Joint Protocol-Channel Decoding for Robust Frame Synchronization

Usman Ali, Michel Kieffer, Pierre Duhamel

Joint Protocol-Channel Decoding for Robust Frame Synchronization

Usman Ali *Member, IEEE,* Michel Kieffer *Senior Member, IEEE,* and Pierre Duhamel, *Fellow, IEEE*

Abstract

In many communication standards, several variable length frames generated by some source coder may be aggregated at a given layer of the protocol stack in the same *burst* to be transmitted. This decreases the signalization overhead and increases the throughput. However, after a transmission over a noisy channel, Frame Synchronization (FS), *i.e.*, recovery of the aggregated frames, may become difficult due to errors affecting the *burst*s.

This paper proposes several robust FS methods making use of the redundancy present in the protocol stack combined with channel soft information. A *trellis*-based FS algorithm is proposed first. Its efficiency is obtained at the cost of a large delay, since the whole burst must be available before beginning the processing, which might not be possible in some applications. Thus, a low-delay and reduced-complexity *Sliding Window*-based variant is introduced. Second, an improved version of an *on-the-fly* three-state automaton for FS is proposed. Bayesian hypothesis testing is performed to retrieve the correct FS. These methods are compared in the context of the WiMAX MAC layer when bursts are transmitted over Rayleigh fading channels.

Index Terms

Cyclic redundancy check, Decoding, Protocols, Wireless communication.

U. Ali, M. Kieffer and P. Duhamel are with the L2S–CNRS-Supelec-Univ Paris-Sud. M. Kieffer is currently on leave at LTCI–CNRS-Telecom ParisTech. U. Ali has been partly supported by Microsoft Research through its PhD European Scholarship program, M. Kieffer has been partly supported by the European NoE NEWCOM++ and by the Institut Universitaire de France. Parts of this work have been presented at IEEE SPAWC 2009, IEEE PIRMC 2010, and IEEE ICC 2011.

I. INTRODUCTION

Frame Synchronization (FS) is an important problem arising at various layers of the protocol stack of several communication systems. The most obvious being, at Physical (PHY) layer, to recover the payload and the side information (headers, etc...) of PHY packets or *frames*. Therefore, FS has received continuous attention since many years [1]–[6].

First results [1]–[3] considered data streams in which regularly spaced fixed patterns or *Synchronization Words* (SW) are inserted to delimit fixed-length frames, as standardized, *e.g.*, at the PHY layer of DVB-H [7] for MPEG2 transport stream frames. Synchronization tools based on the maximization of the correlation between the SW and the received data have been proposed in [1]. The optimal statistic for SW has been proposed for the AWGN case in [2] by taking into account the presence of data around the SW. This was further extended in [4] for more sophisticated transmission schemes and in [8], [9] to provide robustness against frequency and phase errors.

In absence of SW, the Header Error Check (HEC) field of the frame header can be employed for FS, as is the case, *e.g.*, for ATM fixed-length frames or *cells* [10]. This can be done by checking the consistency of the received header sequence candidates with the HEC. FS may be done bit-by-bit or more efficiently, as standardized in [10], using a three-state (3S) automaton.

In many communication systems, frames are of variable-length, see, *e.g.*, the 802.11/802.16 standards [11], [12] for Wireless Local Area Networks (WLANs). In presence of SW, in [6], [13]–[15], several hypothesis testing techniques have been proposed to perform variable-length FS. In absence of SW, assuming a length field is present in the header of the variable-length frame, the conventional *Hard Decision* (HD)-based FS can be performed when the noise is moderate. In [5], the HEC field has been employed to perform FS of variable-length IP frames using a 3S automaton adapted from [10]. In several situations, *e.g.*, when the HEC field is erroneous itself or when its length is too short compared to the header size, false alarms can hamper the efficiency of the algorithm.

The above-mentioned FS techniques work on-the-fly, *i.e.*, at each time, only few additional data samples are processed to perform FS and almost no latency is introduced. Moreover, FS

has mainly been considered at PHY layer, albeit this problem may also occur at upper layers of the protocol stack. In several communication systems, frame aggregation at intermediate layers of the protocol stack have been proposed, see, *e.g.*, 802.11/802.16 standards. This increases the throughput, but, any error upon reception of the burst header may result in the loss of the complete burst. Simultaneous synchronization for several frames are thus considered in [16] to improve FS efficiency. Using a *hold-and-sync(hronize)* approach, where a whole burst is processed at each step, may significantly improve the performance of the FS. This has been proposed in [17] for the segmentation of MAC frames aggregated in WiMAX PHY bursts. However, in this approach, the receiver must wait for whole or at least for a large part of a burst, which is sometimes undesirable in delay-constrained situations.

To be fully efficient, FS techniques require soft information from the channel decoders at PHY layer to reach upper layers, as proposed by [18]. Combined with Joint Source-Channel Decoding (JSCD) (see [19] and the references therein), robust FS techniques allow to significantly reduce the amount of frames that need to be retransmitted.

This paper proposes *Joint Protocol-Channel Decoding (JPCD)* techniques for FS. They exploit soft information at the output of the channel (or channel decoder) as well as redundancy in the protocol layers (SW, known fields, Cyclic Redundancy Check (CRC) or checksums, *etc*).

Section II states FS as a maximum *a posteriori* (MAP) estimation problem. Trellis-based *hold-and-sync* techniques for FS are described first. A modified BCJR algorithm [20] is used to estimate the locations of frame boundaries. Then, an adaptation of the reduced-complexity *Sliding Window* (SlW) [21] variant of the BCJR algorithm is proposed to get a low-delay and reduced-complexity version of the previous algorithm. Second, an improved version of an on-the-fly 3S FS automaton [5] is presented in Section III. It combines robust header recovery techniques from [22] with Bayesian hypothesis testing inspired from [6], [13], [14] to localize frame boundaries via a sample-by-sample search. The techniques presented here are quite general. They are illustrated in Section IV with the FS of WiMAX MAC frames aggregated in bursts transmitted to the PHY layer [12], but are easily extended to other protocols where frame aggregation is performed.

II. MAP ESTIMATION FOR FRAME SYNCHRONIZATION

A memoryless random process is introduced in Section II-A to represent the length of each aggregated frame. This allows to model the cumulated lengths of aggregated frames within a burst by a Markov process, see Section II-B. All possible successions of frames in a burst are then described by a trellis inspired from [23], on which a BCJR algorithm is applied to perform a MAP estimation of the number of aggregated frames and of their boundaries, see Sections II-C and II-D. A reduced-delay FS approach, involving overlapping slices of the burst, is then presented in Section II-E.

A. Frame structure

Consider the n-th variable-length frame at a given protocol layer. Its length in bits is denoted as $\lambda_n = \ell_h + \ell_{p,n}$, where the leading ℓ_h bits represent the frame header, of fixed length, and the remaining $\ell_{p,n}$ bits constitute the variable-length payload. In the header, ℓ_c bits are some HEC bits, CRC, or checksum. The length λ_n is assumed to be a realization of a stationary memoryless process Λ characterized by

$$\pi_\lambda = \Pr(\Lambda = \lambda) \neq 0 \text{ for } \ell_{\min} \leqslant \lambda \leqslant \ell_{\max}, \tag{1}$$

where $\ell_{\min}$ and $\ell_{\max}$ are the minimum and maximum lengths in bits of a frame.

To help FS, the bits of the header $\mathbf{h}_n$ are partitioned into four fields. The *constant* field $\mathbf{k}$, contains all bits that do not change from frame to frame. It includes the SW indicating the beginning of the frame, if available, and other bits that remain constant [22] once the communication is established. The header is assumed to contain a *length* field $\mathbf{u}_n$, indicating the size of the frame (including the header) in bits λ_n. Our task is to estimate the successive values taken by this quantity in all frames of the burst. The *other* field $\mathbf{o}_n$, gathers all bits of the header that are not used to perform FS. Finally, the *HEC* field $\mathbf{c}_n$ is assumed to cover the $\ell_h - \ell_c$ "working" bits of the header, *i.e.*, $\mathbf{c}_n = \mathbf{f}(\mathbf{k}, \mathbf{u}_n, \mathbf{o}_n)$, where $\mathbf{f}$ is some encoding function, *i.e.*, CRC or checksum. The payload (assumed not protected by the HEC), denoted by $\mathbf{p}_n$, is modeled here as a binary symmetric sequence.

