INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 2 (June), pp. 388-395

A Reliability Level List based SDD Algorithm for Binary Cyclic Block Codes

B. Yamuna, T.R. Padmanabhan

B.Yamuna
Assistant Professor,Dept of ECE
Amrita Vishwa Vidyapeetham,
Amrita School of Engineering
Amrita Nagar, Coimbatore. 641 112
Tamil Nadu, India
E-mail: b_yamuna@cb.amrita.edu

Dr. T.R.Padmanabhan
Professor Emeritus, Dept of IT
Amrita Vishwa Vidyapeetham,
Amrita School of Engineering
E-mail: trp@amrita.edu

Abstract: Soft decision decoding (SDD) provides a better coding gain by making use of the unquantized channel output. In this paper we introduce the concept of a Reliability Level List (RLL); based on the RLL a new SDD algorithm for Binary Phase Shift Keying (BPSK) based binary cyclic block codes is proposed. The algorithm guarantees to extract the most reliable codeword in an iterative manner. The formation of the RLL involves a search for the next possible entry into the RLL based on the error probability which is a reflection of the reliability values of the bits of the received word obtained from the channel. The procedure for the formation of RLL which is the central idea of the paper is given as a structured algorithm.
Keywords: cyclic block codes, reliability based decoding, soft decision decoding, probability of error.

1 Introduction

Soft decision decoding (SDD) algorithms in general focus on the extraction of the codeword from the received word by making use of the reliability information available at the channel output which is otherwise discarded in hard decision decoding. SDD algorithms have increased complexity compared to hard decision decoding but they trade off complexity for better error performance.

The idea of exploiting the real channel output to the maximum possible extent instead of quantizing it to 0's and 1's as with hard decision decoding has been the essence of SDD vis-â-vis binary cyclic block codes. Different SDD algorithms have been proposed in the recent years-each with its own trade off between complexity and error performance[1- 3].

The reliability based SDD algorithms aim at decoding a received word which is more reliable even under low signal-to-noise ratio (SNR) conditions. Researchers are focusing on making the best use of the channel output and improving decoding reliability. In these algorithms the channel output is used to quantify reliability values of the bits of the received word. The bits are processed with respect to their reliability values. They can be based on processing the bits from the least reliable end as with Chase decoding algorithms and Generalized-minimum distance (GMD) decoding algorithms [4], [5] or they can be based on processing from the most reliable

end as with Ordered Statistics Decoding algorithms[6].

A SDD algorithm is evolved in this paper. The algorithm uses the channel output reliability in the true sense. The significance of the work is in the fact that the target codeword identified through the algorithm here is the best that reliability based SDD algorithm can lead to.

The method involves searching in a space of 2^n words for the most reliable codeword. But the extent of actual search is confined to a much shorter range as brought out in the paper. The focus of the paper is two-fold: (1) Proving that the codeword extracted is the best possible one from a soft decision point of view and (2) giving a structured algorithm for the extraction of the codeword.

2 Preliminaries of the RLL based SDD Algorithm

The problem of SDD is one of starting with the received word and identifying a codeword which will be the most reliable one; this codeword is called the 'target codeword' here. The process essentially amounts to arranging all the possible 2^n binary bit sequences in the order of increasing reliability and picking the first codeword from it. The sequence so arranged is called the Reliability Level List (RLL) here. Hence RLL represents the search sequence to be followed to find the most reliable codeword.

For an n bit sequence let the pair $\{(b_i,\ m_i)\}$, $0 \leq i \leq n$ represent the sets of the bit values and the corresponding reliability magnitudes. Each bit of hard decision has an associated probability of error given as $\int_{m_i}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx$.

Let $Q(\frac{m_i}{\sigma})$=$\int_{m_i}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx$, σ being the noise variance. Arrange the set $\{Q(\frac{m_i}{\sigma})\}$ in descending order of magnitudes and assign integer values k to them such that k =1 for the bit with the largest value of $Q(\frac{m_i}{\sigma})$; let $q[1]$ represent this $Q(\frac{m}{\sigma})$ value. Similarly $k = 2$ for the next one with $q[2]$ representing the corresponding $Q(\frac{m}{\sigma})$ value and so on until $k = n$ for the bit with the smallest value of $Q(\frac{m_i}{\sigma})$. Let $\{q[k]\}$ be the rearranged set of numbers and $\{p[k]\} = \{1 - q[k]\}$.

