Rateless coding

C. Pouliiat

Abstract

In this chapter, we present the main concepts of rateless coding and some coding schemes that have been proposed in this context. We first review the main fundamental concepts of the fountain coding paradigm, considered as a specific paradigm related to rateless coding. Within this context, we present the most widely used and successful fountain coding solutions based on sparse-graph based codes, namely the LT and Raptor codes. For both solutions, we present their basic properties and we review the related decoding algorithms. These codes have been designed to initially perform on the binary erasure channel for forward error correction at upper layers (e.g. the transport or medium access layers). But they can be also used at the physical layer. Therefore, we present their extensions and the applicability to noisy wireless channels such as, for example, the additive white Gaussian noise channel or fading channels. In these settings, we discuss their main limitations and the solutions that have been proposed to possibly improve their performance, especially in the error floor region. We then consider advanced fountain coding techniques based on sparse graph codes that are some generalizations of the LT or Raptor codes (for example multi-edge type or protograph based fountain codes). We also present some improved decoding algorithms. As stated above, rateless coding is not limited to the fountain coding paradigm. Therefore, we also review rateless coding schemes that have been proposed so far in the literature to implement a rateless coding strategy. Finally, we present some applications where rateless coding can be used efficiently.

Index Terms

rateless coding, fountain coding, LT code, Raptor code, RaptorQ, inactivation decoder, forward-error correction, layered coding, sparse-graph codes
I. INTRODUCTION

In this chapter, we present the main concepts of rateless coding and some coding schemes that have been proposed in this context. We first review the main fundamental concepts of the fountain coding paradigm, considered as a specific paradigm related to rateless coding. Within this context, we present the most widely used and successful fountain coding solutions based on sparse-graph codes, namely the LT and Raptor codes. For both solutions, we present their basic properties and we review the related decoding algorithms. These codes have been designed to initially perform on the binary erasure channel for forward error correction at upper layers (e.g. the transport or medium access layers). But they can be also used at the physical layer. Therefore, we present their extensions and discuss the applicability to noisy wireless channels such as, for example, the additive white Gaussian noise channel or fading channels. In these settings, we discuss their main limitations and the solutions that have been proposed to possibly improve their performance, especially in the error floor region. We then consider advanced fountain coding techniques based on sparse graph codes that are some generalizations of the LT or Raptor codes (for example multi-edge type or protograph based fountain codes). We also present some improved decoding algorithms. As stated above, rateless coding is not limited to the fountain coding paradigm. Therefore, we also review rateless coding schemes that have been proposed so far in the literature to implement a rateless coding strategy. Finally, we present some applications where rateless coding can be used efficiently.

II. THE FOUNTAIN PARADIGM

A. Fountain coding and decoding: definitions and principles

We describe the general concept of Fountain codes and their decoding properties [1], [2]. To do so, we first describe the formal context of their application and we give the abstract properties of an ideal fountain coding scheme. Then, we consider a first pragmatic approach to practically implement a fountain coding engine based on binary random linear coding. Fountain coding
is a coding paradigm that has been introduced for the purpose of scalable and fault-tolerant distribution of data over computer or wireless networks. The aim is to provide efficient coding solutions (eventually distributed) for different transmission scenarios such as point-to-multipoint, multipoint-to-point or multipoint-to-multipoint communications. First practical solutions that have been developed are the LT codes [3]–[5] and online codes [6], [7]. They are an instance of random fountain codes.

Let us consider a block of data, often referred to as source block in the dedicated literature [1]. This source block is then partitioned into $k$ equal size packets, often denoted as source symbols or input symbols. In the following, $k$ will refer to the number of source symbols within a source block. To reliably transmit the source block, the sender will use an encoding engine to generate encoded symbols from the source symbols, also referred to as output symbols. At the receiver side, a decoder is used to retrieve the source symbols of the transmitted source block from the received encoded symbols, even if some symbols are not received due to some losses in the network. Ideally, a fountain code should have the following abstract properties:

- the encoder of a fountain code should be able to produce a limitless number of encoded symbols from the $k$ source symbols of a source block in order to generate as many encoded symbols as required for each receiver;
- at the receiver, the decoder of the fountain code should be able to recover the original source block from any subset of $k$ encoded symbols with high probability;
- from a practical point of view, an ideal decoder should have encoding and decoding complexities (eg. the computation time) that scale linearly with respect to the size of a source block.

The denomination ”fountain” comes from the metaphorical image of a fountain of water: any receiver who aims to receive the source block holds a bucket under the fountain. Only the amount of water (ie. the number of received encoded source symbols) that has been collected
in the bucket is important to recover the original data.

Note that fountain codes are also often referred to as rateless codes in the sense that the encoder of a fountain code can generate on-the-fly a potentially limitless number of encoded symbols. Note also that in the original setting, a source symbol can indifferently refer to a bit or a group of bits. Indeed, all bits or symbols of the same packets are encoded independently the same way.

The main figure of merit to assess the performance of a pair of fountain encoding and decoding engines is the so-called reception overhead, ie. the number of extra encoded symbols needed to recover the \( k \) original source symbols. Let \( n \) be the number of received encoded symbols, then the number of extra symbols \( m \) is given as \( n = k + m \). Thus, an important property of a fountain code is its ability to recover the original data with a little overhead with high probability. The overhead is often given as a fraction of \( k \) for the erasure channel: \( n = (1 + \epsilon)k \) where \( \epsilon \) is the overhead in percent. A fountain coding pair of encoder-decoder will thus be described by a performance curve that gives the decoding failure probability versus the overhead.

