VLSI Architectures for Metric Normalization in the Viterbi Algorithm

C. Bernard Shung
Paul H. Siegel
Almaden Research Center

Gottfried Ungerboeck
Zurich Research Lab

Hemant K. Thapar
General Product Division

IBM Corporation

Abstract

In the realization of Viterbi decoders with finite precision arithmetic, the values of the survivor metrics computed by the Add-Compare-Select (ACS) recursion must remain within a finite numerical range to avoid catastrophic overflow (or underflow) situations. In this paper, we compare several metric normalization techniques which are suitable for VLSI implementations with fixed-point arithmetic. The *modulo* technique is found to be the most *local* and *uniform* approach. An efficient VLSI design of the ACS units based on this technique is discussed.

1 Introduction

The Viterbi algorithm is widely used in communications applications to derive maximum-likelihood sequence estimates of transmitted data on channels with intersymbol interference (ISI) and/or coding. Based upon a trellis description of the channel outputs, the algorithm recursively computes for each state of the trellis the best fit to the received (noisy) sequence among the trellis sequences which end in the specific state. These "best fit" sequences are called "survivor sequences", and associated with each is a measure of how well the survivor matches the received sequence in the sense of a prescribed measure, e.g. squared Euclidean differences.

This numerical measure is called the "survivor metric". At the heart of the algorithm are the Add-Compare-Select (ACS) operations which for each new state in the trellis perform the following functions:

1. The survivor metric of each preceding state with a transition to the new state is augmented by the "branch metric increment" corresponding to this transition, which represents the squared difference between the received sample(s) and the branch label.
2. The resulting metrics are compared to determine which augmented metric is smallest, thus indicating the best fit to the received sample sequence.
3. The smallest metric is selected as the metric for the new state.

The steps mentioned above correspond to minimizing the negative log likelihood function, which is proportional to the squared Euclidean distance, under the assumption of independent, identically-distributed, additive Gaussian noise. An alternative metric definition [9] based on the log likelihood function would lead to selecting the *largest* metric as the metric for the new state. In digital implementations, where metrics are represented as binary fixed-point values in finite-length registers, metric normalization is required to prevent errors due to overflow/underflow during the updating and storage of the metrics. Several methods of metric normalization have been proposed and used in the past. Among these are:

1. **Reset:**

   Redundancy is introduced into the channel input sequence in order to force the survivor sequences to merge after some number N of ACS recursions for each state.
2. **Difference Metric ACS:**

   The Viterbi algorithm is reformulated to keep track only of differences between metrics for each pair of states.
3. **Variable Shift:**

   After some fixed number N of recursions, the minimum survivor metric is subtracted from all of the survivor metrics [9].
4. **Fixed Shift:**

   When all survivor metrics become negative (or all pos-

CH2829-0/90/0000-1723 $1.00 © 1990 IEEE

itive), the survivor metrics are shifted up (or down) by a fixed amount [3].

5. **Modulo Normalization:**

   Use two's complement representation of branch and survivor metrics and modular arithmetic during ACS operations.

Methods 2 through 5 attempt to exploit the fact that, despite the potential unbounded magnitude of survivor metrics, the differences between them remain bounded in magnitude by a fixed quantity Δ, independent of the number of ACS recursion iterations. A bound which has been used [1,4,10] is of the form $2nB$ where n is the minimum number of stages to ensure complete connectivity among the trellis states, and B is the upper bound on the branch metrics. Recently, an improved method for exact calculation of Δ and related bounds for the "extended survivor metrics" has been developed [6].

In section 2, we review the normalization techniques 1-4 mentioned above. These techniques are found to be *global* and/or *non-uniform* and hence unsuitable for VLSI implementation. In section 3, we review the modulo normalization technique [4]. As mentioned in [4], this technique seemed to be known among several experts in the field without being published; it has in fact been used in research prototypes since 1979, and in commercial modem products since 1985 [8]. In [4], the proposed technique cannot be mapped directly into ACS hardware; instead, a subtractor has to be used in place of a comparator. We'll show in section 4 a modified comparison rule that results in a more efficient ACS implementation of the modulo normalization technique, which is shown to be *local* and *uniform* and hence the most attractive choice.