In what follows, the length of a vector $\mathbf{z}$ (in bits) is denoted as $\ell(\mathbf{z})$ and its observation (soft information) provided either by a channel, a channel decoder, or a lower protocol layer is denoted as $\mathbf{y}_z$. $\mathbf{z}_a^b$ represents the sub-vector of $\mathbf{z}$ between indexes a and b (in bits).

B. Aggregated frames within a burst

Consider a burst of L bits consisting of N *aggregated frames*. This burst contains either $N-1$ *data* frames and a *padding* frame containing only padding bits (here 1s), see Figure 1, or N data frames. Here, each data frame is assumed to follow the syntax described in Section II-A.

Assuming that L is fixed *before* frame aggregation and that N is not determined *a priori*, the accumulated length in bits of the n first aggregated frames can be described by a Markov process with state S_n and *a priori* transition probabilities deduced from (1). If $\ell < L$, then

$$p\left(S_n = \ell \,|\, S_{n-1} = \ell'\right) = \begin{cases} \pi_{\ell-\ell'} & \text{if } \ell_{\min} \leqslant \ell - \ell' \leqslant \ell_{\max} \\ 0 & \text{else.} \end{cases} \tag{2}$$

If $\ell = L$, then

$$p\left(S_n = L \,|\, S_{n-1} = \ell'\right) = \begin{cases} 0, \text{ if } L - \ell' > \ell_{\max} \\ 1, \text{ if } 0 < L - \ell' < \ell_{\min} \\ \sum\limits_{k=L-\ell'}^{\ell_{\max}} \pi_k, \text{ else.} \end{cases} \tag{3}$$

In (3), if $0 < L - \ell' < \ell_{\min}$, there is not enough space in the burst to put a data frame and the n-th frame is thus necessarily the (last) padding frame. If $L - \ell' > \ell_{\max}$, there is enough space to put a data frame of any allowed length, thus, the n-th frame cannot end the burst. In the other cases, the last frame is either a frame of exactly $L - \ell'$ (probability $\pi_{L-\ell'}$) or a padding frame of $L - \ell'$ bits. A padding frame is used when the frame that should have been aggregated has a length strictly larger than $L - \ell'$ (probability $\sum\limits_{k=L-\ell'+1}^{\ell_{\max}} \pi_k$). Such frame is then put in the next burst.

With this representation, the successive values taken by S_n can be described by a trellis inspired from that in [23] for the robust decoding of variable-length encoded data, see Figure 2. Dashed transitions represent padding frames and plain transitions represent data frames.

C. Estimators for the number of frames and their boundaries

Consider a *burst* $\mathbf{x}_1^L$ of N aggregated frames and some vector $\mathbf{y}_1^L$ containing soft information (bit *a posteriori* probabilities or likelihood ratios) about the bits of $\mathbf{x}_1^L$. The vector $\mathbf{y}_1^L$ may be obtained from the output of a channel, of a channel decoder, or of a lower transparent protocol layer [18]. Accurate synchronization is assumed at the beginning of the burst: the first entry of $\mathbf{y}_1^L$ corresponds to the first bit of $\mathbf{x}_1^L$. This assumption is further discussed in Section II-D4.

The *joint* MAP estimates $\widehat{N}$ of N and $\widehat{\lambda}_n$ of λ_n, $n = 1, \ldots, \widehat{N}$, from $\mathbf{y}_1^L$ are

$$\left(\widehat{N}, \widehat{\lambda}_1, \ldots, \widehat{\lambda}_{\widehat{N}}\right) = \arg \max_{N, \lambda_1, \ldots, \lambda_N} p\left(N, \lambda_1, \ldots, \lambda_N \,|\, \mathbf{y}_1^L\right). \tag{4}$$

The fact that N is only known to satisfy $N_{\min} \leqslant N \leqslant N_{\max}$, with $N_{\min} = \lceil L/\ell_{\max} \rceil$, $N_{\max} = \lceil L/\ell_{\min} \rceil$, and $\lceil \cdot \rceil$ denoting the upward rounding and the unknown number of padding bits makes the solution of (4) quite difficult. Thus, we resort to a suboptimal estimator consisting in first estimating N before estimating the location of each frame. The MAP estimate $\widehat{N}_{\text{MAP}}$ of N is given by

$$\widehat{N}_{\text{MAP}} = \arg \max_{N_{\min} \leqslant n \leqslant N_{\max}} P\left(S_n = L | \mathbf{y}_1^L\right). \tag{5}$$

Once $\widehat{N}_{\text{MAP}}$ is obtained, the MAP estimate for the index ℓ_n of the last bit of the n-th frame is

$$\widehat{\ell}_n = \arg \max_{\ell} P\left(S_n = \ell | \mathbf{y}_1^L\right), \; n = 1, ..., \widehat{N}_{\text{MAP}}, \tag{6}$$

and the length λ_n of the n-th frame is estimated as

$$\widehat{\lambda}_n = \arg \max_{\ell} P\left(S_n = \ell | \mathbf{y}_1^L\right) - \arg \max_{\ell} P\left(S_{n-1} = \ell | \mathbf{y}_1^L\right), \; n = 1, ..., \widehat{N}_{\text{MAP}}. \tag{7}$$

These estimators are suboptimal when compared to (4), since $\widehat{\lambda}_n$ obtained from (7) is not necessarily consistent with (1).

D. Trellis-based FS Algorithm

A posteriori probabilities $P(S_n = \ell | \mathbf{y}_1^L)$ have to be evaluated in (5), (6), and (7). This may be done with the BCJR algorithm [20], by evaluating first $P(S_n = \ell, \mathbf{y}_1^L) = \alpha_n(\ell)\, \beta_n(\ell)$, where $\alpha_n(\ell) = P(S_n = \ell, \mathbf{y}_1^\ell)$ and $\beta_n(\ell) = P(\mathbf{y}_{\ell+1}^L | S_n = \ell)$.

1) Evaluation of α_n and β_n: Classical BCJR forward and backward recursions are performed with $\alpha_n(\ell) = \sum_{\ell'} \alpha_{n-1}(\ell')\gamma_n(\ell', \ell)$, $\beta_n(\ell) = \sum_{\ell'} \beta_{n+1}(\ell')\gamma_{n+1}(\ell, \ell')$, and $\gamma_n(\ell', \ell) = P(S_n = \ell, \mathbf{y}_{\ell'+1}^{\ell} | S_{n-1} = \ell')$. Being iteratively computed, these quantities must be initialized to some value. Concerning the forward recursion, the value of S_0 is perfectly known, leading to $\alpha_0(\ell = 0) = 1$ and $\alpha_0(\ell \neq 0) = 0$. For the backward recursion, the number of frames is only known to satisfy $N_{\min} \leqslant N \leqslant N_{\max}$, thus there are $N_{\max} - N_{\min} + 1$ possible final states $S_n = L$. We consider two options for the initialization of $\beta_n(L)$.

Coarse Initialization (CI): Assuming all allowed final states to be equally likely, one gets

$$\beta_n(L) = 1/(N_{\max} - N_{\min} + 1),\ N_{\min} \leqslant n \leqslant N_{\max}. \tag{8}$$

All other values of $\beta_n(L)$ are set to 0.

Precise Initialization (PI): More accurate initial values may be obtained as

$$\beta_n(L) = P(S_n = L),\ N_{\min} \leqslant n \leqslant N_{\max}, \tag{9}$$

where

$$P(S_n = \ell) = \sum_{\ell'} P(S_n = \ell | S_{n-1} = \ell') P(S_{n-1} = \ell'), \tag{10}$$

is the *a priori* probability that the n-th frame ends with the ℓ-th bit of the burst. $P(S_n = \ell)$ may be evaluated iteratively with the help of (2) and (3), starting from $n = 1$ till $n = \lceil \ell/\ell_{\min} \rceil$, with initial condition $P(S_0 = 0) = 1$ and $P(S_0 \neq 0) = 0$.

