stream, 400ms latency, REIN, constant sending

### B3.2.2 Results with burst sending arrangement

NOTE: Curves for the “Ideal” block code and Raptor below are for constant rate sending, compared with burst sending for Pro-MPEG.

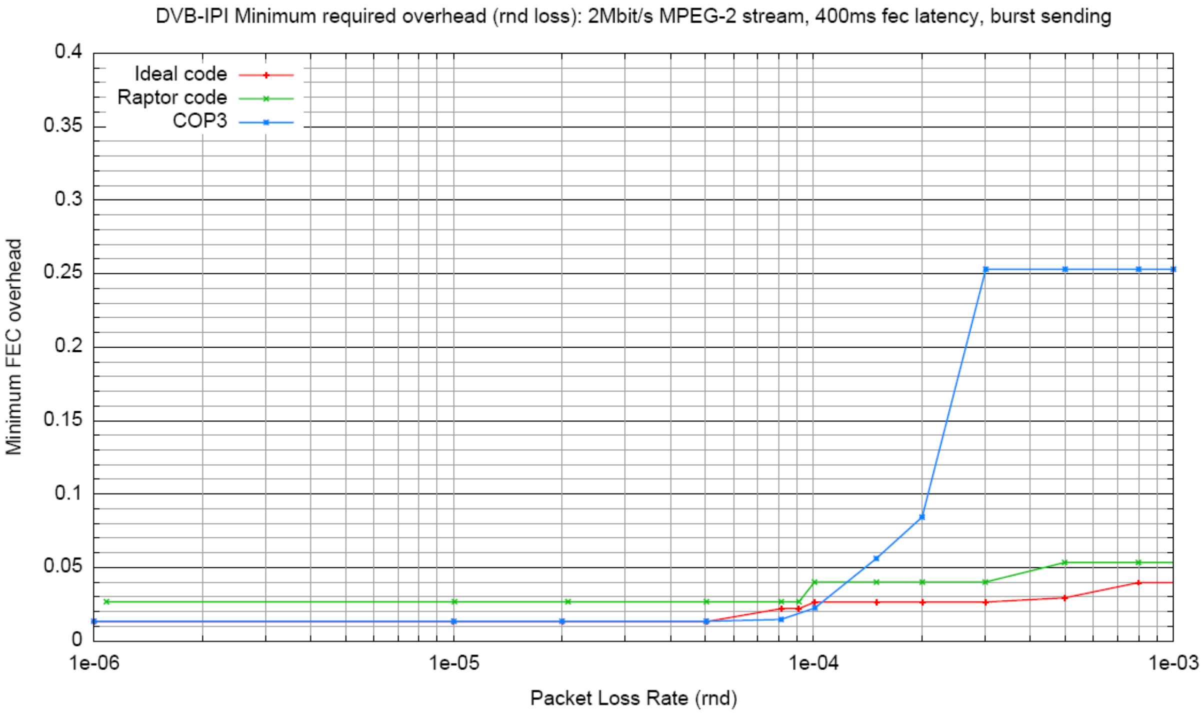


Figure 10: 2Mbit/s MPEG-2 Transport Stream, 400ms latency, random loss, burst sending

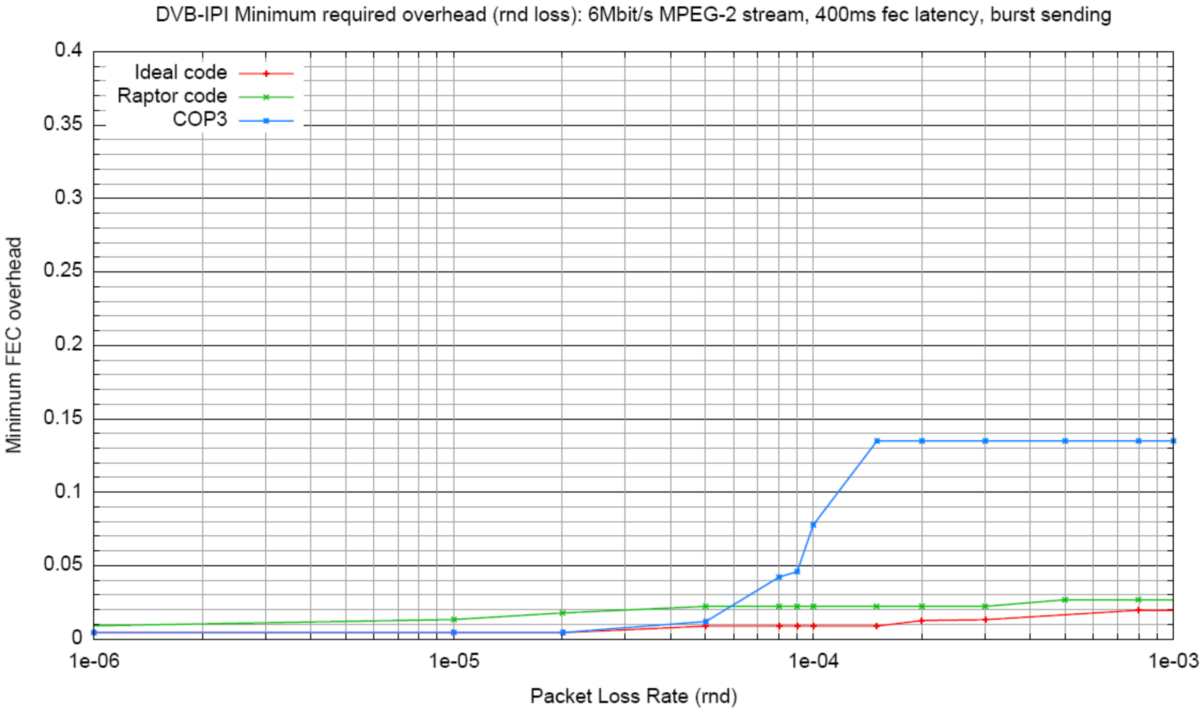


Figure 11: 6Mbit/s MPEG-2 Transport Stream, 400ms latency, random loss, burst sending

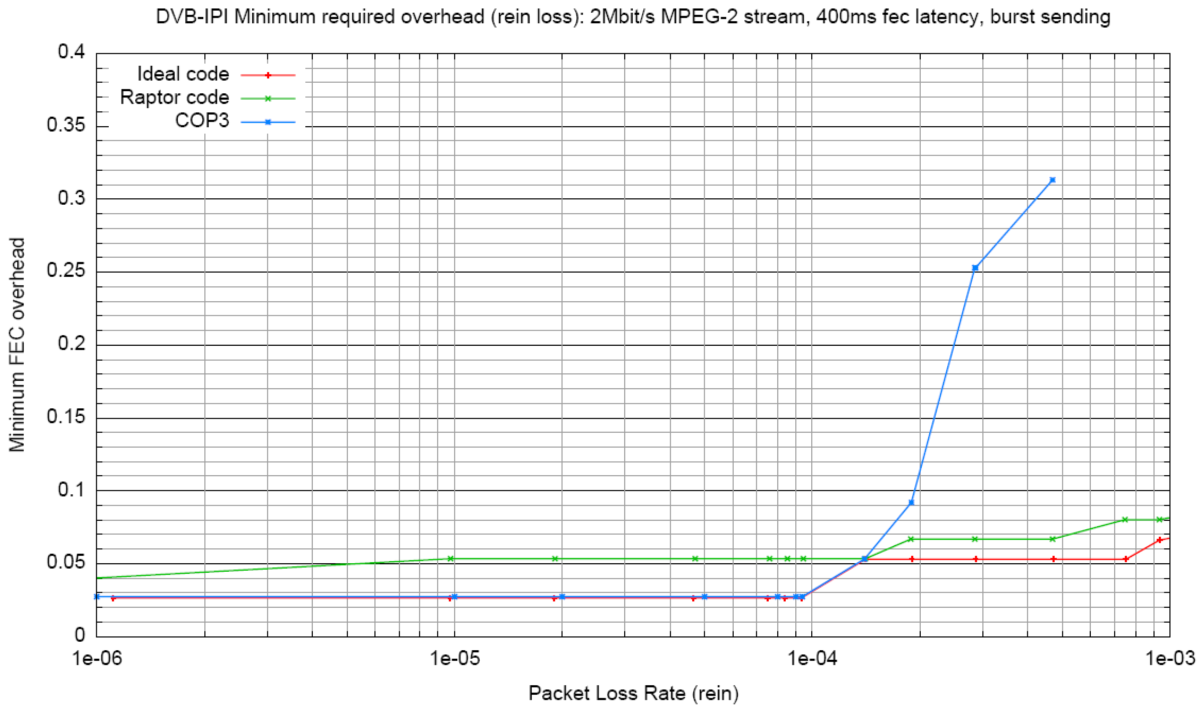


Figure 12: 2Mbit/s MPEG-2 Transport Stream, 400ms latency, REIN loss, burst sending

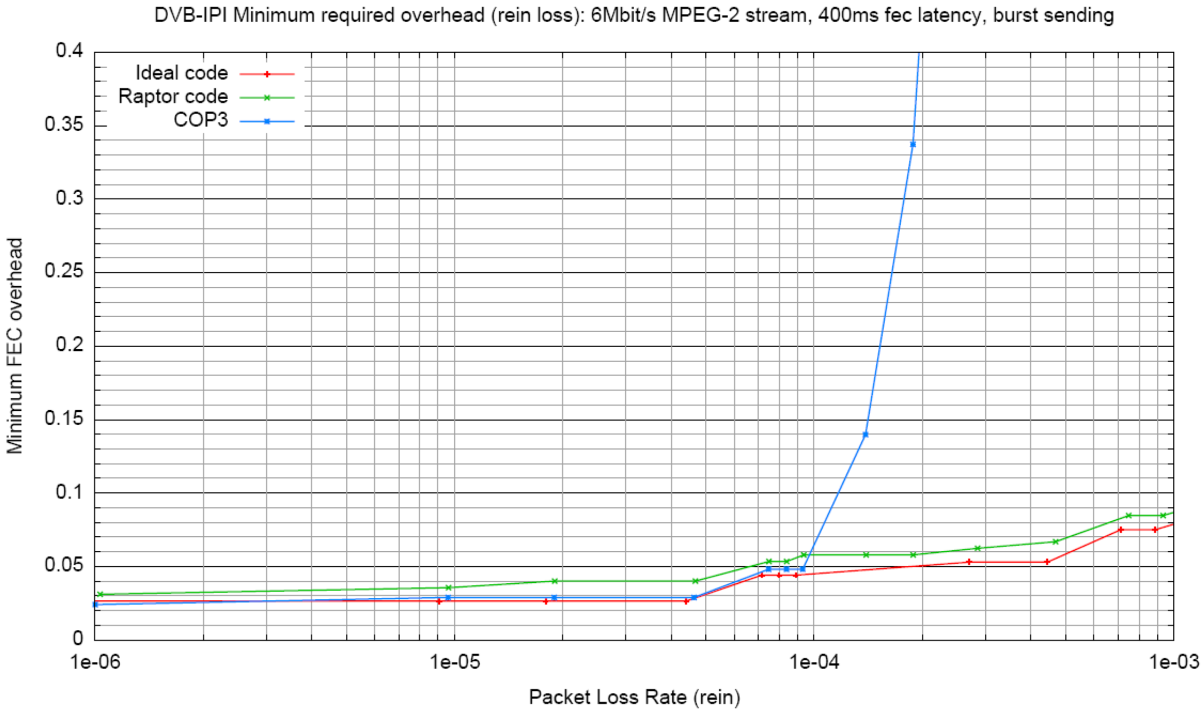


Figure 13: 6Mbit/s MPEG-2 Transport Stream, 400ms latency, REIN loss, burst sending

We note the following from these simulation results:

- The Raptor code consistently requires close to the minimum possible overhead for a block code (as illustrated by the red “ideal” plots).
- The overhead required for the Raptor code increases smoothly as the loss rate increases

- A modest Raptor overhead of 9% provides for FEC protection up to above $10^{-3}$ packet loss in both the random and REIN loss models
- The Pro-MPEG COP3 code with constant sending rate performs close to the ideal code whenever PLR remains under a threshold value around 10-4.and only in the case of random loss this is the case since the Pro-MPEG row code is a simple parity code, which is optimal when only one packet of protection data is needed per block).
- Around $10^{-4}$ packet loss rate for the random loss case, the Pro MPEG code requires higher overhead – around 34% for the 2Mbit/s stream and 20% for the 6Mbit/s stream.
- Depending on the sending arrangement, above around 3 x $10^{-4}$ packet loss for the REIN case, no settings for the Pro-MPEG code which supported the required quality target (measured in mean time between packet losses) could be found. Nevertheless, when using a slightly lowest quality target (same time but measured in mean time between FEC blocks with errors), it is possible to find Pro-MPEG settings to support the required quality target.
- The burst arrangement for the Pro-MPEG code requires somewhat less overhead at high loss rates, although still significantly more than Raptor.
- The burst sending arrangement for the Pro-MPEG code offers significant improvements in the REIN case – in fact improving on the ideal block code (which uses a constant sending arrangement).
- The choice of burst or constant sending arrangement for Raptor makes little difference in the required overhead
- The burst sending arrangement for Pro-MPEG does not allow the quality target to be achieved in the REIN case across the whole loss range. It should be noted that simulation based on a lower quality target *can* be met by ProMPEG.

It should be noted that in the above cases the parameters for the Pro-MPEG code were selected to provide the best performance for each particular loss rate and pattern through a wide search of the possible parameter set. In practice, we expect loss rates and error patterns to be largely unknown in advance.

In particular, for the REIN cases, the Pro-MPEG column code with a number of columns equal to the burst length provides adequate protection so long as events with two error bursts within a protection period happen only once every four hours or less.

This may happen when the overall loss rate is high or when there is strong correlation between bursts. Moreover if random single loss errors happen very close to a burst, they may not be corrected neither.

## B3.3 Unicast case

### B3.3.1 Stored/buffered content

In these cases, content is available at the server in advance of sending to the user: for VOD services the content is stored in its entirety and for live broadcast in trick modes the content is buffered for at least a few hundred ms when the user activates the trick mode by pausing the multicast broadcast.

In these cases the Raptor code incorporates a fast buffer fill technique (called “faststart” in this paper) which allows the protected block size to be gradually increased over the first few seconds of transmission. Note that this technique is possible only because of the independence of block size and overhead supported by Raptor and the possibility to flexibly vary the overhead in single packet increments without impacting the error correction performance of the code.

As above, repeated 96 hour simulations were performed with the FEC overhead again increased for each simulation until the target quality was achieved. The fast-start procedure is repeated every 10 minutes during the simulation to model the impact of repeated channel change or use of trick-modes.