### B. The random binary fountain code

A first pragmatic approach to practically implement the fountain paradigm is to consider random linear fountain codes. Without loss of generality, we consider random binary linear fountain codes. For a given vector of \( k \) source symbols \( (x_1, \ldots, x_k) \), the encoded symbols are generated as follows: at each time \( j \),

1) Generate a binary \( k \)-tuple, noted \( (g_{1j}, \ldots, g_{kj}) \) randomly sampled from a given distribution \( \mathcal{D} \) on the vector space \( \mathbb{F}_2^k \).

2) Calculate the output symbol \( c_j \) as the modulo-2 sum (bitwise XOR) of the source symbols.

This can be written as \( c_j = \sum_i g_{ij}x_i \).

The sampled tuples are independent from an encoded symbol to another. The performance of such a scheme is mainly dependent on the choice of the sampling distribution \( \mathcal{D} \). If the
distribution $\mathcal{D}$ is the uniform distribution on $\mathbb{F}_2^k$, then the fountain code is referred to as the
*random binary fountain code*. From a practical point of view, we also assume that encoder and decoder are synchronized: the decoder should be able to know what are the source symbols that are participating in a received encoded symbol. This can be achieved by mean of an appropriate signalling of seed information using additional headers [1], [8] to “synchronize” the pseudo-random generators of the emitter and the receiver.

We now explain how to recover the source symbols when communicating over a binary memoryless erasure channel. Let $n = k + m$ be the number of collected encoded symbols. As the encoded symbols are linear combinations of the source symbols, we can write the following binary linear system

\[
G^\top x = c \quad \text{with} \quad x = (x_1, \ldots, x_k)^\top \quad \text{and} \quad c = (c_1, \ldots, c_n)^\top
\]

$G$ is a $k \times n$ generator matrix derived from the knowledge of the $k$-tuples $(g_{1j}, \ldots, g_{kj})$ associated with the collected encoded symbols. Due to the uniform sampling, the matrix $G$ is a randomly generated matrix from the set of the $k \times n$ binary matrices. The recovery of the source symbols is now equivalent to solve the associated linear system, and it corresponds to a maximum likelihood decoding rule. The performance of the encoder-decoder pair is then given by the probability of failure that is related to the probability that there exists an invertible $k \times k$ submatrix in $G^\top$. It can be shown that this probability of failure is upper bounded by $2^{-m}$ [1], [2], [9], *for any* $k$. Thus for a given probability of failure, as $k$ increases, the relative overhead becomes negligible. From a performance point of view, random binary linear fountain codes could appear as a quite efficient and practical coding scheme to implement the fountain paradigm since this scheme can achieve good performance. The bottleneck of this approach is the induced complexity. The overall decoding cost is the sum of two costs: \((a)\) the matrix inversion using for example a Gaussian elimination with complexity that scales with $O(k^3)$, and \((b)\) the multiplication by the inverse that has a complexity that scales with $O(k^2)$. This
important limitation has motivated the search for more computationally efficient solutions (for both encoding and decoding) with good overhead-failure probability curves. This is the topic of the next section where the most widely used sparse-graph based solutions are presented.

III. RATELESS SPARSE-GRAPH CODES FOR THE BINARY ERASURE CHANNEL: LT AND RAPTOR CODES

In this section, we present some of the most widely used sparse graph based fountain codes, namely LT codes and their extension, Raptor codes [1], [2], [5], [8]. The rational behind this class of codes is to propose efficient fountain coding schemes (in terms of overhead versus failure probability) with reduced (even better, bounded) encoding and decoding complexities. The aim is to present the main features, properties and parameters of interest of such fountain code families. We first present the class of LT codes and then their practical extension, known as Raptor codes. We then briefly review the literature for efficient decoding of these codes for the finite length regime and present standards where Raptor codes have been adopted for upper-layer forward error correction.

A. LT codes

Luby Tranform (LT) codes [3]–[5] is a class of random linear fountain codes that can be shown to have a sparse bipartite graph representation if the parameters of the encoder are properly chosen. The sparse-graph structure allows the use of an efficient sub-optimal message passing algorithm that is an instance of the belief propagation (BP) or sum-product algorithm dedicated to the erasure channel case (often referred to as peeling decoder [10]). Using this representation, this class of codes can be viewed as an instance of the class of irregular low-density generator-matrix codes (LDGM codes) [10]–[12]. An LT code is defined from its degree distribution \( \Omega(x) = \sum_d \Omega_d x^d \). \( \Omega(x) \) is a probability distribution on integers in the set \( \{1, \ldots, k\} \). \( \Omega_d \) is the probability to assign a degree \( d \) to an encoded symbol, so that \( d \) is the number of source symbols
participating in the encoding of this symbol. Moreover, the $d$ source symbols are assumed uniformly sampled at random. Following the notations of the previous section, the encoder of an LT code is given formally as follows. For an encoded symbol $c_n$ generated from the source symbols $(x_1, \ldots, x_k)$ at time index $n$:

1) randomly sample a degree $d_n$ for the encoded symbol from an LT degree distribution $\Omega(x)$,
2) choose uniformly at random $d_n$ source symbols and compute the encoded symbol as the modulo-2 sum (bitwise XORing) of the $d_n$ selected source symbols.