With an (n,k) code all the 2^k possible codewords are included in the total space of 2^n words. All these codewords appear in the RLL each with its own associated probability of being the target codeword; the very first codeword to appear in the list being the most reliable of the codewords is the target codeword; those further down the RLL being less reliable need not be examined. The first few entries in RLL as defined above can be seen to be as given below:

- The topmost entry is the term $\prod_{k=1}^{15} p[k]$ having the largest magnitude. This represents the received word itself. i.e., none of the bits is in error.
- The next entry in RLL is the product $q[1]\prod_{k=2}^{15} p[k]$ having the next higher magnitude. This represents the received word with the least reliable bit with k =1 being in error. Hence the received word with the least reliable bit flipped is examined and if found to be a codeword it is the target codeword and the search stops. If it is not a codeword then the subsequent entry in RLL has to be made and the process of examining the entry for identification of target codeword is to be continued. The process is thus repeated until target codeword is identified. The third and fourth entries are given below for clarity.
- The third entry in RLL is the term $q[2]p[1]\prod_{k=3}^{15} p[k]$
- The fourth entry in RLL is either $q[3]p[2]p[1]\prod_{k=4}^{15} p[k]$ or $p[3]q[2]q[1]\prod_{k=4}^{15} p[k]$ based on whichever candidate has the higher magnitude. In general the position of a word in RLL is decided by the magnitude of the corresponding product

$$\prod_{k_i} p[k_i] \prod_{k_j} q[k_j] \qquad (1)$$

where k_i and k_j represent the set of k values for which the $p[k_i]$ and $q[k_j]$ values respectively appear in the product. This is the crux of selecting the next entry in to the RLL at every stage. With$s[k] = \frac{q[k]}{p[k]}$ define

$$f = \frac{\prod_{k_j} q[k_j]}{\prod_{k_i} p[k_i]} \tag{2}$$

Using f the product $\prod_{\tau=1}^{n} P[\tau] \times f$ can be used in place of the product in (1) above. This directly leads to the following lemma.

Lemma 1:

Let v represent the rank of an entry in RLL and f_v the corresponding f value as defined in equation (2). Then the v^{th} entry in RLL is such that $f_{v-1} \geq f_v \geq f_{v+1}$ for $0 \leq v \leq n$

The above inequality means that the problem of identifying the rank of an entry in the RLL is the same as computing the f_v's and arranging them in descending order of magnitudes. Since log f_v is a monotonically increasing function of f, the inequality implies log $f_{v-1} \geq$ logf_v $\geq$ log f_{v+1} for $0 \leq v \leq n$.

If an entry in RLL is known its log f_v is known. The problem of deciding the very next entry can be looked upon as the problem of identifying a set of log $s[k]$ such that the corresponding Σ log $s[k]$ satisfies the following two conditions

1. It should be smaller than the Σlog $s[k]$ for the present entry.
2. It should be the largest in magnitude amongst the set yet to be entered in the RLL.

With $W = \Sigma$ log $s[k]$, an entry with rank v in RLL is characterized by its W_v. With this the above result can be formalized as a theorem:

Theorem 1:

For any $v \in \{0, 2^{n-1}\}$ the corresponding entry in RLL has a W_v such that $w_{v-1} \geq w_v \geq w_{v+1}$

Log $s[k]$ being a negative quantity, $W = \Sigma$log $s[k]$ is also a negative quantity. It is more convenient to work with log $(\frac{1}{s[k]})$ = (-log $s[k]$) and the corresponding sum M = -Σ log $s[k]$. With this, Theorem 1 implies Lemma 2 below which is more convenient to work with.

Lemma 2:

For any $v \in \{0, 2^{n-1}\}$ the corresponding entry in RLL has a M_v such that $M_{v-1} \leq M_v \leq M_{v+1}$

3 Procedure for Forming the RLL

The procedure involves forming sequences of n bit binary numbers which would be the entries in the RLL. Let v denote the rank of an entry in the list with v ranging from 0 to 2^n. The step by step procedure for making the entries in the RLL is follows.