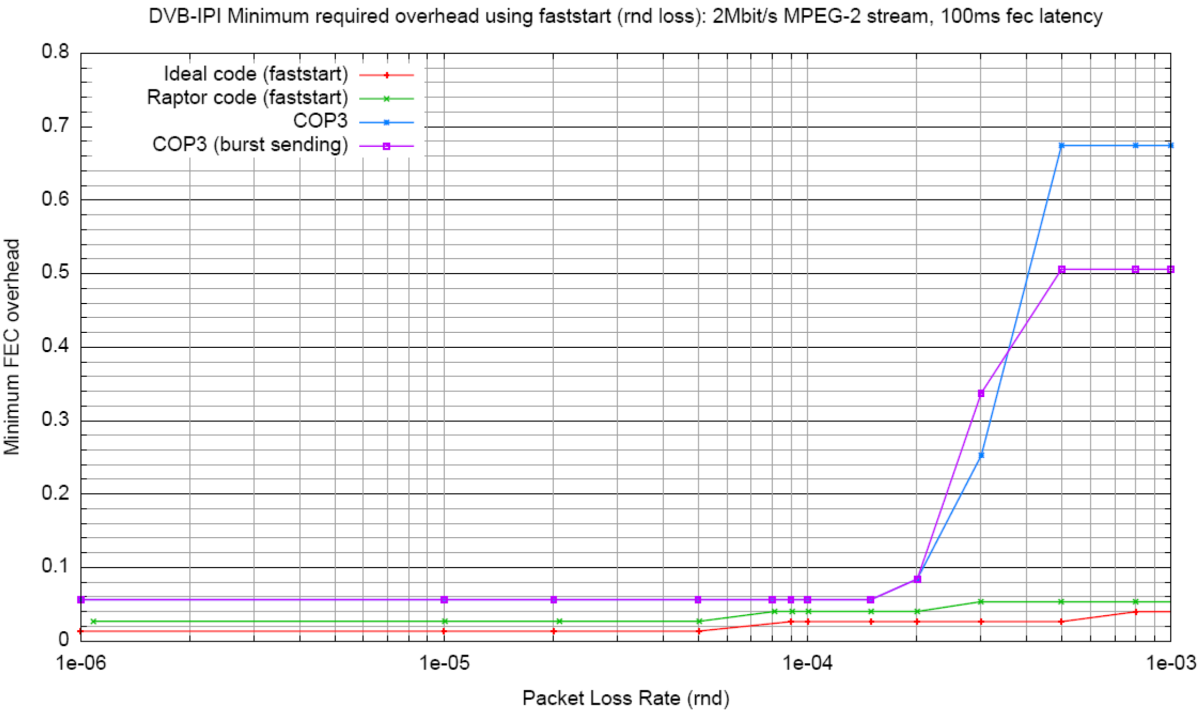


Figure 14: 2Mbit/s MPEG-2 Transport Stream, 100ms latency (stored/buffered content), random loss

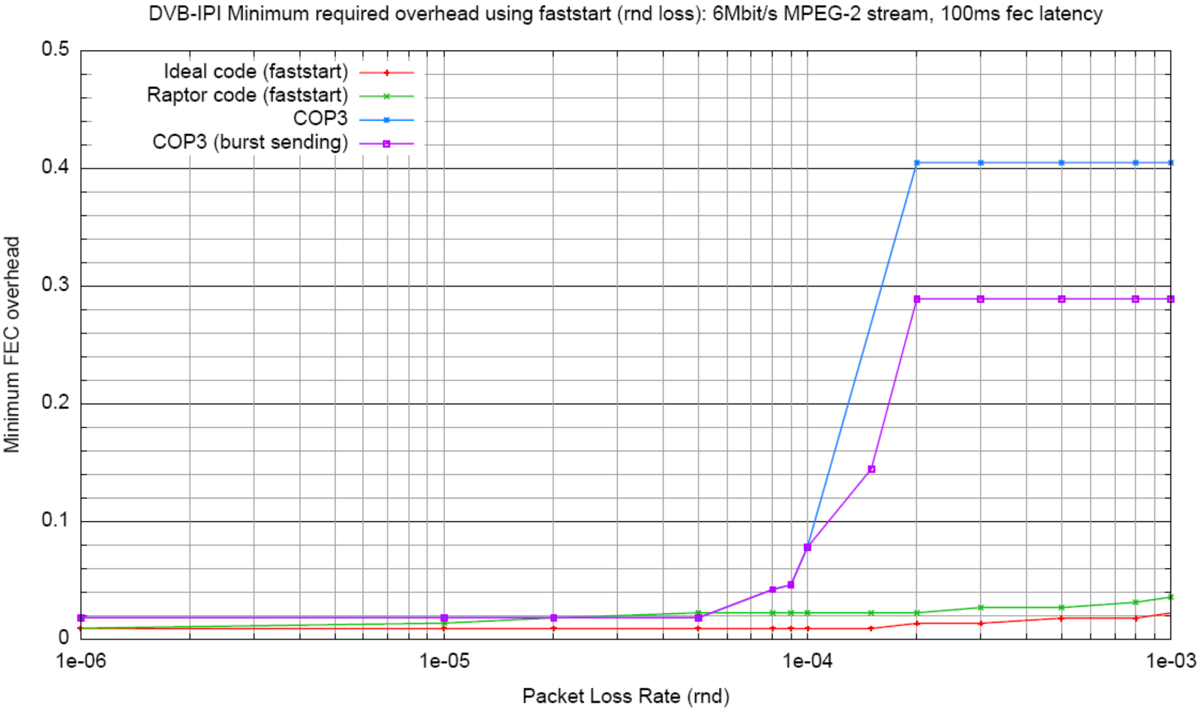


Figure 15: 6Mbit/s MPEG-2 Transport Stream, 100ms latency (stored/buffered content), random loss

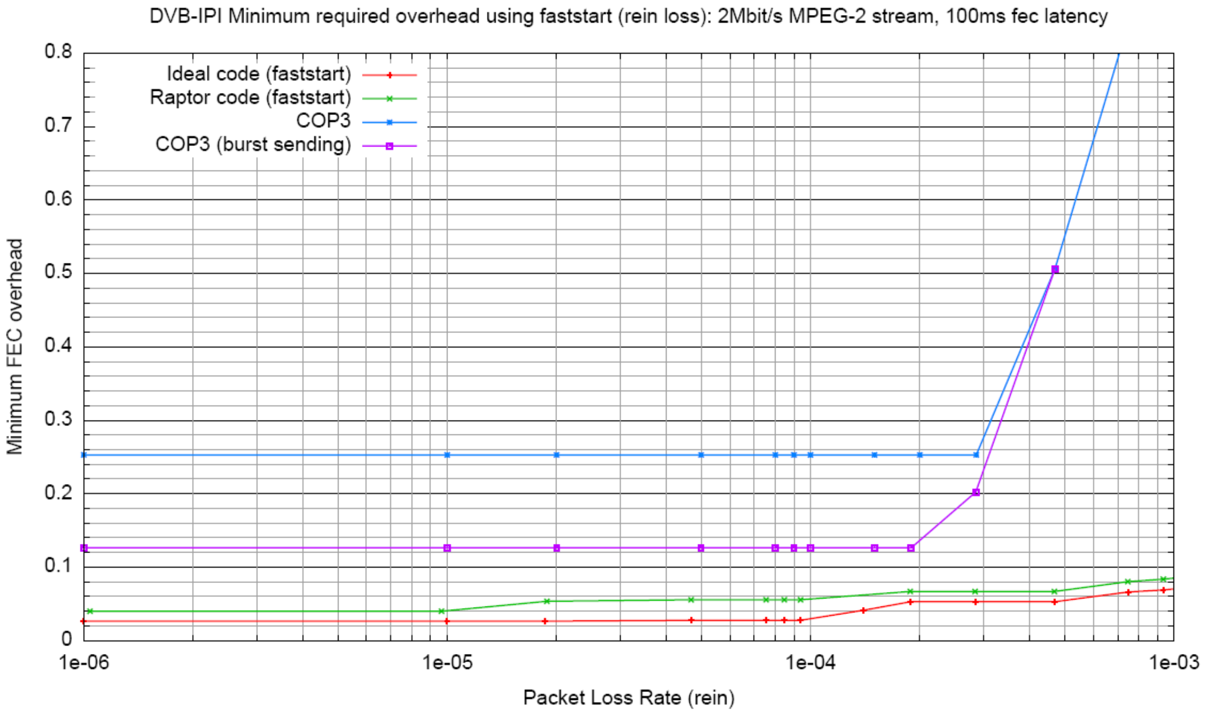


Figure 16: 2Mbit/s MPEG-2 Transport Stream, 100ms latency (stored/buffered content), REIN

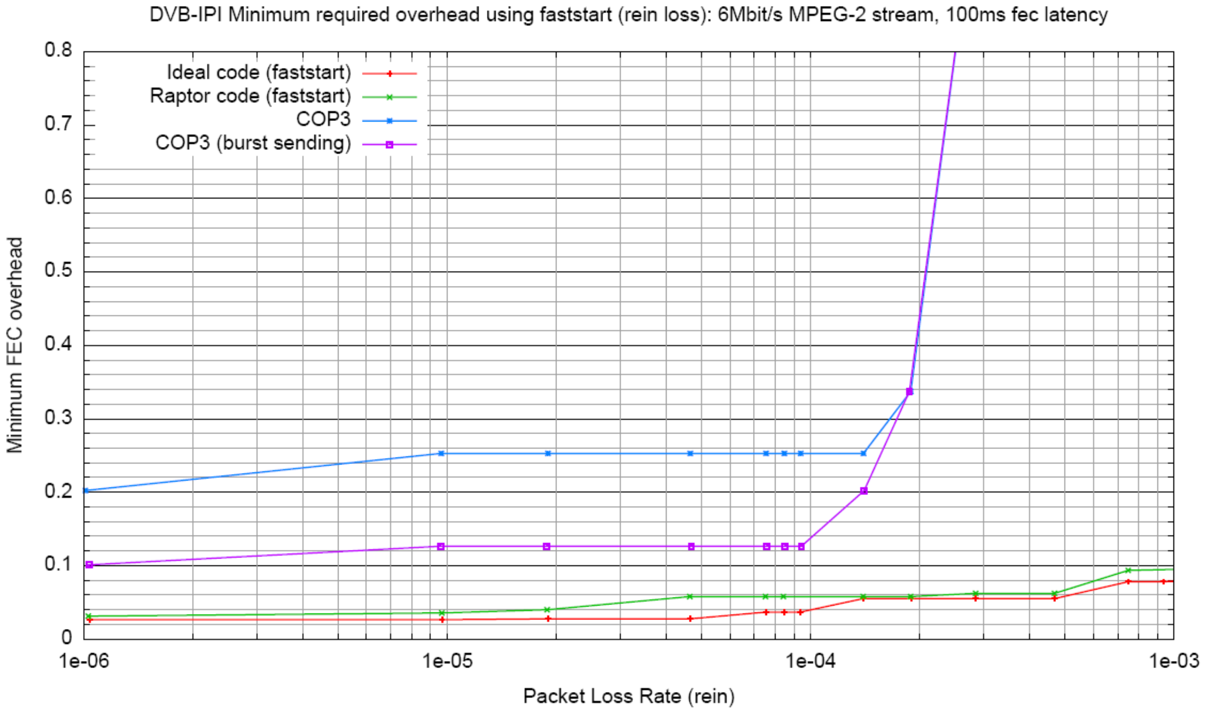


Figure 17: 6Mbit/s MPEG-2 Transport Stream, 100ms latency (stored/buffered content), REIN

### B3.3.2 Live content

In the case of unicast delivery of live content (for example in networks which don’t support multicast) then the block size for the Raptor code is limited by the requirement of a maximum latency due to FEC of 100ms. The following figures show simulation results for this case.

#### B3.3.2.1 Constant sending arrangement

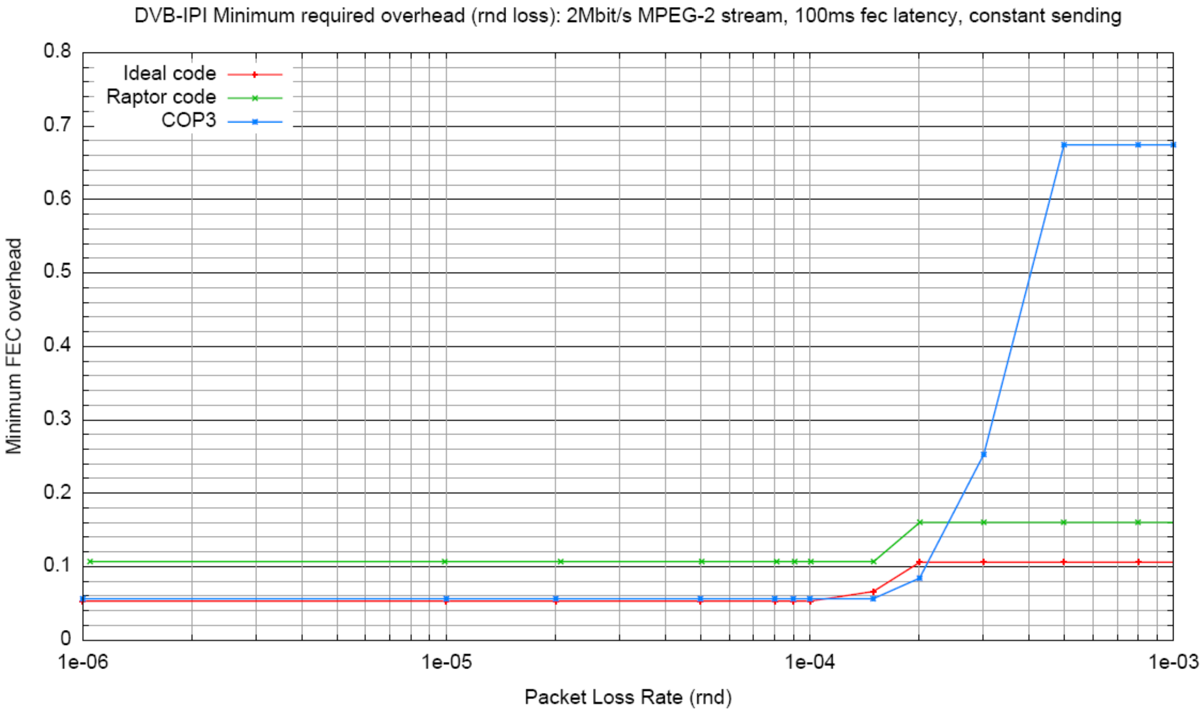


Figure 18: 2Mbit/s MPEG-2 Transport Stream, 100ms latency (live content), random loss, constant sending

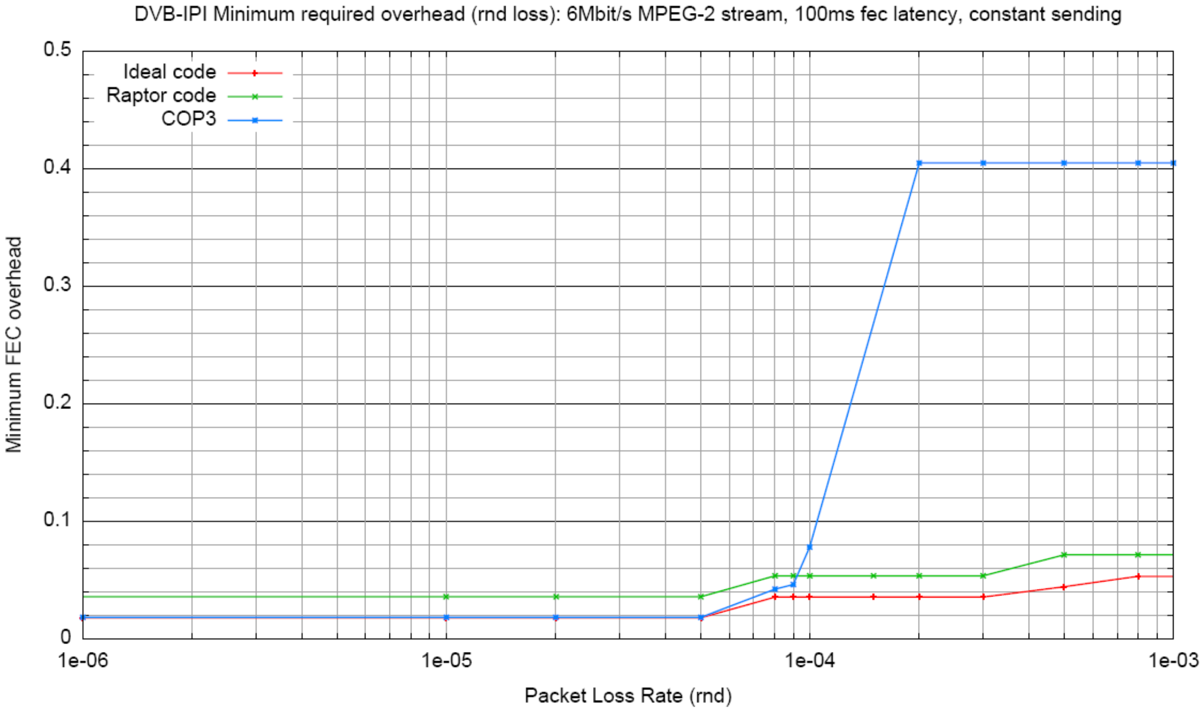


Figure 19: 6Mbit/s MPEG-2 Transport Stream, 100ms latency (live content), random loss, constant sending

