Hard- and Soft-Output Trellis-based Conflict Resolution for Bidirectional Decision Feedback Equalization

Chan Wong Wong, *Student Member, IEEE,* John M. Shea, *Member, IEEE,* and Yumin Lee, *Senior Member, IEEE*

Abstract—**The decision feedback equalizer (DFE) is widely used in practice for intersymbol interference (ISI) mitigation because of its low complexity. The performance of the DFE can be enhanced by arbitrating between the output of a standard DFE and a second DFE that operates on the time-reversed received sequence. The combination is called a bidirectional DFE (BDFE). Several previous papers have discussed how to arbitrate between the outputs of the forward and reverse equalizers to make hard decisions. We show that the arbitration algorithms proposed by other researchers are inefficient and propose a trellis-based conflict resolution (TBCR) algorithm that simultaneously provides better performance and lower complexity. We further show how the TBCR approach can be used to create soft-output BDFE algorithms for use with soft-input decoding of an error-control code. The soft-output BDFE algorithms offer better performance than previously proposed algorithms while requiring lower complexity than the maximum *a posteriori* equalizer. We also propose a novel method to reduce the computational complexity of BDFE algorithms by simultaneously computing the coefficients of the forward and reverse equalizers.**

I. Introduction

Channel equalizers to mitigate ISI are often chosen based on a tradeoff between performance and complexity. Maximum-likelihood sequence estimation (MLSE) [1] minimizes the probability of error but has complexity that grows exponentially in the channel memory. The decision feedback equalizer (DFE) is widely used because of its simple implementation. However, the use of previously detected symbols makes the DFE suffer from error propagation, which causes errors to occur in bursts. This phenomenon is more severe when the tap weights and the number of feedback taps are large and can significantly hinder the performance of the DFE [2]. Thus, many techniques [3–7] have been proposed to mitigate the effect of error propagation.

In [8] and [9], time-reversal of the received sequence followed by a DFE is proposed as a technique to give two alternative equalized versions of a sequence of received symbols. An equalizer that uses a normal mode (forward) DFE and a time-reversal mode (reverse) DFE is called a bidirectional DFE (BDFE). The time-reversal operation is done by reversing the sequential order of the received symbols, in time, prior to equalization. Reversing the sequence of the received symbols is equivalent to reversing the transmitted symbols and the channel impulse response before channel convolution [10]. Hence, the minimum phase roots of the channel become the maximum phase roots and vice-versa. Thus, the forward and reverse DFEs will generally produce different output sequences. Since error propagation is a causal phenomenon, the error events of the forward and reverse DFEs, which are in opposite directions in time, exhibit low correlation. Moreover, even with the assumption of ideal feedback, the noise sequences at the outputs of the forward and reverse DFEs have low correlation [11]. Thus, the performance of equalization can be significantly improved by selecting between the outputs of the forward and reverse DFEs, thereby achieving time-reversal diversity.

Several different algorithms [9, 11–13] have been proposed to utilize time-reversal diversity. As shown in Fig. 1, the existing BDFE algorithms consist of three stages: equalization, reconstruction and arbitration/combination. In the equalization stage, the forward and time-reversed sequences are equalized using conventional DFEs. In the reconstruction stage, the two sets of DFE outputs are input to the estimated channel to generate two different estimates of the channel output sequence in the absence of noise (hereafter referred to as the estimated noise-free received sequences). In the arbitration/combination stage, the reconstructed sequences are compared to the received sequence to aid in selection between, or combination of, the outputs of the forward and reverse DFEs. Most of the previous work on BDFEs focuses on hard-output algorithms [9, 11, 13]. These algorithms differ only in how arbitration is performed (see Section IV-A). The only known soft-output BDFE is the linear combining bidirectional DFE (LC-BDFE) [12], which employs a weighted linear combination of the soft outputs of the forward and reverse DFEs.

In this paper, we propose hard- and soft-output BDFE algorithms that provide better performance and/or lower complexity than the previous algorithms. Fig. 2 illustrates the relationships of various BDFE algorithms. The algorithms that we develop in this paper are in the boxes with dark shadows.

Manuscript received August 9, 2008; revised January 20, 2009; accepted March 9, 2009. The associate editor coordinating the review of this paper and approving it for publication was J. Coon.

C. W. Wong and J. M. Shea are with the University of Florida, Gainesville, FL, 32608 (email: cwwong@ufl.edu, jshea@ece.ufl.edu).

Yumin Lee was with Graduate Institute of Communication Engineering, National Taiwan University, Taipei, Taiwan (email: yuminlee@ieee.org).

This paper was presented in part at the IEEE Military Comm. Conf., Oct. 2007, Orlando FL.

This work was supported by the National Science Foundation under grant number CNS-0626863 and by the Air Force Office of Scientific Research under grant number FA9550-07-10456.

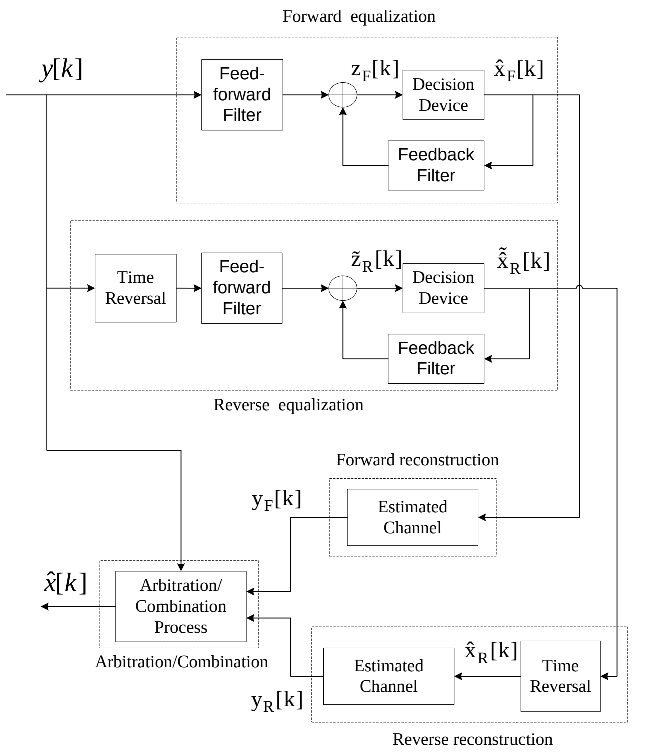

Fig. 1. General bidirectional decision feedback equalizer (BDFE) structure.

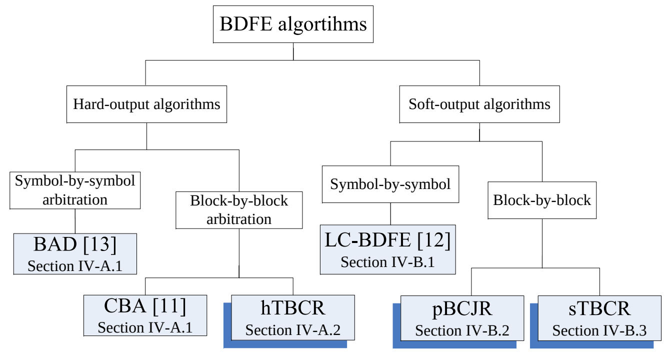

Fig. 2. Classification of bidirectional decision feedback equalizer (BDFE) algorithms.

All of the listed BDFE algorithms are discussed in Section IV. We also propose a new approach to efficiently calculate the coefficients of the reverse DFE.

The rest of this paper is organized as follows. The system model is introduced in Section II. In Section III, the conventional DFE structure is introduced and an algorithm is given to efficiently calculate the coefficients of the reverse DFE. Previous BDFE algorithms and our proposed trellis-based BDFE algorithms are discussed in Section IV. Simulation results and conclusions are given in Sections V and VI, respectively.