This encoding operation defines a particular binary random linear code. Note that in the original setting [5], the distribution $\Omega(x)$ depends on $k$. If we assume that the receiver has the knowledge of the generator matrix $G$ induced by the received encoded symbols, we can define a bipartite graph associated with $G$ as given in Figure 1. In this graph, a class of nodes, referred to as data nodes, is associated with the source symbols while the other class of nodes, referred to as check nodes or also dynamic check nodes, is related to the encoded symbols. Edges connect source symbols to the encoded symbols in which they are participating. For example, in Fig. 1, the first output symbol is obtained from the modulo-2 sum of the first three input symbols. As for LDGM codes, the resulting code is sparse if the average degree of an encoded symbol is small compare to $k$. If the degree distribution of the LT code is properly chosen, the sparse nature of
the resulting graph allows the use of a message passing algorithm on this graph to recover the original source symbols. In order to have some insights on how to choose the LT distribution, let us just remind quickly the main steps to iteratively recover the source symbols. The iterative decoder can simply be described as follows (for more details on iterative decoding for sparse graph codes, please see Chapter 5):

1) find an encoded symbol $c_n$ in the current graph that is connected to only one source symbol $x_i$,
2) if no such node exists, then the decoding fails. Otherwise, set $x_i = c_n$,
3) add $x_i$ (XOR) to all $c_j$ that are connected to $x_i$ and remove all the edges that are connected to the source symbol $x_i$,
4) go to 1 until decoding fails or some $x_i$ are not recovered.

Steps 1 to 3 define a decoding iteration. By doing so, the resulting decoding complexity is linear in the average degree of the degree distribution multiplied by $k$. Note that the denomination of dynamic check nodes is mainly due to the update operation for check nodes at step 3.

Designing a good distribution for LT codes mainly consists in keeping the average degree distribution as small as possible while guaranteeing that the decoder is successful with high probability and with little overhead (ie. with almost $k$ source symbols). In particular, due to the intrinsic nature of the decoder, a good distribution should avoid to have possibly many degree one check nodes at some intermediate decoding step to avoid an increasing of the overhead. However, it should ensure that there exists check nodes of degree one at each decoding step to avoid decoding failure. To design such a distribution, as for LDPC codes, one has to resort to an expectation analysis that is, in this case, slightly different from density evolution [5], [10]. Thus, using an expectation analysis, Luby [1], [5] has shown that we can fulfil the previous requirements by using the so-called \textit{ideal soliton distribution} given by

$$
\Omega_k(x) = \frac{x}{k} + \sum_{d=2}^{k} \frac{x^d}{d.(d-1)}.
$$
For such a distribution, it can be shown that only \( k \) encoded symbols are sufficient to recover the \( k \) source symbols and that on average, at the beginning of an iteration, exactly one encoded symbol has degree one (a degree-one check node appears at the end of this iteration). However, this distribution works quite poorly in practice: due to the variance in the decoding process around the expected behaviour, it becomes very likely that at some step in the decoding process there will be no available degree one check node in the graph, leading to decoding failure. To overcome this problem, the robust Soliton distribution has been proposed in [1], [5]. For this distribution, the decoding complexity is \( k \cdot O(\log(k/p)) \), where \( p \) is the probability of decoding failure, the average degree of an encoded symbol is \( O(\log(k/p)) \) and the overhead is \( O(\sqrt{k} \log^2(k/p)) \).

Further results and insights for the analysis of LT code performance under iterative decoding can be found in [8], [13]–[15].

### B. Raptor codes

Raptor (Rapid Tornado) codes are a class of fountain codes that achieve linear time encoding and decoding [1], [9]. A detailed analysis of this coding scheme can be found in the seminal work [9] that analyses various aspects of the convergence for both finite length and the asymptotic regime, and performs optimization for the code design. Raptor codes can be viewed as an improvement of their LT relatives: the simple idea behind is the use a weakened LT code precoded by an outer code. The rational behind this concatenated scheme is as follows: suppose that the outer precode is an efficient code that can recover up to a fraction \( \delta \) of erasures among the intermediate symbols, i.e., the symbols that are obtained from the input symbols after encoding by the precode. Then, the LT inner part can be designed to recover only \( 1 - \delta \) of the intermediate symbols from the observed output symbols.

A Raptor code is an LT code concatenated with an outer code called precode, which is usually a high rate error correcting block code. The corresponding Tanner graph is given in Figure 2. Thus, source symbols are first precoded as intermediate symbols using a high rate precode.
Then, the encoded output symbols are generated from the intermediate symbols using an LT code with a constant average degree. At the receiver, a classical LT decoder is used. After the decoding process stops, a small fraction of intermediate symbols may remain unrecovered. By a proper choice of the precode, the missing symbols can be recovered using an erasure decoding algorithm applied to the precode. Asymptotically, the precode is assumed to be capacity achieving and the LT distribution is optimized using classical density evolution over the binary erasure channel. When finite length is considered, the precode is usually selected to have good error floor properties, i.e., for the case of the erasure channel, a high stopping distance. The remaining question is how to optimize the LT distribution. This can be achieved by means of density evolution by proper analysis of the so-called ripple: the ripple is the set of intermediate symbols connected to at least one encoded symbol of reduced degree one at a given iteration. From the analysis, it can be shown that the expected fraction of intermediate symbols in the
ripple is $1 - x - e^{(1+\epsilon)\Omega'(x)}$, where $x$ is the fraction of intermediate symbols that have been already recovered and $\epsilon$ is the overhead [9]. To avoid decoding failure, one has to ensure that $1 - x - e^{(1+\epsilon)\Omega'(x)} > 0$ with $x \in [0, 1]$. Let $x_0$ be the smallest root of $1 - x - e^{(1+\epsilon)\Omega'(x)}$ in $[0, 1)$. It follows that $\delta = 1 - x_0$ is the asymptotic expected fraction of unrecovered intermediate symbols. Thus, to design a good distribution, we need to find $\delta$ as small as possible such that $1 - x - e^{(1+\epsilon)\Omega'(x)} > 0$ for $x$ in $[0, 1 - \delta)$. To take into account the variance of the ripple size when finite length is considered, some heuristics can be used to constrain the optimization problem. The preceding equation becomes

$$1 - x - e^{(1+\epsilon)\Omega'(x)} \geq c\sqrt{\frac{1 - x}{k}}.$$  

It can be rewritten as

$$\Omega'(x) \geq -\log \left(1 - x - c\sqrt{\frac{1 - x}{k}}\right) \frac{1}{1 + \epsilon}, \quad x \in [0, 1 - \delta].$$