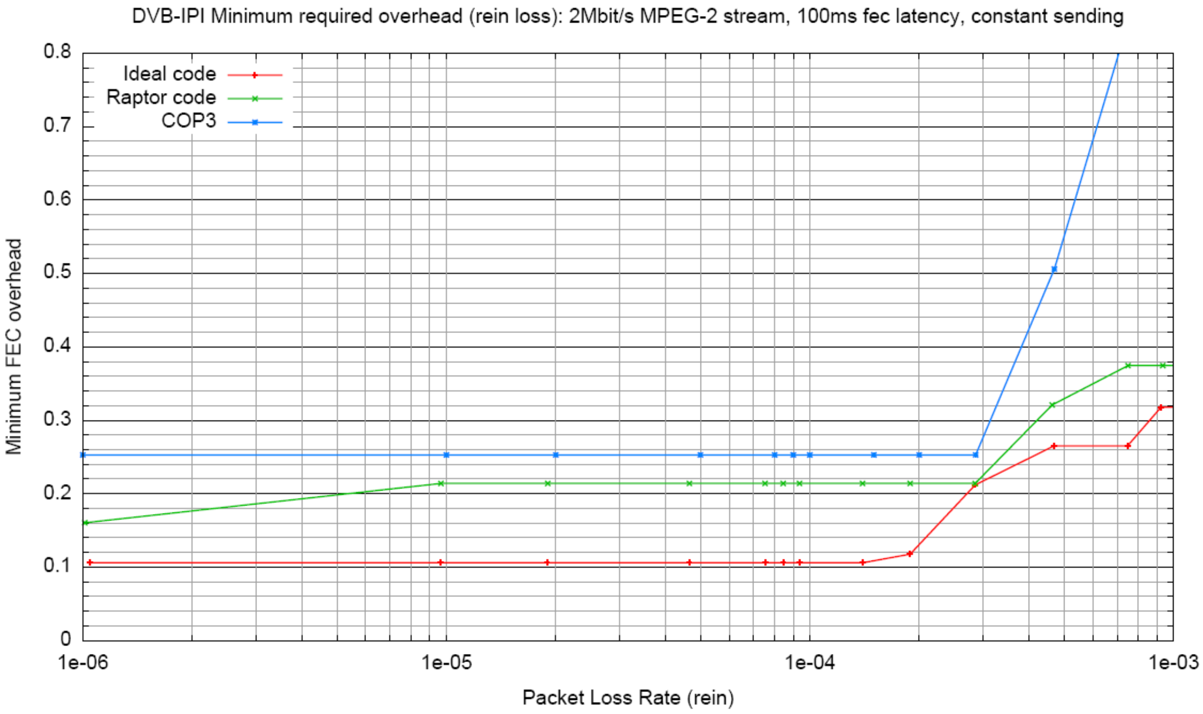


Figure 20: 2Mbit/s MPEG-2 Transport Stream, 100ms latency (live content), REIN, constant sending

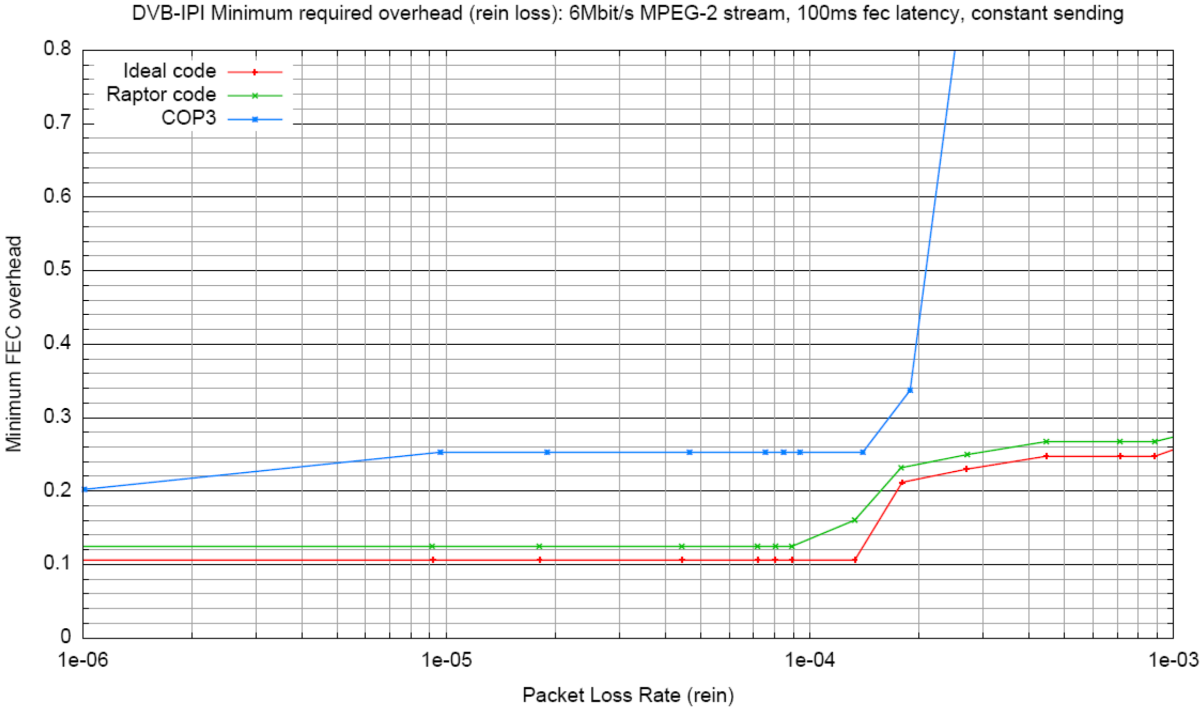


Figure 21: 6Mbit/s MPEG-2 Transport Stream, 100ms latency (live content), REIN, constant sending

### B3.3.2.2 Burst sending

NOTE: Curves for the “Ideal” block code and Raptor below are for constant rate sending, compared with burst sending for Pro-MPEG.

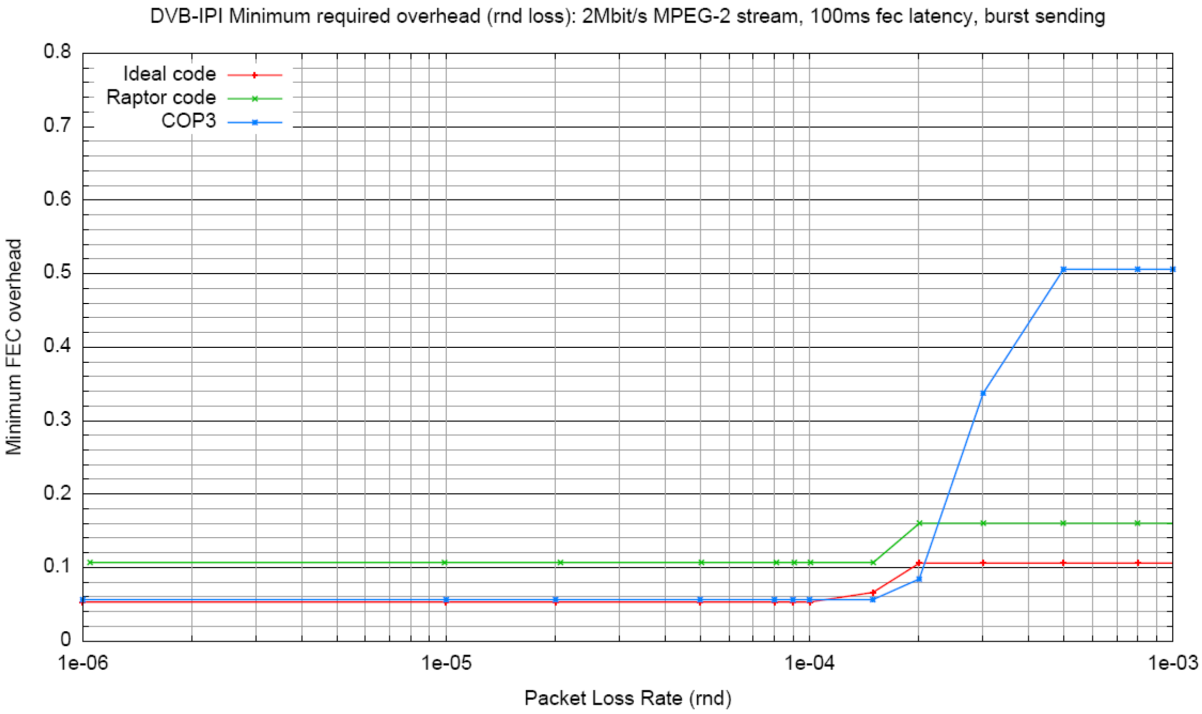


Figure 22: 2Mbit/s MPEG-2 Transport Stream, 100ms latency (live content), random loss, burst sending

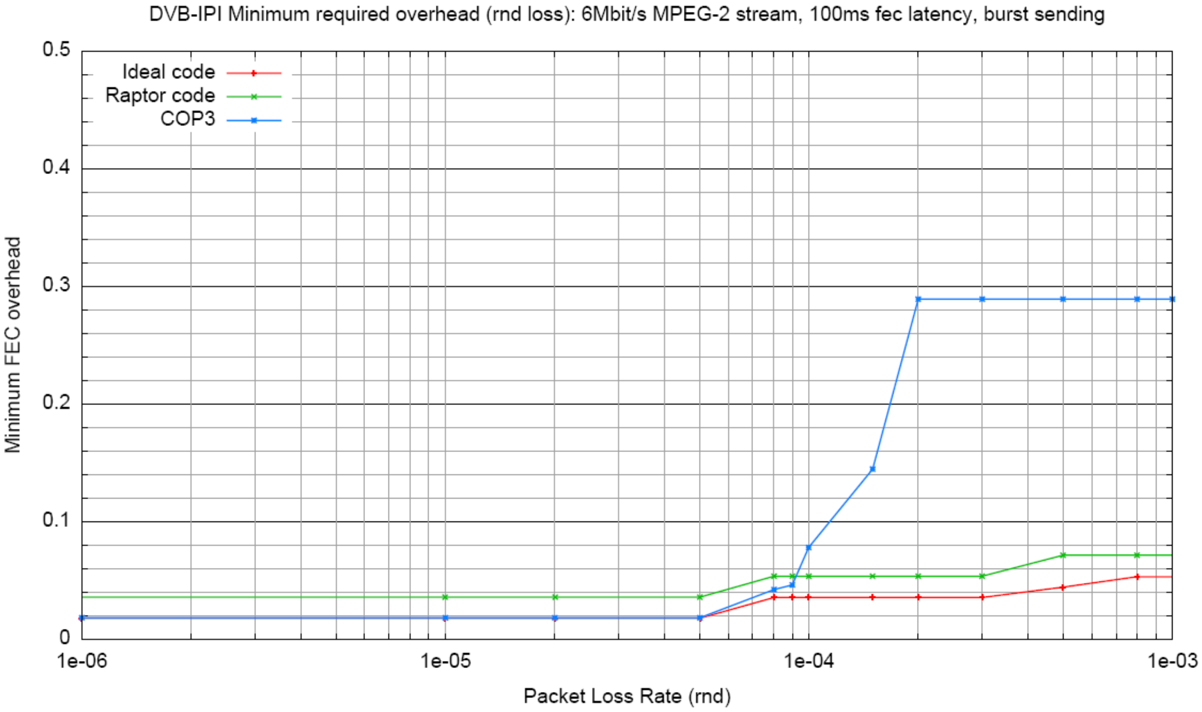


Figure 23: 6Mbit/s MPEG-2 Transport Stream, 100ms latency (live content), random loss, burst sending

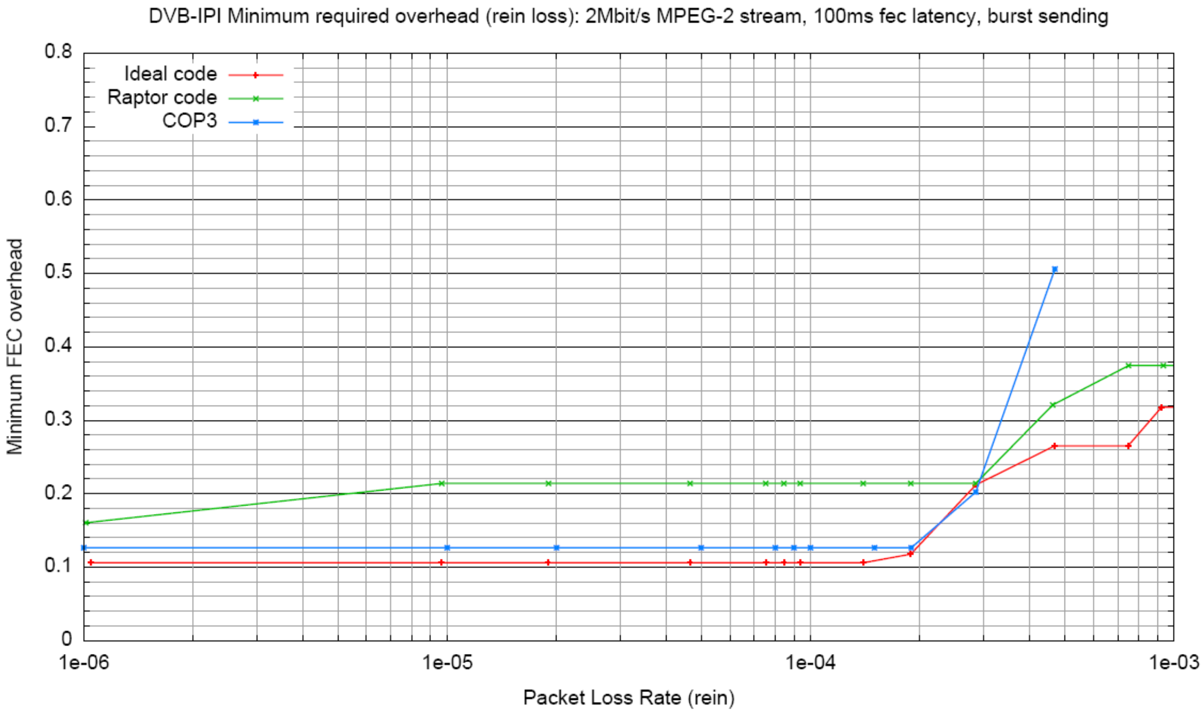


Figure 24: 2Mbit/s MPEG-2 Transport Stream, 100ms latency (live content), REIN loss, burst sending

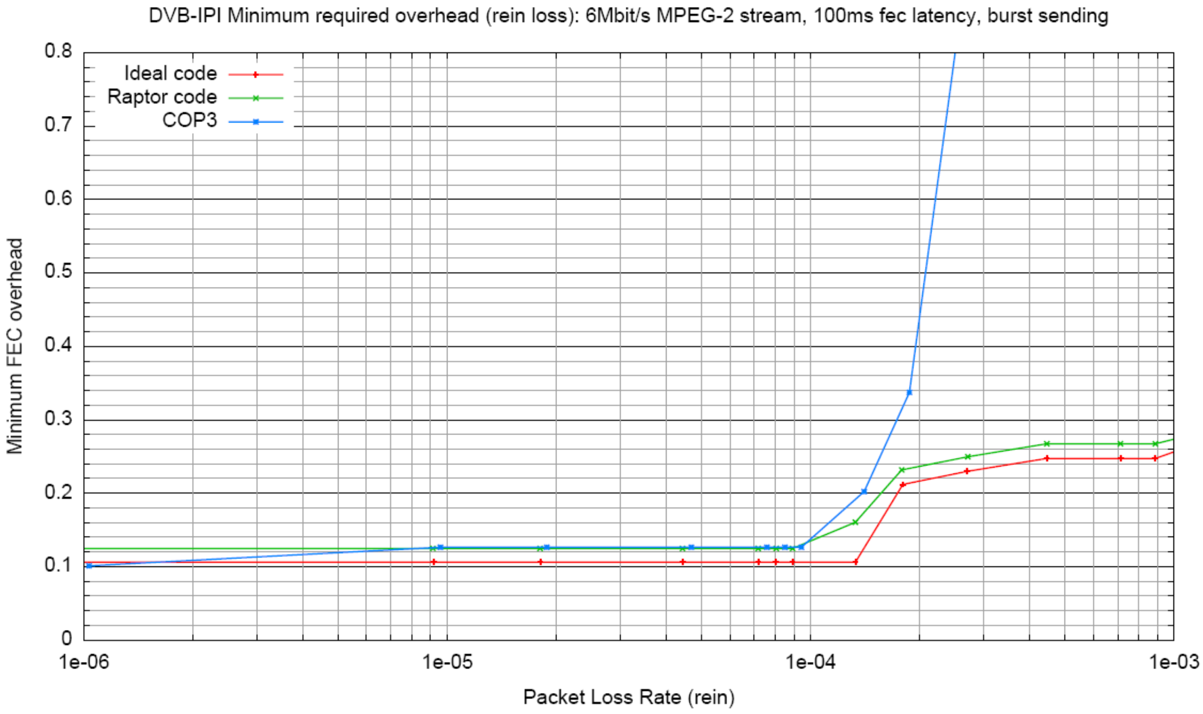


Figure 25: 6Mbit/s MPEG-2 Transport Stream, 100ms latency (live content), REIN loss, burst sending

As in previous cases, the Raptor code meets the quality target at all error rates with overhead close to the minimum possible. The Pro-MPEG code meets the quality target with minimum overhead only in cases where the loss rate is below a threshold which is around $10^{-4}$ packet loss rate.

With the constant sending arrangement, and REIN losses, the Raptor codes requires an overhead which is less than or (approximately) equal to the Pro-MPEG overhead for all loss rates. For other cases (burst sending and/or random loss) the Pro-MPEG code requires marginally less overhead for the loss rates which are below the threshold.

