# Transactions Letters

## Numerically Efficient Algorithm for Modified Trellis Approaches

Hichan Moon, *Member, IEEE,* and Sungroh Yoon, *Member, IEEE*

***Abstract*—In this letter, a numerically efficient algorithm is presented to compute the performance upper bounds for terminated convolutional codes based on a modified trellis diagram. A weight enumerator of one variable is defined with a new set of coefficients. The coefficients of the weight enumerator are shown to form a semiring.**

***Index Terms*—Upper bound, convolutional code, weight enumerator, semiring.**

### I. Introduction

THE performance of a Viterbi decoded convolutional code has been studied based on a transfer function [1], [2]. The conventional upper bounds are obtained on the assumption that the input sequence is infinite and that the error rate of each information bit is the same.

In many communication systems, convolutional codes are used with finite-length input sequences. It is usual to encode an information sequence of finite length followed by additional tail bits. The conventional upper bound becomes loose as the frame length decreases, since the effect of termination is not considered. Furthermore, it was shown that the bit error rate (BER) varies depending on the bit position within a frame [3].

Improved performance upper bounds have been presented for terminated convolutional codes [4], [5]. To compute the upper bounds, a weight enumerator is defined to represent the relation between the Hamming weight of the input bits and the Hamming weight of the coded outputs. In the weight enumerator, codewords composed of more than two error events are not counted, since they are not required for computing the upper bounds [6]. The weight enumerator is computed based on a modified trellis diagram.

The complexity and storage required for computing the weight enumerator are proportional to the number of states in the encoder. They are also proportional to the number of terms used for the upper bounds. The conventional weight enumerator has two variables - one variable for the Hamming weight of the input bits and another variable for the Hamming weight of the coded outputs. The required number of terms with the two dimensional weight enumerator increases, as the number of states in the encoder increases. This is because there generally are many codewords with the same output Hamming weights resulting from different inputs patterns, for a convolutional code with many states. Therefore, it is desirable to reduce the number of terms used for the bounds for a numerically efficient computation.

In this letter, a new weight enumerator of one variable is proposed to reduce the number of terms required for the same performance bounds. In the proposed weight enumerator, only one variable is used to count the Hamming weight of the coded outputs and a new set of coefficients is defined. A coefficient of the proposed enumerator is a pair of integers. Addition and multiplication operations are defined for pairs of integers such that the set of all coefficients forms a semiring. A method is presented to compute the proposed weight enumerator based on a modified trellis diagram. The same performance bounds from the conventional enumerator are obtained from the proposed enumerator with lower computation complexity and storage.

The rest of this letter is organized as follows. Section II describes the upper bounds for terminated convolutional codes computed from the conventional weight enumerator of two variables. Section III presents the performance bounds with the proposed weight enumerator of one variable. Numerical results and conclusions are given in Sections IV and V, respectively.

### II. Conventional Weight Enumerator

In this section, performance upper bounds are presented based on the modified trellis of a terminated convolutional code. A frame is assumed to be composed of $L_1$ information bits followed by $L_2$ tail bits. The input bits are encoded by a rate $k/n$ convolutional code. A rate $k/n$ convolutional code has $2^k$ branches diverging from one state.

For the upper bounds with the modified trellis approach [4], [5], the conventional weight enumerator is defined with two variables $B$ and $D$. Even though only rate $1/n$ convolutional codes are considered in [4], [5], extension to a rate $k/n$ convolutional code is easy, if the exponent of the variable $B$ is set to the Hamming weight of the input information bit(s) in each branch. A large number of terms are needed in computing the upper bounds with the conventional enumerator, since it is necessary to store two dimensional polynomials. For a convolutional code terminated by length $L_1 + L_2$, the conventional weight enumerator $W(B, D)$ is defined as

$$W(B,D) = \sum_j \left( \sum_i c_{i,j} B^i \right) D^j, \tag{1}$$

Paper approved by E. Ayanoglu, the Editor for Communication Theory and Coding Applications of the IEEE Communications Society. Manuscript received February 20, 2009; revised June 19, 2009 and August 15, 2009.

H. Moon is with the Telecom. R&D center, Samsung Electronics Co., Suwon, South Korea (e-mail: hchan.moon@samsung.com).

S. Yoon is with the School of Electrical Engineering, Korea University, Seoul, South Korea (e-mail: sryoon@korea.ac.kr).

S. Yoon was supported in part by the National Research Foundation of Korea funded by the Ministry of Education, Science and Technology (Grant No. 2009-0079888).