II. System Model

We consider the single-input single-output (SISO) communication system shown in Fig. 3, although our techniques can also be applied to some multiple-input, multiple-output

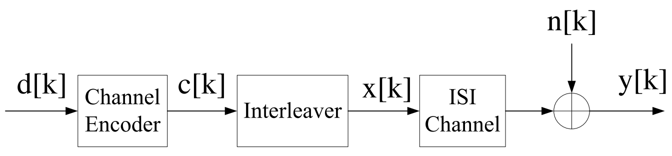

Fig. 3. System model: transmitter and channel.

(MIMO) systems by using vectors for the inputs and outputs and matrices for the channel. Without loss of generality, we use a baseband equivalent model for the analysis and discussions. The information symbol $d[k]$ is encoded with a channel encoder yielding the coded symbol $c[k]$, where k is the time index. An interleaver is used to permute $c[k]$ to provide the channel input $x[k]$. This operation is denoted as $x[k] = \Pi\left(c[k]\right)$, where $\Pi(\cdot)$ is the permutation function. We omit the modulator in this model to simplify the discussion. Unless mentioned otherwise, we assume that $x[k]$ takes values from signal alphabet $\mathcal{B} = \{+\sqrt{E_x}, -\sqrt{E_x}\}$ with equal probability. The channel is modeled as an ISI channel with $\nu+1$ taps. The channel output is corrupted by additive Gaussian white noise (AWGN) $n[k]$, whose variance is $N_0/2 = \sigma^2$ and is uncorrelated with $x[k]$, to form the received symbol $y[k]$.

In the SISO discrete time baseband equivalent model, the received signal $y[k]$ is given by

$$y[k] = \sum_{m=0}^{\nu} p[m]x[k-m] + n[k], \tag{1}$$

where $p[0]\ldots\ldots p[\nu]$ is the (ν+1)-tap discrete-time equivalent channel.

The BDFE takes $\{y[k]\}$ as input and produces soft or hard outputs for the transmitted symbols $\{x[k]\}$. As shown in Fig. 1, the first stage of the BDFE is to apply a conventional DFE to the forward and reversed received sequences. We use the minimum mean-square error decision feedback equalizer (MMSE DFE), which is discussed further in Section III. During the second stage, reconstruction, the decisions $\hat{x}_F[k]$ and $\hat{x}_R[k]$ of the forward and reverse DFEs are filtered through the estimated channel impulse response to reconstruct the estimated noise-free received sequences, $y_F[k]$ and $y_R[k]$. In the final stage, arbitration or combination is performed on $\hat{x}_F[k]$ and $\hat{x}_R[k]$ to produce the hard or soft output. When forward error-correction coding is used, the soft inputs to the channel decoder are the log-likelihood ratios (LLR) for the coded symbols $\{c[k]\}$, which depend on the particular soft-output equalizer that is used (see Section IV-B).

Before proceeding, we introduce some frequently used notations. Vectors are written in bold letters and assumed to be column vectors. Matrices are specified by bold capital letters. We use $\mathbf{y}$ to denote the whole sequence of $y[k]$ with length N. Subsequences of $\mathbf{y}$ are denoted by $\mathbf{y}_j^{\ell} = [\ y[j]\ y[j+1]\ \ldots\ y[\ell]\ ]^T$ for $j \leq \ell$. $\mathbf{0}_{i\times j}$ is the $i \times j$ zero matrix. $\mathbf{I}_{i\times j}$ is the $i \times j$ identity matrix. $\mathbf{E}(\cdot)$ denotes expectation. Time reversal of a sequence is denoted by $\widetilde{(\cdot)}$. Transpose, Hermitian transpose, and conjugate are denoted by $(\cdot)^T$, $(\cdot)^H$ and $(\cdot)^*$, respectively. We use $(\cdot)_F$ and $(\cdot)_R$ to represent the equivalent terms in the forward and reverse

DFEs.

III. Conventional DFE architecture

We consider a MMSE DFE, which is composed of a feed-forward filter (FFF), a feedback filter (FBF) and a decision device. The block diagram of the conventional MMSE DFE is the same as the "Forward equalization" part of the BDFE shown in Fig. 1. Let $\hat{x}[k]$ be the decision for $x[k]$, Δ be the decision advance, and w_i and b_i be the i-th coefficients of the FFF and FBF, which are chosen to minimize $\mathbf{E}\left[|z[k]-x[k]|^2\right]$. Here $z[k]$ is the output of the MMSE DFE, i.e. the decision statistic for $x[k]$, given by

$$z[k]=\sum_{i=0}^{N_f-1} w_i^* y[k+\Delta-i]+\sum_{i=1}^{N_b} b_i^* \hat{x}[k-i].$$

By assuming perfect decision feedback and invoking the orthogonality principle [14], the coefficients of the DFE are given by

$$\begin{aligned} \mathbf{w} &= \left(\mathbf{P}\mathbf{P}^H-\mathbf{H}\mathbf{H}^H+\frac{\mathbf{R}_N}{E_x}\right)^{-1}\mathbf{p}_{\Delta+1}, \text{and} \\ \mathbf{b} &= -\mathbf{H}^H\mathbf{w}, \end{aligned} \tag{2}$$

where $\mathbf{w}=\left[w_0\ w_1\ \ldots\ w_{N_f-1}\right]^T$ and $\mathbf{b}=\left[b_1\ b_2\ \ldots\ b_{N_b}\right]^T$ are $N_f\times 1$ and $N_b\times 1$ vectors, respectively. Here, $\mathbf{R}_N=\sigma^2\mathbf{I}_{N_f\times N_f}$ is the $N_f\times N_f$ autocorrelation matrix of $n[k]$ and $\mathbf{P}$, $\mathbf{H}$ and $\mathbf{p}_{\Delta+1}$ are $N_f\times(N_f+\nu)$, $N_f\times N_b$ and $N_f\times 1$ matrices defined by

$$\begin{aligned} \mathbf{P} &= \begin{bmatrix} p[0] & \cdots & p[\nu] & & \\ & \ddots & \cdots & \ddots & \\ & & p[0] & \cdots & p[\nu] \end{bmatrix}, \\ \mathbf{H} &= \mathbf{P}\begin{bmatrix} \mathbf{0}_{(\Delta+1)\times N_b} \\ \mathbf{I}_{N_b\times N_b} \\ \mathbf{0}_{(N_f+\nu-N_b-\Delta-1)\times N_b} \end{bmatrix}, \text{and} \\ \mathbf{p}_{\Delta+1} &= \mathbf{P}\begin{bmatrix} \mathbf{0}_{\Delta\times 1} \\ 1 \\ \mathbf{0}_{(N_f-\Delta-1)\times 1} \end{bmatrix}. \end{aligned}$$

In the BDFE, the coefficients of the forward and reverse DFEs are computed using (2). When the number of filter coefficients is large, the complexity in solving for $\mathbf{w}$ is high. Fortunately, the matrices involved have a structure that can be exploited to significantly reduce the complexity by using the concept of displacement structure [15] and the generalized Schur algorithm [16]. In a naive approach, the use of forward and reverse DFEs requires a two-fold increase in the complexity of computing the DFE coefficients in comparison to the conventional single-pass DFE. We first show how the relation between the coefficients of the two DFEs can be used to reduce the complexity of generating the coefficients of the reverse DFE.

