International Journal of Computer and Communication Technology

Volume 6 | Issue 4 Article 3

October 2015

DESIGN OF TCM DECODERS USING OPTIMIZED VITERBI DECODER

D. KOKILANAYAKI
Sasurie academy of Engineering, Coimbatore, kokilanayaki@yahoo.com

DESIGN OF TCM DECODERS USING OPTIMIZED VITERBI DECODER

D.KOKILANAYAKI

Sasurie academy of Engineering, Coimbatore.
Email: kokilanayaki@yahoo.com

Abstract-The use of forward error correcting codes in the digital communication is used to overcome the data corruptions. Although widely-used, the most popular communications decoding algorithm, the Viterbi algorithm, requires an exponential increase in hardware complexity to achieve greater decode accuracy. The Viterbi decoder that forms the dominant module of TCM decoders are responsible for the overall power consumption. We have analyzed the original T-algorithm with that of the precomputation architecture's of the T-algorithm. The T-algorithm reduces the complexity of the original viterbi algorithm and reduces the power consumption by 70%.This can be done with the reduction in the negligible clock speed.

General Terms-*Trellis coded modulation (TCM), viterbi decoder, T-Algorithm*

1. INTRODUCTION

Covolutional codes that are widely used in wireless communication system allows for efficient soft decision decoding. Trellis coded modulation (TCM) schemes that are used in many bandwidth-efficient systems, employs a high-rate convolutional code, which leads to a high complexity of the Viterbi decoder (VD) for the TCM decoder, even if the constraint length of the convolutional code is moderate. Viterbi decoder is the heart of the TCM modulation. Viterbi decoder uses various efficient algorithms in order to select the shortest path from trellis. A Viterbi decoder is an important target for power reduction in many low-power communications devices. It can account for more than one-third of power consumption during baseband processing in current generation cellular telephones.

The power consumption in Viterbi decoder is achieved using various algorithms. The Reduced state sequence decoding (RSSD) [1] is one of such approach. RSSD reduces state trellis of a channel by combining states into classes. RSSD is in general not as efficient as the M-algorithm [2],[3].In M-algorithm states examined should be drawn according to their probabilities to be the correct path. In extreme case, we can just sort the states according to such probabilities, and only visit a pre-specified fixed number of those whose being-the-correct-path probabilities are higher. This is exactly the idea behind the M algorithm. Only M most likely states are kept, and the remaining states are deleted. At the end of the trellis, the M algorithm selects the best path with the largest path metric as the Viterbi algorithm does, and outputs the respective path labels of the located best path. In practical implementation, however the T-algorithm [4],[5] is more popular than the M-algorithm for two reasons: 1) It can adaptively adjust the number of purged states according to run-time channel conditions, and 2) it has lower computational complexity since the T-algorithm only needs to find out the optimal PM (maximum or minimum value, depending on different implementations), while the M-algorithm involves a sorting process for values. However, the T–algorithm still requires the search of the optimal PM in the ACS loop. The extra comparison operation will affect the clock speed of the entire design

2. VITERBI DECODER

The viterbi decoder is used to reconstruct the original transmitted signal from the encoded noisy signal. The functional block diagram of Viterbi decoder mainly comprises of three blocks: 1) Branch metric unit (BMU) 2) Path Metric Unit (PMU) 3) Survivor path Memory unit (SMU). The BMU computes the metric between the received noisy symbol and the output symbol of the state transition. This can be calculated using either the hamming distance or the Euclidean distance. The path metric unit combines with Adder Compare and Select Unit (ACSU).The branch metrics(BMs) are passed on to the ACSU unit. The ACSU recursively computes the path metrics by adding the branch metrics to the state metrics of the previous time instant. The smaller one of the two is selected to be the new state metric for each state. After that, the decision bits are stored in and retrieved from the SMU in order to decode the source bits along the final survivor path. The PMs of the current iteration are stored in the PM unit (PMU).

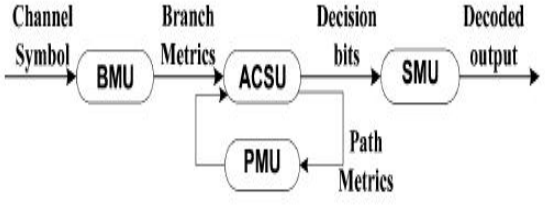

Fig 1: Functional diagram of Viterbi decoder

There are two basic schemes for SMU implementation:
The register-exchange (RE) scheme and the trace-back (TB) scheme. Because of the maximum-likelihood (ML) path searching in SMU, the VD usually causes a decoding latency proportional to the constraint length. The RE scheme is well suited for low-latency high-speed applications, while the TB scheme consumes less power. Although there exist extensive works on reducing the latency of the TB scheme, none of them could achieve the latency of the RE scheme, even at the expense of a huge power consumption overhead. Another advantage of using the RE scheme is that the -algorithm on PM can be efficiently exploited in RE design.