2 Normalization Techniques 1-4

In VLSI implementation of the normalization techniques, the two most desirable properties are *locality* and *uniformity*. Metric normalization is considered local if it is accomplished within each ACS without information from other ACS's. It is considered uniform if the ACS operation is not interrupted to perform metric normalization. Locality minimizes the global signal communications and hence reduces the wiring area. Uniformity not only simplifies the control but also minimizes the variety of required functional units. Moreover, a non-uniform technique usually reduces the achievable data rate. The following comparison of the first four normalization techniques focuses on these two properties.

2.1 Reset

The reset technique periodically resets the encoder to a *ground* state. It was proposed [5] for generating independent data blocks from the same data source to exploit the decoder concurrency. Without loss of generality, let the state 0 be the ground state. Reset can be accomplished by (1) shifting in zeros in case of a convolutional encoder or (2) forcing the encoder into state 0 in case of a finite-state machine encoder. The resultant trellises are shown in Figure 1(a)(b) [5]. The reset technique is relatively simple, but not uniform. Moreover, the precision required in the ACS depends on the reset period, unless another normalization technique is employed within each period.

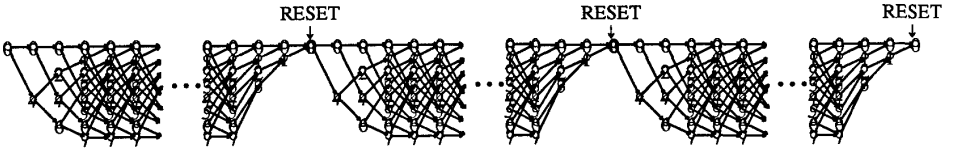

Figure 1(a). Trellis Diagram with Zero-shift

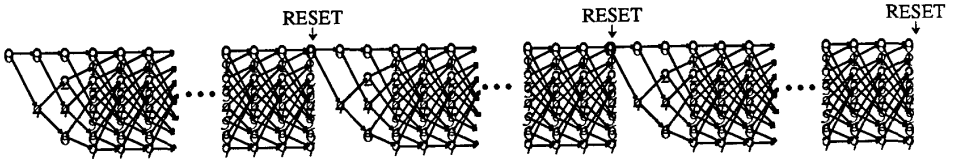

Figure 1(b). Trellis Diagram with Abrupt Reset

2.2 Difference Metric ACS

The Viterbi algorithm is reformulated to keep track only of differences between metrics for each pair of states. Based on the "bounded difference" property of the Viterbi algorithm, the technique renders metric normalization unnecessary. It was proven useful for binary partial response channels [2], such as $1 \pm D$. However, as Ferguson concluded [2], "The extension to multilevel signaling destroys the beauty and simplicity of the binary scheme". In general this technique is difficult to formulate and offers no clear advantage when the number of trellis states exceeds 2.

2.3 Variable Shift

After some fixed number N of recursions, the minimum survivor metric is subtracted from all of the survivor metrics. Its ACS architecture is shown in Figure 2. A subtractor is required to subtract the minimum survivor metric. If metric normalization is performed in every trellis stage, then the subtractor increases the ACS delay. If metric normalization is performed occasionally by interrupting the ACS operation, which results in a variable rate decoder, then the control becomes more complex.

This technique is non-uniform and extremely global, but is probably also the most frequently used. In addition to the hardware penalty for the subtractors and minimum metric selection, it calls for a significant amount of signal communications to select the minimum survivor metric, and to distribute the minimum metric to all ACS units to be subtracted.

2.4 Fixed Shift

When the Hamming distance is used as branch metric, the survivor metrics tend to increase. On the other hand, when the Euclidean distance (with the square of the received sample term cancelled) is used as branch metric, the survivor metrics tends to decrease. This technique shifts all the survivor metrics up (or down) by a fixed amount when all the survivor metrics become negative (or all positive).