1. M_0, M_1, M_2, where $M_0 = 0, M_1$ = - log $s[1]$, M_2 = - log $s[2]$ are formed as explained earlier in section 2 and the corresponding binary sequences designated N_0, N_1, N_2 are entered in the RLL.
2. Define a 'pending list' - P_L as the collection of all contenders for the next entry to the RLL.
3. The entries in the P_L are characterized by the following.
   (a) Each entry is a collection of indices.
   (b) Each entry has an associated characteristic magnitude - M.

4. Once the P_L is fully known, the next entry to the RLL can be decided by comparing the M values for each entry in the P_L. It also implies that the left over elements in the P_L will also be members for the P_L for the subsequent entry in the RLL.

5. Once an entry to the RLL is decided, the P_L for the next entry to the RLL is formed through the following steps.

   (a) All the entries left over after the previous entry to the RLL are also to be in the new P_L.

   (b) Additional sets are to be added to the new P_L depending on the present entry to the RLL. This essentially amounts to scanning the indices in the set for the present entry and altering them. A perusal of the set of indices shows that the additional element sets can be formed as follows:

      i. If the least index K_1 for the set is not 1 add 1 to the index set.

      ii. For every index X_i in the set if $X_i+1 < X_{i+1}$ replace X_i with X_i+1. This amounts to adding a set to the P_L where the contribution to M by X_i is increased by the least possible amount in the neighborhood of X_i.

The procedure to form the RLL as a sequence of 2^n binary numbers explained above can be cast as structured algorithm. The list of notations used in the algorithm is given below:

- K: reliability index assigned to the bits of the received word.
- N_v : binary number in RLL with rank v.
- $s[k] = \frac{q[k]}{1-q[k]}$
- n : block length of the codeword
- P_L : Pending list after updating
- P_u : pending list before updating
- T_i : i^{th} set in P_L
- J_{max} :total number of T_i's in P_L
- β_i: number of elements in T_i
- F : flag
- U_i, μ, Δ: Temporary symbols used for T_i's, U_i , and J_{max} respectively

4 RLL based SDD Algorithm

1. **Input**: {-log $s[k]$}, n.
2. **Output**: Sequential entries in RLL
3. **Initial Condition**: $N_0 = \{0\}$, $N_1 = \{1\}$, $N_2 = \{2\}$, $P_L = \{T_1, T_2\}$ where $T_1 = \{3\}$, $T_2 = \{1, 2\}$,$J_{max} = 2$, $F = 0$.
4. Do For v=3 to 2^n

5. Do For i=1 to J_{max}
   Compute $M_{v_i} = -\sum_{\alpha=1}^{\beta_i} log(s[X_\alpha])$
   EndDo i

6. M_{min} = min{ M_{v_i} for i = 1 to J_{max}}

7. Let i_m = the i for which M_{v_i} is minimum

8. Form a binary sequence N_v whose 1 bits are decided by the indices in the set T_{im}

9. $\beta_v = \beta_{im}$

10. Do for i=1 to J_{max}
    If($i \neq i_m$) $U_i = T_i$
    EndDo i
    (Removing T_{im} from the pending list and reassigning the T_i's to the U_i's.)

11. $P_u = \{0\}$

12. Do for $i = i$ to i_{m-1}
    $P_u = \{ P_u : U_i\}$
    EndDo i

13. Do for $i = i_m$ to J_{max}

    (a) $U_i = U_{i+1}$

    (b) $P_u = \{ P_u : U_i\}$

    (c) $\beta_i = \beta_{i+1}$

    EndDo i

14. In N_v (Updating the pending list by forming candidate entries; processing done with present entry)

15. Do for d = 1 to n

    (a) If $K_1 \neq 1$, (If LSB of N_v is 0 add 1 to the present entry)
    {
    i. $\mu = \{ N_v : 1\}$
    ii. if μ is not in P_u ,
    iii. $U_i = \mu$ for $i = J_{max}$
    iv. $\Delta = J_{max}$
    v. $\beta_{J_{max}} = \beta_{J_{max}} + 1$
    vi. $F = 1$
    }