3. VITERBI ALGORITHM

Decoding of convolutional codes are done efficiently by viterbi algorithm.Viterbi algorithm is Maximum likelihood algorithm(ML), that is a recursive sequential minimization algorithm that can be used to find the least expensive way to route symbols from one edge of a state diagram to another. The encoder for a rate 1/2 code generates two output codeword bits as a function of the input information bits and the encoder state (stored bits in the registers). The output bits are then transmitted over a noisy channel. The Viterbi algorithm estimates the most likely sequence of encoder state transitions given the received noisy samples.

The Viterbi algorithm solves the minimization problem by applying recursively equation,

$$PM[j](t) = \min (PM[i](t-1) + BM[i, j](s))$$

PM [j](t) is the path metric associated to the minimum cost path leading to state j at time t. BM [i, j](s) is the branch metric associated to the transition from state i to state j. BM[i,j](s) is the distance between the received symbol s and the symbol associated to that transition from state i at time t-1 to sate j at time t. .The use of exhaustive search makes the Viterbi decoder essentially not power efficient, particularly for applications demanding large trellises. In contrast, limited search trellis decoding algorithms perform limited (or nonexhaustive) search, as suggested by the name, on the original trellis. They have much less computational complexities than the VA, yet still achieve ML or near-ML performance. One such algorithm being the T-algorithms perform nonexhaustive breadth-first search, where the number of survivor paths at each decoding depth is typically much less than the total number of trellis states, leading to much less computational complexities.

4. T-ALGORITHM

The T-algorithm belongs to the family of breadth-first decoding algorithms. The T-algorithm is similar to VA except that the number of the survivor paths is not constant. Unlike the traditional VA which retains all the 2^{k-1} states in every trellis stage, only some of the most-likely paths are kept at each stage in the T-algorithm. Every surviving path at the trellis stage is expanded and its successors at stage are kept if their corresponding path metric values are smaller or equal to dm +T, where T is a preset pruning threshold determined by user and dm is the best path metric of all the survivor states. By setting a proper threshold T, significant amount of the paths are pruned, while the BER performance is maintained. In the T-algorithm a threshold is set, and the difference between each PM and the optimal one is calculated. The algorithm compares the differences with T; only the states with a difference that is less than T survive and are used for the calculation in the next cycle. The corresponding ACS computations for the pruned paths are saved, thus the computation complexity is reduced.

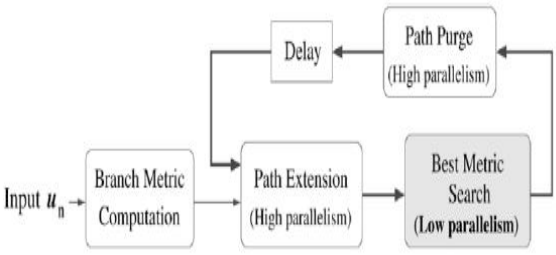

Fig 2: Data flow diagram of T-algorithm

Broadly speaking, breadth-first decoding algorithms extend all the survivor paths at each trellis depth at once, purge some paths according to certain criterion, and then continue on to the next trellis depth. Various breadth-first algorithms primarily differ on the purging rules. In the T-algorithm, at each decoding depth, all the paths whose cumulative path metric falls outside of a retention band will be purged.

5. T-ALGORITHM ON BMS

Unlike low-rate codes (e.g., 1/2 and 1/3) that have two incoming paths for each state, high-rate convolutional codes generate many more paths in state transitions (eight in the aforementioned case due to the 3 input bits). When applying Viterbi decoding on these codes, more BMs are involved (16 in the aforementioned case because of the 4 output bits). Motivated by the conventional T-algorithm, we found that eliminating redundant additions is an effective way to reduce the computational complexity.
In our case, since the number of BMs is so large, BMs also play a very important role in determining the overall complexity. Since PMs and BMs are evenly distributed in ACSU calculations, purging a BM is equivalent to purging four PMs in terms of equivalent number of additions being eliminated.

Therefore, the T-algorithm could be applied on BMs instead of PMs. The process is the same as in the conventional T–algorithm applied on PMs: first, finding the optimal BM the maximum value of all 16 BMs in our case and setting up a threshold, and then comparing the difference between each BM and the maximum value. If the difference is larger than the threshold, the corresponding BM is purged, which means that, in ACSU, the additions involving the purged BMs are not performed.