It can be implemented by inspecting the sign bit of all survivor metrics and resetting (or presetting) all the sign bits when all of them become 1 (or 0). The ACS architecture is shown in Figure 3. Compared to the Variable Shift, this method requires much less global signal communications. However, because the sign bit monitoring and the subsequent fixed shifts are performed within the ACS recursion, this approach reduces the data rate that can be

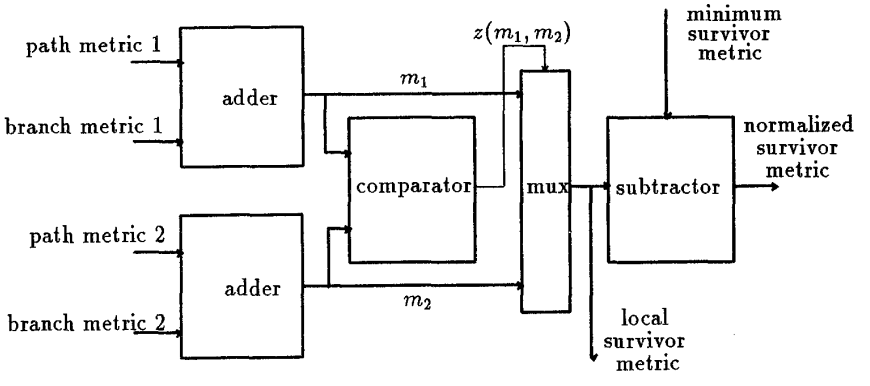

Figure 2. ACS Architecture for Variable-Shift Normalization

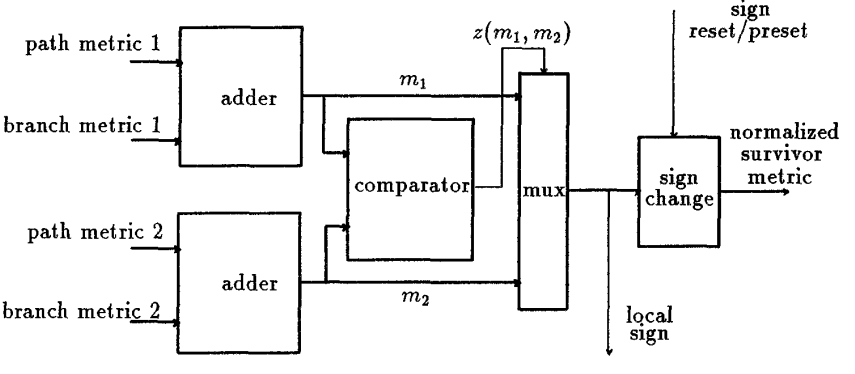

Figure 3. ACS Architecture for Fixed-Shift Normalization

achieved. Moreover, this technique requires one extra bit in precision.

3 Modulo Normalization

In modular arithmetic the metric m_j is replaced by the normalized metric $\overline{m}_j \equiv (m_j + \frac{C}{2}) \pmod{C} - \frac{C}{2}$, as in Figure 4. One can think of the quantities $\{\overline{m}_j\}$ as positions on the circumference of a circle with diameter $\frac{C}{\pi}$, with $\overline{m}_j$ at angle $\frac{2\pi m_j}{C}$. Note that $-\frac{C}{2} \leq \overline{m}_j < \frac{C}{2}$. The normalized metrics are obtained by wrapping the real line around the circle, with the real number 0 mapped to the point at angle 0 on the circle, and the direction of increasing real numbers corresponding to counterclockwise motion around the circle. It was found [4,8] that the differences of the survivor metrics remain unchanged using modular arithmetic provided that $\Delta \leq \frac{C}{2}$, where Δ is the bound of the differences between survivor metrics.

Proposition: (its proof is straightforward, as can be seen from Figure 4): Let m_1, m_2 be real numbers with

$$|m_1 - m_2| < \frac{C}{2}.$$

Let α be the angle swept out by counterclockwise motion along the arc from the projection $\overline{m}_1$ to $\overline{m}_2$. Then,

$$m_1 < m_2$$

if and only if

$$\alpha < \pi.$$

□

We can interpret the proposition in less mathematical terms by thinking of the survivor metrics as "runners in a race". The constraint that the difference between metrics is less than half of the circumference is like saying it is a very competitive race. At all times all the runners are running in one half of the circumference. Therefore, it is very easy to determine which one is leading.