By discretizing the interval $[0, 1 - \delta]$, we obtain a set of inequalities. Then, by minimizing the cost function $\Omega'(1)$ (ie. the average degree distribution), we can obtain good degree distribution by solving the corresponding linear program. In practice, the precode can be itself a concatenation of error correcting codes. For example in [9], an efficient Raptor codes is proposed where the precode is the concatenation of an extended Hamming code with a LDPC code. This is mainly motivated by the fact that the extended Hamming code can correct stopping sets of relatively small sizes that can hinder decoding of the high rate LDPC code.

Originally, Raptor codes are not systematic codes. However, a lot of applications may require the coding engine to be systematic. Therefore, [9] has proposed a method to design properly a systematic Raptor code. For more information, please refer to [1], [9]. Interesting tutorials on Raptor codes can be found in [1], [2], [8]. The connection between the maximum a posteriori and BP decoding is investigated in [16] by defining EXIT (extrinsic information transfer) functions for both LT and Raptor codes [10].

November 27, 2013 DRAFT
C. Decoding algorithms for the BEC

As seen in the previous section, the performance of a sparse-graph based fountain code is mainly dependent on the number of source symbols $k$. In particular, for small $k$, message passing algorithms become inefficient, requiring a relatively large overhead to achieve a small probability of failure of the decoding process. The alternative, however, cannot be to resort to a pure maximum likelihood decoding strategy that can be far too complex for increasing $k$. To achieve a reasonable trade-off between performance and complexity for a wide range of $k$, some hybrid decoding algorithms have been proposed. The rational behind these approaches is as follows: the decoder tries to combine the computational efficiency of a message passing based algorithm with the optimality of the Gaussian elimination. The main idea is to make use of iterative decoding each time it is possible while resorting to a Gaussian elimination (i.e. a linear system inversion) only when required. The inactivation decoder has been proposed as an efficient algorithm to decode fountain codes. The main concepts have been briefly described in [1], [8], but further details are available in [17]–[20]. Thus, as $k$ increases, the efficiency of the BP decoder will also increase and the Gaussian elimination that will be used to recover the few remaining unrecovered source symbols will also have a decreasing complexity in average. This algorithm is part of the recommendations for the different standards where Raptor codes have been adopted for upper-layer forward error correction. Several papers [21]–[24] have considered the improvement of the algorithm initially proposed in [20]. As pointed out in [9], a noticeable consequence is that the analysis, the distribution optimization and the finite length code design, as originally proposed for a BP decoder, are no longer optimal. Thus, one has to design specific distributions for this new decoding scheme. Unfortunately, if a lot of references are considering the design of distributions for iterative decoding, only a few ones are really addressing the design of codes under inactivation decoding (or more generally hybrid ML+iterative decoders).

In [1], the design and requirements of standardized Raptor codes are finally explained, that give
enlightening insights for their practical design. More recently, [25] proposes an explicit method to design Raptor codes for inactivation decoding. Generalizing the approach taken in [26], one can also cite [27], [28] that can be an instructive complement where efficient pivoting algorithms exploiting sparsity are presented and where code design is addressed for LDPC code decoding over the erasure channel. For further details on decoding algorithms for the erasure channel refer to Chapter 5.

D. Raptor codes in standards

The Raptor codes have been adopted in several standards. Two code families have been commercially deployed, namely the R10 Raptor code and the RQ (RaptorQ) code [29]. The R10 code has been adopted in the following standards: 3GPP Multimedia Broadcast Multicast Service [20], IETF [30], [31], and ETSI for DVB applications [32]. For more details, you can refer to [1], [8], [29], [30].

IV. EXTENSIONS TO NOISY CHANNELS

In this section, we explain how LT and Raptor codes can be extended to other channels. As particular instances of sparse-graph based codes such as LDPC or LDGM codes, iterative decoding of LT or Raptor codes on noisy channel is straightforward by instantiating the BP decoding algorithm for a specific channel [10], [33], [34]. For this specific channel, the update equations of the BP decoder can be used to describe the asymptotic behaviour by means of density evolution or its tailored approximations [10].