2) Evaluation of γ_n: When evaluating $\gamma_n(\ell', \ell)$, one implicitly assumes that the n-th frame starts at the $(\ell' + 1)$-th bit and ends at the ℓ-th bit of a burst. When $\ell < L$, the n-th frame is not the last one, it is thus a data frame. When $\ell = L$, depending on the value of ℓ', data and padding frames have to be considered simultaneously (parallel plain and dashed transitions in Figure 2) or only padding frames have to be taken into account (dashed transitions in Figure 2), see Appendix A for more details.

3) Complexity evaluation: The complexity of the FS algorithm of Section II-D is proportional to the number of nodes or to the number of transitions within the trellis on which FS is performed.

From Figure 2, one may easily show that the number of nodes in the trellis is

$$\mathcal{N}_n = \sum_{n=0}^{N_{\min}-1} n\left(\ell_{\max}-\ell_{\min}\right) + \sum_{n=N_{\min}}^{N_{\max}} \left(L-n\ell_{\min}\right). \tag{11}$$

Taking $N_{\min} \approx L/\ell_{\max}$ and $N_{\max} \approx L/\ell_{\min}$, (11) simplifies to

$$\mathcal{N}_{\mathrm{n}} = \frac{L^2}{2}\left(\frac{\ell_{\max}-\ell_{\min}}{\ell_{\max}\ell_{\min}}\right) = \mathcal{O}\left(L^2\right). \tag{12}$$

From each node, at most $\ell_{\max}-\ell_{\min}$ transitions may emerge. Thus, from (12), the number of transitions $\mathcal{N}_{\mathrm{t}}$ may also be approximated as $\mathcal{N}_{\mathrm{t}} = \mathcal{O}\left(L^2\right)$.

4) Limitations: The plain trellis-based FS technique presented in this section is based on the knowledge of the beginning and length of the *burst*. This requires an error-free decoding of the headers of lower protocol layers, which contain this information. This may be done using methods from [22], which enable lower protocol layers to forward soft information to the layer where it is processed. The main drawback of the proposed FS technique in terms of implementation is the increase in memory requirements for storing the soft information, estimated in [18] to be three to four times that of storing hard bits. Moreover, this hold-and-sync FS technique requires buffering the whole burst, inducing buffering and processing delays proportional to L^2, see (12). To alleviate these problems, a low-delay and less-complex FS variant is now proposed.

E. Sliding window-based FS

Low-latency variants of the BCJR algorithm considering a SlW [21], [24] have been proposed for the decoding of convolutional codes. A classical BCJR decoding is performed within a window, shifted bit-by-bit [21] or by several bits [24] depending on the complexity and efficiency target. From one window to the next, only the results obtained during the forward iteration are reused.

The trellis in Figure 2 has a variable number of states for each value of the frame index n. One may apply directly the SlW ideas, but due to the increase of the size of the trellis (at least for small values of n), this would still need very large trellises to be manipulated, with an increased computation time.

Here, the SlW-based approach of [21], [24] is adapted. The burst is still divided into overlapping windows. For each window, a *reduced-size* trellis is built, the beginning of which corresponds to the end of the frame deemed reliably synchronized in the previous window. As in [24], considering overlapping windows allows a better reuse of already computed quantities.

The initialization of β is performed as in Section II-D1, since no knowledge from the previous window can be exploited. The initialization of α and the evaluation of γ toward the window boundary may depend on the location of the window inside a burst. Three types of window locations are considered: the *first* window at the start of a burst, the *intermediate* windows in the middle of the burst, and the *last* window at the end of the burst.

1) Sliding windows and corresponding trellises: Consider the first window ($m = 1$) of $L_m < L$ bits starting at bit index $\varepsilon_0 = 0$. The trellis representing all possible successions of frames within L_m bits is similar to the beginning of the trellis in Figure 2. The decoding approach, including the initialization of α, is similar to that presented in Section II-D, except for the computation of $\gamma_{\bar{n}}\left(\bar{\ell}', \bar{\ell}\right)$, where $\bar{n}$ and $\bar{\ell}$ are the local trellis coordinates. Normal data frames, leading to $\gamma_{\bar{n}}^{\mathrm{d}}\left(\bar{\ell}', \bar{\ell}\right)$, have to be distinguished from truncated data frames toward the boundary of the window, leading to $\gamma_{\bar{n}}^{\mathrm{t}}\left(\bar{\ell}', L_m\right)$, see Appendix B.

Once $P(S_{\bar{n}}^m = \bar{\ell} | \mathbf{y}_{\varepsilon_{m-1}+1}^{\varepsilon_{m-1}+L_m})$ is evaluated, one can apply the estimators (5), (6), and (7) to determine the number of frames $\widehat{N}_m$ in the m-th window (including the last truncated frame), the beginning, and the length of each frame. Nevertheless, the evaluation of the beginning of truncated frames is not as reliable as that of complete frames, especially when the HEC has been truncated, and the localization quality of frames immediately preceding a truncated frame may also be affected. Thus, we choose to consider only the $\widehat{N}_m^c < \widehat{N}_m$ *complete* frames ending in the first $L_m - \ell_{\max} - \ell_h$ bits of the window as reliably synchronized. The unreliable region toward the boundary of the window is dashed in Figure 3.

As a consequence, L_m has to be such that $L_m \geqslant 2\ell_{\max} + \ell_h$ to ensure that at least one frame is deemed as reliably synchronized in an SlW.

Consider now the m-th window ($1 < m < M$) containing the bits from $\varepsilon_{m-1} + 1$ to $\varepsilon_{m-1} + L_m < L$, see Figure 3. The index ε_{m-1} of the last bit of the last frame deemed

reliably synchronized (*i.e.*, the $\widehat{N}_{m-1}^{c}$-th frame) in the $m-1$-th window, corresponds in the local coordinates of the $m-1$-th SlW to $\bar{\ell} = \varepsilon_{m-1} - \varepsilon_{m-2}$. The m-th trellis starts at the local coordinates $\left(\bar{n} = \widehat{N}_{m-1}^{c}, \bar{\ell} = \varepsilon_{m-1} - \varepsilon_{m-2}\right)$ of the $m-1$-th trellis. The computation of $\gamma_{\bar{n}}\left(\bar{\ell}', \bar{\ell}\right)$ for the intermediate window is identical to that of the first window. Following [21], up to $\ell_{\max}$ initial values for $\alpha_0^m\left(\bar{\ell}\right)$ are propagated from the $m-1$-th window to the m-th window, see Section II-E2. This allows a better FS in case of erroneous FS in the $m-1$-th window.

Finally, for the last window ($m = M$), only the potential presence of a padding frame has to be taken into consideration, see Figure 4. Decoding is performed as in the trellis-based approach (Section II-D), except for the initialization of $\alpha_0^M\left(\bar{\ell}\right)$, which is similar to that of the intermediate windows case.

As a consequence, the m-th and $m+1$-th windows overlap over L_m^o bits, with $\ell_h + \ell_{\max} \leqslant L_m^o < \ell_h + 2\ell_{\max}$.

2) Initialization of α in the trellis of each SlW: In the SlW-BCJR algorithm proposed in [21], the α^ms evaluated in the m-th window are deduced from those in the $m-1$-th window. Here, since the number of states S_n evolves with n, $\alpha_{\bar{n}}^m$ cannot be obtained that easily from $\alpha_{\bar{n}}^{m-1}$.

In the $m-1$-th window, one has evaluated $\alpha_{\bar{n}}^{m-1}\left(\bar{\ell}\right)$, with $0 \leqslant \bar{\ell} \leqslant L_{m-1}$ and $0 \leqslant \bar{n} \leqslant \lceil L_{m-1}/\ell_{\min} \rceil$. We choose to propagate at most $\ell_{\max}$ values of α from $\bar{n} = \widehat{N}_{m-1}^{c}$ in the $m-1$-th window to $\bar{n} = 0$ in the m-th window (for $\bar{\ell} = 0, ..., \ell_{\max} - 1$) as follows

$$\alpha_0^m(\bar{\ell}) = \kappa\, \alpha_{\widehat{N}_{m-1}^{c}}^{m-1}\left(\varepsilon_{m-1} - \varepsilon_{m-2} + \bar{\ell}\right), \tag{13}$$

where κ is some normalization factor chosen such that the $\alpha_0^m(\bar{\ell})$s sum to one. This allows the first frame of the m-th window to start at any bit index between $\varepsilon_{m-1}+1$ and $\varepsilon_{m-1}+\ell_{\max}$.