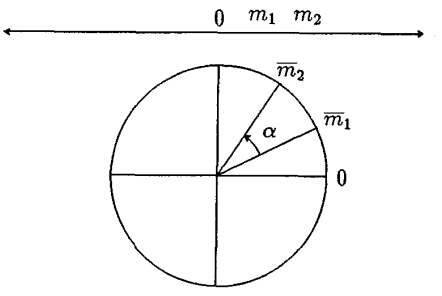

Figure 4. Modulo comparison of normalized metrics

The modular arithmetic can be implemented by 2's-complement adders and subtractors [4,8]. In particular, a subtractor is used in place of a comparator as in a normal ACS. This is based on the observations that (1) 2's complement subtraction produces α, the angle between $\overline{m}_1$ and $\overline{m}_2$, and (2) $\overline{m}_1 - \overline{m}_2 = m_1 - m_2$ if $|m_1 - m_2| < \frac{C}{2}$. The sign bit (1 if $\alpha > \pi$, 0 otherwise) can be used to drive the multiplexer to perform the selection.

The VLSI architecture of an ACS using modulo normalization is shown in Figure 5. This technique normalizes the survivor metrics locally (inside the 2's complement adder and subtractor) and uniformly (no interruption for metric normalization). The penalty is the extra bit required in the ACS because $C = 2\Delta$. In the next section we will present a more efficient ACS architecture for modulo normalization. It is based on a *modified comparison rule* because direct (either signed or unsigned) comparison cannot determine the relative order of $\overline{m}_1$ and $\overline{m}_2$ due to the modular arithmetic.

4 Modified Comparison Rule

Let $z(\overline{m}_1, \overline{m}_2)$ denote the logical result of the comparison of the survivor metrics m_1 and m_2, given by

$$z(\overline{m}_1, \overline{m}_2) = \begin{cases} 1, & \text{if } m_1 \leq m_2 \\ 0, & \text{otherwise.} \end{cases}$$

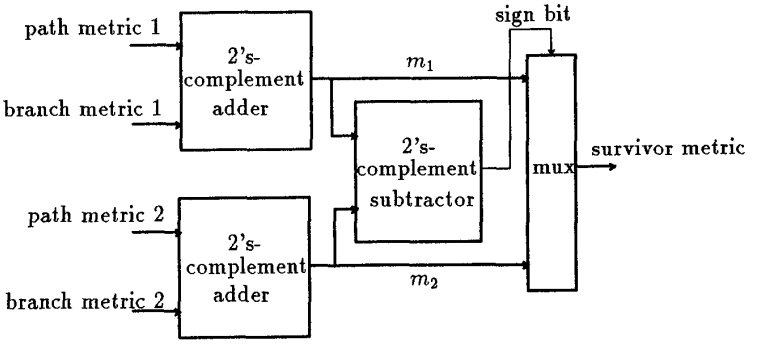

Figure 5. VLSI architecture for Modulo Normalized ACS

Let $\overline{m}_1 = (\overline{m}_{1p}, \ldots, \overline{m}_{10})$ and $\overline{m}_2 = (\overline{m}_{2p}, \ldots, \overline{m}_{20})$ be the two's-complement representations of the normalized metrics $\overline{m}_1$ and $\overline{m}_2$. Or

$$\overline{m}_1 = \sum_{j=0}^{p} \overline{m}_{1j} 2^j$$

$$\overline{m}_2 = \sum_{j=0}^{p} \overline{m}_{2j} 2^j$$

Let $\hat{m}_j \equiv m_j \pmod{\frac{C}{2}}$, or

$$\hat{m}_1 = \sum_{j=0}^{p-1} \overline{m}_{1j} 2^j$$

$$\hat{m}_2 = \sum_{j=0}^{p-1} \overline{m}_{2j} 2^j$$

Let $\oplus$ denote the exclusive-or operation.

Modified Comparison Rule:

$$z(\overline{m}_1, \overline{m}_2) = \overline{m}_{1p} \oplus \overline{m}_{2p} \oplus y(\hat{m}_1, \hat{m}_2)$$

where $y(.,.)$ denotes the *unsigned* comparison. In words, $z(\overline{m}_1, \overline{m}_2)$ equals $y(\hat{m}_1, \hat{m}_2)$ (or logical inverse of $y(\hat{m}_1, \hat{m}_2)$) if $\overline{m}_1$ and $\overline{m}_2$ have the same (or opposite) sign.