When computing the filter coefficients for the DFE, a very critical process in complexity reduction [16] is to develop fast algorithms to invert the matrix $\mathbf{\Lambda}=\left(\mathbf{P}\mathbf{P}^H-\mathbf{H}\mathbf{H}^H+\mathbf{R}_N/E_x\right)$ from (2). We first write $\mathbf{\Lambda}=\mathbf{\Theta}-\mathbf{\Psi}$, where $\mathbf{\Theta}=\mathbf{P}\mathbf{P}^H+\mathbf{R}_N/E_x$ and $\mathbf{\Psi}=\mathbf{H}\mathbf{H}^H$. In order to use the information of the forward DFE to generate the coefficients of the reverse DFE, it is helpful to study the matrices $\mathbf{\Theta}$ and $\mathbf{\Psi}$ to determine the relation between $\mathbf{\Lambda}_F$ and $\mathbf{\Lambda}_R$. First of all, the matrix $\frac{\mathbf{R}_N}{E_x}$ in $\mathbf{\Theta}$ can be ignored without loss of generality since it is the same for the forward and reverse DFEs. It is straightforward to show that $\mathbf{\Theta}_F=(\mathbf{\Theta}_R)^*$, thus eliminating the need to compute $\mathbf{\Theta}_R$ once $\mathbf{\Theta}_F$ is known.

Now consider the matrix $\mathbf{\Psi}=\mathbf{H}\mathbf{H}^H$. Taking $\Delta=N_f-1$ and $N_b=\nu$, $\mathbf{H}$ has non-zero elements in only the last N_b columns. Thus $\mathbf{\Psi}$ is a sparse matrix with non-zero elements only inside the $N_b\times N_b$ lower right submatrix. Thus, $\mathbf{\Lambda}_F$ and $\mathbf{\Lambda}_R$ only differ significantly in the $N_b\times N_b$ lower right submatrix (the other elements are conjugates of each other). In the normal DFE architecture, N_f is usually much larger than N_b. This relation between $\mathbf{\Lambda}_F$ and $\mathbf{\Lambda}_R$ can be used to compute the inversion of the matrix $\mathbf{\Lambda}_R$ at only a fraction of the complexity of direct inversion. For example, this relation can be embedded into many fast algorithms that efficiently compute the coefficients of the DFE. In the Appendix, we demonstrate the appropriate modifications to the well-known and easy-to-understand Jacobi algorithm. This approach can also be applied to the generalized Schur algorithm [16].

IV. Proposed Trellis-based Conflict Resolution (TBCR) Algorithms

In this section, we review the existing BDFE algorithms and the algorithms that we propose. Hard-output and soft-output algorithms are considered in Sections IV-A and IV-B, respectively.

A. Hard-output BDFE Algorithms

1) Existing Hard-output BDFE Algorithms: Bidirectional arbitrated DFE (BAD) [13] is a hard-output BDFE algorithm that performs arbitration on a symbol-by-symbol basis. For each symbol for which $\hat{x}_F[k]\neq\hat{x}_R[k]$, arbitration is performed using a window of the reconstructed symbols $\mathbf{y}_F$ and $\mathbf{y}_R$ around k. Let $\mathrm{d_E}()$ denote Euclidean distance, and $d_F=\mathrm{d_E}(\mathbf{y_F}_{k-W}^{k+W},\mathbf{y}_{k-W}^{k+W})$ and $d_R=\mathrm{d_E}(\mathbf{y_R}_{k-W}^{k+W},\mathbf{y}_{k-W}^{k+W})$ denote the windowed distance metrics for the forward and reverse DFE, respectively. The output of the DFE which produces the smaller metric is selected as the final output; i.e.,

$$\hat{x}[k]=\begin{cases}\hat{x}_F[k], & d_F\leq d_R\\ \hat{x}_R[k], & d_F>d_R.\end{cases} \tag{3}$$

Several other versions of BAD, namely partial suppression DFE, partial data DFE, and better BAD are proposed in [17] to focus on generating more candidate data sequences to further improve the performance of BAD [18]. However, the performance improvement does not merit the increased complexity generated by those new algorithms [17].

Instead of using symbol-by-symbol arbitration, a block-by-block arbitration technique called BDFE with contradictory block arbitration (CBA) is proposed in [11]. This algorithm partitions the hard outputs of the two DFEs into consistent blocks and contradictory blocks. A consistent block consists of consecutively identical hard outputs between the two DFEs, while a contradictory block is made of consecutively different

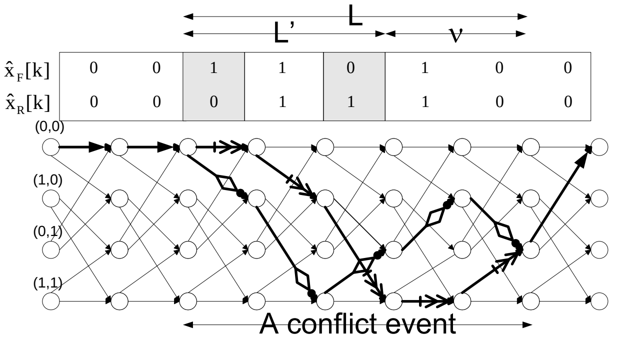
L
L'
ν
$\hat{x}_F[k]$ 0 0 1 1 0 1 0 0
$\hat{x}_R[k]$ 0 0 0 1 1 1 0 0
(0,0)
(1,0)
(0,1)
(1,1)
A conflict event

Fig. 4. An example that illustrates how the trellis can be used to interpret differing sequences from the forward and reverse DFEs.

hard outputs between the two DFEs. Then CBA performs arbitration on a block-by-block basis over the contradictory blocks. For each contradictory block between indices m_1 and m_2, the windowed distance metrics for the forward and reverse DFEs in CBA are $d_F = \mathrm{d_E}(\mathbf{y}_F{}_{m_1-W}^{m_2+W}, \mathbf{y}_{m_1-W}^{m_2+W})$ and $d_R = \mathrm{d_E}(\mathbf{y}_R{}_{m_1-W}^{m_2+W}, \mathbf{y}_{m_1-W}^{m_2+W})$. The final outputs of CBA between indices m_1 and m_2 in each contradictory block are determined using the same minimum distance criterion as in (3).

2) Proposed Hard-output TBCR Algorithm: To motivate our proposed algorithm, we provide an example to explain why there is an experimental value W in CBA and BAD. Fig. 4 shows the outputs of the forward and reverse DFEs and the trellis diagram of a 3-tap ($\nu = 2$) channel. By using the definition in [11], the contradictory blocks and the consistent blocks are shown as blocks in grey and white, respectively, in Fig. 4. Consider symbols 3 to 7, which are divided into two consistent and two contradictory blocks. For this example, CBA and BAD work exactly the same because there are no consecutively different outputs between the two DFEs. Since the output of the DFE depends not only on the received symbol at that particular time instance, but also on the surrounding symbols, CBA and BAD calculate the distances on a window of $2W + 1$ symbols around each arbitration unit (symbols 3 and 5 in the example). However, the choice of W is usually based on using simulations to evaluate the performance and then making a tradeoff between performance and complexity.

We note that the outputs of the two DFEs determine two paths through the trellis diagram for the channel, as shown in Fig. 4. Using the trellis diagram, we observe that the two paths through the trellis only differ from symbols 3 to 7. Thus, only these symbols contribute to the difference in Euclidean distances between the received sequence and the two estimated noise-free sequences over the entire trellis. Thus, by comparing the outputs of the forward and reverse DFEs as paths through the channel trellis, we can determine exactly which symbols should be grouped as an arbitration unit. We define a *conflict event* as the portion of the trellis diagram where two paths first diverge from a common state until they first merge back to a common state. Let L denote the length of a given conflict event. It is easy to see that L is at least $\nu+1$, the length of the channel response and that conflict events are always isolated from each other. This resolves the need to choose a window W around contradictory symbols/blocks and also resolves the effect described in [11], in which the author claims that if two contradictory blocks are very close to each other, their block arbitrations would affect each other.