3) Complexity: To evaluate the complexity of the SlW approach, assume that M windows of approximately the same size L^w have been obtained. These windows are assumed to overlap in average on $L^o = \ell_h + 1.5\ell_{\max}$ bits. For sufficiently large L, one has thus to process $M \approx \frac{L}{L^w - L^o}$ overlapping windows, each one with $\mathcal{N}_{\mathrm{n}}^w \approx \frac{(L^w)^2}{2}\left(\frac{\ell_{\max}-\ell_{\min}}{\ell_{\max}\ell_{\min}}\right)$ nodes. The total number of nodes to process is then

$$M\mathcal{N}_{\mathrm{n}}^w \approx \frac{L}{L^w - L^o}\frac{(L^w)^2}{2}\left(\frac{\ell_{\max} - \ell_{\min}}{\ell_{\max}\ell_{\min}}\right). \tag{14}$$

This decoding complexity is smaller than that of Section II-D3. Table I provides a comparison for different values of L for window size $L^w = 480 + L^o$, showing that the complexity gain increases with L. Choosing small L^w for the window lengths L_m reduces the latency as well as the complexity at the cost of some loss in decoding performance. The limitation is that L^w cannot be chosen smaller than $\ell_h + 2\ell_{\max}$, see Section II-E1.

III. Three-State FS Automaton

With tight delay constraints, a classical method is the 3S automaton proposed in [10] for the synchronization of fixed-length ATM cells and adapted to variable-length frames by [5].

The automaton presented in [5] consists of three states: *SYNCH*, *HUNT*, and *PRESYNCH*. The automaton remains in the *SYNCH* state as long as no FS error is detected using HEC. If HEC detects an FS error, one first tries to correct errors in the header. In case of failure, the automaton switches to *HUNT*, where the automaton hunts for FS by searching *bit-by-bit* for an HEC consistent with the corresponding header fields. Once an agreement is found, the automaton switches to *PRESYNCH*, where a *frame-by-frame* verification is performed to ensure that the FS performed in the *HUNT* state is correct. This is done by checking the correctness of the HEC for $\delta > 0$ consecutive frames. Once δ consecutive correct HECs have been obtained, the automaton returns to *SYNCH* or switches again to *HUNT* in case of error.

We propose several improvements to Ueda's 3S automaton, see Figure 5. First, instead of performing hard CRC error correction in the *SYNCH* state, the robust header recovery technique presented in [22], and briefly recalled in Section III-A, for correcting corrupted headers (exploiting intra and interlayer redundancies), is employed to estimate the length field of the frame. In case of inconsistent HEC in some header with estimated length field (HEC verification is performed after replacing the received noisy length field with the estimated length field), the automaton switches to *HUNT*, where Bayesian hypothesis testing is performed to search for correct FS, see Section III-B. Operations performed in the *PRESYNCH* state remain unchanged.

Note that alternatively, an automaton with a single *HUNT* State (*HUNT* State Alone, HSA) may be considered to perform FS, without using the frame length field present in the header.

A. SYNCH State: Header Recovery

In [22], a MAP estimator is proposed to determine some fields in the frame header. In case of FS with frames of variable lengths, one is mainly interested in the length field denoted by $\mathbf{u}_n$ for the n-th frame. The MAP estimate $\hat{\mathbf{u}}_n$ of $\mathbf{u}_n$ is given by

$$\hat{\mathbf{u}}_n = \arg\max_{\mathbf{u}} P\left(\mathbf{u} \,|\, \mathbf{k}, \mathbf{y}_k, \mathbf{y}_u, \mathbf{y}_o, \mathbf{y}_c\right). \tag{15}$$

Using the notations and hypothesis of Section II-D2, (15) simplifies to

$$\hat{\mathbf{u}}_n = \arg\max_{\mathbf{u}\in\Omega_u} P\left(\mathbf{y}_u|\mathbf{u}\right) \sum_{\mathbf{o}} P\left(\mathbf{y}_o|\mathbf{o}\right) P\left(\mathbf{o}\right) P\left(\mathbf{y}_c|\mathbf{c}=\mathbf{f}\left(\mathbf{k},\mathbf{u},\mathbf{o}\right)\right), \tag{16}$$

where $\Omega_u = \{\ell_{\min}, ..., \ell_{\max}\}$ is the set of lengths which may be taken by the length field. The evaluation of (16) may be done with a reduced-complexity algorithm, see [22] for more details.

B. HUNT State: Bayesian hypothesis test

In [6], [13], [14], several hypothesis tests based on a Neyman-Pearson (NP) criterion are introduced to determine whether a frame starts at a given bit index. Only the presence of a SW is assumed. This technique suffers from limitations when the SW is short.

This section is devoted to the construction of Bayesian hypothesis tests exploiting all sources of redundancy present in the header along with the soft information provided by the channel. This allows to build more efficient tests, especially when the SW is short.

Consider bit index ℓ of a burst. Under the hypothesis H_h that a frame header $\mathbf{h}_n = [\mathbf{k}, \mathbf{u}_n, \mathbf{o}_n, \mathbf{c}]$ starts at ℓ, one may interpret the corresponding channel output as $\mathbf{y} = [\mathbf{y}_k, \mathbf{y}_u, \mathbf{y}_o, \mathbf{y}_c]$ and

$$P(\mathbf{y}|H_h) \;=\; \sum_{\mathbf{h}} P\left(\mathbf{y}|\mathbf{h}, H_h\right) P\left(\mathbf{h}|H_h\right). \tag{17}$$

With the hypothesis of Section II-D2, only headers starting with $\mathbf{k}$ have to be considered, thus

$$P(\mathbf{y}|H_h) = P\left(\mathbf{y}_k|\mathbf{k}\right) \sum_{\mathbf{u}\in\Omega_u} \left(P\left(\mathbf{y}_u|\mathbf{u}\right) P\left(\mathbf{u}\right) \sum_{\mathbf{o}} P\left(\mathbf{y}_o|\mathbf{o}\right) P\left(\mathbf{o}\right) P\left(\mathbf{y}_c|\mathbf{c}=\mathbf{f}\left(\mathbf{k},\mathbf{u},\mathbf{o}\right)\right)\right). \tag{18}$$

Under the hypothesis H_d that ℓ does not correspond to the beginning of a frame, $\mathbf{y}$ is the channel output when data bits $\mathbf{d}$ are transmitted. Assuming balanced data symbols, one gets

$$P(\mathbf{y}|H_d) = \sum_{\mathbf{d}} P\left(\mathbf{y}|\mathbf{d}, H_d\right) P\left(\mathbf{d}|H_d\right) \;=\; \sum_{\mathbf{d}} P\left(\mathbf{y}|\mathbf{d}, H_d\right) 2^{-\ell(\mathbf{d})}. \tag{19}$$

A Bayesian hypothesis test can now be defined as

$$\Lambda(\mathbf{y}) = \frac{P(\mathbf{y}|H_d)}{P(\mathbf{y}|H_h)} \underset{D_d}{\overset{D_h}{\lessgtr}} \frac{P_a(\ell, H_h)}{P_a(\ell, H_d)}, \tag{20}$$

where D_h or D_d correspond to deciding H_h or H_d respectively. $P_a(\ell, H_h)$ and $P_a(\ell, H_d)$ are the *a priori* probabilities of the hypothesis at the bit index ℓ .

When $L - \ell < \ell_{\max}$, ℓ may also represent the start of a padding frame. Thus, an additional hypothesis H_p, corresponding to the presence of a padding frame, has to be considered. The Bayesian hypothesis test for deciding between H_p and H_d is given by

$$\Lambda(\mathbf{y}) = \frac{P(\mathbf{y}|H_d)}{P(\mathbf{y}|H_p)} \underset{D_d}{\overset{D_p}{\lessgtr}} \frac{P_a(\ell, H_p)}{P_a(\ell, H_d)}. \tag{21}$$

Under H_p, $P(\mathbf{y}|H_p) = P(\mathbf{y}|\mathbf{1})$, where $\mathbf{1}$ is a vector of $\ell(\mathbf{y})$ ones. $P(\mathbf{y}|H_d)$ is given by (19).