For low loss rates and in the presence of random loss, the ProMPEG code is simple a 1D parity code, which is well known to be ideal. In these cases ProMPEG achieves lower overhead than Raptor.

## B3.4 A note on latency, jitter and traffic shaping

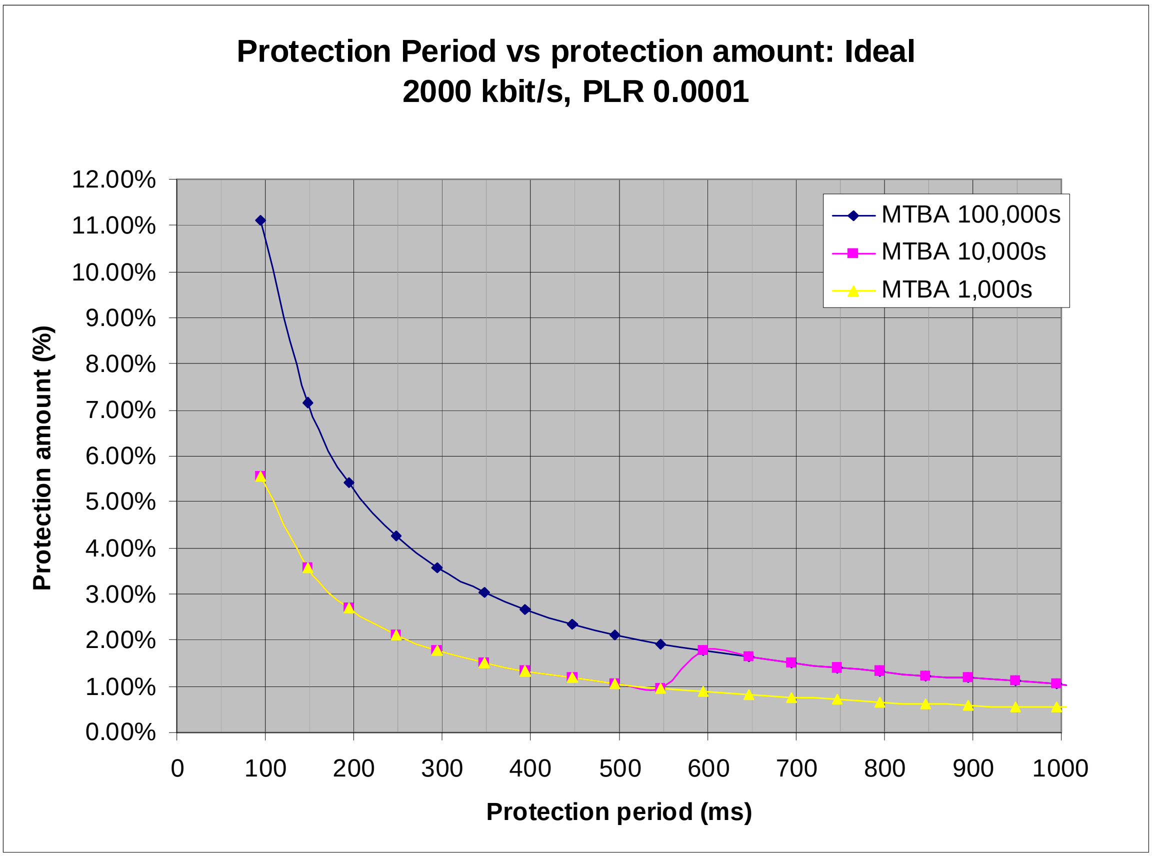
All the above simulations assume that the sent traffic should maintain a constant bit-rate (although it is accepted that the constant–bitrate ProMPEG scheme actually doubles the instantaneous bit-rate each time a repair packet is sent, this is only visible as a variation in bit-rate over very short time periods. However for the burst sending arrangement, the variation is significant and over a longer period of time).

In order to support legacy receivers in the case of multicast, whenever this is feasible, the use of FEC should not introduce significant additional jitter in the source packets. Using the sending arrangement proposed for Raptor codes does introduce a small amount of additional jitter to the arrival of source packets at the receiver. Using the constant sending arrangement proposed for Pro-MPEG avoids such jitter, however using the burst sending arrangement proposed for Pro-MPEG will introduce a small amount of additional jitter as the bursts are traffic shaped on the access link. Sending arrangements are interchangeable between the codes, so there are many possibilities - See Annex A for more details.

In the simulations above, the maximum additional jitter in the case of Raptor is around 40ms for the 400ms latency cases and in most cases significantly less. Finally, “latency” in these simulations has been interpreted as the additional latency introduced between the source and the playout due to the use of FEC. This is equivalent to the size of the FEC data buffer assumed to exist at the receiver. This figure adds directly to the response time for user actions, such as channel change, re-wind, forward-wind etc.

In the case of live content, the Raptor scheme as proposed adds a small additional amount to the time between the event actually occurring at the sender and the presentation to the user (distinct from the response time for user actions, referred to above). In the cases above this is at most around 40ms and in general considerably less. Since the overall end-to-end delay is general much higher than 40ms, this additional delay is not considered significant, especially since it does not contribute to the response time for user actions. The Raptor scheme is sufficiently flexible that this delay could be reduced if required. Targets on this end-to-end delivery time have not been discussed and again could be included in a further phase of this evaluation if necessary, but again it is unlikely to significantly affect the results.

Finally, the only two latency figures (100ms and 400ms) were tested in these evaluations. It is instructive to consider the trade-off involved in selection of an FEC latency figure. Lower latency results in shorter channel change time but has a cost in that a higher FEC overhead is required for a given level of protection. Conversely, a longer latency budget results in longer channel change time in return for a lower FEC overhead. Figure 26 illustrates this trade-off for an “ideal” code and for several quality targets (“Mean Time Between Artifacts”). This figure suggests that a significant bandwidth saving is available if the latency budget is increased from 100ms to (say) 200ms, but that there is little to be gained by increasing the latency above 400ms. In particular, this figure throws doubt on the practical validity of the 2MBit/s, 100ms case evaluated above: an operator who was sufficiently bandwidth-constrained to use 2Mbit/s encoding would surely also take advantage of the FEC bandwidth savings that could be achieved with a 200ms latency budget.



Figire 26: latency/FEC bandwidth trade-off

## B3.5 Summary of simulation results

We summarise the above results according to the sending arrangement and type of loss:

**Summary for multicast and unicast live video:**

- There is a “loss rate threshold” in each case: below this threshold, the Pro-MPEG overhead is very low and close to Raptor (sometimes higher, sometimes lower) and above this threshold, the Pro-MPEG overhead is significant (always much higher than Raptor overhead).
- The threshold is around 1e-4 Packet Loss Rate (actually between 5e-5 and 2e-4), depending on the case

Constant sending arrangement, random loss

- Below the threshold, the Pro-MPEG overhead is slightly less than the Raptor overhead and above this threshold, the Raptor overhead is much less than the Pro-MPEG overhead.

Constant sending arrangement burst (REIN) loss

- Below the threshold, the Raptor overhead is slightly less than the Pro-MPEG overhead and above this threshold, the Raptor overhead is much less than the Pro-MPEG overhead. Please note that in this case, Raptor overhead is always the lowest.

Burst sending arrangement, random loss

- Burst sending does not have much effect on results below the threshold.
- The Pro-MPEG overhead is reduced above the “threshold” compared to constant sending arrangement, but is still much greater than the Raptor threshold
- Burst sending does not have much effect on the Raptor overhead

Burst sending arrangement, burst (REIN) loss

- The Pro-MPEG overhead is reduced both above and below the “threshold”, but above the threshold is still much greater than the Raptor threshold
- Below the threshold the Pro-MPEG overhead is slightly less than the Raptor overhead
- Burst sending does not have much effect on the Raptor overhead

**Summary for unicast stored or buffered content:**

- In the particular case of unicast stored or buffered content, Raptor code can use the faststart sending arrangement so as to use significantly less bandwidth than Pro-MPEG in all cases.
- When faststart mechanism is not used, results are the same as multicast and unicast live video.

In all cases, the results plotted above show the overhead required by the “best” configuration parameters for the Pro-MPEG COP3 code according to guidelines for setting Pro-MPEG parameters and the specification in [3]. These were chosen by searching through the various possible configurations (including row packets only, column packets only, both row and column packets and different matrix sizes) and reporting only the lowest overhead which achieved the required quality. This means that the choice of code was based implicitly on complete knowledge of the loss rates and patterns in each case.

In summary, the requirements on network quality (target end-to-end loss rates) depend significantly on the choice of FEC code (Pro-MPEG or Raptor): network quality requirements are much more stringent if Pro-MPEG is chosen since it works well only as long as the packet loss rate remains under the previously defined threshold (around 1e-4).

## B4 Flexibility

The FEC evaluation criteria for flexibility states:

”Flexibility:

- Changing the overhead or/and the block size dynamically (within or between FEC blocks)
- Range of protection periods
- Suitability for use with a wide variety of FEC sending strategies “

The Raptor code provides complete flexibility in terms of overhead (protection amount) and block size (protection period). These parameters can be set independently according to application requirements and the error correction performance of the code remains just as close to ‘ideal’ whatever the parameter settings. Parameter settings can easily be changed dynamically and protection periods from 10s to 1000s of milliseconds can be efficiently supported.

For the Pro-MPEG code, the protection period and protection amount are related and constrained and in practice only certain combinations are supported Nevertheless, the possible number of combinations is large enough to offer many different levels of protections.

# B5 Processing and Memory requirements

The Raptor code has been designed to have very modest computational complexity such that it is easy to implement in software on resource constrained devices such as Set-Top Boxes and mobile devices. Techniques for efficient hardware implementation for high capacity encoders have also been presented and many options exist for hardware-assisted implementations for decoders.

The Pro-MPEG code has been designed to have very low computational complexity such that it is easy to implement it in software or in hardware.

For both Raptor and Pro-MPEG, the complexity of encoding is comparable with the complexity of decoding. For Raptor, both scale linearly with the volume of data to be encoded/decoded, making the overall computational requirements proportional the service bit-rate and to a large extent independent of the losses or level of protection.

Raptor encoding complexity for the scenarios considered here is in the region of 2 MIPS per Mbit/s – so a 6Mbit/s stream would require ~12 MIPS of processing power to encode, although in practice the encode time is also dependent on memory bus speed and cache/DMA availability. For example, Digital Fountain has demonstrated an off-the-shelf rack-mounted server with a Pentium processor running at 3 GHz performing Raptor encoding at 2Gbit/s – the equivalent of 1000 2Mbit/s video streams. Further optimizations for the specific case of video stream encoding and platform-specific optimization could be expected to increase this encoding speed significantly. Leading Pro-MPEG COP3 processing cards encode at around 400Mbit/s and so similar performance could be easily achieved with Raptor with modest processing requirements.

Hardware optimizations of Raptor codes in the form of hardware assist for XOR operations or complete implementation of the code in hardware are also possible and can further improve capacity. The application of the Raptor code for streaming has been designed so that for a given stream rate/latency the block size and structure from the encoders point of view is the same for every block. Thus the sequence of operations required to encode repair packets for a block can be calculated or stored in advance and executed quickly (in software or hardware) for each block. This is true even if the actual block size (in terms of packets) differs between protection periods.

The number of primitive symbol XOR operations required for Raptor encoding or decoding for the scenarios considered here is around 12-14 operations for each source symbol.

The number of primitive symbol XOR operations required for Pro-MPEG encoding or decoding for the scenarios considered here is 1 operation for each source symbol in Pro-MPEG 1D and 2 operations for each source symbol in Pro-MPEG 2D.

Nevertheless, in practice, for each symbol, these operations are performed on-chip (in cache) and so the bottleneck is the speed with which data can be moved between memory and the processor, rather than the precise number of XOR operations. All modern processors employ pipelining and so can perform the XOR operations on-chip concurrently with moving data for future operations between off- and on-chip memory. This means a reduction in XOR operations doesn’t necessarily translate into a significant increase in speed of encoding or decoding.

With Raptor, minimum memory requirements for data to be encoded/decoded at both encoder and decoder are slightly greater than the source block size. At the decoder, received data (which is a mix of source data and repair data) may be transformed “in-place” into the recovered source block. Thus, these memory requirements are less than 350KB for the largest block size considered in this evaluation.

With Pro-MPEG, the encoder only needs to have buffers so as to store the repair packets of a protection block. Since amount of protection is always much lower than the amount of data, it means a Pro-MPEG encoder requires memory much smaller than the source block size. On the decoder, Pro-MPEG only requires enough memory to store the current protection block and its repair packets. Therefore it means a Pro-MPEG decoder requires memory slightly greater than the source block size.

Note also, that depending on the sequencing arrangement used, the decoder may need more memory. For instance, when repair packets are arranged within the block after the one they protect, the decoder would need twice as much memory to store the current and following protection blocks.

Note that for decoders, this memory requirement is still very modest compared to the memory required, for example, for storing a single HD frame after decoding.

## B6 Support of layered transmission

The evaluation criteria include “Ability to add or remove FEC correction packets“ and further discussions have highlighted interest in layered transmission of FEC data so that unnecessary sending of unnecessary FEC data over DSL lines with low error rates can be avoided even in the multicast case.

The Raptor code is capable of generating an effectively unlimited supply of FEC repair packets for any given block size. The incremental computational cost for generating additional repair packets is very low. Raptor repair packets are generated by a pseudo-random process, meaning that different packets are essentially equivalent to receivers in terms of their usefulness for decoding. As a result such packets can be distributed amongst IP Multicast groups in an arbitrary fashion and receivers can join or leave according to the quantity of repair packets they require.

The above mechanism cannot be implemented with the same efficiency/flexibility using the Pro-MPEG code, since repair packets are generated according to a specific simple structure, are limited in supply and are sent to the same multicast group. Receivers need to be sent all packets of a given Pro-MPEG code to achieve the error correction performance illustrated above – listening to only a partial transmission will result in much poorer performance compared to listing to a transmission which was initially designed with a lower overhead.

## B7 Additional criteria

The following additional criteria are included in the evaluation criteria document:

- Continued functioning of existing STB products in presence of FEC data
- Option for new STB products to use or ignore FEC data
- Confirmation of FEC scheme IPR compliance with DVB rules
- Support of combined protection of different streams (such as when audio and video packets are sent in two separate streams)

Raptor is compliant to all these criteria.

Pro-MPEG is compliant to the first two criteria and believed to be compliant to the third (IPR compliance is currently being clarified by SMPTE).

The Pro-MPEG code does not support combined protection of different streams – separate protection streams are required for each RTP flow. Specifically in the case of audio streams, which have much lower bandwidth than the video streams, then high quality protection will be extremely difficult to achieve if latency needs to be kept very small.

In general, combined protection is more efficient than separate protection and in particular separate protection of the relatively low bit-rate audio stream can be extremely inefficient.

Combined protection can also encompass the RTCP packets that provide time synchronisation information between the audio and video streams.

# B8 Content Download

It has been suggested that the FEC solution chosen for streaming services should also be suitable for use in content download applications. It should be noted that it has not yet been agreed, (or even discussed in detail), that Forward Error Correction is required for Content download - other solutions do exist. An evaluation of these solutions should be carried out by the TM-IPI Content Download System (CDS) taskforce.

However, solutions based on forward error correction have a number of significant advantages over other solutions in the multicast case. The Raptor code proposed for DVB-IPI streaming applications is highly suitable for content download applications as well (and has been adopted for such applications by 3GPP and DVB CBMS). The same code could therefore be used for both streaming and content download.

No description is available of whether and how the Pro-MPEG code could be applied to content downloading: it was clearly designed for streaming services in extremely low packet loss cases only. The Pro-MPEG code is by nature a short block code and for content downloading a large block code is much more efficient if FEC is to be used.