    (b) If $K_d + 1 < K_{d+1}$
    {
    If F =1
    {
    i. Form μ with K_d replaced by K_d +1 in N_v
    ii. If μ is not in P_u ,

iii. $U_{\Delta+1} = \mu$
iv. $\beta_{\Delta+1} = \beta_v$
}
Else
{

15.2.5 Form μ with K_d replaced by K_d +1 in N_v
15.2.6 If μ is not in P_u
15.2.7 $U_\Delta = \mu$
15.2.8 $\beta_\Delta = \beta_{J_{max}}$

}

16. Increment Δ
}
17. $J_{max} = J_{max}$ +1
18. EndDo d
19. $P_L = P_u$ for i =1 to Δ (Pending list updated)
20. $F = 0$.
21. EndDo v

5 Example

Extensive simulation with a variety of binary cyclic block codes like (15, 7), (31, 16), and (127, 64) have been carried out; in each case a few thousand transmissions over an AWGN channel was done for different values of σ and the erroneous received words decoded using the proposed algorithm. In all the cases the resultant target codeword was found to be the most reliable one. Details of a representative set are discussed briefly here.

A (15, 7) binary cyclic block code with $d_{min} = 5$ and t $= \lfloor d_{min}/2 \rfloor = 2$ is considered. The message 030h has been encoded as 304eh and transmitted over an AWGN channel with σ= 0.8. The received word is 78cch. There are 4 errors at bit positions 1, 7, 11, 14. With the input as the set {-log $s[k]$}, $n = 15$ and the Initial Condition as:$N_0 = \{0\}$ $N_1 = \{1\}$, $N_2 = \{2\}$, $P_L = \{T_1, T_2\}$ where $T_1 = \{3\}$, T_2= {1, 2},$J_{max} = 2$, $F = 0$, following the steps of the algorithm the RLL was formed. A representative segment of the RLL is given in Table 1. The table gives the rank v, k value, M_{min} , and the contents of T_i with the corresponding M_{v_i} (both separated by a hyphen in the table).

As seen in Table 1 in each row the specific T_i in italics has the minimum value of M_{v_i} and forms the candidate identified for the next entry. Further in each row the entries in bold represent the updated candidates that are formed by applying the rules in row 15 in the algorithm given above in section 4. For example considering the row with rank 82, the entry T_{36}= {*2,3,6,7 - 0.977*} - with k=2, k = 3, k = 6, k = 7 as 1ι's and Σ log $s[k]$ as 0.977 - is the next entry with rank as 83 ; and this is the one with minimum of M_{v_i}. The entries in the same row with rank 82, namely T_{38} = {**1,4,6,7 - 1.198**} and T_{39} = {**1,3,6,8 - 1.4992**}are the ones formed on applying the rules in rows 15.1 and 15.2.1 of the algorithm given in section 4 respectively.

Table 1: Partial RLL

Rank v	K	M_{min}	Details of T_i's in the Updated pending list P_L
1	1	0.0832	
2	2	0.0957	T_1= {*3 - 0.1223*}, T_2= {1,2 - 0.1789}
3	3	0.1223	P_L= { T_1, T_2, T_3}; T_1= {*1,2 -0.1789*}, T_2= {**1,3 -0.2055**}, T_3= {**4,-0.3558**}
4	1,2	0.1789	T_1= {*1,3 - 0.2055*}, T_2= {4 - 0.3558}
5	1,3	0.2055	T_1= {4 - 0.3558}, T_2= {**2,3 - 0.218**}, T_3= {**1,4 - 0.439**}
.	.	.	.
.	.	.	.
82	1,3,6,7	0.9645	T_1= {1,8 - 0.9978}, T_2= {2,8 - 1.0103}, T_3= {3,8 - 1.0369}, T_4= {1,2,8 - 1.0935}, T_5= {1,3,8 - 1.1201}, T_6= {2,3,8 - 1.1326}, T_7= {1,2,3,8 - 1.2158}, T_8= {4,8 - 1.2704}, T_9= {5,8 - 1.2727}, T_{10}= {6,8 - 1.2937}, T_{11}= {1,4,8 - 1.3536}, T_{12}= {1,5,8 - 1.3559}, T_{13}= {2,4,8 - 1.3661}, T_{14}= {2,5,8 - 1.3684}, T_{15}= {1,6,8 - 1.3769}, T_{16}= {2,6,8 - 1.3894}, T_{17}= {3,4,8 - 1.3927}, T_{18}= {4,5,6 - 1.093}, T_{19}= {4,5,7 - 1.0938},T_{20}= {3,5,8 -1.395}, T_{21}= {4,6,7 - 1.1148}, T_{22}= {3,6,8 - 1.416}, T_{23}= {1,2,4,8 - 1.4493}, T_{24}= {1,2,5,8 - 1.4516}, T_{25}= {1,2,3,4,5 - 1.0151}, T_{26}= {1,2,6,8 - 1.4726}, T_{27}= {1,9 - 1.0262}, T_{28}= {10 - 1.0421}, T_{29}= {1,2,3,4,6 - 1.0361}, T_{30}= {1,2,3,4,7 - 1.0369}, T_{31}= {2,3,4,8 - 1.4884}, T_{32}= {1,2,3,5,6 - 1.0384}, T_{33}= {2,4,5,6 - 1.1887}, T_{34}= {1,2,3,5,7 - 1.0392}, T_{35}= {2,4,5,7 - 1.1895}, T_{36}= *{2,3,6,7 - 0.977}*, T_{37}= {2,3,5,8 - 1.4907}, **T_{38}= {1,4,6,7 - 1.198 }, T_{39}= {1,3,6,8 - 1.4992 }**
83	2,3,6,7	0.977	T_1= {1,8 - 0.9978}, T_2= {2,8 - 1.0103}, ...T_{37}= {1,4,6,7 - 1.198 }, T_{38}= {1,3,6,8 - 1.4992 },**T_{39}={1,2,3,6,7 - 1.0602}, T_{40}={2,4,6,7 - 1.2105}, T_{41}={2,3,6,8 - 1.5117}.**