The global procedure is as follows: (20) is first used to choose between header and data till one reaches the bit index $\ell = L - \ell_{\min}$. If data has been decided (*i.e.*, decision is D_d) for this bit index, then (21) is used for the bit indexes $\ell > L - \ell_{\max}$ starting from the last correct FS bit index, to see whether the data correspond to a padding packet. Finally, the best bit index $\ell > L - \ell_{\max}$ is selected to signal the start of the padding packet.

To evaluate the a priori probabilities $P_a(\ell, H_h)$ and $P_a(\ell, H_p)$, one knows that

$$P(\ell) = P_a(\ell, H_h) + P_a(\ell, H_p) = P(\ell)P(H_h|\ell) + P(\ell)P(H_p|\ell),$$

where $P(\ell)$ is the *a priori* probability that a frame (be it a data and/or a padding frame) starts at a bit index ℓ of a burst of L bits, and $P(H_h|\ell) = P(P_n = 0|S_{n-1} = \ell)$ and $P(H_p|\ell) = P(P_n = 1|S_{n-1} = \ell) = 1 - P(H_h|\ell)$ are the conditional *a priori* probabilities of H_h and H_p.

To determine $P(\ell)$ consider again the trellis in Figure 2. One may write

$$P(\ell) = \sum_{1 \leqslant n \leqslant \lceil \ell/\ell_{\min} \rceil} P(S_n = \ell), \tag{22}$$

where $P(S_n = \ell)$ is calculated using (10). The *a priori* probability $P_a(\ell, H_d) = \overline{P}(\ell)$ of an absence of the start of frame at bit index ℓ is $P_a(\ell, H_d) = \overline{P}(\ell) = 1 - P(\ell)$.

C. Complexity

The complexity increase of the proposed techniques is mainly due to (16) and (18). Both require the evaluation of some probability for all $\mathbf{u} \in \Omega_u$ and marginalizing over $\mathbf{o}$. A reduced-complexity algorithm introduced in [22] may be used to evaluate these quantities. Moreover, (16) and (18) are only employed in the SYNCH state in presence of errors and in the HUNT state. When the channel is relatively clear, (16) and (18) are then not too frequently evaluated, and the complexity remains of the same order of magnitude as that of the techniques presented in [5] or in [6], [13], [14]. This is no more true when the channel is very noisy.

IV. Simulation results

In the WiMAX standard [12], down-link (DL) sub-frames begin with a frame control section that contains the DL map (DL-MAP) for the current DL frame. The DL sub-frames are divided into bursts. Each burst is filled with several fixed-size or variable-size MAC frames until there is not enough space left. Padding bytes (0xFF) are then added at the end of the burst. Each MAC frame begins with a fixed-length header ($\ell_h = 48$ bits), followed by a variable-length payload and ends with an optional CRC, see Figure 6. When considering only the DL case, the connection is already established. The MAC frames belonging to a burst contain then only Convergence Sublayer (CS) data, so only the *generic* MAC header [12], as shown in Figure 7, is possible inside a burst.

Some assumptions are made in what follows for the sake of simplicity. CRC, ARQ, packing, fragmentation, and encryption are not used for the MAC frames. Some fields are already fully determined in the MAC header and, with the considered assumptions, fields such as Header Type (HT), Encryption Control (EC), sub-headers and special payload types (Type), Reserved (Rsv), CRC Indicator (CI), and Encryption Key Sequence (EKS) remain constant. The LEN field, representing the length in bytes of the MAC frame, and the Connection IDentifier (CID) have variable contents. The Header Check Sequence (HCS), an 8-bit CRC, is used to detect errors in the header and is a function of the content of all header fields.

The considered simulator consists of a burst generator, a BPSK modulator, a channel, and a receiver. Simulations are carried over a Rayleigh fading channel with known channel state

information. The modulated signal is subject to zero mean and unit variance fast (bit) Rayleigh fading plus zero-mean AWGN noise. For performance analysis, the Erroneous Frame Location Rate (EFLR) is evaluated as a function of the channel Signal-to-Noise Ratio (SNR). A frame is deemed correctly recovered when its start *and* end are both correctly determined.

In our simulations, the burst size L is assumed to be received without any error as it is transmitted in DL-MAP, which is protected with a more robust modulation and coding scheme. Data frames are randomly generated with a length uniformly distributed between $\ell_{\min} = 50$ bytes and $\ell_{\max} = 200$ bytes and are concatenated in burst of 1800 bytes. Each burst is then BPSK modulated and sent over the channel. Since WiMAX MAC frames are byte-aligned, *i.e.*, the LEN field of the frame is in bytes and all MAC frames contain an integer number of bytes, α, β, and γ are evaluated for ℓs corresponding to the beginning of bytes.

First, the proposed hold-and-sync FS techniques are compared with the HD-based FS technique. Simulation results for the trellis-based FS technique (Section II-D), the low-complexity SlW-based approach (Section II-E), and the HD-based FS are shown in Figure 8. For the SlW-based approach, the burst is divided into three windows, with $L_1 = 600$ bytes, $L_2 = 600 + L_1^o$ bytes, and $L_3 = 600 + L_2^o$ bytes. A significant gain in SNR is achieved with the proposed trellis-based FS techniques compared to the HD-based FS using the CI (8) for β. It is even larger with the PI (9). Compared to the trellis-based FS (CI), the SlW-based approach shows a performance degradation of about 1 dB, but reduces delay and complexity. On average, the overlap is about 277 bytes and a decrease in complexity by a factor of 1.7 is observed.

Second, the proposed on-the-fly 3S FS method is compared with on-the-fly methods such as HD-based method and Ueda's method [5], serving as references. Since the MAC header uses an HEC of only 8 bits, a modified version of Ueda's method, denoted by MU[1], is applied. Figure 9

[1]**Modified Ueda's (MU) method:** Ueda's Method [5] is well-suited to long HECs, like CRC-16, because in this case there are no more than two candidates for two-bit error syndromes. For short HECs, like CRC-8, many more candidates can be found, thus Ueda's method needs some modification. We propose to search for the best candidate by shifting the bit stream by the potential frame length and then calculating HEC over the next header sequence at the position indicated by the corresponding candidate. If no candidate is correct (*i.e.*, corresponds to correct HEC), then the candidate that gives one-bit error syndrome is selected as the best candidate. In case of failure, the FS automaton switches to *HUNT*.

shows that the proposed 3S FS method performs better than MU method thanks to the use of soft hunt operations and to robust header recovery. The MU method performs poorly, especially at low SNR, due its use of hard HEC detection/correction and to the small HCS (8 bits) compared to the size of the header (48 bits), leading to more than 10 candidates for two bit error syndrome to consider. The 3S FS automaton performs better at low SNR due to the effectiveness of Bayesian hypothesis testing in the *HUNT* state, which can retrieve FS quickly. The difference between the 3S FS and MU decreases at high SNR, since even though header recovery performs well, erroneous FS by Bayesian hypothesis testing degrades performance because even in good channel conditions one can wrongly assume correct FS due to the simulation of header by random data. The proposed 3S FS technique clearly provides improved FS compared to the state-of-the-art algorithms, thus allowing a compromise between performance and latency. The HSA FS method gives some FS error floor at high SNR, as expected, due to unavoidable and persistent false alarms, as the payload data can simulate the header.

V. Conclusion

We have proposed JPCD techniques for robust FS of aggregated frames. They exploit the redundancy present in the frame headers and channel soft information. A trellis-based technique processing the whole burst and all successions of frames within a burst has first been proposed. Then, a low-delay, reduced-complexity SlW variant has been proposed, allowing some performance-complexity trade-off. Finally, a less-efficient but on-the-fly FS using an improved 3S automaton has been proposed.

The proposed techniques were evaluated to perform FS at MAC layer of the WiMAX standard. The trellis-based technique provides the best performance, at the cost of a delay equivalent to the length of the burst. The SlW approach significantly reduces FS delay and computational complexity. The price to be paid is a slight performance degradation. FS using the proposed 3S automaton works on the fly, but is less efficient. Simulations clearly demonstrate the performance improvement compared to other on the fly FS methods. All proposed algorithms reduce significantly the amount of frames that need to be retransmitted compared to traditional techniques.