By operating on conflict events in the channel trellis, we show in Section V that the proposed hard-output algorithm can simultaneously reduce the computational complexity and improve the performance in comparison to the schemes in Section IV-A1. Since we identify the conflict events through operating on the trellis, this hard-output algorithm is referred to as hard-output trellis-based conflict resolution (hTBCR). The proposed hTBCR algorithm is summarized as:

1) Process the received sequence $y[k]$ using the BDFE structure and obtain two hard output sequences $\hat{x}_F[k]$ and $\hat{x}_R[k]$.
2) Identify the conflict events in the trellis diagram.
3) For those $x[k]$ not belong to any conflict event, output the results of the forward DFE.
4) For each conflict event, which starts from time index m_1 and ends in m_2, reconstruct the estimated noise-free received signals $y_F[k] = \sum_{m=0}^{\nu} p[m]\hat{x}_F[k-m]$ and $y_R[k] = \sum_{m=0}^{\nu} p[m]\hat{x}_R[k-m]$.
5) The hTBCR output between indices m_1 and m_2 can be determined using (3) with $d_F = \mathrm{d_E}(\mathbf{y}_F{}_{m_1}^{m_2}, \mathbf{y}_{m_1}^{m_2})$ and $d_R = \mathrm{d_E}(\mathbf{y}_R{}_{m_1}^{m_2}, \mathbf{y}_{m_1}^{m_2})$.

B. Soft-output BDFE Algorithms

It is well known that the maximum *a posteriori* (MAP) equalizer is the optimal detection scheme for minimizing the probability of symbol error. For the MAP equalizer, the goal is to estimate the *a posteriori* probabilities (APPs) $\Pr(x[k] \mid \mathbf{y})$. Knowledge of the APPs allows us to generate the log *a posteriori* probability (log-APP) ratio as

$$L_E(x[k]) = \ln \frac{\Pr\left(x[k] = +\sqrt{E_x} \mid \mathbf{y}\right)}{\Pr\left(x[k] = -\sqrt{E_x} \mid \mathbf{y}\right)}. \tag{4}$$

By applying the well known Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm [19] on the trellis for the ISI channel, we rewrite $L_E(x[k])$ as

$$L_E(x[k]) = \ln \frac{\sum_{\mathcal{U}^+} f\left(S[k-1] = s', S[k] = s, \mathbf{y}\right)}{\sum_{\mathcal{U}^-} f\left(S[k-1] = s', S[k] = s, \mathbf{y}\right)}, \tag{5}$$

where $S[k]$ is the trellis state at time k. Here, $\mathcal{U}^+$ and $\mathcal{U}^-$ is the set of branches (denoted by the previous and next state pairs (s', s)) that correspond to the events $x[k] = +\sqrt{E_x}$ and $x[k] = -\sqrt{E_x}$, respectively. In [19], it is shown that the probability density function $f(S[k-1] = s', S[k] = s, \mathbf{y})$ can be factored as $f(S[k-1] = s', S[k] = s, \mathbf{y}) = \alpha_{k-1}(s')\gamma_k(s', s)\beta_k(s)$, where

- $\alpha_{k-1}(s') = f\left(S[k-1] = s', \mathbf{y}_1^{k-1}\right)$ is the forward-looking state probability, which is calculated using forward recursion from time 0 to time $k-1$.
- $\beta_k(s) = f\left(\mathbf{y}_{k+1}^{N} | S[k] = s\right)$ is the backward-looking state probability, which is found using reverse recursion from time N to time $k+1$, and

- $\gamma_k(s',s) = f\left(\mathbf{y}_k^k, S[k]=s|S[k-1]=s'\right)$ is the state-transition probability, which can be calculated directly from the channel likelihood for the received symbol at time k.

The BCJR algorithm first computes the state-transition probabilities for all possible transition pairs according to the trellis diagram and then generates the forward-looking and reverse-looking state probabilities recursively [19]. The (soft) output $L_E(x[k])$ of the MAP equalizer can be computed by (5), and its sign is the hard output of the MAP equalizer.

1) Existing Soft-output BDFE Algorithms: The only existing soft-output BDFE algorithm of which we are aware is the linear combining bidirectional DFE (LC-BDFE) [12]. LC-BDFE is a soft-decision symbol-by-symbol linear-combining technique. LC-BDFE generates its output as a weighted linear combination of the soft outputs $z_F[k]$ and $z_R[k]$ of the forward and reverse DFEs, i.e. $az_F[k]+(1-a)z_R[k]$. The weighting factor a is chosen to minimize the mean square error between the transmitted symbol and the combined soft output [12].

2) Proposed Piecewise BCJR (pBCJR) Algorithm: The definition of conflict event not only divides the outputs of the forward and reverse DFEs in a more efficient way than other BDFE algorithms, but also specifies in which states the two paths spilt and remerge in the trellis (see Fig. 4). Hence, a full BCJR can be performed to compute the log-APP ratios for each $x[k]$ in a conflict event. Assume that a conflict event starts at time m_1 and ends at time m_2, then

$$L_E(x[k]) = \ln \frac{\sum_{\mathcal{U}^+} f\left(S[k-1]=s', S[k]=s, \mathbf{y}_{m_1}^{m_2}\right)}{\sum_{\mathcal{U}^-} f\left(S[k-1]=s', S[k]=s, \mathbf{y}_{m_1}^{m_2}\right)}. \quad (6)$$

We refer to this sub-optimal algorithm as piecewise BCJR (pBCJR). pBCJR differs from the conventional BCJR in that the BCJR computes the log-APP ratio for every $x[k]$ using the whole received sequence $\mathbf{y}$, while the pBCJR computes the log-APP for only those symbols in a conflict event using only those received symbols in that conflict event.

The pBCJR algorithm is summarized as follows:

1) Process the received sequence $y[k]$ using the BDFE structure and obtain two hard output sequences $\hat{x}_F[k]$ and $\hat{x}_R[k]$.
2) Identify the conflict events in the trellis diagram.
3) For each conflict event, which starts in state s_1 at time index m_1 and ends in state s_2 at time index m_2, a full BCJR algorithm is performed using only those symbols in the conflict event. Note that unlike the conventional BCJR, which is usually initialized and terminated in state 0, the trellis is initialized and terminated to states s_1 and s_2, respectively.
4) For those $x[k]$ not belong to any conflict event, the LC-BDFE is used to compute the log-APP ratios.

3) Proposed Soft-output TBCR Algorithm: Although the pBCJR performs additional computations over only the conflict events, for those symbols in a conflict event, it uses a full BCJR algorithm and thus includes all the possible state transitions in computing the log-APP ratios. Hence, the complexity of pBCJR can be much higher than hTBCR. There are many existing techniques to reduce the complexity of the BCJR algorithm [20–22]. Here we focus on how the BDFE structure can be used to reduce complexity. The complexity can be significantly reduced by limiting the BCJR algorithm to operate on only those paths that correspond to the outputs of the forward and reverse DFEs. Since these paths are selected by the forward and reverse DFEs, they may be more reliable than other paths through the same section of the trellis. The hope is that by using these two paths, performance close to pBCJR can be achieved at a greatly reduced complexity. Since this approach has a structure that is similar to hTBCR, we call the current approach soft-output trellis-based conflict resolution (sTBCR).

In sTBCR, we modify (5) to compute the log-APP ratio[1] of $x[k]$ as

$$L_E(x[k]) = \ln \frac{\sum_{\mathcal{V}^+} f\left(S[k-1]=s', S[k]=s, \mathbf{y}_{m_1}^{m_2}\right)}{\sum_{\mathcal{V}^-} f\left(S[k-1]=s', S[k]=s, \mathbf{y}_{m_1}^{m_2}\right)}, \quad (7)$$

for all the symbols in the conflict event for which $\hat{x}_F[k] \neq \hat{x}_R[k]$. Here, $\mathcal{V}^+$ is the set of branches (denoted by the previous and next state pairs (s',s)), which correspond to the event of either $\hat{x}_F[k] = +\sqrt{E_x}$ or $\hat{x}_R[k] = +\sqrt{E_x}$. $\mathcal{V}^-$ is similarly defined. For example, at symbol 3 of Fig. 4, we compute the log-APP ratio of $x[3]$ by using $\mathcal{V}^+ = ((0,0),(0,0))$ and $\mathcal{V}^- = ((0,0),(1,0))$. For those symbols where $\hat{x}_F[k] = \hat{x}_R[k]$, (7) cannot be used (because either the numerator or denominator is zero), and thus, LC-BDFE is used to generate the log-APP.