# B9 Raptor vs Pro-MPEG Summary

The table below summarises the results described above. The green font identifies the best result while the red font identifies the worst result. When the result between codes is very close, an orange font is used to identify the code that only performs slightly less well.

| **Criteria** | **Pro-MPEG Constant** | **Pro-MPEG Burst** | **Raptor** | **Comments** |
|---|---|---|---|---|
| **Bandwidth cost – loss rates >~1e-4** | | | | |
| - SD MPEG-2 TS broadcast (400ms) | **High** | **High** | **Low** | |
| - HD MPEG-2 TS broadcast (400ms) | **High** | **High** | **Low** | |
| - SD MPEG-2 TS unicast (100ms) | **High** | **High** | **Low** | Thanks to its fast-start mechanism, Raptor achieves very low overhead in case of stored/buffered content |
| - HD MPEG-2 TS unicast (100ms) | **High** | **High** | **Low** | Thanks to its fast-start mechanism, Raptor achieves very low overhead in case of stored/buffered content |
| **Bandwidth cost – loss rates <~1e-4** | | | | |
| - SD MPEG-2 TS broadcast (400ms) | Low | **Lowest** | Low | |
| - HD MPEG-2 TS broadcast (400ms) | Low | **Lowest** | Low | |
| - SD MPEG-2 TS unicast (100ms) | **Modest** | **Lowest** | Low | Thanks to its fast-start mechanism, Raptor achieves very low overhead achieved in case of |

| | | | | |
|---|---|---|---|---|
| | | | | stored/buffered content |
| - HD MPEG-2 TS unicast (100ms) | **Modest** | **Lowest** | **Low** | Thanks to its fast-start mechanism, Raptor achieves very low overhead achieved in case of stored/buffered content |
| | | | | |
| Support of target quality for evaluated packet loss range/patterns | **See comment** | | **Yes** | Pro-MPEG COP3 could not provide a Mean Time Between Packet Loss of 4 hours for a number of the burst loss cases. However, a slighty weaker target of Mean Time Between Artifacts (visible errors) of 4 hours could be achieved. |
| Further packet losses that could occur in the core network due to congestion and/or the home environment e.g. wireless technologies. | **-** | | **-** | Not yet evaluated |
| Flexible engineering of code parameters | **Yes (but fixed number of combinations and direct correlation between overhead and protection block size)** | | **Yes (fully)** | |
| Computational complexity | **Lowest** | | **Modest** | |
| Scalability (e.g. encoding of 1000s of streams) | **Yes** | | **Yes** | |
| Memory requirements (encoder) | **Lowest** | | **Modest** | |
| Memory requirements (decoder) | **Modest** | | **Modest** | |
| Visibility of artifacts after FEC decoding | **-** | | **-** | Both codes could perform partial correction. |
| Continued functioning of existing STB products in presence of FEC data | **Yes** | | **Yes** | |
| Option for new STB products to use or ignore FEC data | **Yes** | | **Yes** | |
| Confirmation of FEC scheme IPR compliance with DVB rules | **Yes** | | **Yes** | Pro-MPEG IPR compliance is currently under SMPTE process. |
| Efficient support of direct encapsulation of audio/video in RTP (as defined in TS 102.005) : Support of combined | **No** | | **Yes** | Raptor can protect several RTP and RTCP streams |

| | | | |
|---|---|---|---|
| protection of audio and video packets | | | together whereas Pro-MPEG has to consider each RTP and RTCP streams separately. |
| Efficient support of direct encapsulation of audio/video in RTP (as defined in TS 102.005) : support of variable length packets | **Yes (but less efficient)** | **Yes** | |
| Suitable for Content Download Service | **No (much less efficient)** | **Yes** | |

## B10 Conclusions

The sending arrangement chosen has a significant impact on the performance / bandwidth cost.

The comparison of the two codes also differs depending on the packet loss rate.

In the case that burst sending is used and for loss rates below a threshold (between 5e-5 and 2e-4), the Pro-MPEG code requires slightly less bandwidth than Raptor code.

In the case that burst sending is not used and for loss rates below a threshold (between 5e-5 and 2e-4), both Pro-MPEG and Raptor codes requires similar bandwidth overhead although there are differences depending on the precise case (see section B3.5).

For loss rates above a threshold (between 5e-5 and 2e-4), Raptor code requires much less bandwidth than Pro-MPEG code.

The threshold indentified through these simulations depends on quality target, source stream bitrate, latency budget and loss patterns.

When the Raptor fast-start mechanism is used for unicast/buffered content, Raptor requires less overhead than Pro-MPEG.

Regarding implementation aspects (complexity, memory requirements,…), though there are differences between codes (see section B9), no significant issues were identified with either code.

Both codes meet the requirement for backward compatibility with existing equipments.

The Raptor code support various future requirements which the Pro-MPEG does not (see section B9).

Since neither of these two codes is optimal in all cases, an hybrid code with performance similar to the best of either was defined (see Appendix 1 for simulation results).

# Annex A Sending arrangements

## A.1 DF Raptor default sending arrangement

The sending arrangement proposed for the DF Raptor code is illustrated in Figure A1. In this sending arrangement the overall sending rate is kept constant and the source packets of each block are sent before any of the repair packets of the block. This approach requires that the sending rate of the source packets be increased marginally to make space for the repair packets at the end of the block.

It's important to note that the sequencing of packets is determined by the FEC procedures which operate "below" the RTP layer. The contents of the packets, in particular the RTP timestamps, are not modified compared to the contents in the case in which FEC is not applied and therefore the correct timing for the packets can be reconstructed with the usual procedures.

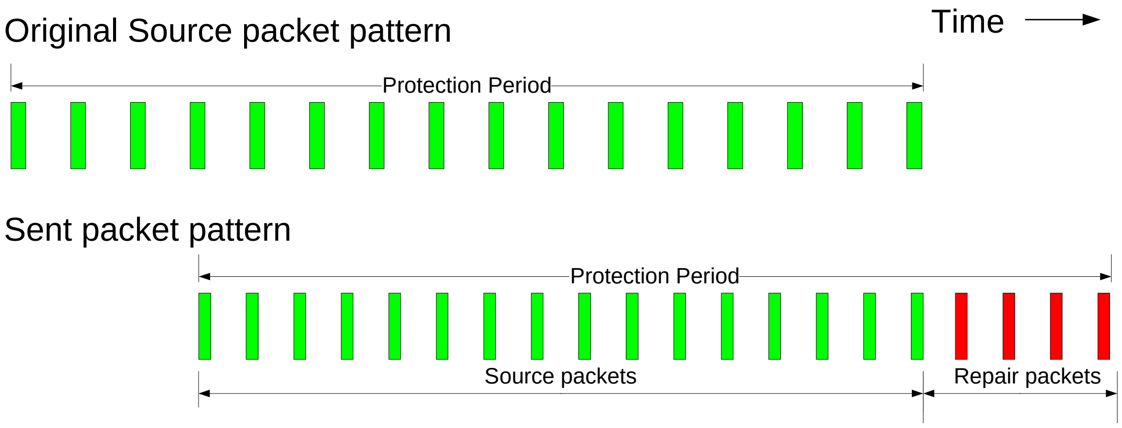


Figure A.1: DF Raptor sending arrangement

Note that while this arrangement ensures a global constant bitrate, it actually modifies the rate at which source packets are sent and consequently creates a small amount of additional jitter on the transmission.

Other sending arrangements are also possible for DF Raptor but were not investigated.

Pros and cons:

+ global sending rate is constant
+ full latency budget available for FEC protection
- source data sending rate is different from original source data sending rate
- insertion of repair packets introduces small amount of jitter on all source packets

## A.2 Pro-MPEG COP3 fully interleaved sending arrangement

Annex C of the Pro-MPEG specification proposes a sending arrangement as illustrated in Figure 2. In this sending arrangement the overall sending rate is kept constant and the sending rate of source packets is also kept constant.

Because this sending arrangement distributes repair packets for one block over the entire duration of the next block, then the maximum block size is limited to one half of the latency budget. As a result, the overhead required by the code is increased. This is illustrated in the "constant sending arrangement" results above.

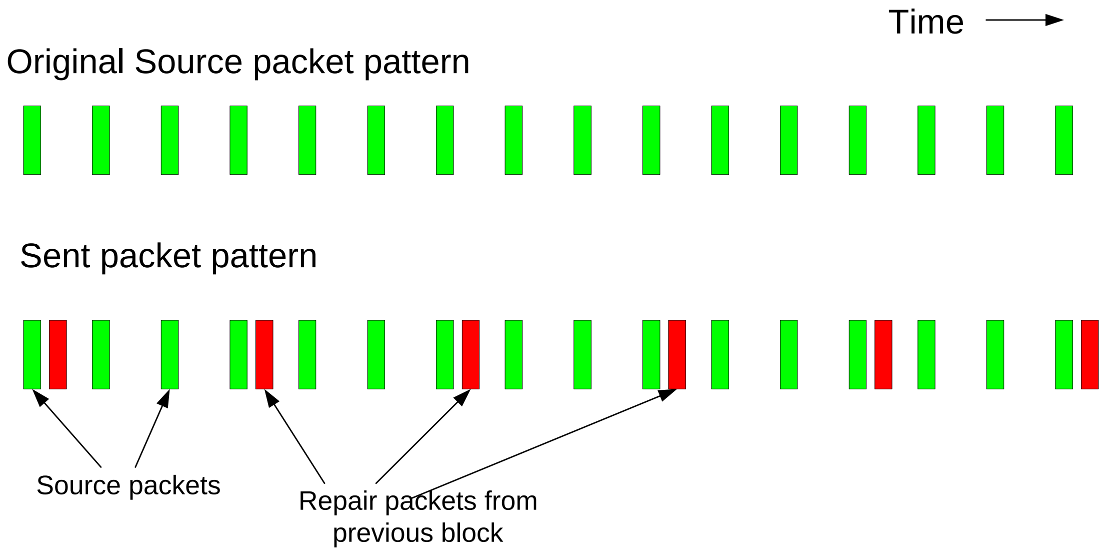

Time
Original Source packet pattern
Sent packet pattern
Source packets
Repair packets from previous block

Figure A.2: Pro-MPEG COP3 fully interleaved sending arrangement

Pros and cons:

- + source data sending rate is the same as original source data sending rate
- + global sending rate is kept constant
- - only half of latency budget is available for FEC protection
- - insertion of repair packets introduces very small amount of jitter at the beginning when total stream bandwidth is close to available channel bandwidth

## A.3 Pro-MPEG COP3 burst sending arrangement

This arrangement is illustrated in Figure A3. In this case, repair packets for one block are interleaved with the first few packets of the next block. As a result, the instantaneous sending rate during these first few packets is significantly increased. However, the block size may now be set almost as large as the latency budget, which reduces the required overhead. This is illustrated in the “burst sending” results above.

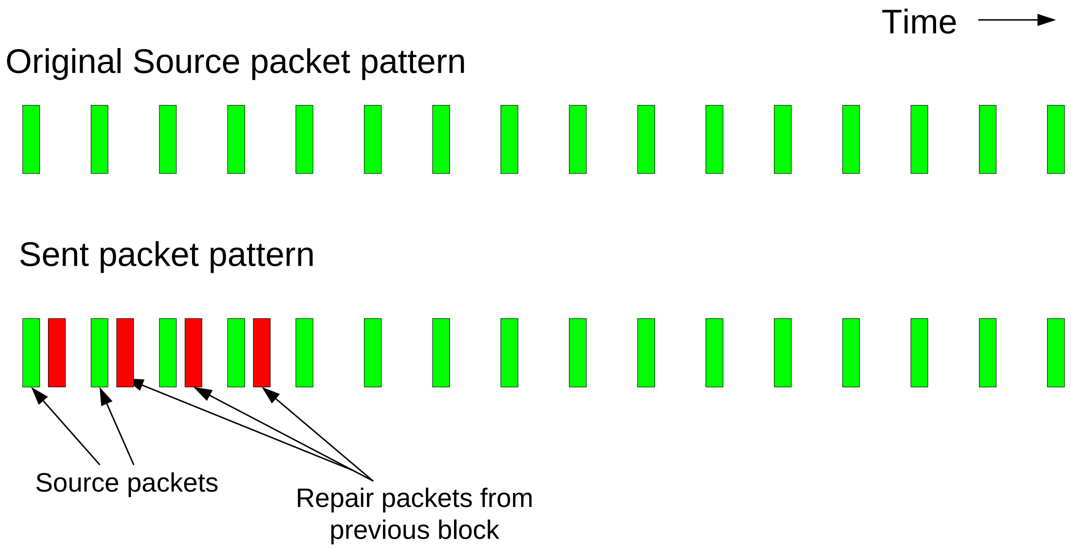

Time
Original Source packet pattern
Sent packet pattern
Source packets
Repair packets from previous block

Figure A.3: Pro-MPEG COP3 burst sending arrangement

Pros and cons:

- + source data sending rate is the same as original source data sending rate
- + almost all of latency budget is available for FEC protection
- - global sending rate is very bursty (and therefore not constant)
- - insertion of repair packets introduces small amount of jitter at the beginning when total stream bandwidth is close to available channel bandwidth

## A.4 Concurrent Interleaved sending

In the case of Video on Demand, or if additional latency at the encoder is acceptable, a sending arrangement as depicted in Figure A4 is possible. In this case, repair packets are interleaved within the block that they protect. This is possible in the Video on demand case because the data to be protected is available for FEC calculations to be performed slightly in advance of sending the data. Alternatively, a live stream can be buffered at the encoder for long enough for the FEC calculations to be performed before beginning to send the source packets of the block.

This sending arrangement could also be used for live content with a penalty that buffering equal to the block size would be required at the sender. This buffering contributes additional end-to-end delay to the playout of live streams i.e. the delay between a live event occurring and being presented on the user's screen. However it would not contribute additional channel change delay. This option may be important if there is existing equipment which is affected by changes in the timing of source packets. The procedures for timing recovery specified in TS102 034 Annex A allow MPEG 2 timing to be recovered even in the presence of significant IP packet arrival jitter – however, if these procedures have not been correctly implemented then equipment may be adversely affected by the additional jitter introduced by some of the other sending arrangements described here.

This sending arrangement has the desirable properties that both the source packet data rate and the total data rate are constant. However, in the Pro-MPEG case, unlike the constant data rate arrangement in 9.2, the whole latency budget can be used for a single source block.

New simulation results are presented for this sending arrangement in Annex B. Note that only the Pro-MPEG column code was tested, not the 2D code.

For random loss, the results are similar to the comparison between Raptor with constant sending and Pro-MPEG with burst sending – i.e. Pro-MPEG uses slightly less overhead below the loss rate threshold than Raptor does. However, for burst loss, the Pro-MPEG code is significantly affected by interleaving of repair packets with the source packets they protect. For the 2Mbit/s stream, this pushes the threshold where Pro-MPEG performs well down to 1e-5 or below. For the 6Mbit/s stream, the quality target was not achievable: it is easy to see why, since a burst loss of 6 source packets will often hit a repair packet as well, and it is not possible with only 6 repair packets per block to avoid that the burst hits a source packet that is protected by that repair packet.

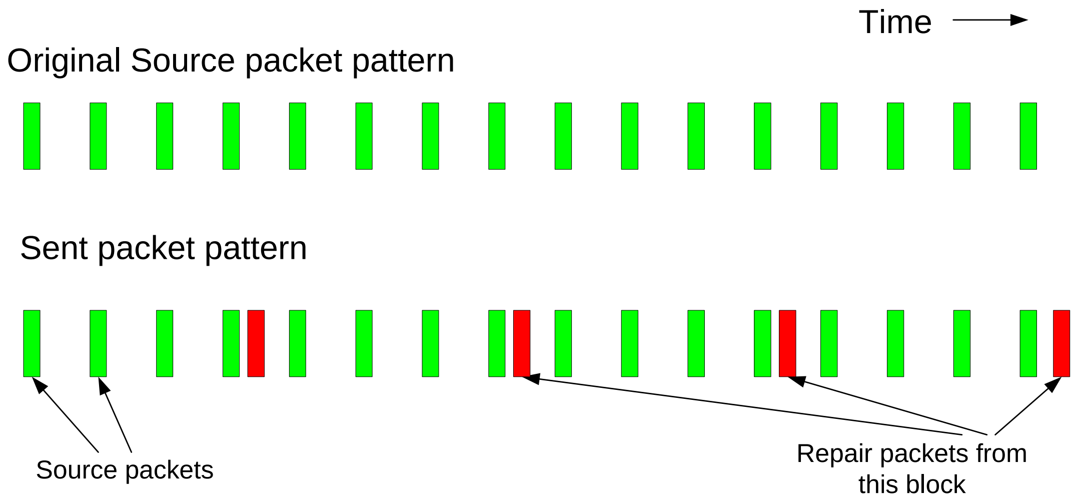

Time
Original Source packet pattern
Sent packet pattern
Source packets
Repair packets from this block

Figure A.4: Interleaved sending for VoD

Pros and cons:

- + source data sending rate is the same as original source data sending rate
- + all of latency budget is available for FEC protection
- - global sending rate is kept constant
- - insertion of repair packets introduces very small amount of jitter at the beginning when total stream bandwidth is close to available channel bandwidth
- - not resilient to burst losses for the Pro-MPEG FEC.

