
Rationale for MBMS AL-FEC Enhancements

- Source: Qualcomm Incorporated
- Agenda Item: 5
- Document for discussion

Agenda

- Definitions
- Background on application layer FEC and raptor codes
- Relative performance of raptor codes
 - Flexibility and convenience
 - Efficiency
 - Computational complexity
 - Memory utilization
- Summary and Recommendations

Definitions and Background

- A **source symbol** is a collection of unique, typically contiguous, bytes from the source data (file or stream) to be transmitted
- The source data is divided into **k** total source symbols
- An **encoded symbol** is the result of the encoding process
- **n** is the total number of encoded symbols sent
- **k/n** is the code rate
- **r** is the number of correct symbols received

- **Linear time encoding and decoding** is the property that the computation complexity of encoding or decoding the source data is proportional to the source data size
- In an **erasure channel**, a transmitter sends a symbol, and the receiver either receives the symbol correctly together with a symbol identifier or it does not receive the symbol, i.e. the symbol was lost or erased
- An **erasure code** is a forward error correction (FEC) code for an **erasure channel**
- Application layer FEC functions in an **erasure channel** and utilizes **erasure coding**

More Definitions and Background

- An **optimal erasure code** can recover the entire source data with any k encoded symbols
 - A Reed Solomon code is an optimal erasure code, but it does not have the linear time encoding and decoding property

- A **fountain code** is an erasure code with the property that a large number of unique encoded symbols can be created relative to the k source symbols

- A fountain code is also known as a **rateless erasure code**
 - Given flexibility in k essentially any code rate, k/n , can be achieved

- **A raptor code**
 - A raptor code is a fountain code with the property of linear time encoding and decoding
 - Raptor codes are suboptimum, i.e. k symbols or slightly more than k symbols must be received to assure decoding
 - The property of linear encoding and decoding make Raptor codes ideal for large files or streaming

Existing MBMS FEC Status

- The original FEC for MBMS is defined in RFC 5053: Raptor Forward Error Correction Scheme for Object Delivery
 - Raptor RFC 5053 is the currently specified FEC for RFC 3926: File Delivery over Unidirectional Transport (FLUTE) in 3GPP MBMS
 - Raptor RFC 5053 is also specified for streaming in 3GPP MBMS
 - eMBMS incorporates Raptor RFC 5053 by reference

- In the 5 to 6 years since RFC 5053 was adopted there has been significant progress in the design of erasure codes
 - An enhanced raptor code has been documented at IETF
 - This newer variant is known as RaptorQ
 - The following slides highlight what can be accomplished by current technology

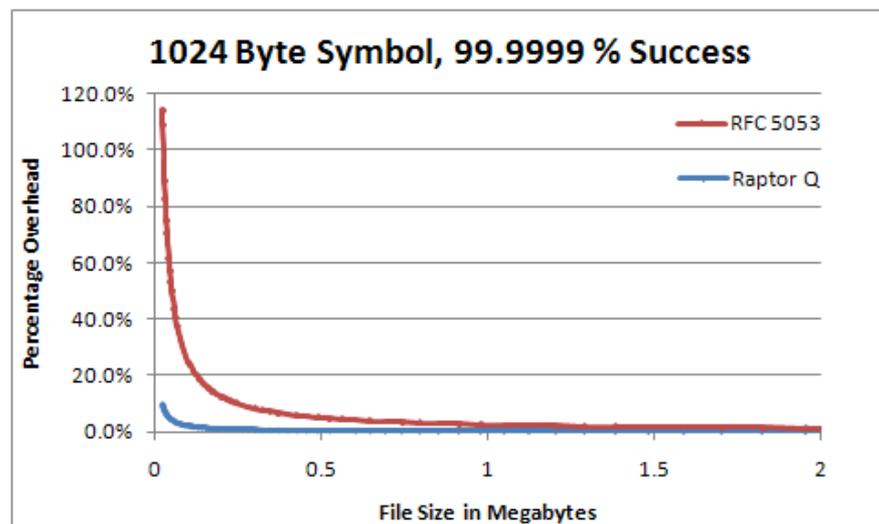
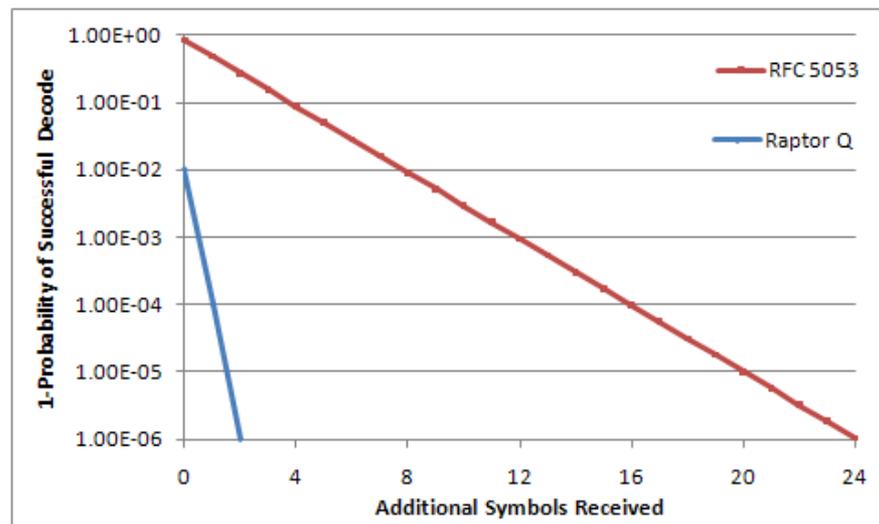
Raptor RFC 5053 and RaptorQ Properties