A. The additive white Gaussian noise channel

First investigations on fountain coding for noisy channels have been done in [35]–[37] where the performance of LT and Raptor codes are investigated for both the binary symmetric channel (BSC) and the binary additive white Gaussian noise channel (AWGN). Based on simulations,
it is shown that LT codes exhibit severe error floors while Raptor codes seems to have a better behaviour in the error floor region for reasonably large codeword lengths. The error floor problem of LT codes is in fact typical for codes that belong to the class of LDGM codes [10]. Performance and design of Raptor codes over general binary memoryless symmetric channels have been investigated in [33]. In particular, it is shown that Raptor codes are not universal for any other channels than the binary erasure channels, as it has been shown by [9]. However, while they are not universal, Raptor codes can achieve a constant gap to capacity. Thus, in theory, they can work relatively close to capacity for a wide range of channel parameters. Bounds on the proportion of encoded symbols with degree one and two have been also derived that are the equivalent of the stability condition for LDPC codes. As for LDPC or LDGM codes, one are able to perform an asymptotic analysis of the iterative decoding such as density evolution or one of its approximations. For the AWGN channel, if unidimensional analysis is considered, it is possible to derive an asymptotic analysis based on a Gaussian approximation (GA) of the log-likelihood messages that are propagated during the BP iterative process (see Chapter 6 for more details on asymptotic analysis of sparse-graph codes). We can then derive formally the update equations of the decoder by tracking mono-dimensional quantities associated with Gaussian approximated messages. Such quantities can be the mean of the messages as initially proposed by [38] or entropy/mutual information quantities defining EXIT based asymptotic analysis [10]. Thus, to optimize, the authors in [33] have considered an asymptotic analysis based on a refined Gaussian approximation [39]. With their method, they have been able to design very efficient distributions for the AWGN channel. Based on a density evolution analysis, some practical aspects of the design process of good Raptor codes for finite block lengths for general memoryless symmetric channels using a simple model of the convergence behaviour has been investigated in [40]. In [41], the authors have improved the approach taken in [33] by designing generalized Raptor codes that can achieve different rates (ie. operating with different channel parameters) in a rate-compatible way. In [42], [43], the optimization method proposed in [33] is studied in details.
Some new insights are given on how to set the different parameters and their domain of validity. More recently, [44] has investigated a ripple-based design of LT codes for the AWGN channel including explicitly the finite length constraint. The analysis leads to similar results as reported in [9] for the BEC case and allows to derive an elegant optimization method based on a simple linear programming approach.

Other approaches have been considered based on a EXIT analysis such in [45], [46]. The EXIT based asymptotic analysis has been derived for Raptor codes for both serial or joint decoding. Serial decoding refers to as decoding first the LT then the precode, while joint decoding considers the decoding of both the LT and the precode using the joint Tanner graph. Note that LT codes can be seen as a particular case. Some bounds on degree one and two encoded symbols are derived, similarly to [33] but in the EXIT context. As in [42] for the mean evolution, the influence of different parameters for the design has also been investigated. Using this approach, an alternative method for the LT distribution optimization has been derived in [45]. Based on this analysis, a pragmatic approach for the finite length design has also been proposed in [46], [47]. The method is based on the fact that high rate precodes have poor cycle spectrum properties when \( k \) decreases for a fixed rate, leading to higher error floors. Thus, when \( k \) decreases, lowering the rate of the precode while carefully optimizing the LT distribution to minimize the resulting overhead leads to an efficient method to design Raptor codes under joint decoding.

Some studies have also been dedicated to the analysis and optimization of LT codes over the AWGN channel to predict and combat the error floor phenomenon. In [48], an EXIT analysis has been used to derive upper and lower bounds on the asymptotic average error probability that appear quite tight to predict the asymptotic performance in the error floor region. The inherent trade-off between overhead and error floor performance when asymptotically designing LT distributions is pointed out. Then, a method to design the precode of a Raptor code is proposed when serial decoding of the LT and the precode is used. The design is based on the fact that the equivalent channel for the precode can be seen as a mixture of channels with known
parameters that are assumed to be Gaussian when an EXIT analysis is used. More recently, bounds for the error floor prediction have been derived in [49] following a similar approach. Based on their analysis, the authors have proposed a modified LT encoding process that enhances the performance in the error floor region with almost no threshold degradation for large range of signal-to-noise ratios (SNRs). The proposed scheme intends to limit the numbers of source symbol nodes with low average degree. The performance of LT codes over the AWGN channel is also investigated in [50]: the ensemble weight distribution of LT codes is derived and maximum likelihood decoding performance is upper-bounded by a refined version of the union bound.

B. Fading channels

As for LDPC codes, the analysis of LT/Raptor codes can be extended to time varying wireless fading channels. Several works have addressed these topics for the two main classes of fading channels: the uncorrelated fast fading channels that belongs to the class of ergodic memoryless channels and non ergodic block fading channels (including quasi static block fading channels). A detailed study and design of LT and Raptor codes over the uncorrelated Rayleigh fading channel is presented in [51]. Using an EXIT approach slightly adapted to the Rayleigh case, it is shown that the optimization is very similar to that of the AWGN case. It can be easily extended to any kind of fading distributions (Rice or Nakagami for example). When considering non ergodic time varying channels, fountain coding appears as an attractive solution. It can be of real interest for time varying channels when no channel state information is available at the transmitter or when no feedback is possible. Moreover for multicast communications, different users may experience different fading conditions, and thus, fountain coding may appear as a possible coding scheme to cope with multiple simultaneous unknown fading realisations. The use and interest of rateless coding in the context of non ergodic channels has been first introduced and motivated in [52]. By means of simulations, it is shown that fountain coding implemented by means of Raptor codes has some advantages in terms of efficiency, reliability and robustness over conventional
fixed-rate codes. A more theoretical study for analysing the fountain coding scheme on block fading channels is given in [53] based on some information theoretic arguments. It is shown the existence of efficient and reliable rateless codes for quasi-static fading channels and block-fading channels with large coherence times. Thus, there exists a rateless code that can operate (ie. having an a posteriori rate) within a constant gap to the capacity of any instantaneous realization of the channel. This is often referred to as an almost universal or a robustness property. In [51], [54], the analysis and performance of Raptor codes for the quasi-static block fading channel have been investigated. It is shown that, by constraining the optimization problem simultaneously for different channel capacities, it is possible to design Raptor codes that perform close to the theoretical outage probability limits. The influence of imperfect channel state information at the receiver has also been investigated, showing that the use of channel log-likelihood ratios using channel estimation accuracy to avoid mismatched decoding is crucial to ensure a good convergence behaviour.