Each path is disjoint and consists of a single consecutive set of branches, so (7) simplifies to

$$L_E(x[k]) = \ln \frac{\prod_{t=m_1+1}^{m_2} \gamma_t(s_{t-1}^{k+}, s_t^{k+})}{\prod_{t=m_1+1}^{m_2} \gamma_t(s_{t-1}^{k-}, s_t^{k-})}, \quad (8)$$

where s_t^{k+} and s_t^{k-} are the states at time t for the path with $x[k] = +\sqrt{E_x}$ and $x[k] = -\sqrt{E_x}$, respectively. This expression can further be simplified as

$$L_E(x[k]) = \frac{\mathrm{d_E}(\mathbf{y}_{m_1}^{m_2,k-}, \mathbf{y}_{m_1}^{m_2})}{2\sigma^2} - \frac{\mathrm{d_E}(\mathbf{y}_{m_1}^{m_2,k+}, \mathbf{y}_{m_1}^{m_2})}{2\sigma^2}, \quad (9)$$

where $\mathbf{y}_{m_1}^{m_2,k+}$ and $\mathbf{y}_{m_1}^{m_2,k-}$ denote the estimated noise-free paths (i.e., one is a sub-sequence of $\mathbf{y}_F$ and the other is a sub-sequence of $\mathbf{y}_R$) corresponding to $x[k] = +\sqrt{E_x}$ and $x[k] = -\sqrt{E_x}$, respectively. Note that one of the two estimated noise-free sequences terms corresponds to the output of the forward equalizer and the other corresponds to the output of the reverse equalizer. Thus, the magnitude of L_E is equal for all of the symbols with $x_F[k] \neq x_R[k]$ in a conflict event. Moreover, the sign of the log-APP corresponds to selecting the hard output for the symbol on the path with the minimum Euclidean distance, thus providing a further connection of sTBCR with hTBCR. Thus, the overall sTBCR algorithm is as follows:

1) Process the received sequence $y[k]$ using the BDFE structure and obtain two hard output sequences $\hat{x}_F[k]$ and $\hat{x}_R[k]$.
2) Identify the conflict events in the trellis diagram.

[1]To simplify the discussion, we use the term "log-APP ratio" even though only a subset of trellis transitions are considered in sTBCR.

TABLE I
COMPUTATIONAL COMPLEXITY IN PRODUCING OUTPUT, PER ARBITRATION BLOCK OF LENGTH L' FOR HARD-OUTPUT BDFE ALGORITHMS AND PER SYMBOL FOR SOFT-OUTPUT BDFE ALGORITHMS

Hard-output BDFE Algorithms	**# of Additions per Arbitration Block**	**# of Multiplications per Arbitration Block**
BAD (IV-A1)	$2L' + 4W - 1$	$L' + 2W$
CBA (IV-A1)	$2L' + 4W - 1$	$L' + 2W$
hTBCR (IV-A2)	$2L' + 2\nu - 1$	$L' + \nu$
Soft-output BDFE Algorithms	**# of Additions per Symbol**	**# of Multiplications per Symbol**
MAP equalizer (IV-B)	$3 \cdot 2^{\nu+1}$	$4 \cdot 2^{\nu+1}$
LC-BDFE (IV-B1)	1	3
pBCJR (IV-B2)	$3 \cdot 2^{\nu+1}$	$4 \cdot 2^{\nu+1}$
sTBCR (IV-B3)	$3 \cdot 2$	$4 \cdot 2$

3) For each conflict event, compute the Euclidean distances between the estimated noise-free sequences and the received sequences (cf. (9)).
4) For each symbol in the conflict event such that $x_F[k] \neq x_R[k]$, set the magnitude of the log-APP to the absolute value of the weighted difference of the Euclidean distances (cf. (9)). Set the sign according to the path with the minimum Euclidean distance (i.e., positive if the path with the minimum distance has $x[k] = +\sqrt{E_x}$, and negative otherwise).
5) For those symbols in a conflict event with $x_F[k] = x_R[k]$ and those symbols that do not belong to any conflict event, use LC-BDFE to compute the log-APP ratios.

C. Complexity Comparisons of Different BDFE algorithms

Selecting an equalization technique involves a tradeoff between performance and complexity; thus, this section is focused on the relative complexities of the various hard- and soft-output BDFE algorithms discussed in Sections IV-A and IV-B. However, we first briefly discuss some general complexity issues of BDFEs versus the conventional DFE. One significant difference between BDFEs and the conventional DFE is that extra memory may be needed to store the received symbols in a block to perform the time-reversal operation. The use of two DFEs also implies doubling the computational complexity for the FFFs and FBFs. A naive implementation also requires twice the complexity in generating the DFE coefficients. However, as mentioned in Section III, by using the information of the forward DFE's coefficients, we show that the computational complexity of generating the reverse DFE's coefficients can be significantly reduced (also see the Appendix). Finally, the computation complexity of BDFE algorithms is also determined by their arbitration/combining technique, as further discussed below.

We first evaluate the computational complexity for the hard-output BDFE algorithms in terms of the number of required DSP operations for an arbitration block with length L', with other parts of BDFE being equal. For TCBR, the last ν symbols that make up a conflict event agree in value, and thus the length of the arbitration block is $L' = L - \nu$. For BAD, $L' = 1$, and for CBA, $L' \leq L - \nu$; however, BAD and CBA generally divide each conflict event into multiple

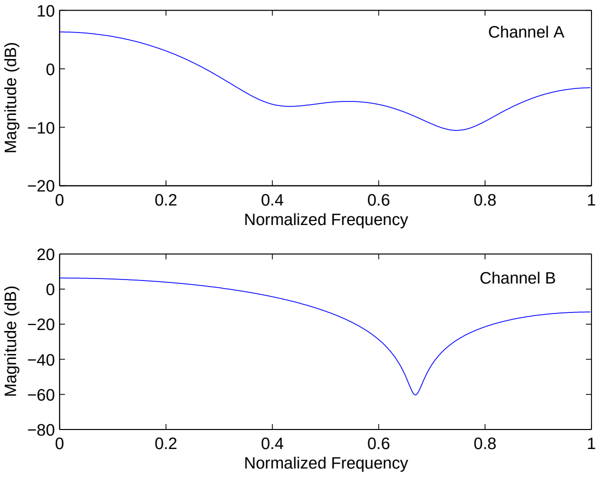

Fig. 5. Frequency responses of channels A and B.

arbitration blocks. Table I shows the computational complexity required to compute the output for various hard-output BDFE algorithms of an arbitration block with length L' (cf. Table II of [13]). It can be seen that the computational complexity of both BAD and CBA is proportional to the value of W. For a 5-tap ($\nu = 4$) channel [11, 13], the typical value of W for BAD and CBA is 15 and 10. Taking into account that BAD and CBA may divide a single conflict event into several arbitration blocks, the overall complexity of BAD and CBA is usually around 10 and 5 times higher than that of hTBCR (cf. [23] for supporting results).

We evaluate the computational complexity of the soft-output BDFE algorithms in terms of the number of DSP operations per received symbol required to generate the log-APP. Table I also shows the computational complexity per symbol for various soft-output BDFE algorithms with binary modulation (cf. Table I of [7]). LC-BDFE has the lowest complexity, since it is just a linear combining technique. As mentioned before, pBCJR has the same complexity per symbol as the MAP equalizer, while sTBCR reduces the complexity of pBCJR by a factor of 2^ν.