With each such entry to the RLL the corresponding candidate word is formed by complementing the bits indicated by the entry. The candidate word is examined to check if it is a codeword; if yes it is the desired target codeword. Once the target codeword is identified the process of filling the RLL can be discontinued since any codeword found below this in the RLL is less reliable. For the specific case here the target code word is obtained at the rank of 83 with errors at bits 7, 1, 11, and 14 with the corresponding K values as 2, 3, 6, and 7 respectively.

6 Conclusion

A structured, iterative, and reliability based Soft Decision Decoding algorithm is proposed. The algorithm uses the reliability value of the received word as a soft metric and outputs the desired target codeword through an ordered search in the 2^n word space. The algorithm identifies an entry in the RLL such that the corresponding word is the most reliable yet. For each entry in the RLL the corresponding word formed has to be checked to see if it is a code word and if so it is the desired target codeword. The proposed algorithm guarantees to return *the best* in terms of reliability because of the very fact that any other codeword found below the identified one will be less reliable. The structured algorithm can be easily coded and used for decoding any (n, k) binary cyclic block code.

Bibliography

[1] Wenyi Jin and Marc.P.C.Fossorier,Fellow,IEEE, Reliability-Based Soft-Decision Decoding with Multiple Biases , *IEEE Trans. Inform. Theory*, Vol. 53, pp. 105-120, Jan. 2007

[2] Ye Liu, Member, IEEE, Shu Lin, Fellow, IEEE, and Marc.P.C.Fossorier,Fellow,IEEE, MAP algorithms for Decoding linear block codes based on sectionalized Trellis diagrams, *IEEE Trans. Communications*, Vol. 48, pp. 577-587, Apr. 2000

[3] Yuansheng Tang, Member, IEEE, San Ling and Fang - WeiFu, On the Reliability - Based Soft - Decision Decoding Algorithms for Binary Linear Block Codes, *IEEE Trans. Inform. Theory*, Vol. 52, pp. 328-335, Jan. 2006

[4] David Chase, Member, IEEE, A Class of Algorithms for Decoding Block Codes With Channel Measurement Information, *IEEE Trans. Inform. Theory*, Vol. IT-18, pp.170-182, Jan 1972

[5] G. D. Forney, Jr., Generalized minimum distance decoding, *IEEE Trans. Inform. Theory*, Vol. IT-12, pp. 125-131, Apr. 1966

[6] Marc P. C. Fossorier, Member, IEEE, and Shu Lin, Fellow, IEEE, Soft-Decision Decoding of Linear Block Codes based on Ordered Statistics,*IEEE Trans. Inform. Theory*, Vol. 41, pp. 1379-96, Sep 1995