APPENDIX

A. Evaluation of γ_n, trellis associated to the whole burst

When $\ell < L$, the transition corresponding to the n-th frame represents a data frame. Assuming that $\ell_{\min} \leqslant \ell - \ell' \leqslant \ell_{\max}$, the bits between $\ell'+1$ and ℓ may be interpreted as $\mathbf{x}_{\ell'+1}^{\ell} = [\mathbf{k}, \mathbf{u}_n, \mathbf{o}_n, \mathbf{c}, \mathbf{p}]$, where $\mathbf{u}_n = \mathbf{u}(\ell - \ell')$ is the binary representation of $\ell - \ell'$. The corresponding observation is $\mathbf{y}_{\ell'+1}^{\ell} = [\mathbf{y}_k, \mathbf{y}_u, \mathbf{y}_o, \mathbf{y}_c, \mathbf{y}_p]$. With these notations, for $\ell \neq L$, the transition metric $\gamma_n(\ell', \ell) = \gamma_n^{\mathrm{d}}(\ell', \ell)$ accounting only for data frames is

$$\begin{aligned}\gamma_n^{\mathrm{d}}(\ell', \ell) = P(S_n = \ell, \mathbf{y}_{\ell'+1}^{\ell} | S_{n-1} = \ell') &= \sum_{\mathbf{x}_{\ell'+1}^{\ell} \neq \mathbf{1}} P\left(S_n = \ell, \mathbf{y}_{\ell'+1}^{\ell}, \mathbf{x}_{\ell'+1}^{\ell} | S_{n-1} = \ell'\right) \\ &= p(S_n = \ell | S_{n-1} = \ell') \, \varphi\left(\mathbf{y}_{\ell'+1}^{\ell}, \mathbf{x}_{\ell'+1}^{\ell}\right), \end{aligned} \tag{23}$$

where

$$\varphi\left(\mathbf{y}_{\ell'+1}^{\ell}, \mathbf{x}_{\ell'+1}^{\ell}\right) = \sum_{\mathbf{x}_{\ell'+1}^{\ell} \neq \mathbf{1}} P\left(\mathbf{y}_{\ell'+1}^{\ell} | \mathbf{x}_{\ell'+1}^{\ell}, S_{n-1} = \ell', S_n = \ell\right) P\left(\mathbf{x}_{\ell'+1}^{\ell} | S_{n-1} = \ell', S_n = \ell\right). \tag{24}$$

The sum in (24) is over all possible $\mathbf{x}_{\ell'+1}^{\ell}$, except $\mathbf{x}_{\ell'+1}^{\ell} = \mathbf{1}$, which corresponds to the content of a padding frame. However, only the $\mathbf{x}_{\ell'+1}^{\ell}$s starting with $\mathbf{k}$ and $\mathbf{u}_n = \mathbf{u}(\ell - \ell')$ have to be considered since these fields are fully determined. Moreover, assuming that the channel is memoryless, taking into account the fact that $\mathbf{k}, \mathbf{u}_n, \mathbf{o}_n$, and $\mathbf{c}$ do not depend on $\mathbf{p}$, and the fact that the HEC $\mathbf{c}$ is fully determined by $\mathbf{k}$, $\mathbf{u}_n$, and $\mathbf{o}_n$, (24) simplifies to

$$\begin{aligned}\varphi\left(\mathbf{y}_{\ell'+1}^{\ell}, \mathbf{x}_{\ell'+1}^{\ell}\right) &= P(\mathbf{y}_k | \mathbf{k}) \, P(\mathbf{y}_u | \mathbf{u}(\ell - \ell')) \sum_{\mathbf{p}} P(\mathbf{y}_p | \mathbf{p}) \, P(\mathbf{p}) \\ &\quad \sum_{\mathbf{o}} P(\mathbf{y}_o | \mathbf{o}) \, P(\mathbf{y}_c | \mathbf{c} = \mathbf{f}(\mathbf{k}, \mathbf{u}(\ell - \ell'), \mathbf{o})) \, P(\mathbf{o}). \end{aligned} \tag{25}$$

Under the assumptions above, and for a memoryless AWGN channel with variance σ^2, one gets

$$P(\mathbf{y}_u | \mathbf{u}(\ell - \ell')) = P(\mathbf{y}_u | \mathbf{u}) = \prod_{i=1}^{\ell(\mathbf{u})} \frac{1}{\sqrt{2\pi}\sigma} e^{-(\mathbf{y}_u(i) - \mathbf{u}(i))^2 / 2\sigma^2}.$$

Assuming that the values taken by $\mathbf{p}$ are all equally likely, one gets $P(\mathbf{p}) = 2^{-\ell(\mathbf{p})}$.

In (25), the sum over all possible $\mathbf{o}$ may be quite complex to evaluate for long $\mathbf{o}$. It may be calculated using a trellis construction consisting in iteratively grouping the combinations of $\mathbf{o}$

leading to the same HEC, as proposed in [22], with a complexity of $\mathcal{O}(\ell(\mathbf{o})\, 2^{\ell(\mathbf{c})})$. A reduced-complexity algorithm for evaluating this sum can also be found in [22]. The calculation of $\gamma_n(\ell', \ell)$ for each transition has a similar complexity.

When $\ell = L$ and $L - \ell' < \ell_{\max}$, the n-th frame is the last one, and $\mathbf{x}_{\ell'+1}^{L} = \mathbf{1}$ has also to be considered in $\gamma_n(\ell', L)$, leading to

$$\gamma_n(\ell', L) = \gamma_n^{\mathrm{d}}(\ell', L)\, P(P_n = 0 | S_{n-1} = \ell') + \gamma_n^{\mathrm{p}}(\ell', L)\, P(P_n = 1 | S_{n-1} = \ell'), \tag{26}$$

where P_n is a random variable indicating whether the n-th frame is a padding frame with

$$P(P_n = 1 | S_{n-1} = \ell') = \begin{cases} 0, \text{ if } L - \ell' \geq \ell_{\max} \\ 1, \text{ if } 0 < L - \ell' < \ell_{\min} \\ \sum_{\lambda = L-\ell'+1}^{\ell_{\max}} \pi_\lambda, \text{ else.} \end{cases} \tag{27}$$

and $P(P_n = 0 | S_{n-1} = \ell') = 1 - P(P_n = 1 | S_{n-1} = \ell')$. In (26),

$$\begin{aligned} \gamma_n^{\mathrm{p}}(\ell', \ell) &= p\left(S_n = L, \mathbf{y}_{\ell'+1}^{L} | S_{n-1} = \ell', P_n = 1\right) \\ &= p(S_n = L | S_{n-1} = \ell', P_n = 1)\, P\left(\mathbf{y}_{\ell'+1}^{\ell} | P_n = 1, S_{n-1} = \ell', S_n = \ell\right) \end{aligned} \tag{28}$$

accounts for the padding frame, with $p(S_n = L | S_{n-1} = \ell', P_n = 1) = 1$. While,

$$\begin{aligned} \gamma_n^{\mathrm{d}}(\ell', L) &= p\left(S_n = L, \mathbf{y}_{\ell'+1}^{\ell} | S_{n-1} = \ell', P_n = 0\right) \\ &= p(S_n = L | S_{n-1} = \ell', P_n = 0)\, \varphi\left(\mathbf{y}_{\ell'+1}^{\ell}, \mathbf{x}_{\ell'+1}^{\ell}\right) \end{aligned}$$

accounts for the data frame, with $p(S_n = L | S_{n-1} = \ell', P_n = 0) = \pi_{L-\ell'} / \sum_{\lambda = \ell_{\min}}^{L-\ell'} \pi_\lambda$.