V. SIMULATION RESULTS

In this section, we present performance results on the bit error rate (BER) of the various equalizers introduced in Section II and Section IV. The BERs are found through simulation with the following parameters.

- Two real-valued channels [13] are simulated, whose frequency responses are shown in Fig. 5. Channel A and channel B are a 5-tap maximal phase channel and a 5-tap symmetric channel, respectively, with channel impulse responses specified by $H_A(z) = 0.227 + 0.227z^{-1} + 0.46z^{-2} + 0.46z^{-3} + 0.688z^{-4}$ and $H_B(z) = 0.227 + 0.46z^{-1} + 0.688z^{-2} + 0.46z^{-3} + 0.227z^{-4}$. The channels are perfectly known to the receiver.
- A rate 1/2, constraint-length 7 convolutional code is used. The generator polynomials of the code are $g_0(x) = 1 +$

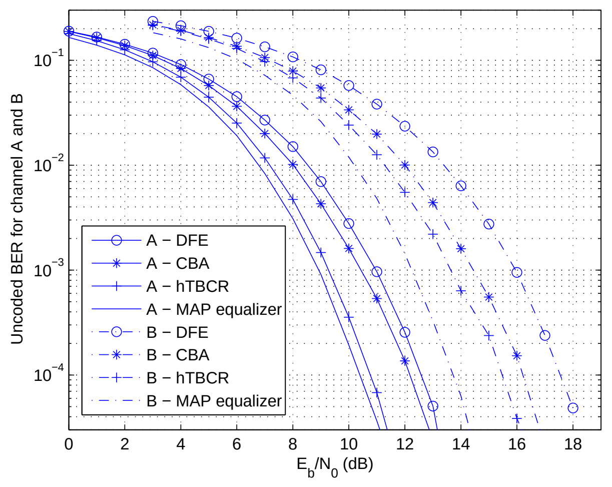

Fig. 6. Bit error rate (BER) performance of uncoded BPSK over channel A (a maximum phase channel) and channel B (a symmetric channel).

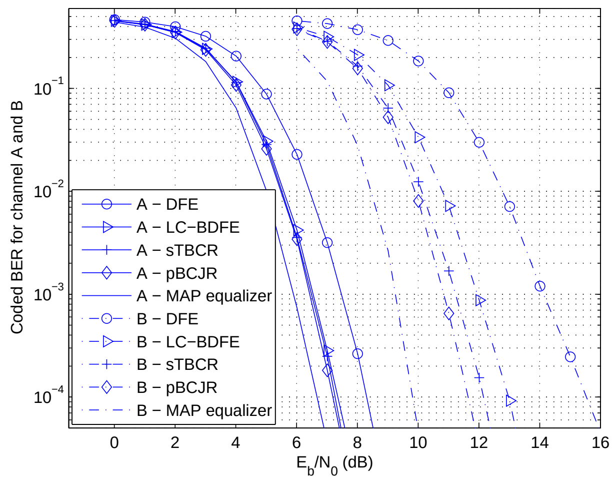

Fig. 7. Bit error rate (BER) performance of coded BPSK over channel A (a maximum phase channel) and channel B (a symmetric channel).

$D^2+D^3+D^5+D^6$ and $g_1(x)=1+D+D^2+D^3+D^6$. Soft-input Viterbi decoding is employed at the receiver.

- A rectangular interleaver of depth 25 is implemented.
- For channel length of ν, $N_b=\nu, N_f=2\nu+1$ and $\Delta=N_f-1$ are used for all DFEs.
- We only provide the performance results for CBA since CBA usually outperforms BAD with lower complexity [23]. We set the extension of the computation window in CBA to $W=\frac{\nu}{2}$ to make the complexity of CBA comparable to hTBCR[2].
- Uncoded and coded BER performance are reported in terms of signal-to-noise ratio (SNR) per channel bit and SNR per information bit, respectively.

[2]For the typical value of 10 that is used in other work, the performance of CBA will be the same as hTBCR, but the complexity will be much higher.

The BER performance of uncoded binary phase-shift keying (BPSK) with different equalization techniques is shown in Fig. 6. Consider first the results for channel A, which is a maximal-phase channel; i.e., all the poles are inside the unit circle, and the reverse channel results in a minimum-phase channel. The BDFE is particularly beneficial in such a scenario [9]. We can see that hTBCR is only 0.4 dB away from the MAP equalizer, while it outperforms CBA by more than 1.2 dB for BERs less than 10^{-3}. The complexity of CBA (based on the number of DSP operations in the arbitration stage, cf. [23]), is around 1.2 times higher than that of hTBCR. Hence, performing arbitration on conflict events in the channel trellis can simultaneously give better performance and lower complexity than previously proposed techniques.

Fig. 6 also shows the performance of different hard-output equalization techniques on channel B, which is a symmetric channel. From Fig. 5 we can see that this channel contains a spectral null and thus induces severe ISI. The conventional DFE suffers a lot in this channel because of error propagation. There is more than a 2 dB gap between hTBCR and the conventional DFE for BERs less than 10^{-2}, and this shows the power of BDFE even if the channel is symmetric. CBA has around 0.8 dB performance degradation from hTBCR, and the complexity is also 1.4 times higher. Once again we can see that hTBCR is a more efficient algorithm than other hard-output BDFE algorithms. However, there is at least 1.3 dB gap from the MAP equalizer for the various BDFE algorithms for BERs less than 10^{-2}. This represents the fundamental limit of the BDFE algorithms. BDFE relies on the difference between the outputs of the forward and reverse DFEs to improve the BER performance, but in this severe ISI channel, there is a high probability that both the forward and reverse DFEs make incorrect decisions, in which case the BDFE cannot provide any advantage. Nevertheless, these results demonstrate that hTBCR is a more desirable BDFE algorithm than the other approaches in terms of both performance and complexity.

The performance of coded BPSK with various soft-output equalization techniques and soft-input channel decoding is shown in Fig. 7. Consider first the results for channel A. LC-BDFE outperforms the conventional soft-decision DFE by around 0.8 dB for BERs less than 10^{-2}. Although LC-BDFE does not give much gain in the form of a hard-output algorithm as reported by many previous works [11, 13], this result shows that it can provide significantly better performance than the conventional soft-decision DFE when followed by a soft-input decoder. On the other hand, sTBCR and pBCJR perform only slightly better than LC-BDFE. We can see that the performance of LC-BDFE is only 1 dB away from MAP equalizer. The results imply that in channel A, LC-BDFE gives close to optimal performance with low computational complexity. For channel B, sTBCR provides around 2.8 dB better performance than the conventional DFE for a BER of 10^{-3}, and outperforms LC-BDFE by 0.8 dB. A further 0.5 dB gain can be achieved by using pBCJR over channel B, and it is less than 2 dB away from the optimal MAP equalizer.

The performance of uncoded quadrature phase-shift keying (QPSK) and 16-point quadrature amplitude modulation (16-QAM) over channel A with various equalization techniques

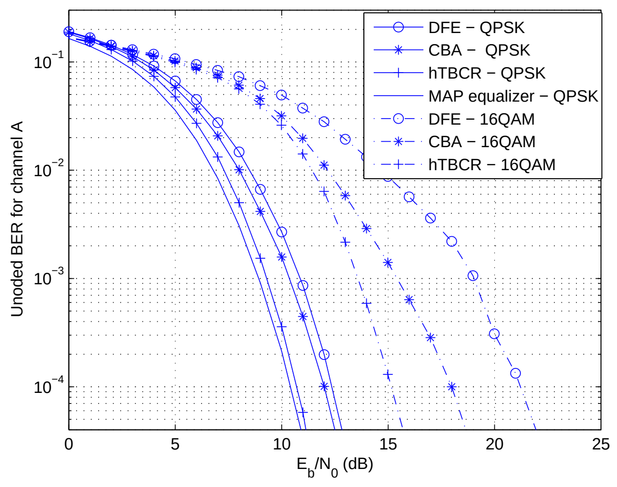

Fig. 8. Bit error rate (BER) performance of uncoded QPSK and 16-QAM over channel A (a maximum phase channel).