## A.5 DF Raptor faststart sending for stored/buffered content

An additional sending arrangement for stored or buffered content (i.e. VoD and trick modes on live content) was proposed and simulated for DF Raptor. This sending arrangement is illustrated in Figure A5. In this arrangement, source data is sent slightly faster than the nominal stream rate at the start of the session or when trick modes are used. This allows the buffering period to be gradually increased without introducing additional channel change latency.

Two variants of this approach were simulated:

- “faststart with constant rate sending” – in which the additional source data bandwidth is obtained by reducing the FEC bandwidth at the beginning of the stream. As a result the total stream rate remains constant, but stream quality is reduced for these few initial seconds.

- “faststart with variable rate sending” – in which the overall stream rate at the beginning of the stream is somewhat higher than the nominal stream rate (e.g. 20% higher) for the initial few seconds of the stream, but as a result the stream quality is maintained.

The second variant provided the best results.

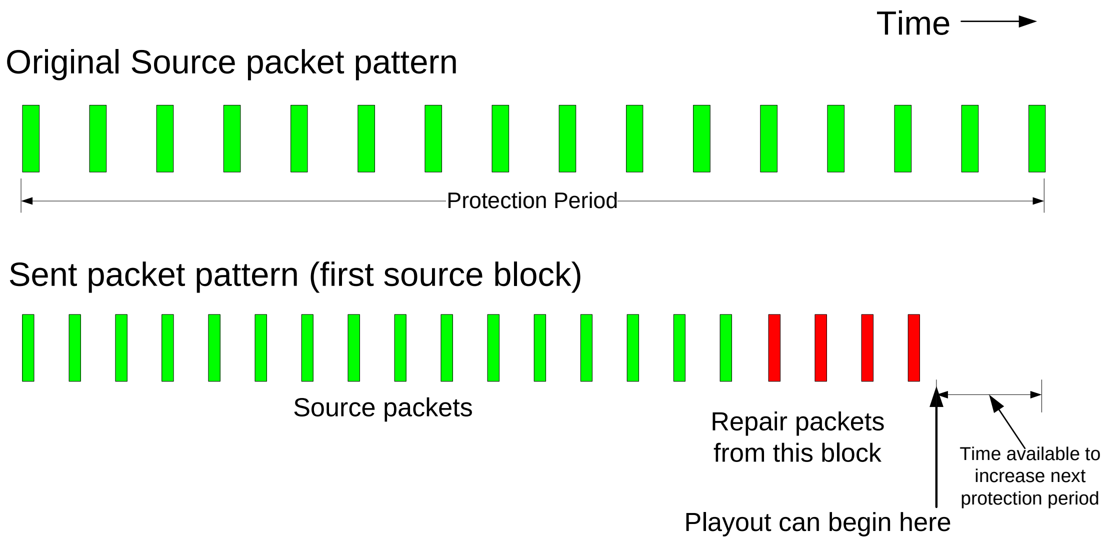


Figure A.5: DF Raptor faststart sending arrangement

Pros and cons:

| | |
|---|---|
| + | FEC protection period can be increased to much greater than the latency budget |
| - | only applicable to unicast/buffered content for Raptor |

# Annex B: Concurrent interleaving results

This Annex presents simulation results for the sending arrangement described in A.4 in which both the source packet rate and the total stream rate are kept constant, whilst also allowing the full latency budget to be used for the FEC block.

Note that, due to lack of time, these results do not include the Pro-MPEG 2D code. It might be expected that in some of the cases where a result is not shown with the 1D code then the 2D code could provide the target quality, but at a relatively high overhead.

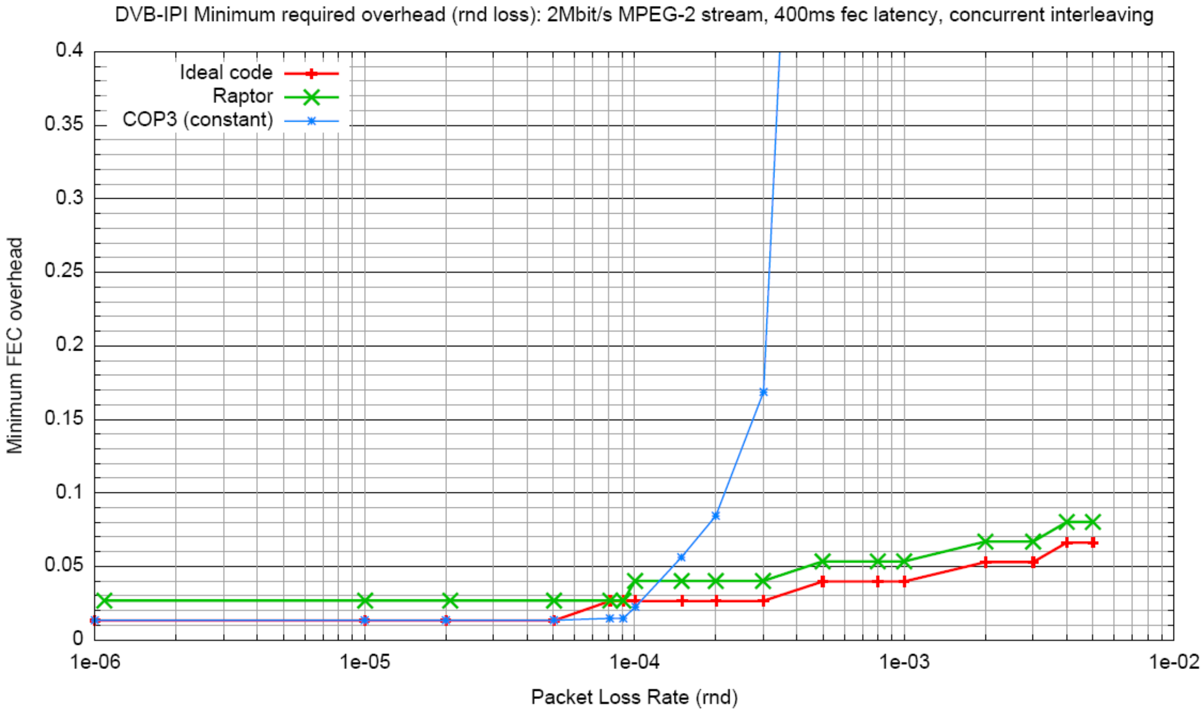


Figure B.1: 2Mbit/s MPEG-2 Transport Stream, 400ms latency, Random Loss, concurrent interleaving

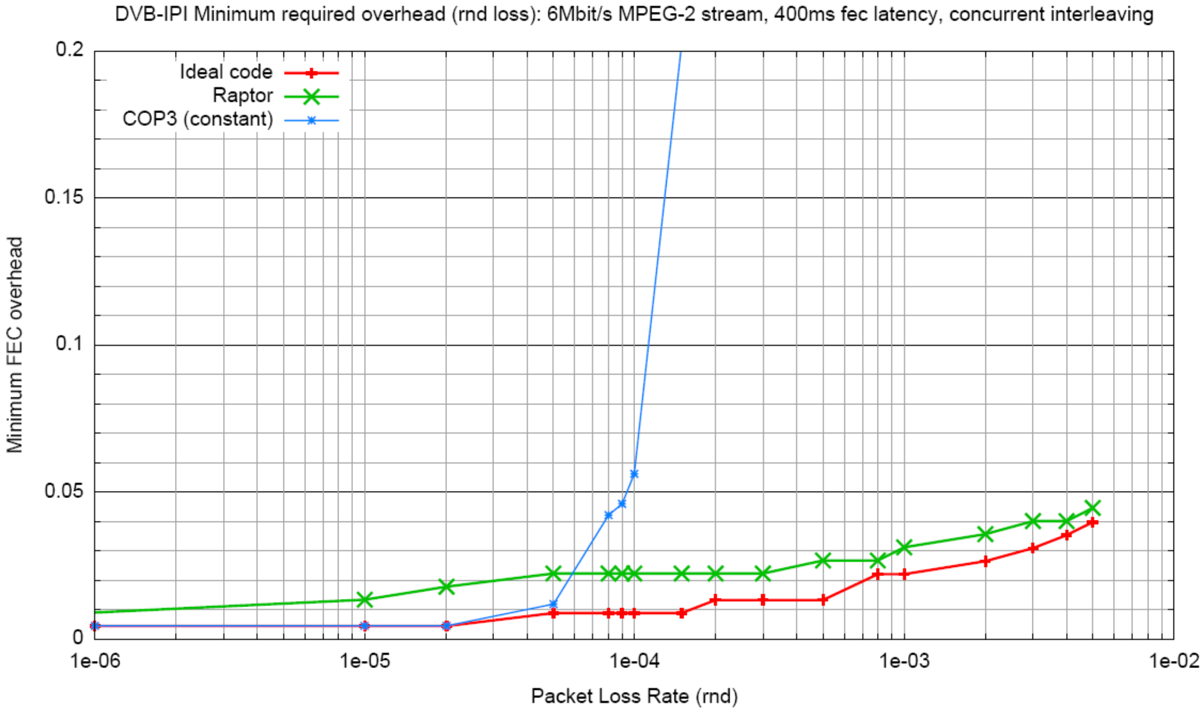


Figure B.2: 6Mbit/s MPEG-2 Transport Stream, 400ms latency, Random Loss, concurrent interleaving

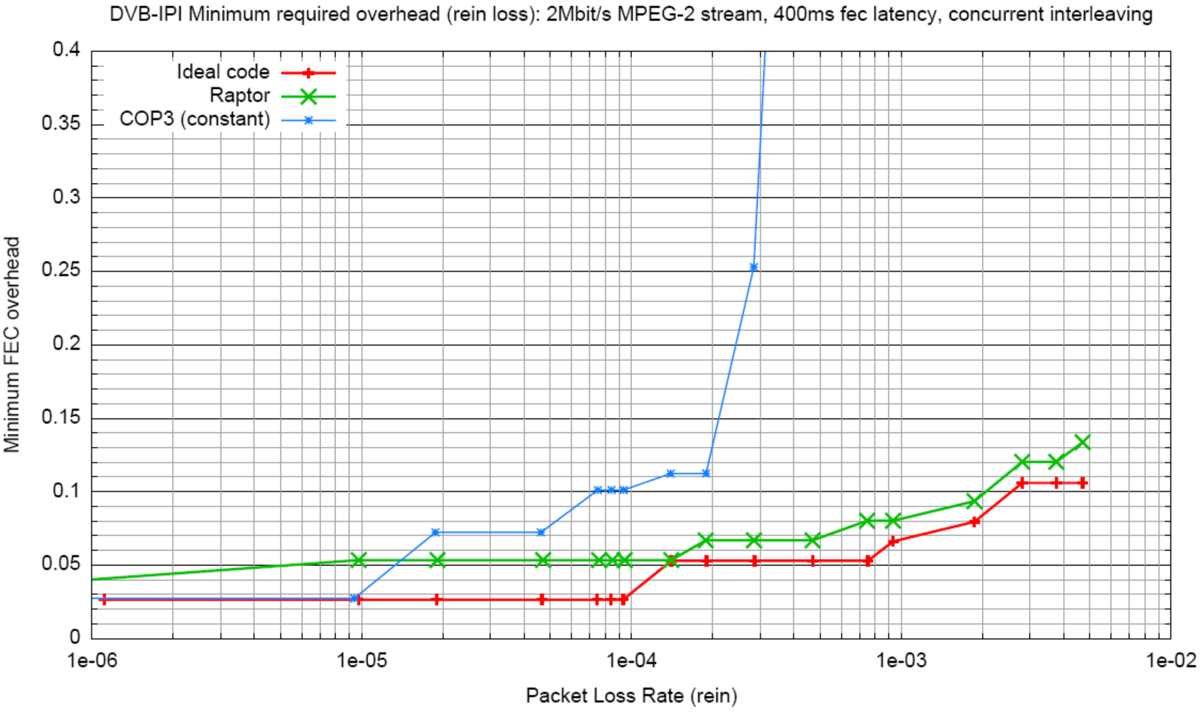


Figure B.3: 2Mbit/s MPEG-2 Transport Stream, 400ms latency, REIN Loss, concurrent interleaving

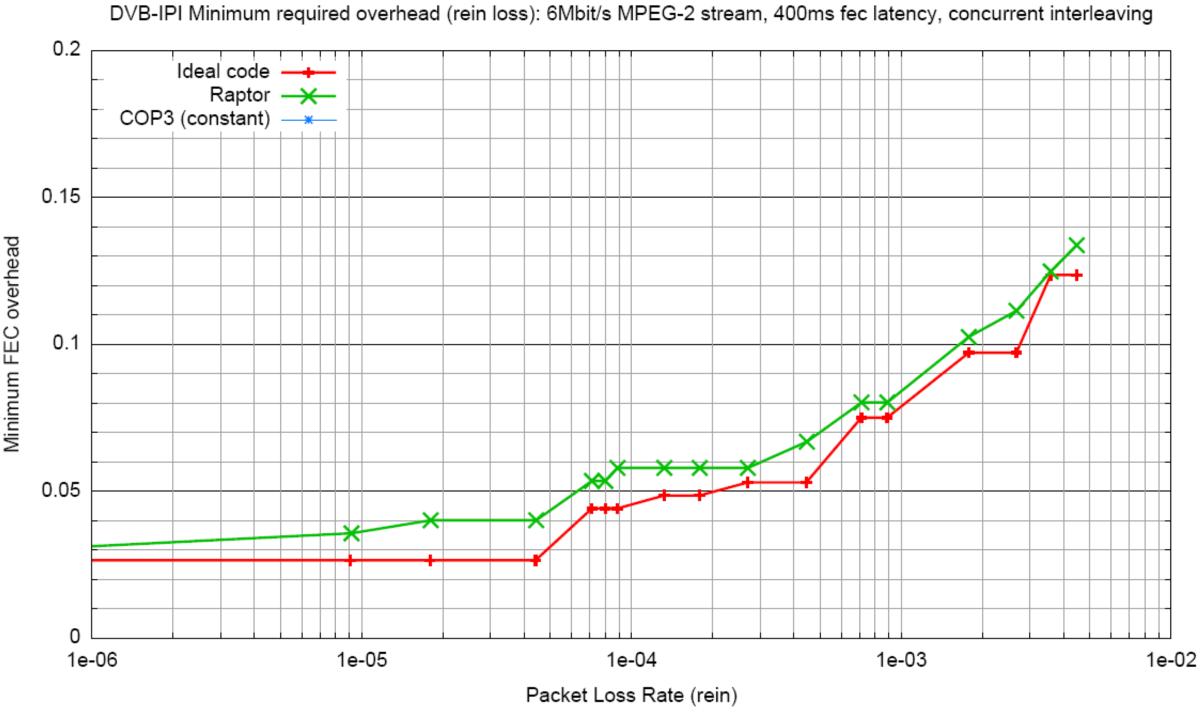


Figure B.4: 6Mbit/s MPEG-2 Transport Stream, 400ms latency, REIN Loss, concurrent interleaving

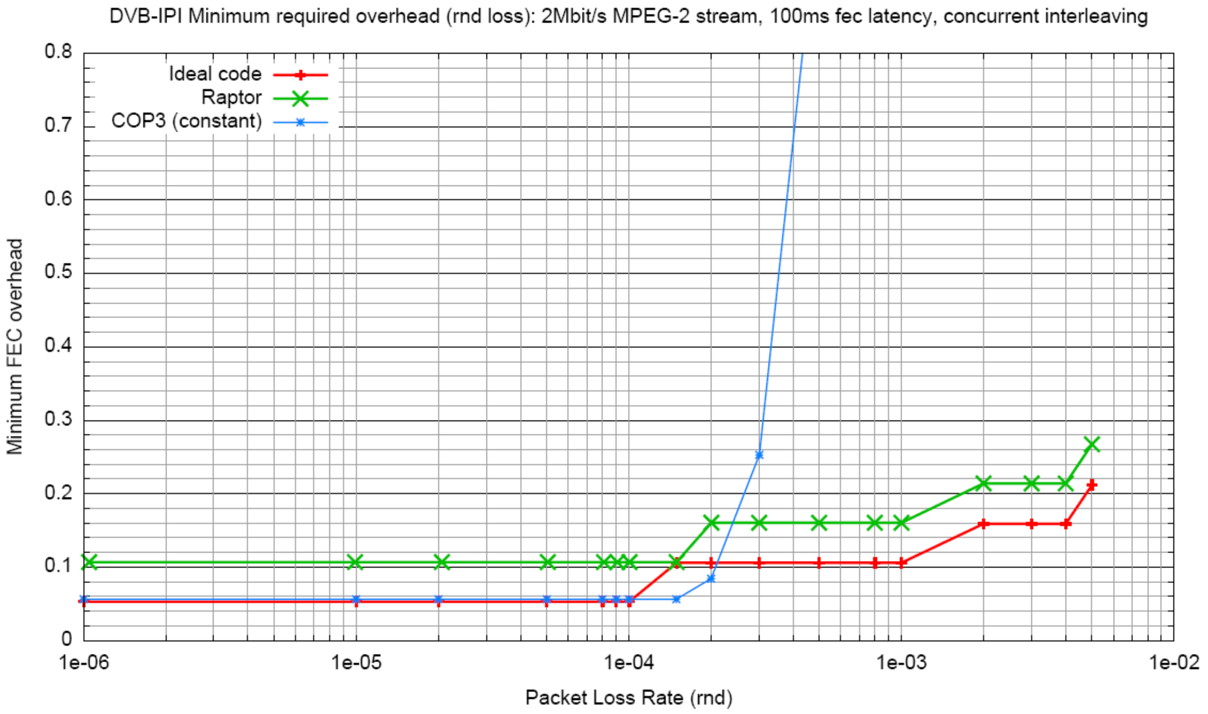


Figure B.5: 2Mbit/s MPEG-2 Transport Stream, 100ms latency, Random Loss, concurrent interleaving