- A couple of important aspects of Raptor codes
 - Maximum number of source symbols possible
 - Maximum number of unique encoded symbols possible
- RaptorQ can more easily support large files than RFC 5053 Raptor
 - 6.8 x the size with a single source block
- RaptorQ has many more encoding symbols available
 - 256 x
- Expanding the range of these two parameters simplifies the application of the FEC code

Property	Raptor RFC 5053	RaptorQ
Maximum Number of Source Symbols	8,192	56,403
Maximum Number of Encoded Symbols	65,536	16,777,216
Maximum Symbol Size	65,536	65,536

Raptor RFC 5053 and RaptorQ Properties Continued

- Probability of a successful decode as a function of received symbols is a key property of a raptor code
- The upper bound for the probability p of a failed decode for each code is shown below
- RFC 5053 $p = 0.85^{*(0.567(r-k))}$
 - 99.9999% success for $r = k+24$
- RaptorQ $p = 0.01^{*(0.01(r-k))}$
 - 99.9999% success for $r = k+2$
- RaptorQ is notably more efficient for smaller $k(s)$
 - This is a crucial property for small files and streaming applications



Computational Complexity

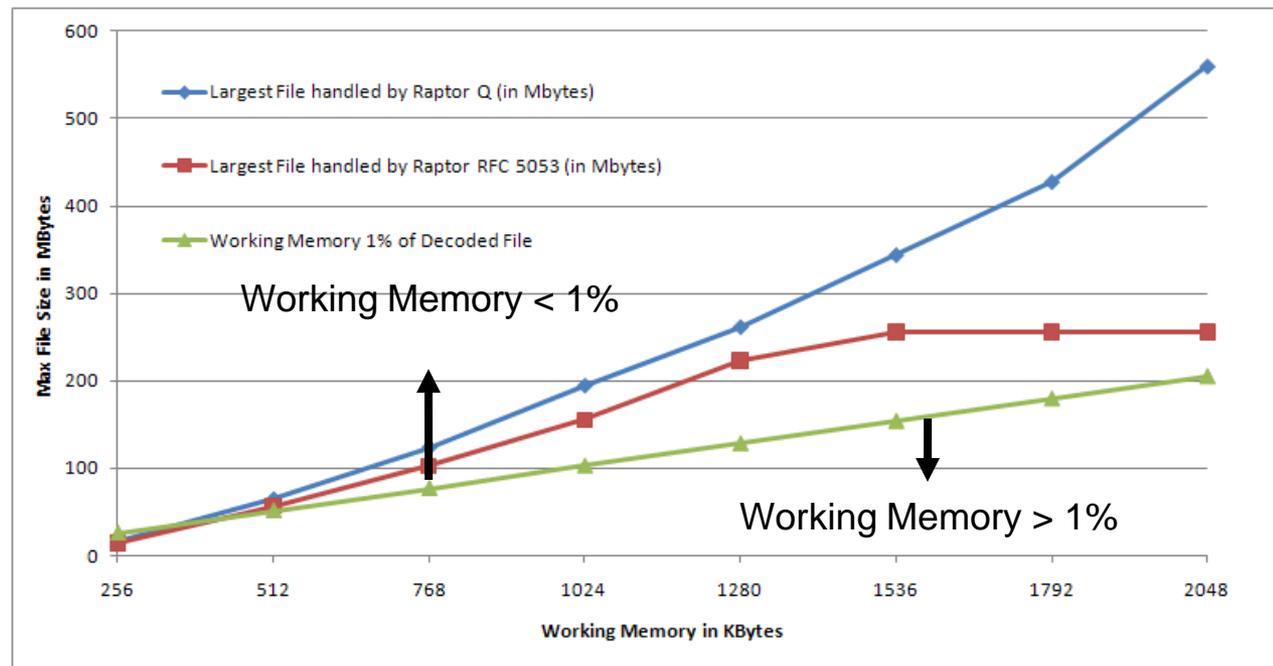
- The table shows the decode rates achieved on an Android OS device
 - Same file sizes
 - Same overhead rate
 - 50% symbol loss rate
 - Raptor RFC 5053 symbol size of 85 bytes
 - Raptor Q symbol size of 1024 bytes
- The RaptorQ decode rate is 10% to 100% better than Raptor RFC 5053 under similar conditions
- Rapid decoding is critical for large files and streaming applications

File Size Bytes	Raptor RFC 5053 Decode Rate Mbps	RaptorQ Decode Rate Mbps	Overhead Rate
32768	49.6	54.1	6.3%
65536	48.6	69.7	3.1%
131072	42.2	76.0	1.6%
262144	37.3	74.7	0.8%
524288	33.1	69.8	0.4%

Memory Utilization

The decode of Raptor encoded files requires working memory

- This graph shows the largest file size decodable as a function of working memory space
- For files larger than 50 MB both codes use less than 1% of object size



RaptorQ is slightly more efficient in its use of working memory than Raptor RFC 5053

Summary and Recommendations

- Sufficient progress has been made with respect to FEC for the application layer that a work item should be opened to consider improved options for MBMS / eMBMS
- Given the nascent nature of MBMS and eMBMS deployments to date, it may be possible to accomplish a system upgrade ahead of any significant deployments