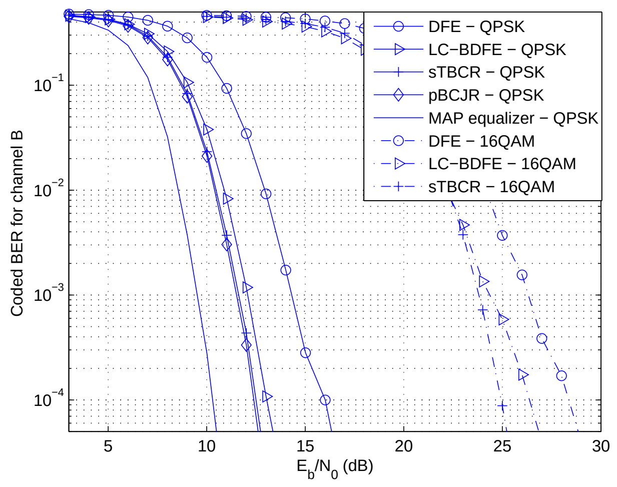

Fig. 9. Bit error rate (BER) performance of coded QPSK and 16-QAM over channel B (a symmetric channel).

is shown in Fig. 8 [3]. hTBCR outperforms CBA, achieving up to a 1.5 dB and 2.5 dB performance advantage with QPSK and 16-QAM, respectively. Note that for the standard DFE with 16-QAM, the channel coefficient matrix $\mathbf{\Lambda}$ often becomes ill-conditioned and further degrades the performance at higher values of E_b/N_0. As with BPSK, hTBCR offers little performance advantage over CBA on channel B but has lower complexity.

The performance of coded QPSK and 16-QAM over channel B with various equalization techniques is shown in Fig. 9. In comparison to LC-BDFE, sTBCR provides around 0.75 dB gain for QPSK and approximately 1.6 dB gain with 16-QAM. However, pBCJR offers little additional gain. As with the uncoded results, the channel coefficient matrices for the forward DFE often become ill-conditioned at high E_b/N_0, and this degrades the performance of both the standard DFE and, for 16-QAM, the LC-BDFE. As with BPSK, sTBCR and pBCJR offer little performance gain over LC-BDFE on channel A with QPSK and 16-QAM.

In summary, hTBCR and sTBCR offer significant improvement over previously proposed BDFE algorithms. However, their performance is still dominated by those symbols over which both the forward and reverse DFEs make incorrect decisions, especially for severe ISI channels. The fact that the performance advantage of sTBCR over LC-BDFE varies significantly with the channel suggests that an adaptive scheme could be used in which the more-complicated sTBCR algorithm is used only when the channel satisfies certain conditions for which sTBCR will give a significant performance gain over LC-BDFE[4].

VI. Conclusion

We propose trellis-based algorithms for both hard- and soft-output BDFE. We address the problem of inefficiency for current existing hard-output BDFE algorithms and propose a novel definition of the unit in the arbitration process, namely the conflict event. Simulation results show that the proposed hard-output trellis-based conflict resolution algorithm (hTBCR) can simultaneously provide better performance and lower complexity than other hard-output BDFE algorithms. The piecewise BCJR algorithm (pBCJR) and soft-output trellis-based conflict resolution (sTBCR) algorithms were proposed for use with soft-input decoding of an error control-code and shown to bring a 0.75 dB to 1.6 dB performance improvement over the only existing soft-output BDFE algorithm in a severe ISI channel. In addition, we showed how the relation between the coefficients of the forward and reverse DFEs can be used to develop efficient algorithms to reduce the computational complexity for computing the DFE filter coefficients.

Appendix

Modified Jacobi Algorithm

Recall the set of linear equations given by (2) and rewrite them as $\mathbf{\Lambda w} = \mathbf{p}_{\Delta+1}$. Cholesky factorization of the matrix $\mathbf{\Lambda}$ can first be performed to find a lower triangular matrix $\mathbf{L}$ such that $\mathbf{\Lambda} = \mathbf{L}\mathbf{L}^H$. To obtain $\mathbf{w}$, forward substitutions can be applied to solve $\mathbf{Lx} = \mathbf{p}_{\Delta+1}$ for $\mathbf{x}$, and $\mathbf{w}$ can be found by solving $\mathbf{L^H w} = \mathbf{x}$ with backward substitutions. Thus the ability to find $\mathbf{L}$ efficiently is crucial. Based on the relation described in Section III, the first $N_f - N_b$ columns of the lower triangular matrix $\mathbf{L}$ are conjugate between the forward and reverse DFEs. Hence, the intermediate products of the Cholesky factorization of $\mathbf{\Lambda}_F$ can be reused in the Cholesky factorization of $\mathbf{\Lambda}_R$ as shown below.

The Jacobi algorithm [24] is an efficient technique to find the Cholesky factorization of any positive definite matrix. We modify the Jacobi algorithm to efficiently compute the Cholesky factorization of $\mathbf{\Lambda}_F$ and $\mathbf{\Lambda}_R$ simultaneously. The

[3]The performance of the MAP equalizer (and later the pBCJR algorithm) is not shown for 16-QAM because the number of states in the trellis (65536) precluded simulation.

[4]This adaptation was suggested by an anonymous reviewer.

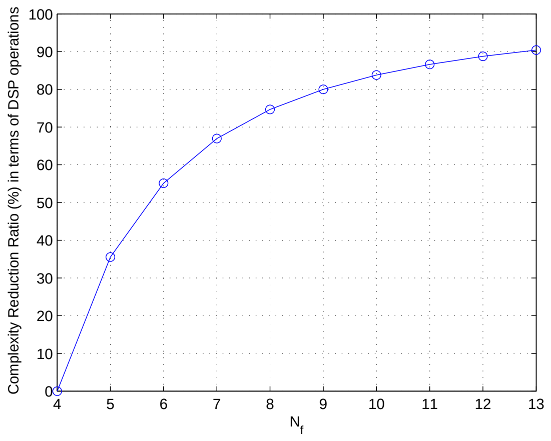

Fig. 10. Complexity reduction ratio of the modified Jacobi algorithm.

modifications to the Jacobi algorithm are summarized as follows:

1) Let $\mathbf{m}_F^{(i)}$ and $\lambda_F^{(i)}$ be the i-th column and the (i,i) element of $\mathbf{\Lambda}_F$ respectively.
2) Let $\mathbf{m}_R^{(i)}$ and $\lambda_R^{(i)}$ be the i-th column and the (i,i) element of $\mathbf{\Lambda}_R$ respectively.
3) For $i = 0 : N_f - N_b - 1$
   a) $\mathbf{\Lambda}_F = \mathbf{\Lambda}_F - \mathbf{m}_F^{(i)}(\lambda_F^{(i)})^{-1}(\mathbf{m}_F^{(i)})^H$ and $\mathbf{\Lambda}_R = \mathbf{\Lambda}_R - \left(\mathbf{m}_F^{(i)}(\lambda_F^{(i)})^{-1}(\mathbf{m}_F^{(i)})^H\right)^*$
   b) Then the ith columns of $\mathbf{L}_F$ and $\mathbf{L}_R$ are $\mathbf{l}_F^{(i)} = \frac{\mathbf{m}_F^{(i)}}{\sqrt{|\lambda_F^{(i)}|}}$ and $\mathbf{l}_R^{(i)} = \frac{(\mathbf{m}_F^{(i)})^*}{\sqrt{|\lambda_F^{(i)}|}}$, respectively.
4) For $i = N_f - N_b : N_f - 1$
   a) $\mathbf{\Lambda}_F = \mathbf{\Lambda}_F - \mathbf{m}_F^{(i)}(\lambda_F^{(i)})^{-1}(\mathbf{m}_F^{(i)})^H$ and $\mathbf{\Lambda}_R = \mathbf{\Lambda}_R - \mathbf{m}_R^{(i)}(\lambda_R^{(i)})^{-1}(\mathbf{m}_R^{(i)})^H$
   b) Then the ith columns of $\mathbf{L}_F$ and $\mathbf{L}_R$ are $\mathbf{l}_F^{(i)} = \frac{\mathbf{m}_F^{(i)}}{\sqrt{|\lambda_F^{(i)}|}}$ and $\mathbf{l}_R^{(i)} = \frac{\mathbf{m}_R^{(i)}}{\sqrt{|\lambda_R^{(i)}|}}$, respectively.