B. Evaluation of $\gamma_{\bar{n}}$, trellis associated to a SlW

When $m < M$, transitions corresponding to truncated frames have to be considered at the end of the window. When the size of the truncated frame is larger than ℓ_h, the header is entirely contained in the truncated frame. In this case $\gamma_{\bar{n}}\left(\bar{\ell}', L_m\right) = \gamma_{\bar{n}}^{\mathrm{t}}\left(\bar{\ell}', L_m\right)$, with

$$\gamma_{\bar{n}}^{\mathrm{t}}\left(\bar{\ell}', L_m\right) = p\left(S_{\bar{n}}^{m} = L_m | S_{\bar{n}-1}^{m} = \bar{\ell}'\right) \varphi^{\mathrm{t}}\left(\mathbf{y}_{\bar{\ell}'+1}^{L_m}, \mathbf{x}_{\bar{\ell}'+1}^{L_m}\right). \tag{29}$$

In (29), since truncated frames have to be considered, $p\left(S_{\bar{n}}^{m}=L_m|S_{\bar{n}-1}^{m}=\bar{\ell}'\right)$ is given by (3). Moreover, the length of the frame, *i.e.*, the content of the length field $\mathbf{u}_n$, is now only known to be between $\max(L_m-\bar{\ell}',\ell_{\min})$ and $\ell_{\max}$ bits. Thus

$$\varphi^{\mathrm{t}}\left(\mathbf{y}_{\bar{\ell}'+1}^{L_m},\mathbf{x}_{\bar{\ell}'+1}^{L_m}\right)=P\left(\mathbf{y}_k|\mathbf{k}\right)\sum_{\mathbf{p}}P\left(\mathbf{y}_p|\mathbf{p}\right)P\left(\mathbf{p}\right)$$
$$\sum_{\ell=\max(L_m-\bar{\ell}',\ell_{\min})}^{\ell=\ell_{\max}}P\left(\mathbf{u}\left(\ell\right)\right)\sum_{\mathbf{o}}\left(P\left(\mathbf{y}_u|\mathbf{u}\left(\ell\right)\right)P\left(\mathbf{y}_o|\mathbf{o}\right)P\left(\mathbf{y}_c|\mathbf{c}=\mathbf{f}\left(\mathbf{k},\mathbf{u}\left(\ell\right),\mathbf{o}\right)\right)P\left(\mathbf{o}\right)\right). \quad (30)$$

When the size of the truncated frame is strictly less than ℓ_h, for the sake of simplicity, all bits of the truncated header are assumed equally likely. In such case $\gamma_{\bar{n}}\left(\bar{\ell}',L_m\right)=\gamma_{\bar{n}}^{\mathrm{e}}\left(\bar{\ell}',L_m\right)$, with

$$\gamma_{\bar{n}}^{\mathrm{e}}\left(\bar{\ell}',L_m\right)=p\left(S_{\bar{n}}^{m}=L_m|S_{\bar{n}-1}^{m}=\bar{\ell}'\right)\sum_{\mathbf{x}_{\bar{\ell}'+1}^{L_m}}P\left(\mathbf{y}_{\bar{\ell}'+1}^{L_m}|\mathbf{x}_{\bar{\ell}'+1}^{L_m}\right)P\left(\mathbf{x}_{\bar{\ell}'+1}^{L_m}\right), \quad (31)$$

where $p\left(S_{\bar{n}}^{m}=L_m|S_{\bar{n}-1}^{m}=\bar{\ell}'\right)$ is still given by (3) and $P(\mathbf{x}_{\bar{\ell}'+1}^{L_m})=2^{-\ell(\mathbf{x}_{\bar{\ell}'+1}^{L_m})}$, since all beginning of headers are assumed equally likely.

For $m=M$, the evaluation of $\gamma_{\bar{n}}^{M}$ is as in Section II-D2.

REFERENCES

[1] R. H. Barker, *Group synchronization of binary digital systems in Communication Theory.* Butterworth, London, 1953.

[2] J. L. Massey, "Optimum frame synchronization," *IEEE Trans. on Comm.*, vol. 20, no. 4, pp. 115–119, 1972.

[3] R. A. Scholtz, "Frame synchronization techniques," *IEEE Trans. on Comm.*, vol. 28, no. 8, pp. 1204 – 1213, 1980.

[4] G. L. Lui and H. H. Tan, "Frame synchronization for Gaussian channels," *IEEE Trans. on Comm.*, vol. 35, no. 8, pp. 818–829, 1987.

[5] U. Ueda, H. Yamaguchi and R. Watanabe, "Reducing misframe frequency for HEC-based variable length frame suitable for IP services," in *Proc. IEEE ICC 2001*, pp. 1196 – 1200, 2001.

[6] M. Chiani and M. G. Martini, "On sequential frame synchronization in AWGN channels," *IEEE Trans. Comm.*, vol. 54, no. 2, pp. 339 – 348, 2006.

[7] ETSI, "Digital video broadcasting (DVB); framing structure, channel coding and modulation for digital terrestrial television," tech. rep., ETSI EN 300 744 v1.5.1, jun. 2004.

[8] Z. Y. Choi and Y. H. Lee, "Frame synchronization in the presence of frequency offset," *IEEE Trans. Comm.*, vol. 50, no. 7, pp. 1062–1065, 2002.

[9] D.-U. Lee, P. Kim, and W. Sung, "Robust frame synchronization for low signal-to-noise ratio channels using energy-corrected differential correlation," *EURASIP J. Wirel. Commun. Netw.*, vol. 2009, pp. 1–8.

[10] "B - ISDN user-network interface - Physical layer specification: General characteristics," 1999.

[11] "IEEE802.11n part 11: Wireless LAN Medium Access Control (MAC) and physical layer (PHY) specifications: Enhancements for higher throughput," March 2006.

[12] ANSI/IEEE, "802.16: IEEE standard for local and metropolitan area networks, air interface for fixed broadband wireless access systems," tech. rep., 2004.

[13] M. Chiani and M. Martini, "Practical frame synchronization for data with unknown distribution on AWGN channels," *IEEE Communication Letter*, vol. 9, no. 5, pp. 456 – 458, 2005.

[14] M. G. Martini and M. Chiani, "Optimum metric for frame synchronization with Gaussian noise and unequally distributed data symbols," in *Proc. IEEE SPAWC*, (Perugia, Italy), 21-24 June 2009.

[15] M. G. Martini and C. Hewage, "Cross-layer frame synchronization for H.264 video over WiMAX," in *Proc. IEEE SPAWC*, (Marrakech, Morocco), June 2010.

[16] E. Bastaki, H. Tan, Y. Shi, and K. Letaief, "Frame synchronization based on multiple frame observations," *IEEE Trans. Wirel. Comm.*, vol. 9, no. 3, pp. 1097–1107, 2010.

[17] U. Ali, M. Kieffer, and P. Duhamel, "Joint protocol-channel decoding for robust aggregated packet recovery at WiMAX MAC layer.," in *Proc. IEEE SPAWC*, (Perugia, Italy), pp. 672 – 676, 21-24 June 2009.

[18] G. Panza, E. Balatti, G. Vavassori, C. Lamy-Bergot, and F. Sidoti, "Supporting network transparency in 4G networks," in *Proc. IST Mobile and Wireless Communication Summit*, 2005.

[19] P. Duhamel and M. Kieffer, *Joint source-channel decoding: A cross-layer perspective with applications in video broadcasting*. Academic Press, 2009.

[20] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Info. Theory*, vol. 20, no. 2, pp. 284–287, 1974.

[21] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Algorithm for continuous decoding of turbo codes," *Electronics Letters*, vol. 32, no. 4, pp. 314 – 315, 1996.

[22] C. Marin, Y. Leprovost, M. Kieffer, and P. Duhamel, "Robust mac-lite and soft header recovery for packetized multimedia transmission," *IEEE Trans. on Communications*, vol. 58, no. 3, pp. 775–784, 2010.

[23] R. Bauer and J. Hagenauer, "Symbol-by-symbol MAP decoding of variable length codes," in *Proc. 3rd ITG Conference Source and Channel Coding*, (München), pp. 111–116, 2000.

[24] J. Gwak, S. K. Shin, and H. M. Kim, "Reduced complexity sliding window BCJR decoding algorithms for turbo codes," in *IMA - Crypto & Coding'99* (M. Walker, ed.), pp. 179–184, Springer-Verlag, 1999.

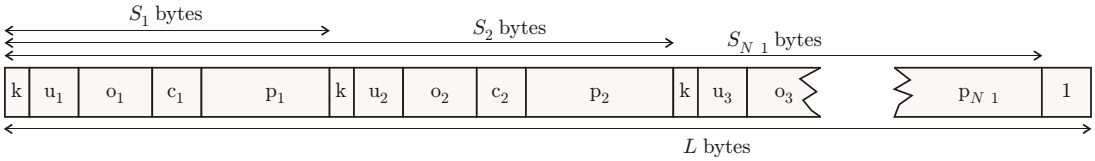

Figure 1. Aggregated frames in a WiMAX burst

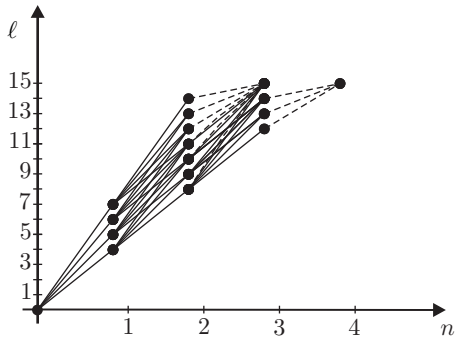

Figure 2. Trellis depicting the allowed total length in bits L vs. the number of frames n in a *burst* of $L = 15$ bits with $\ell_{\min} = 4$ bits and $\ell_{\max} = 7$ bits. Dashed lines correspond to padding bits.