C. Other channels

Sparse graph codes for fountain coding have also been investigated for a wide class of channels. Benefiting from the numerous existing studies for LDPC codes, several applications to other kind of channels have been investigated. In [55], a more realistic model for the erasure channel is studied by means of simulations. In this context, the erasure rates are packet length dependent. It is shown by simulations that the packet lengths when using rateless codes in these particular settings can be optimized. The performance of rateless codes over the BSC with hard decoding algorithms has been investigated in [35], [36]. Their optimization has been addressed in [40] and [56] with two different approaches. The particular case of a piecewise stationary BSC channel is investigated in [57]. Rateless coding for multiple inputs - multiple outputs (MIMO) channels have also been investigated. In [58], the optimization of Raptor codes for MIMO channels is derived using the EXIT approach [45], [59]. This method can be also applied to parallel
Gaussian channels. The application of LT-based coding schemes to systems using different MIMO transmit diversity systems has been investigated in [60], [61] based on a specific systematic LT coding scheme [62], [63]. Raptor codes for hybrid error-erasure channels with memory have been considered in [64], [65]. Finally, the design of Raptor codes is addressed for some different multiple access channels in [66]–[71].

D. Link to fixed rate counterparts

The performance of sparse graph based fountain codes is often given in terms of error rate versus overhead to assess their ability to operate close to the optimal channel rate. However for a fixed overhead, these codes can be considered as fixed length codes and can be then compared to other fixed rate counterparts. For example in [72], fixed-rate Raptor codes have been shown to provide performances that are comparable with or better than that of several state-of-the-art codes over memoryless fading channels. Fountain coding is a coding paradigm that inherently enables incremental redundancy transmissions with or without an available feedback. Hence, it suggests to compare the performance of LT/Raptor codes to that of fixed rate coding schemes enabling incremental redundancy. In practice, they are often compared to rate-compatible fixed rate code families such as rate-compatible LDPC or irregular repeat accumulate (IRA) codes as in [73]. In their settings, it is shown that rate-compatible quasi-cyclic LDPC or IRA codes perform better for high SNRs while performing almost similarly at low SNRs.

V. Advanced sparse graph based rateless coding schemes

In the previous sections, we have presented mainly results for LT and Raptor codes: their structure, their performance analysis, their decoding algorithms and their optimization for different channels. In this section, we provide a review of the different extensions that have been proposed. As an instance of sparse graph codes, one can think of applying any extension that is applicable to LDPC-like codes. We first review the different extensions and generalizations of
LT/Raptor coding schemes that have been proposed so far in the literature. Then we introduce a more general view of rateless coding by considering coding schemes or studies related to rateless coding that do not only resort to sparse graph codes with iterative or ML decoding.

A. LT/Raptor codes extensions

1) Improved decoding algorithms and sequence designs: Several studies have considered the improvement of the iterative decoding to improve performance. Most of the studies are considering improvements of LT decoding to lower the error floor or to achieve a better complexity-performance trade-off, especially in the context of noisy channels (eg. AWGN, BSC or fading channels). Most of previous presented studies have considered BP iterative decoding. In practical scenarios on noisy channels, the transmitter sends successive blocks of \( p \) bits. Each time a block is received, the BP decoder is applied on the Tanner graph that is built from the received encoded symbols. Thus, the Tanner graph is dynamically updated from one block to another. A simple strategy is to reset the decoding process at each newly received encoded symbols block (all messages in the graph except observation messages are set to zero). This strategy is sometimes referred to as message reset decoding [74], [75]. By doing so, each decoding attempt is independent and does not make use of the soft information produced in the previous decoding attempt. Another strategy consists of considering incremental decoding: the decoding is initialized with messages obtained from the previous decoding attempt [74], [75]. Only messages from newly added edges in the graph are set to zero. Thus, with this strategy, the BP decoder “continues” from the decoding results of the last decoding attempt. This results in a decrease of complexity since the total number of decoding iterations to decode a codeword on average is reduced. Some improved versions have been proposed in [76], [77] that combines incremental decoding with a particular scheduling called informed dynamic scheduling. For this strategy, early termination is achieved through the use of a new stopping criterion. An extension of the preceding method that takes into account the trapping sets of the LT codes is proposed in [78],
[79]. Recently, in [80], the ripple analysis has been rethought in the BEC context, enabling decreasing ripple size during the decoding process. This new strategy leads to a new design procedure for fixed $k$, providing distributions that show significant performance improvements compared to that of existing LT distributions.

2) Generalization of LT/Raptor codes: As for LDPC codes, several extensions of the structure of LT/Raptor codes have been proposed. In a series of works [81]–[84], several direct extensions of the original LT coding scheme have been proposed to design LT code families with good error floor properties. In [81], it is first proposed to consider accumulate LT codes that can be viewed as a particular case of Raptor codes with an accumulator as a pre-code. The proposed scheme can be systematic or non systematic. It is shown that these rateless codes perform as efficiently as LT codes do. They can reliably recover a constant fraction of information bits. Although rateless, there are not universal. To further improve this scheme, a modified version is proposed, named doped accumulate LT codes, which can be regarded as the parallel concatenation of a doping code and an LT code using an accumulate precode. In this scheme, the parity bits of the accumulator are doped with those generated by a semi-random LPDC code as proposed in [85], [86]. The rational behind is that the doping bits can help the decoder to reliably remove the residue loss rate of the accumulate LT code part. This scheme has competitive performance compared to that of Raptor codes. Mainly inspired by the latter scheme, a design of systematic LT codes with linear complexity using a doped accumulator to address the error floor problem is proposed in [84]. The systematic version of [81] is investigated in details in [82], [83] where a quasi-systematic version is proposed. Recently, an enhanced rateless version of [85], [86] has been proposed to design capacity approaching systematic rateless codes for the erasure channel [87]. The main principle is to design an efficient erasure code as in [85], combined with a constrained scrambling technique. Note that the resulting family falls into the family of rateless irregular repeat accumulate codes for binary erasure channels.