Digital Object Identifier 10.1109/TCOMM.2010.02.090024

where $c_{i,j}$ is the number of codewords composed of a single error event with output Hamming weight $j$ and input Hamming weight $i$. If the weight enumerator is considered as a polynomial of the variable $D$, the set of all coefficients $\sum_i c_{i,j}B^i$ forms a semiring with the operations of addition and multiplication. A semiring is an algebraic structure which satisfies all the requirements of a ring except that of the inverse for addition [7], [8].

From the weight enumerator, the upper bound on average BER is obtained as

$$P_b \leq \frac{1}{L_1}\sum_j \sum_i i \cdot c_{i,j} \cdot P_j, \tag{2}$$

where $P_j$ is the probability that a codeword at distance $j$ from the transmitted codeword is chosen at the decoder. If the coded symbols are transmitted over an AWGN (additive white Gaussian noise) channel, then $P_j$ is obtained for BPSK or QPSK modulation as

$$P_j = Q\left(\sqrt{\frac{2E_S \cdot j}{N_0}}\right), \tag{3}$$

where $E_S$ is the energy per coded symbol. The weight enumerator can be used for the upper bound on frame error rate (FER). The upper bound on FER is obtained as

$$P_{FE} \leq \sum_j \sum_i c_{i,j} \cdot P_j. \tag{4}$$

## III. Proposed Weight Enumerator

### A. Definition of Coefficients

An algebraic structure is presented for the coefficients of the proposed weight enumerator. Consider a polynomial of the following form

$$f(B) = \sum_i a_i B^i, \tag{5}$$

where $a_i$, $i = 0, 1, \cdots$ are non-negative integers. It is easy to show that all possible polynomials form a semiring with the operation of conventional addition and multiplication operations. An example of semirings is the set of all non-negative integers $Z^*$ with the addition and multiplication operations. $Z^*$ satisfies all the requirements of a ring except that of an inverse element for addition.

Consider the set of all possible pairs of non-negative integers $Z^* \times Z^* = \{(\alpha, \beta) | \alpha, \beta \in Z^*\}$. Define a mapping $\mathcal{T}$ from a polynomial of $B$ to a component $(\alpha, \beta)$ in $Z^* \times Z^*$ as

$$\begin{aligned} \mathcal{T}\{f(B)\} &= \mathcal{T}\{\sum_i a_i B^i\} \\ &= \left(\sum_i a_i, \sum_i i \cdot a_i\right) \equiv (\alpha, \beta). \end{aligned} \tag{6}$$

The first and second components of $(\alpha, \beta)$ are the number of terms and the sum of coefficients multiplied by the corresponding exponents of the polynomial $f(B)$, respectively. For example, with the mapping defined in (6), polynomials 1, $B$, $B^2$ and $B + B^2$ are mapped to $(1, 0)$, $(1, 1)$, $(1, 2)$ and $(2, 3)$, respectively.

Define another polynomial $g(B) = \sum_i b_i B^i$. Then, its mapping $\mathcal{T}\{g(B)\}$ is $(\gamma, \delta)$, where $\gamma = \sum_i b_i$ and $\delta = \sum_i i b_i$.

The addition of two polynomials $f(B) + g(B)$ is mapped as

$$\begin{aligned} \mathcal{T}\{f(B) + g(B)\} &= \mathcal{T}\{\sum_i (a_i + b_i) B^i\} \\ &= \left(\sum_i a_i + b_i, \sum_i i(a_i + b_i)\right) \\ &= (\alpha + \gamma, \beta + \delta). \end{aligned} \tag{7}$$

The multiplication of two polynomials $f(B) \cdot g(B)$ is mapped as

$$\begin{aligned} \mathcal{T}\{f(B) \cdot g(B)\} &= \mathcal{T}\{\sum_i \sum_j a_i b_j B^{i+j}\} \\ &= \left(\sum_i \sum_j a_i b_j, \sum_i \sum_j (i + j) a_i b_j\right) \\ &= \left(\sum_i a_i \cdot \sum_j b_j, \sum_i \sum_j [i a_i b_j + j a_i b_j]\right) \\ &= (\alpha\gamma, \beta\gamma + \alpha\delta). \end{aligned} \tag{8}$$

Therefore, the addition and multiplication of two elements in $Z^* \times Z^*$ are defined as

$$\begin{aligned} (\alpha, \beta) + (\gamma, \delta) &= (\alpha + \gamma, \beta + \delta), \\ (\alpha, \beta) \cdot (\gamma, \delta) &= (\alpha\gamma, \beta\gamma + \alpha\delta). \end{aligned} \tag{9}$$

With operations defined in (9), the identity elements are $(0, 0)$ and $(1, 0)$ for addition and multiplication, respectively. It is easy to show that $Z^* \times Z^*$ forms a semiring with the defined addition and multiplication operations.