The benefit of applying the T-algorithm on BMs is obvious .In our case, finding the maximum value of 16 BMs is more efficient than searching for the maximum value of 64 PMs, not only because the number of BMs is less than the number of PMs but also due to the fact that each BM is the sum of the Euclidian metrics from four signal sets. The maximum value of BMs is equivalent to the sum of the maximum Euclidian metrics from each signal set. Finding four maximum values and then adding them is much more convenient than finding the maximum value of 16 arbitrary data. Additionally, the processes of finding the maximum value of BMs and calculating 16 BMs could take place in parallel since the maximum BM is derived directly from the Euclidian metrics. This property can be explored to eliminate the latency caused by the process of calculating the maximum value in a normal case that needs to obtain the values of all BMs first.

For the T-algorithm on BMs, when the BMs involved for calculating a new PM are all purged and the PM cannot get a valid value, the corresponding state can be treated as "purged." Again, the registers associated with the "purged" state will not be updated in SMU. in the case of the T-algorithm on PMs, whereas in the case of the T–algorithm on BMs, the number of purged computation remains almost the same, regardless of the channel condition. It may be noted that the computational complexity of the proposed T-algorithm applied on BMs looks more like that of the M-algorithm rather than the conventional T-algorithm. Although the proposed T-algorithm on BMs cannot save as many computations as the conventional T-algorithm on PMs can in the high-SNR region, it still has the advantages of lower complexity, higher throughput, and better BER performance.

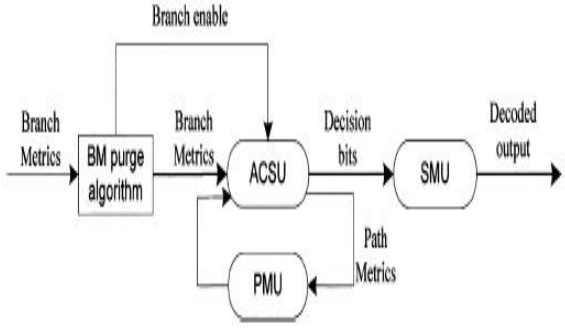

Fig 3: Functional diagram of T-algorithm on BMs

6. HYBRID T-ALGORITHM

The proposed T-algorithm on BMs not only provides an alternative to the conventional T-algorithm on PMs but also given insights leading to techniques that can further reduce computations.

From Fig. 3, we observe that implementing the T–algorithm on BMs does not require any change to the architecture of the conventional T-algorithm on PMs. This feature makes it possible to combine the two schemes together and purge more computations in ACSU. The functional diagram of the resulting scheme is shown in Fig. 4, which is called the "hybrid T-algorithm". The PM at n time slot is calculated from the sum of PMs at n-1 time slot and BMs at time slot as

$$PM_n^j = \max\Big(PM_{n-1}^{j1} + BM_n^{j1}; PM_{n-1}^{j2} + BM_n^{j2}; PM_{n-1}^{j3} + BM_n^{j3}; PM_{n-1}^{j4} + BM_n^{j4}; PM_{n-1}^{j5} + BM_n^{j5}; PM_{n-1}^{j6} + BM_n^{j6}; PM_{n-1}^{j7} + BM_n^{j7}; PM_{n-1}^{j8} + BM_n^{j8}\Big)$$

where the subscript denotes the time slot and the superscript j denotes the label of any arbitrary state. Each PM is selected from eight candidates. The addition operation is performed only if both the PM and BM involved in the operation are retained. Since the hybrid T-algorithm is developed based on the conventional T-algorithm on PMs, we keep the threshold of 0.3 for PMs. the complexity of the T-algorithm on PMs and that of the T-algorithm on BMs are also shown. Compared with the conventional T-algorithm on PMs, the computational complexity is reduced by more than 50% when the threshold is set to 0.3.

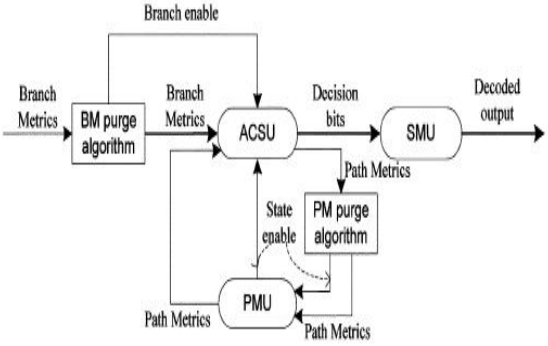

Fig 4: Functional diagram of T-algorithm on BMs

These simulation results indicate that, with the hybrid -algorithm, more calculations can be eliminated, while its BER performance is almost the same as that of the conventional T-algorithm on PMs. However, a simple straightforward combination of the conventional T-algorithm on PMs with the proposed T-algorithm on BMs will lose the advantage of speed of the latter. To retain the advantage of speed, we need a novel architecture. Here, we take

advantage of the SPEC T- algorithm proposed in [6] to reduce the length of critical path. The key idea of the SPEC T-algorithm is to use an estimated optimal PM value derived from the optimal BM value, instead of searching for the optimal PM in each cycle.