As an example, in Figure 6, $\overline{m}_1$ and $\overline{m}_2$ have opposite sign, $\hat{m}_1 (= \overline{m}_1) > \hat{m}_2 (= \overline{m}_2 - \frac{C}{2})$, $y(\hat{m}_1, \hat{m}_2) = 0$. From the modified comparison rule, we conclude that $z(\overline{m}_1, \overline{m}_2) = 1$, or $m_1 < m_2$. Since $\alpha < \pi$, the proposition confirms this conclusion.

The VLSI architecture for an ACS which implements modulo normalization with modified comparison rule is shown in Figure 7. In general the comparator is smaller and faster than the subtractor (Figure 6). In addition, the computation of $y(\hat{m}_1, \hat{m}_2)$ and $\overline{m}_{1p} \oplus \overline{m}_{2p}$ can be performed in parallel. Thus the implementation based on the modified comparison rule further removes one-bit delay. As a result, the VLSI architecture for ACS based on the modified comparison rule is more efficient in *area* and *speed* than the one based on 2's-complement subtraction.

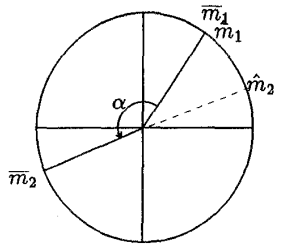

Figure 6. An Example for the Modified Comparison Rule

The modulo normalization technique with the modified comparison rule has been used in the VLSI design of a Viterbi decoder for partial-response signals encoded with a rate-8/10 trellis code [7].

5 Conclusion

Five metric normalization techniques and their implementations are described and compared. The modulo normalization technique offers the most *local* and *uniform* architecture and hence is the most suitable for VLSI implementation. The modified comparison rule is found to produce a more efficient ACS architecture than previous results based on subtraction.

Acknowledgment

The authors wish to thank C. Cox and F. Dolivo for many helpful discussions.

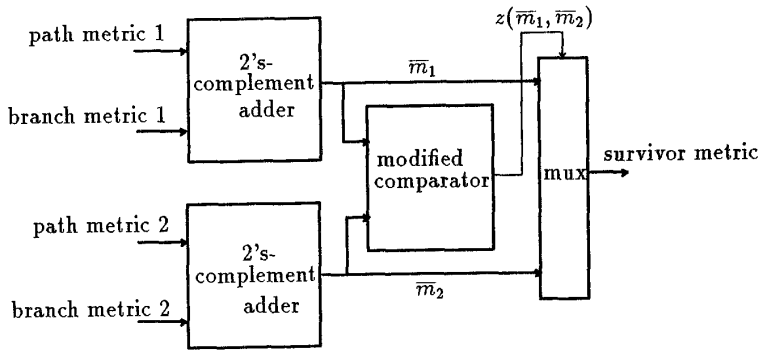

Figure 7. VLSI Architecture for Modulo Normalized ACS with Modified Comparison Rule

References

[1] G. C. Clark and J. B. Cain. *Error Correction Coding for Digital Communication.* Plenum Press, New York, 1981.

[2] M. J. Ferguson. Optimal reception for binary partial response channels. *Bell System Technical Journal*, 51(2):493–505, February 1972.

[3] G. D. Forney Jr. The Viterbi algorithm. *Proc. IEEE*, 61:268–278, Mar 1973.

[4] A. P. Hekstra. An alternative to metric rescaling in Viterbi decoders. *IEEE Trans. Communications*, 37(11):1220–1222, November 1989.

[5] H-D Lin. *Architectures for Viterbi Decoding and ML Estimators for Controllable Markov Sources.* Master's thesis, University of California at Berkeley, May 1988.

[6] C. B. Shung, P. H. Siegel, T. D. Howell, and H. K. Thapar. Exact bounds on metric differences in Viterbi algorithms. In preparation.

[7] C. B. Shung, P. H. Siegel, H. K. Thapar, and R. Karabed. Implementation issues for the design of a rate 8/10 trellis code chip for partial response channels. In *Proc. 3rd Workshop on ECC*, pages 213–225, San Jose, CA, Sept 1989.

[8] G. Ungerboeck. Unpublished work.

[9] G. Ungerboeck. Adaptive maximum-likelihood receiver for carrier-modulated data-transmission systems. *IEEE Trans. Communications*, 22(5):626–636, May 1974.

[10] A. J. Viterbi and J. K. Omura. *Principles of Digital Communication and Coding.* McGraw–Hill, New York, 1979.