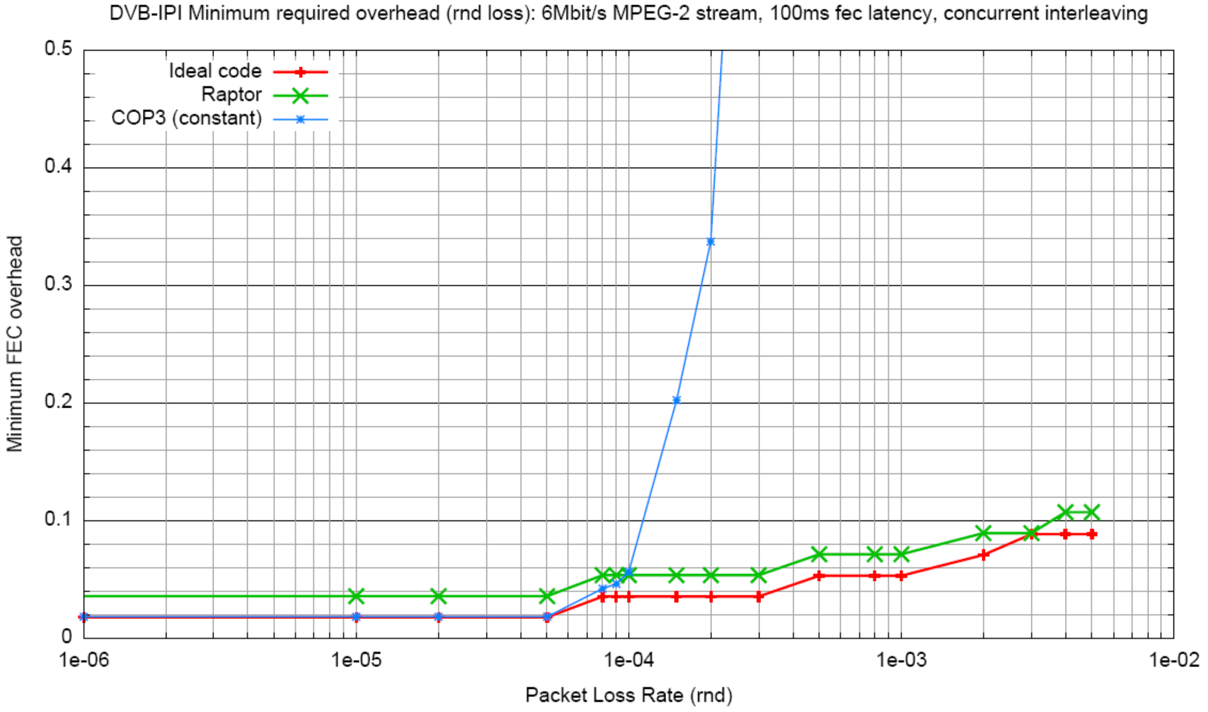


Figure B.6: 6Mbit/s MPEG-2 Transport Stream, 100ms latency, Random Loss, concurrent interleaving

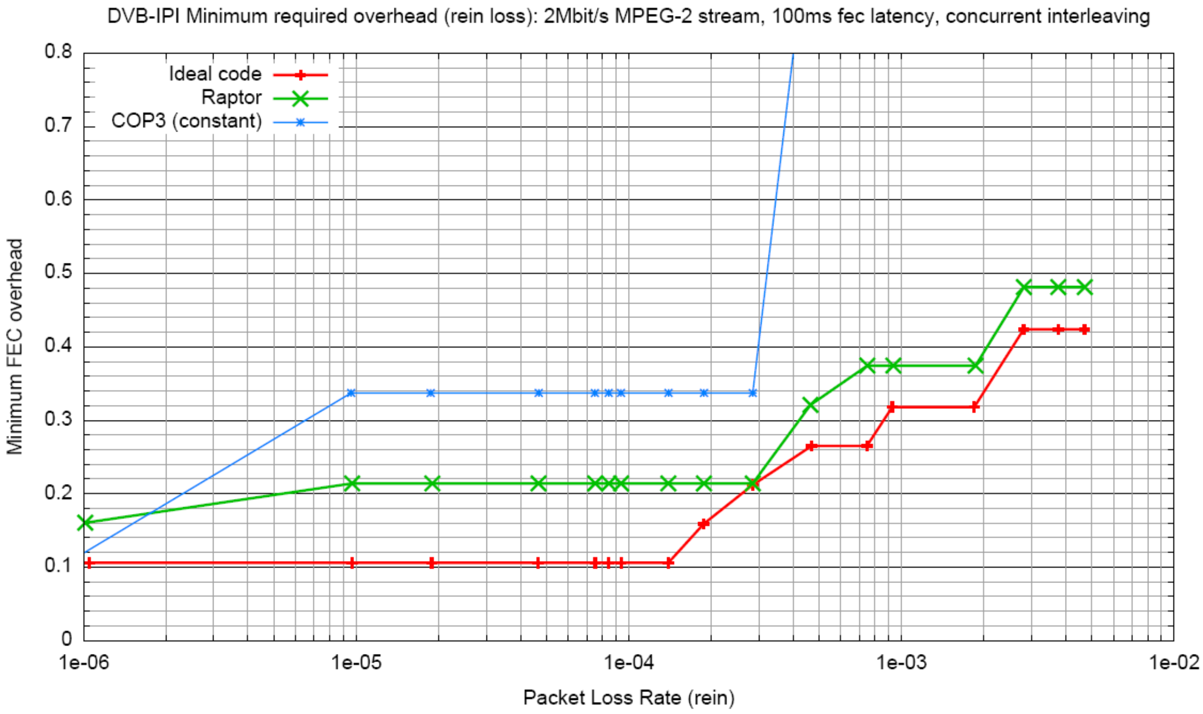


Figure B.7: 2Mbit/s MPEG-2 Transport Stream, 100ms latency, REIN Loss, concurrent interleaving

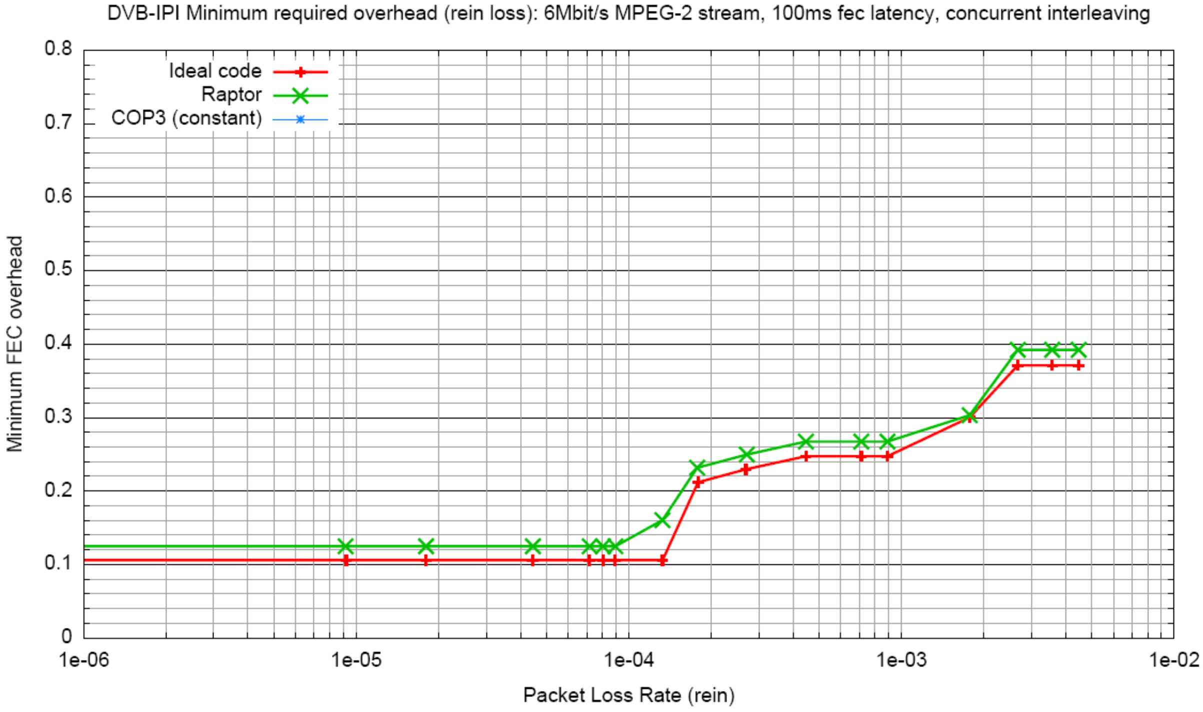


Figure B.8: 6Mbit/s MPEG-2 Transport Stream, 100ms latency, REIN Loss, concurrent interleaving

# Appendix 1 : Hybrid code

A hybrid of the Pro-MPEG 1D column code and the Raptor code was proposed in order to provide a single scalable FEC solution with performance similar to the best of either the Pro-MPEG or Raptor codes in any given case.

## Hybrid code results

This Annex presents results for the Hybrid code. The hybrid cases are denoted “Raptor P<n>” where <n> is the number of parity packets used. The value of <n> chosen in each case is the smallest such that the quality target can be achieved with Pro-MPEG packets alone at loss rates of 1e-5 and lower.

The sending arrangement of Annex A section A.1 was used for these simulations.

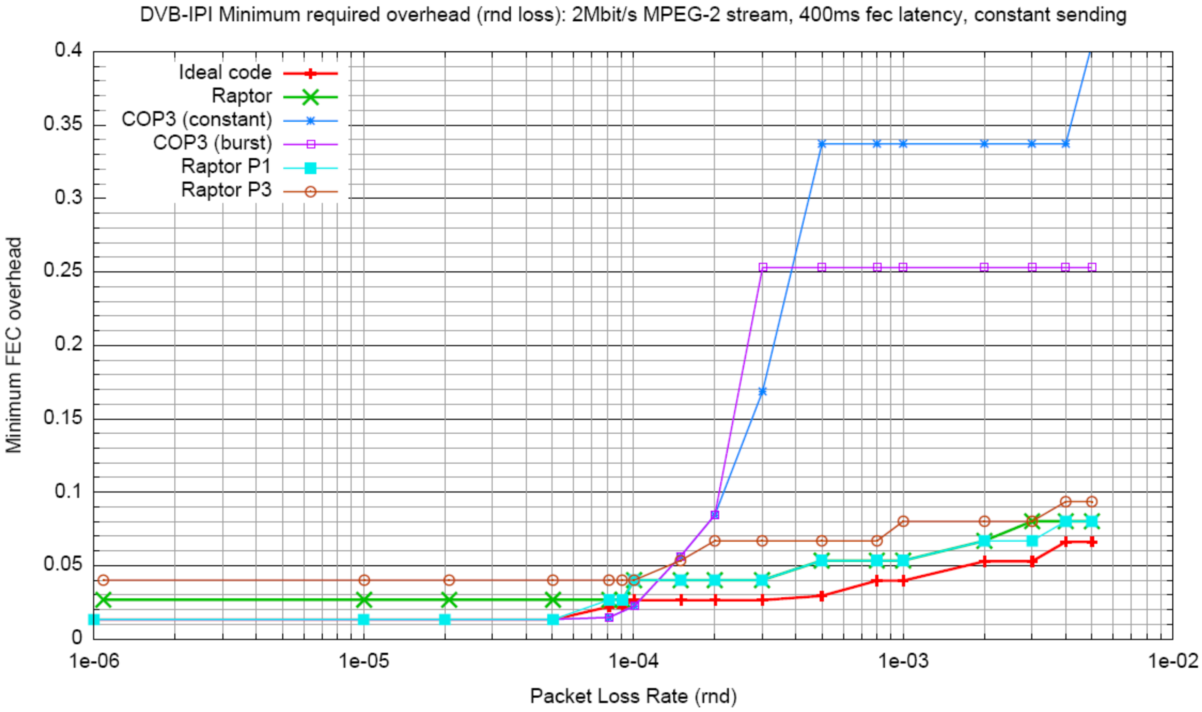


Figure 1.1: 2Mbit/s MPEG-2 Transport Stream, 400ms latency, Random Loss, constant sending

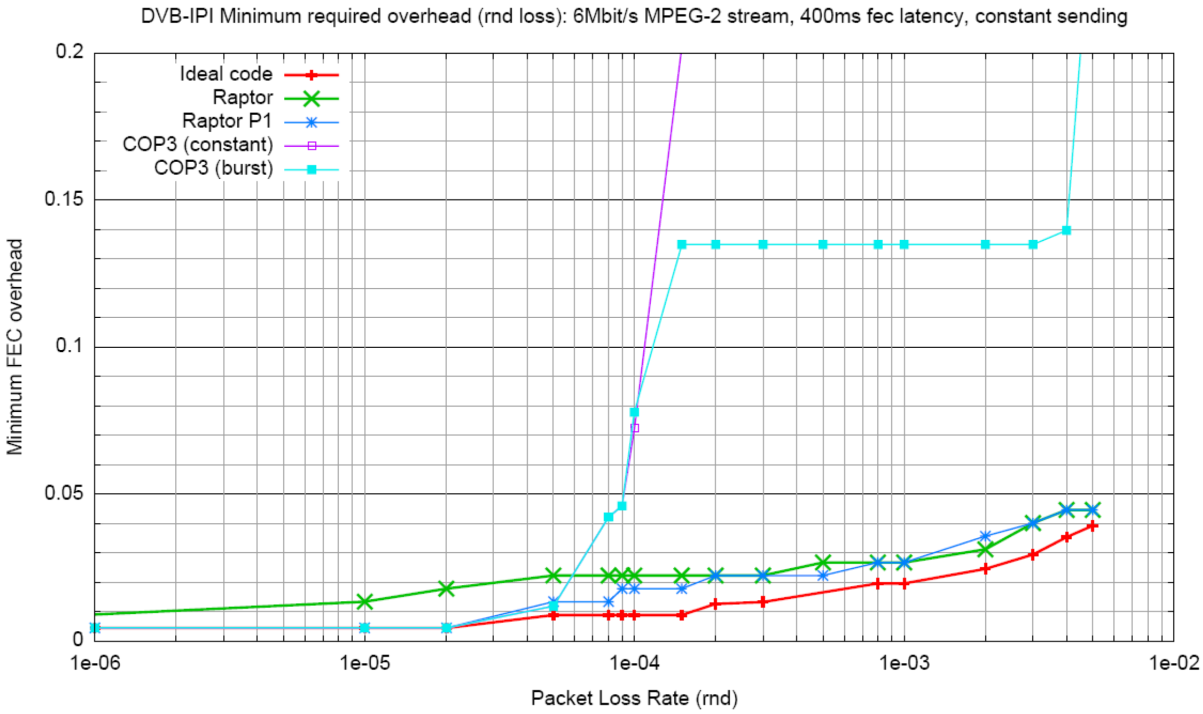


Figure 1.2: 6Mbit/s MPEG-2 Transport Stream, 400ms latency, Random Loss, constant sending