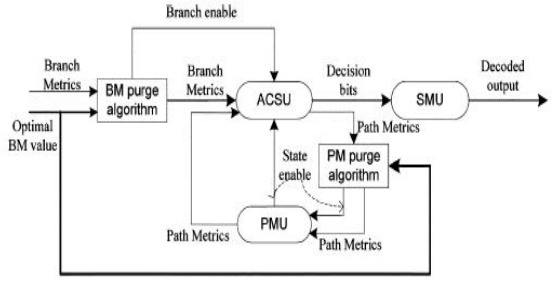

Fig 5: Hybrid T-algorithm VD with a SPEC-T algorithm.

Since the T-algorithm on BMs will generate an optimal BM anyway, very little additional logic is needed. The functional diagram of the resultant scheme is shown in Fig.5. The optimal BM value is forwarded to the PM purging unit to calculate the estimated optimal PM value. For most of the time, the VD uses the estimated optimal PM value to perform the T-algorithm on PM. In the meantime, some compensation schemes are needed to adjust the estimated optimal value regularly. A straightforward compensation scheme is to find the real optimal PM in a number of cycles and compute the estimation error for the next adjustment. The clock speed will not be affected by this searching process since it can be pipelined into several clock cycles outside the ACSU loop.

7. 2-STEP PRECOMPUTATION

In the 1-step pre-computation architecture, odd numbered states are extended by odd BMs, while even numbered states are extended by even BMs. Furthermore, the even states all extend to states with higher indices in the trellis transition, while the odd states extend to states with lower indices. This information allows us to obtain the 2-step pre-computation data path. The fig.6 shows the functional diagram of 2-step precomputation architecture.

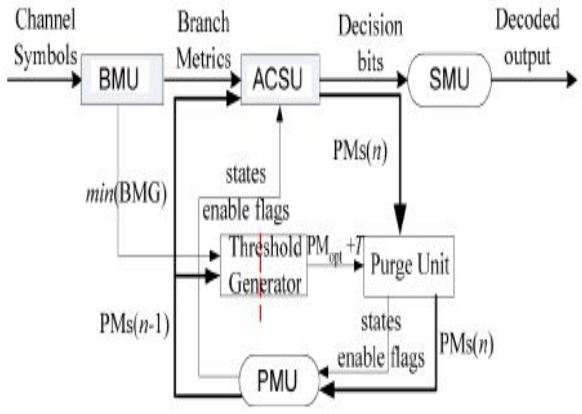

Fig 6: Functional diagram of 2-step T-algorithm

The states are further grouped into 4 clusters and the BMs are categorized in the same way

$$\text{cluster0} = \{PM_m \mid 0 \le m \le 63,\ m \bmod 4 = 0\}$$
$$\text{cluster1} = \{PM_m \mid 0 \le m \le 63,\ m \bmod 4 = 2\}$$
$$\text{cluster2} = \{PM_m \mid 0 \le m \le 63,\ m \bmod 4 = 1\}$$
$$\text{cluster3} = \{PM_m \mid 0 \le m \le 63,\ m \bmod 4 = 3\}$$

$$BMG0 = \{BM_m \mid 0 \le m \le 15,\ m \bmod 4 = 0\}$$
$$BMG1 = \{BM_m \mid 0 \le m \le 15,\ m \bmod 4 = 2\}$$
$$BMG2 = \{BM_m \mid 0 \le m \le 15,\ m \bmod 4 = 1\}$$
$$BMG3 = \{BM_m \mid 0 \le m \le 15,\ m \bmod 4 = 3\}$$