Following ideas from [41], reconfigurable rateless codes that select suitable LT distributions
according to the SNR are proposed in [88]. An adaptive rateless coding scheme is designed in an incremental manner to be able to operate over a wide range of SNRs. The limitation is however the need for a feedback in the original setting. Similarly, in [89], a class of rateless LDPC codes for the erasure channel with dynamic degree distributions is proposed. The method is based on the design of constrained layered rate compatible degree distributions for a concatenated code. In the same spirit, fountain codes with varying probability distributions during the encoding process have been proposed in [90]. Using a more involved theoretical framework it is shown that it is possible to design a set of varying distributions that can lead to a significant reduction of the overhead over the BEC.

All the previously mentioned improvements were mainly dedicated to the LT part or to the precode part using the initial structure as proposed in [5], [9]. As for the case of LDPC codes, all generalizations that have been applied to LDPC/LDGM or turbo-codes can be used for the design of sparse graph based fountain codes. Thus, as a more general family, one can think of the multi-edge type family [10] that encompasses a wide class of sparse graph based codes. Several multi-edge type approaches have been considered so far [91]–[97]. Among them, a well-studied class is the class of generalized fountain codes, also denoted as weighted fountain codes [91]–[93]. The $k$ source symbols are partitioned into $n_c$ sets, also referred to as classes, with $k_i$ source symbols for the $i$-th class such that $\sum_i k_i = k$. The source symbols chosen to produce an encoded symbol are not selected uniformly at random. During the encoding process, the source symbols from the different classes are assigned different probabilities of being chosen, after the degree of an encoded symbol has been sampled. This encoding procedure induces different recovery properties for the different classes. In the original setting, the assignment of the probabilities is done such that input source symbols from the “more important” classes are more likely to be chosen in generating the encoded symbols. This results in enhanced unequal error recovery/protection (UEP) properties. An improved design method based on a ripple analysis is given in [98]. Another class of multi-edge type codes is the so-called windowed fountain
codes also referred to as expanding windowed fountain codes [94], mainly inspired from [99]. This class differs from the preceding class in that the partition of the \( k \) source symbols is done in a different manner: the proposed partition determines a sequence of subsets of the entire source symbols set with strictly increasing cardinality. Each subset is called a \textit{window} of source symbols. The \( i \)-th window consists of the first \( k_i \) symbols that will be used to produce the encoded symbols associated with the \( i \)-th class of encoded symbols. An expanding windowed fountain code is a fountain code which assigns each encoded symbol to the \( i \)-th window with probability \( \Gamma_i \) and encodes this chosen window using an LT code with distribution \( \Omega^{(i)}(x) = \sum_{j=1}^{k_i} \Omega_j^{(i)} x^j \). This also leads to UEP properties. A generalization of both preceding classes using the multi-edge framework is proposed in [95]. Protograph based Raptors-like codes have been studied in [96], [97]. Spatially-coupled rateless codes have been investigated in [100], [101] showing that universality can be achieved using spatial-coupling.

The concatenation of sparse-graph based rateless codes (LT or Raptor) with modulation has been often considered in the literature leading to rateless bit-interleaved coded-modulation (BICM) or multi-level coded modulation schemes with high spectral efficiency. The proposed approaches are mostly based on an EXIT chart-based design [59], [102]–[107].

3) \textit{Distributed LT codes:} Distributed LT codes have been first introduced in [108], [109]. Distributed LT codes are used to independently encode data from multiple sources in a network. The resulting encoded packets can be then combined at a relay node. Distributed LT codes are designed to ensure that the resulting equivalent degree distribution seen at the destination is a good approximation of an LT code. A complementary study is given in [110] where the design is generalized. The distributed approach has also been considered for distributed storage as in [111]–[115]

4) \textit{Non binary fountain codes:} Fountain codes are often referred to as binary fountain although symbols can refer indifferently to bit or group of bits (packet). This is mainly due to the fact that, in the original setting, fountain coding assumes a bitwise XORing of the packets when
producing the encoded symbols (packets). Consequently, we actually only resort to the additive 
subgroup of $\mathbb{F}_2^k$. The fountain coding approach can be easily extended to the non binary case 
by considering that symbols are $p$-tuples seen as elements of the finite field $\mathbb{F}_q$, where $q = 2^p$ 
is the field order. Encoding is performed using an equivalent generating matrix whose non zero 
entries are randomly chosen from $\mathbb{F}_q$. For a given vector of $k$ source symbols in $\mathbb{F}_q^k$, the encoded 
symbols are generated as follows: At each time $j$,

1) Generate a non binary $k$-tuple, noted $(g_{1j}, \ldots, g_{kj})$ randomly sampled from a given distri-
bution $D$ on the vector space $\mathbb{F}_q^k$; generally each element is independently sampled with 
uniform probability defining a random non binary fountain code,

2) Calculate the output symbol $c_j$ as $c_j = \sum_i g_{ij}x_i$, where all operations are performed on 
$\mathbb{F}_q$.

The fundamental performance of random non binary fountain codes has been investigated in 
[116] using hybrid decoding. The performance of some random linear fountain codes over higher 
order Galois fields under maximum likelihood decoding is studied in [117], [118]. The parallel 
concatenation of maximum distance separable codes with random linear non binary fountain 
codes is considered in [119]. Based on simple non binary codes enabling multiplicatively repeated 
non-binary parity symbols, a very simple rateless/incremental redundancy scheme that performs 
well over noisy channels has been designed in [120], [121]. The decoding used is non binary 
BP decoding.