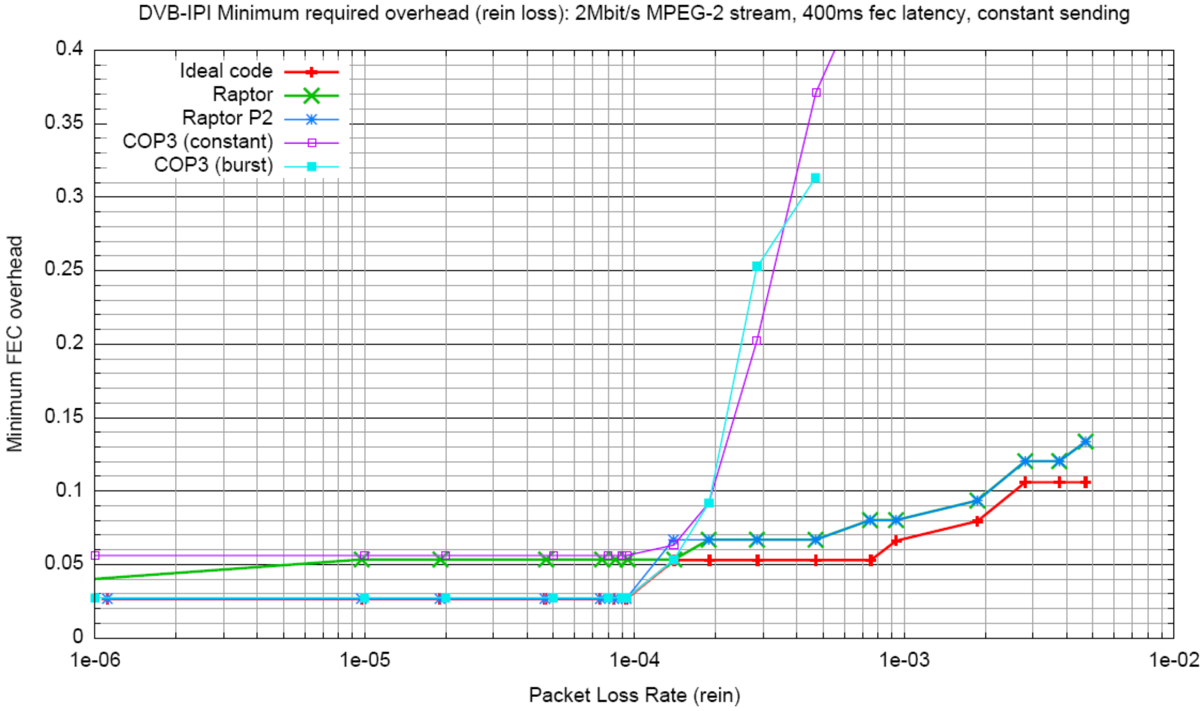


Figure 1.3: 2Mbit/s MPEG-2 Transport Stream, 400ms latency, REIN Loss, constant sending

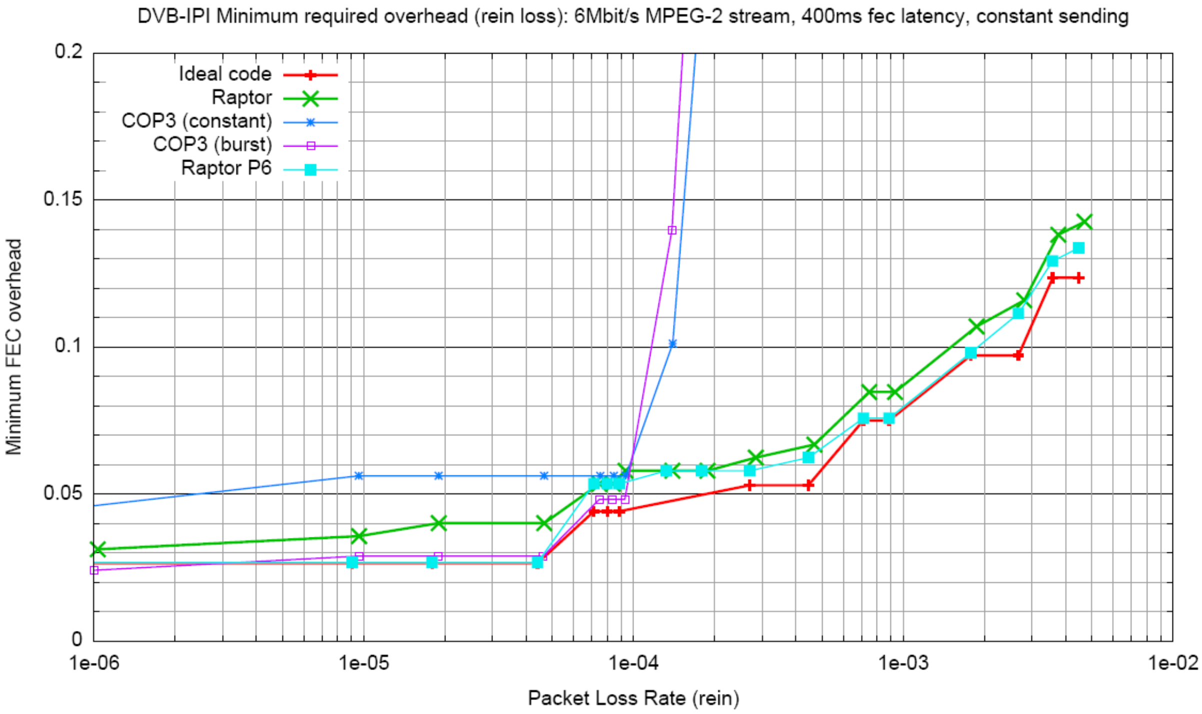


Figure 1.4: 6Mbit/s MPEG-2 Transport Stream, 400ms latency, REIN Loss, constant sending

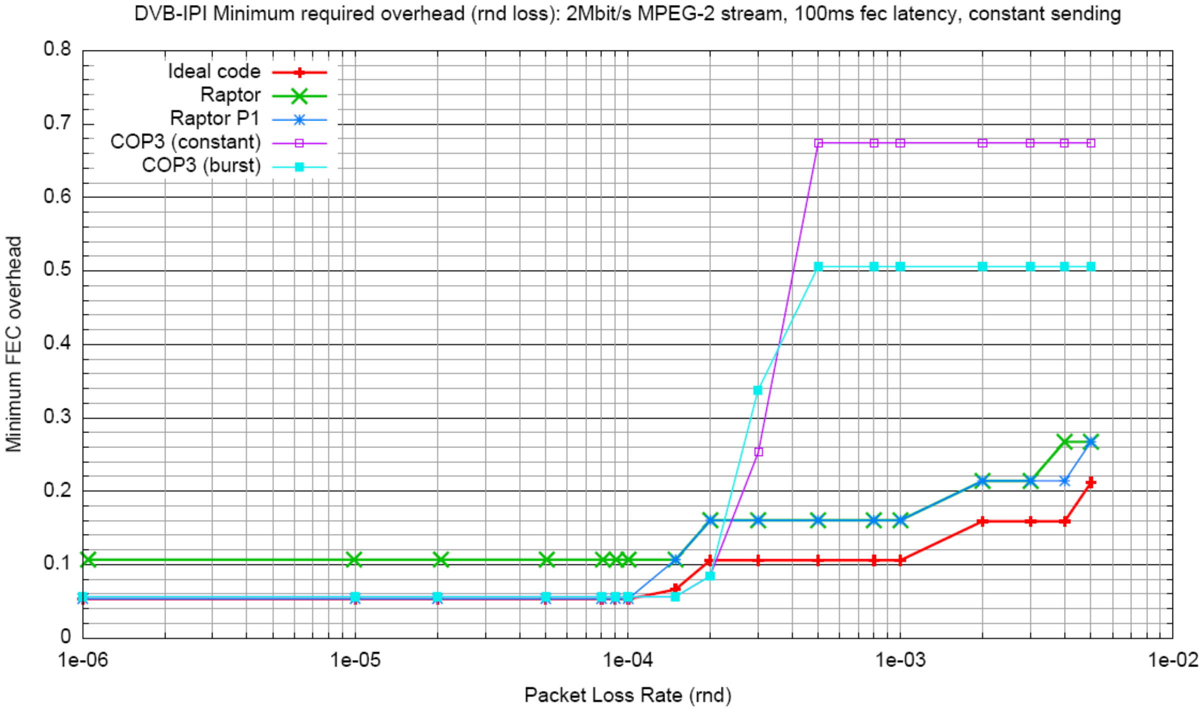


Figure 1.5: 2Mbit/s MPEG-2 Transport Stream, 100ms latency, Random Loss, constant sending

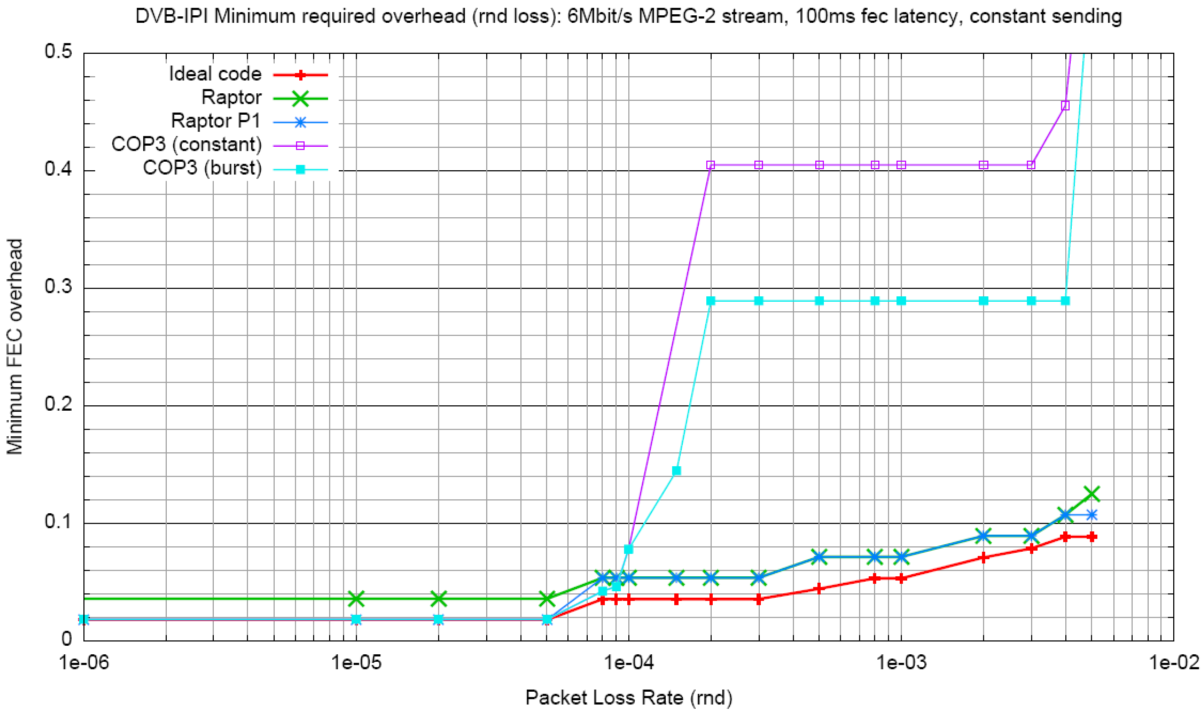


Figure 1.6: 6Mbit/s MPEG-2 Transport Stream, 100ms latency, Random Loss, constant sending

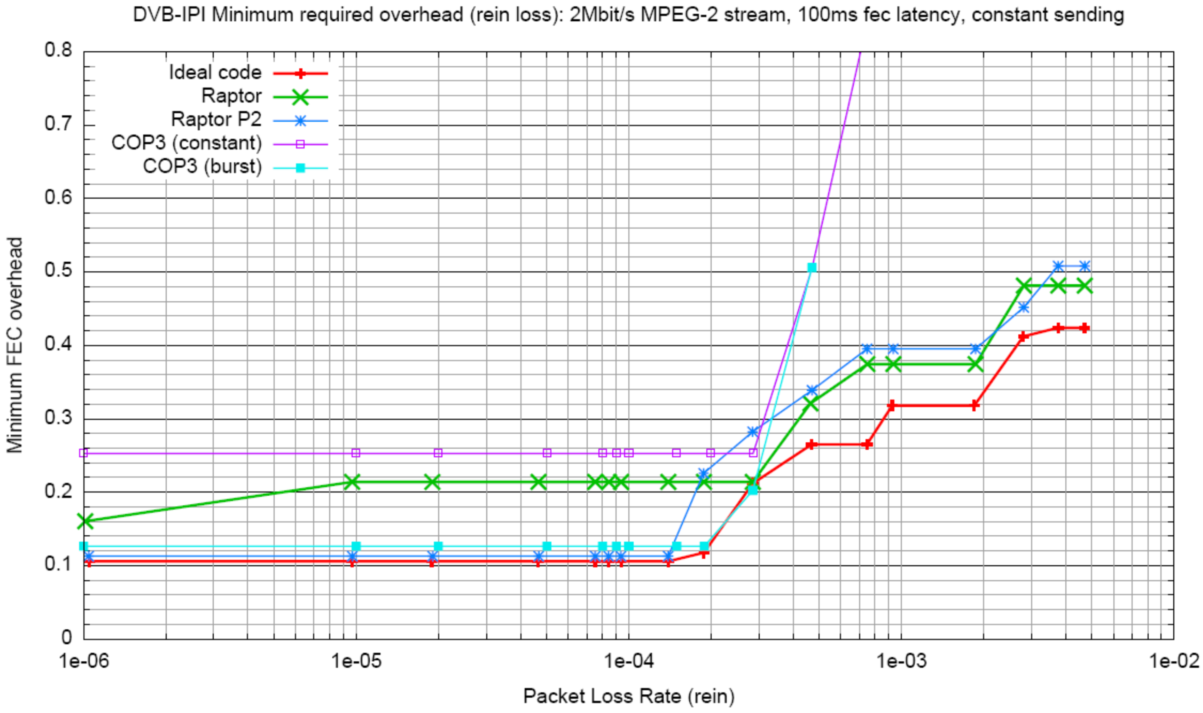


Figure 1.7: 2Mbit/s MPEG-2 Transport Stream, 100ms latency, REIN Loss, constant sending

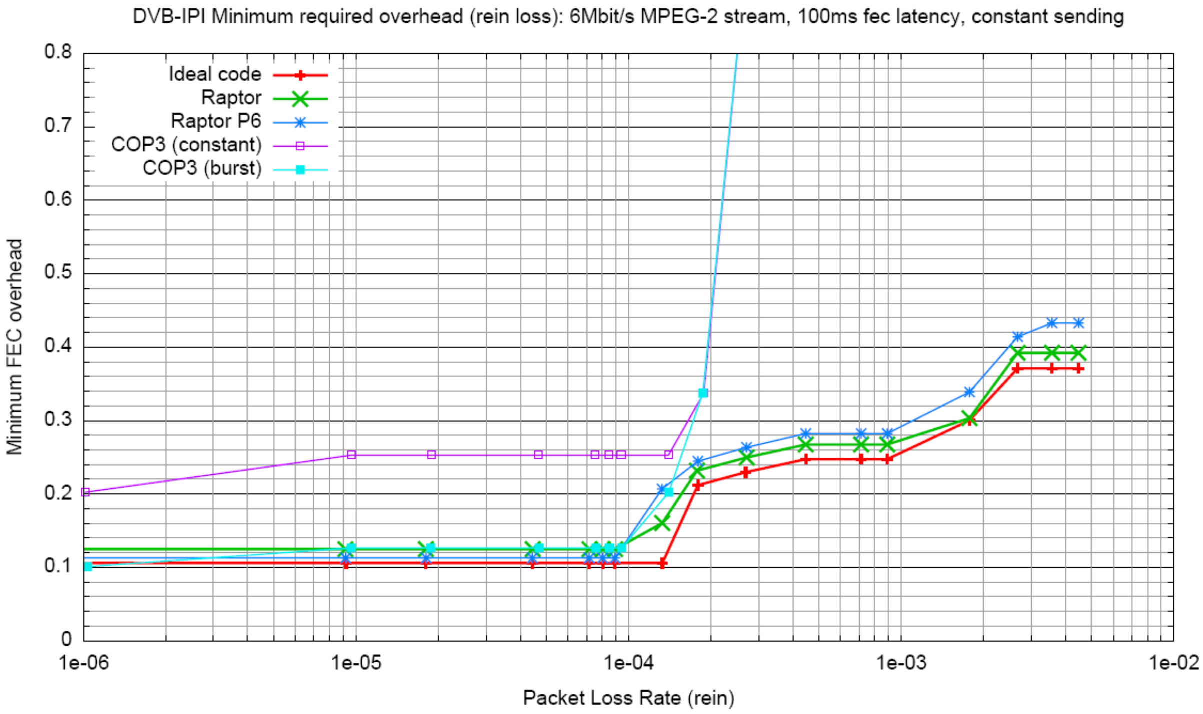


Figure 1.8: 6Mbit/s MPEG-2 Transport Stream, 100ms latency, REIN Loss, constant sending