The results in Fig. 10 show the complexity reduction ratio in computing the Cholesky factorization of $\mathbf{\Lambda}_R$ by the modified Jacobi algorithm with $N_b = \nu = 4$. It is observed that using the information of the forward DFE can reduce the complexity by more than 80% when N_f is large.

References

[1] G. D. Forney, Jr., "Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference," *IEEE Trans. Inform. Theory*, vol. 18, no. 3, pp. 363–378, May 1972.

[2] J. G. Proakis, *Digital Communications*, 4th ed. McGraw-Hill, 2000.

[3] M. Tomlinson, "New automatic equaliser employing modulo arithmetic," *IEE Electron. Lett.*, vol. 7, no. 5, pp. 138–139, Mar. 1971.

[4] M. Eyuboglu, "Detection of coded modulation signals on linear, severely distorted channels using decision-feedback noise prediction with interleaving," *IEEE Trans. Commun*, vol. 36, no. 4, pp. 401–409, Apr. 1988.

[5] S. Ariyavisitakul and Y. Li, "Joint coding and decision feedback equalization for broadband wireless channels," *IEEE. J. Sel. Areas Commun.*, vol. 16, no. 9, pp. 1670–1678, Dec. 1998.

[6] A. Chan and G. Wornell, "A class of block-iterative equalizers for intersymbol interference channels: Fixed channel results," *IEEE Trans. Commun*, vol. 49, pp. 1966–1976, Nov. 2001.

[7] M. Tüchler, R. Koetter, and A. Singer, "Turbo equalization: Principles and new results," *IEEE Trans. Commun*, vol. 50, no. 5, pp. 754–767, Mar. 2002.

[8] A. Higashi and H. Suzuki, "Dual-mode equalization for digital mobile radio," in *Proc. IEICE*, vol. 74-B-II, Mar. 1991, pp. 91–100.

[9] S. Ariyavisitakul, "A decision feedback equalizer with time-reversal structure," *IEEE. J. Sel. Areas Commun.*, vol. 10, no. 3, pp. 599–613, Apr. 1992.

[10] J. Balakrishnan and C. R. Johnson, Jr., "Bidirectional decision feedback equalizer: Infinite length results," in *Proc. Asilomar Conf. Signals, Syst. and Comput.*, vol. 2, 2001, pp. 1450–1454.

[11] X.-G. Tang and Z. Ding, "Contradictory block arbitration for bi-directional decision feedback equalizers," in *Proc. IEEE Global Commun. Conf.*, vol. 1, Nov. 2002, pp. 283–286.

[12] J. Balakrishnan and C. R. Johnson, Jr., "Time reversal diversity in decision feedback equalization," in *Proc. Allerton Conf. Commun., Control, and Computing*, Monticello, IL, Oct. 2002.

[13] J. Nelson, A. Singer, U. Madhow, and C. McGahey, "BAD: Bidirectional arbitrated decision-feedback equalization," *IEEE Trans. Commun*, vol. 53, no. 2, pp. 214–218, Feb. 2005.

[14] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear Estimation*. Upper Saddle River, New Jersey: Prentice Hall, 2000.

[15] T. Kailath and J. Chun, "Generalized displacement structure for Block-Toeplitz, Toeplitz-Block, and Toeplitz-derived Matrices," *SIAM J. Matrix Anal. Appl.*, vol. 15, no. 1, pp. 114–128, Jan. 1994.

[16] N. Al-Dhahir and A. Sayed, "The finite-length multi-input multi-output MMSE-DFE," *IEEE Trans. Signal Process.*, vol. 48, no. 10, pp. 2921–2936, Oct. 2000.

[17] G. Barriac, N. Jacobsen, and U. Madhow, "Beyond BAD: A parallel arbitration framework for low-complexity equalization," in *Proc. Allerton Conf. Commun., Control, and Computing*, Monticello, IL, Oct. 2001.

[18] J. Nelson, A. Singer, and U. Madhow, "Asymptotic efficiency of the BAD algorithm," in *Proc. IEEE Statistical Signal Process. Workshop*, Sept. 2003, pp. 86–89.

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

[20] V. Franz and J. Anderson, "Concatenated decoding with a reduced-search BCJR algorithm," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 2, pp. 186–195, Feb. 1998.

[21] G. Colavolpe, G. Ferrari, and R. Raheli, "Reduced-state

BCJR-type algorithms," *IEEE J. Sel. Areas Commun.*, vol. 19, no. 5, pp. 848–859, May 2001.

[22] C. Vithanage, C. Andrieu, and R. Piechocki, "Novel reduced-state BCJR algorithms," *IEEE Trans. Commun.*, vol. 55, no. 6, pp. 1144–1152, June 2007.

[23] C. W. Wong, J. M. Shea, and Y. Lee, "Trellis-based conflict resolution for bidirectional decision-feedback equalization," in *Proc. IEEE Military Commun. Conf. (MILCOM)*, Orlando, FL, Oct. 2007, pp. 1–7.

[24] G. Golub and C. V. Loan, *Matrix Computations*. Baltimore, MD: John Hopkins Univ. Press, 1996.

Chan Wong Wong (S'07) received the B.S. and the M.S. in electrical engineering from National Taiwan University, Taipei, Taiwan in 2002 and 2004, respectively. Chan Wong is currently pursuing his PhD degree at University of Florida, Gainesville, FL. From 2002 to 2004 he was a teaching assistant at Graduate Institute of Communications Engineering, National Taiwan University. During 2003 to 2006 he was with Afa Technologies, Inc., Taipei, Taiwan, as a DSP system engineer in developing demodulators for various digital video broadcasting standards. His current research interests lie in the area of communication theory applied to equalization and coding for wireless communication. Chan Wong is the member of The Phi Tau Phi Scholastic Honor Society of the Republic of China.

John M. Shea (S'92-M'99) received the B.S. (with highest honors) in computer engineering from Clemson University in 1993 and the M.S. and Ph.D. degrees in electrical engineering from Clemson University in 1995 and 1998, respectively.

Dr. Shea is currently an Associate Professor of electrical and computer engineering at the University of Florida. Prior to that, he was an Assistant Professor at the University of Florida from July 1999 to August 2005 and a post-doctoral research fellow at Clemson University from January 1999 to August 1999. He was a research assistant in the Wireless Communications Program at Clemson University from 1993 to 1998. He is currently engaged in research on wireless communications with emphasis on error-control coding, cross-layer protocol design, cooperative diversity techniques, and hybrid ARQ.

Dr. Shea was selected as a Finalist for the 2004 Eta Kappa Nu Outstanding Young Electrical Engineer Award. Dr. Shea was a National Science Foundation Fellow from 1994 to 1998. He received the Ellersick Award from the IEEE Communications Society for the Best Paper in the Unclassified Program of the IEEE Military Communications Conference in 1996. He has been an Editor for the IEEE Transactions on Wireless Communications since 2008. He was an Associate Editor for the IEEE Transactions on Vehicular Technology from 2002 to 2007.