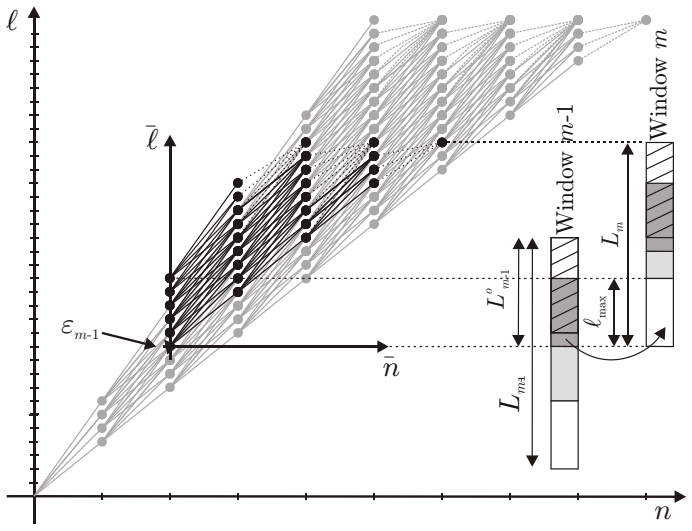

Figure 3. m-th decoding window and corresponding trellis, the original trellis is in gray

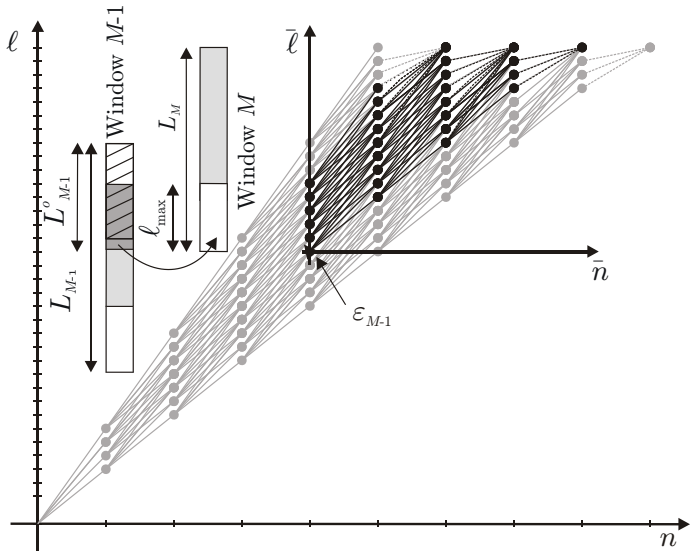

Figure 4. Trellis for the last decoding SlW, the original trellis is in gray

L (bytes)	1800	8000	16000	24000
Trellis-based (# of Nodes)	24300	480000	1920000	4320000
SlW-based (# of Nodes)	17400	77200	154450	231700
Complexity Gain	1.4	6.2	12.5	18.6

Table I

FS COMPLEXITY COMPARISON FOR $L^w = 480 + L^o$

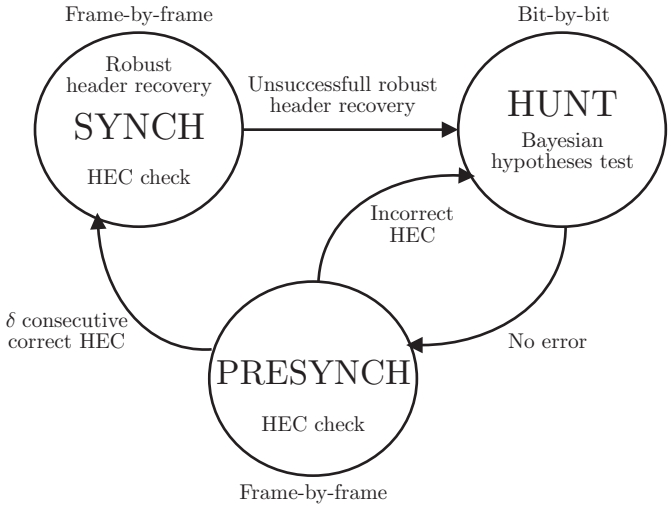

Figure 5. 3S FS automaton

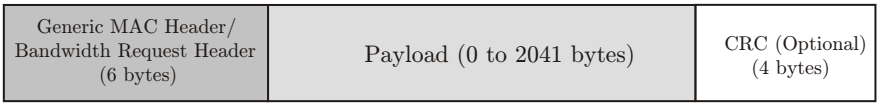

Figure 6. Typical WiMAX MAC Frame

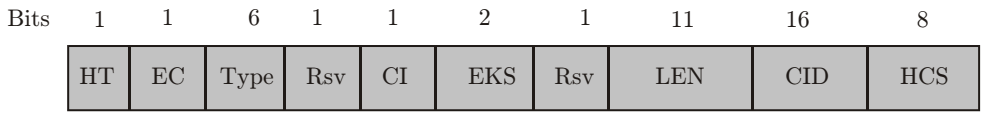

Figure 7. Generic MAC header as specified in IEEE 802.16-2004

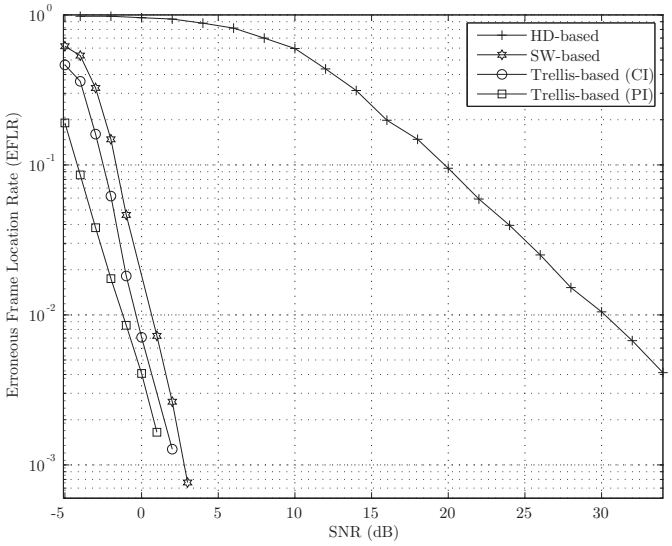

Figure 8. Hold-and-sync FS methods for bursts transmitted over Rayleigh channel

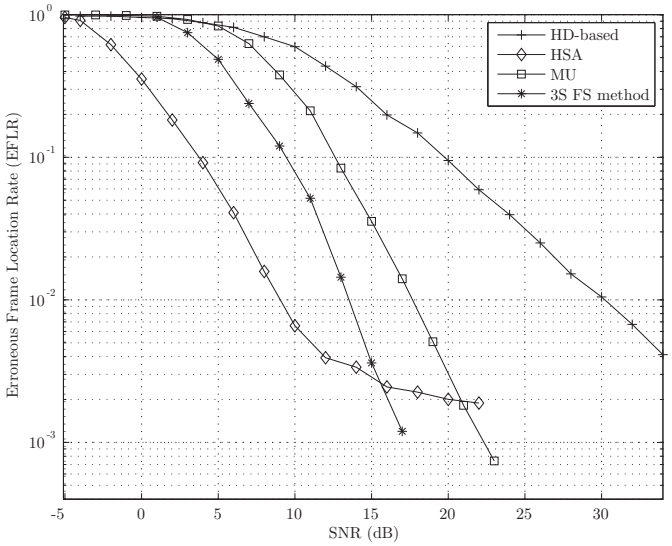

Figure 9. On-the-fly FS methods for bursts transmitted over Rayleigh channel