5) “Turbo”-based fountain coding: Turbo fountain codes have been introduced in [122], 
[123], extending the turbo principle to the rateless coding context. For this scheme, parallel turbo-
codes are considered and a possibly limitless number of parallel systematic recursive convolutive 
encoders is used to practically implement the fountain paradigm in a “turbo” context. In this 
setting, turbo fountain codes can be viewed as a subclass of Raptor codes, and in particular, 
as a subclass of precode-only (PCO) Raptor codes, where the precoder is a parallel turbo-code. 
Several works have considered this particular setting with some extensions [124]–[126].
B. Other rateless coding schemes: beyond sparse graph codes

Fountain sparse graph codes are not the only coding paradigm to implement rateless coding. Apart from the previously presented coding schemes, other approaches for the physical layer have been investigated for broadcast applications over noisy channels. In this context, rateless coding is a communication paradigm that enables to transmit a common information content and serve different users with different wireless link qualities without feedbacks. Most of these approaches are dedicated to the design of rateless schemes that are in fact incremental redundancy schemes, mainly inspired from rate-compatible coding schemes as used for example in Hybrid Automatic Repeat reQuest (HARQ) retransmission protocols. Apart from rate-compatible coding schemes derived for the most popular code families (see Chapter 12 for more details on rate-compatible code design), different rateless coding schemes have been proposed in the literature. A first approach is to consider a layered coding scheme as proposed initially in [127]–[129]. Taking an information theoretical approach, the authors have investigated a rateless coding scheme based on layered encoding and successive decoding. The proposed scheme is an incremental redundancy scheme using at most $M$ transmitted redundancy blocks. Each of the $M$ redundancy blocks is a linear combination (in the signal space) of $L$ codewords issued from $L$ distinct codebooks, each of which being associated with a given layer $l$, $l = 1 \cdots L$. At the receiver, decoding is achieved using successive decoding. The proposed scheme defines a rate compatible scheme that enables communication for different SNR thresholds related to the effective rate $R/m$ associated with the $m$ received redundancy blocks, with $R$ being the rate associated with the first transmitted redundancy block. In [128]–[130], it is shown that perfect codes can be designed for some rate $R$ with $L = M$. The initial constructions are provided for the Gaussian channel and are also considered for time varying channels. This solution can be implemented using common efficient error correcting schemes such as LDPC or turbo codes. The extension of this approach to the MIMO context has been considered in [131]–[134]. This coding approach has been practically
implemented in [135]. Recently, a new rateless coding approach has been proposed in [136]–[138], called spinal codes. This rateless coding scheme has a totally new structure and is proved to be capacity achieving for some noisy channels such as the AWGN channel. The main feature for encoding is a sequential application of a hash function over the message bits. The decoding is achieved using maximum likelihood decoding over a particular tree.

VI. APPLICATIONS OF RATELESS CODING

A. Rateless coding versus IR-HARQ

Fountain codes or layered rateless coding schemes have been naturally considered for applications such as HARQ retransmission protocols with incremental redundancy (IR), referred to as IR-HARQ. Indeed, Fountain codes or layered rateless coding schemes are in essence IR schemes. Numerous works have compared pros and cons of rateless coding schemes compared to that of rate compatible fixed rate code based approaches. Each system has different advantages. While rate compatible fixed rate codes can operate closely to capacity for rates above the mother code rate (minimum designed rate), fountain codes, although not universal for noisy channels, can also operate close to capacity even for low SNRs (no low rate limit). However, for performance at high SNRs, they seem limited compared to their fixed rate counterparts, especially at small lengths due to the precode rate penalty. For more details see for example [139]–[144].

B. Multimedia communications and broadcasting

1) Unequal error protection: Due to their inherent structure, generalized Raptor codes [91]–[93] or expanding windows fountain codes [94] have been initially design to provide UEP [95]. Since then, numerous works have considered extensions or improved fountain coding schemes to handle UEP for multimedia communications. [145]–[158].

2) Multimedia video communications: The main application of rateless coding is multimedia broadcasting for which Raptor codes have been standardized [159], [160]. Numerous references
are available on the subject and describe how rateless coding can be efficiently implemented to enable efficient and reliable multimedia communications. Video multimedia delivery is one of the most successful applications of the rateless paradigm [24], [146], [147], [152], [153], [161]–[168].

C. Wireless networks and relays

Due to its inherent nature to provide low complexity distributed coding schemes or its ability to provide incremental redundancy, fountain coding has been widely studied for applications such as wireless sensor networks or cooperative communications in relay networks. Thus, numerous studies have considered analysis and/or optimization of fountain codes or rateless coding schemes for cooperative relay networks [167], [169]–[187]. Rateless coding has also been investigated for cognitive radio networks in [188]–[191]. Other related kind of applications can be found in [177], [192]–[199].

D. Distributed storage and data dissemination

Other natural and popular applications of rateless coding are related to distributed data storage [111], [113], [114] or data dissemination or collection [200], [201]. For more details about these kinds of applications, please see the corresponding references.

E. Source and source-channel coding

A broad class of applications are related to source and joint source-channel coding. Several works are investigating the use of rateless coding for compression or quantization [202]–[205]. Others are considering distributed source coding [206]–[209]. Finally joint source and channel coding is considered in [165], [210]–[212].
REFERENCES


[31] ETSI TS 102 472 V1.3.1, Digital Video Broadcasting (DVB); IP Datacast over DVB-H: Content Delivery Protocols, June 2009.