The optimal PM at time *n* is calculated as

$$\begin{aligned}PM_{opt}(n) = min\ [&\\ &min\ \{min\ (\text{cluster0}\ (n\text{-}2)) + min\ (BMG0(n\text{-}1)),\\ &\quad min\ (\text{cluster1}\ (n\text{-}2)) + min\ (BMG1(n\text{-}1)),\\ &\quad min\ (\text{cluster2}\ (n\text{-}2)) + min\ (BMG3(n\text{-}1)),\\ &\quad min\ (\text{cluster3}\ (n\text{-}2)) + min\ (BMG2(n\text{-}1))\\ &\}+ min\ (\text{even BMs}(n)),\\ &min\ \{min\ (\text{cluster0}\ (n\text{-}2)) + min\ (BMG1(n\text{-}1)),\\ &\quad min\ (\text{cluster1}\ (n\text{-}2)) + min\ (BMG0(n\text{-}1)),\\ &\quad min\ (\text{cluster2}\ (n\text{-}2)) + min\ (BMG2(n\text{-}1)),\\ &\quad min\ (\text{cluster3}\ (n\text{-}2)) + min\ (BMG3(n\text{-}1))\\ &\}+ min\ (\text{odd BMs}(n))\\ &\ ].\end{aligned}$$

Calculating PMopt(*n*) from PM(*n*-2) means that the calculation can be completed within 2 cycles . Thus, the process is pipelined as two stages as indicated by the dashed line in Fig.7. Again, we need to exam the critical path of each stage. The critical paths of the first stage (left side of the dashed line in Fig.7) and the second stage (right side of the dashed line) are given by

$$\mathbf{T\ (stage1)_{2\text{-}step\text{-}pre\text{-}T} = T_{adder} + 2T_{4\text{-}in\_comp}}$$

$$\mathbf{T\ (stage\ 2)_{2\text{-}step\text{-}pre\text{-}T} = 2T_{adder} + T_{4\text{-}in\_comp} + 2T_{2\text{-}in\_comp}}$$

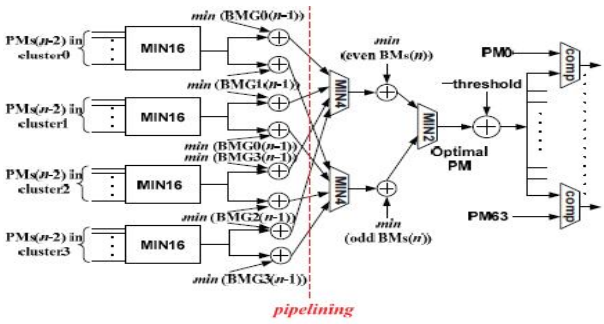

Fig 7: 2-step pre-computation *T*-algorithm with redundant operations.

The ACSU using several different schemes: 1) Full trellis, 2) T-algorithm on BMs, 3) SPEC-T algorithm 4) Hybrid T- algorithm and 5) 2-step pre-computation are compared. The synthesis results are summarized in Table I.

Table 1. Table captions should be placed above the table

	Number of 4 input LUTs	**Number of flip flops**	**Max speed(MHz)**
Full trellis	18522(13%)	576(0%)	103.8
T-algorithm on BMs	18626(13%)	576(0%)	42.5
SPEC-T algorithm	21716(16%)	1842(1%)	90.8
Hybrid-T algorithm	22342(16%)	1842(1%)	88.2
2-step pre-computation	22258(16%)	686(0%)	446.4(11%)

Table 2.Power Estimation Results

	Power(mw)	
Full trellis	21.473(100%)	
2-step pre-computation	Tpm=0.75	20.069

REFERENCES

[1] J. B. Anderson and E. Offer, "Reduced-state sequence detection with convolutional codes," *IEEE Trans. Inf. Theory*, vol. 40, no. 3, pp.965–972, May 1994.

[2] C. F. Lin and J. B. Anderson, "M-algorithm decoding of channel convolutional codes," presented at the Princeton Conf. Info. Sci. Syst.,Princeton, NJ, Mar. 1986.

[3] J. B. Anderson and S.Mohan, "Senquential coding algorithm:A survey and cost analysis," *IEEE Trans. Commun.*, vol. COM-32, no. 6, pp. 169–176, Feb 1984.c

[4] S. J. Simmons, "Breadth-first trellis decoding with adaptive effort," *IEEE Trans. Commun.*, vol. 38, no. 1, pp. 3–12, Jan. 1990.

[5] Chan, F. and Haccoun, D., "Adaptive Viterbi decoding of convolutional codes overmemoryless channels," *IEEE Trans. commun.*, vol. 45, no. 11, pp. 1389–1400, Nov. 1997.

[6] F. Sun and T. Zhang, "Parallel high-throughput limited search trellis decoder VLSI design," *IEEE Trans. Very large Scale Integr. (VLSI) Syst.*, vol. 13, no. 9, pp. 1013–1202, Sep. 2005.