### B. Weight Enumerators

A numerically efficient algorithm is presented for the modified trellis approach. The proposed algorithm is based on a new weight enumerator with only one variable $D$

$$W'(D) = \sum_j (\alpha_j, \beta_j) D^j, \tag{10}$$

where $\alpha_j$ and $\beta_j$ are defined by $\alpha_j = \sum_i c_{i,j}$ and $\beta_j = \sum_i i \cdot c_{i,j}$.

From the new weight enumerator $W'(D)$, the upper bounds on FER $P_{FE}$ and average BER $P_b$ are computed as

$$P_{FE} \leq \sum_j \alpha_j \cdot P_j, \tag{11}$$

$$P_b \leq (1/L_1) \cdot \sum_j \beta_j \cdot P_j. \tag{12}$$

A method is presented to compute the proposed weight enumerator based on a modified trellis diagram. Consider a rate $k/n$ convolutional code with $M$ memory elements in the encoder and denote the states at trellis depth $t$ as $S_t$ $(S_t = 0, 1, \cdots, 2^M - 1)$. To obtain the proposed weight enumerator, it is necessary to produce a modified trellis diagram for the terminated convolutional as in [4]. The modification is summarized as follows:

(i) If there is a branch merging into the state $S_t = 0$ in the original trellis diagram, define a new state $S_t = 2^M$ and redirect the branch to the new defined state. Connect all the

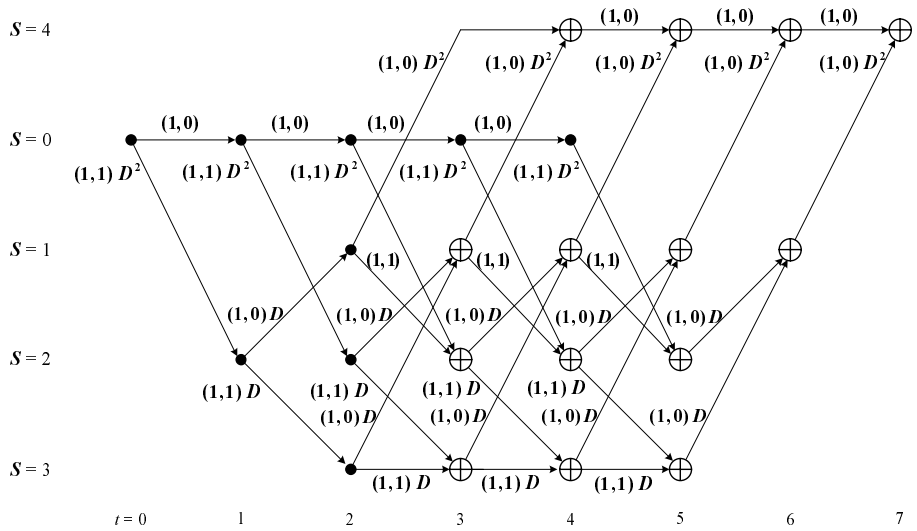


Fig. 1. Modified trellis diagram for convolutional code $C_1$.

new defined states with a branch of gain (1,0) from trellis depth $t-1$ to $t$.
(ii) If there is a branch diverging from the state $S_t = 0$, connect the zero states with a branch of gain (1,0) from trellis depth $t-1$ to $t$. Otherwise, remove the state $S_t = 0$.
(iii) Place an adder at the state node where more than two branches are merging.

In [4], [5], the gain of the branch between trellis depths $t-1$ and $t$ is a polynomial of two variables $B$ and $D$. The gain is in the form of $B^{k_1}D^{k_2}$, where $k_1$ and $k_2$ are the Hamming weight of the information bits and the Hamming weight of the coded output of the branch, respectively. However, the gain of each branch should be modified considering the new coefficients defined in this letter. $G_{S_{t-1},S_t}(D)$ is the gain of the branch between the states $S_{t-1}$ and $S_t$. In the proposed method, the gain is transformed to a polynomial with one variable $D$ as

$$G_{S_{t-1},S_t}(D) = (1,k_1)\, D^{k_2}. \tag{13}$$

For the special case of a rate $1/n$ convolution code, $k_1$ is either 0 or 1.

$F_{S_t}(D)$ is a temporary polynomial to store the coefficient information at state $S_t$. To obtain the proposed weight enumerator, initialize $F_{S_0}(D)$ at trellis depth $t = 0$ as

$$F_{S_0}(D) = \begin{cases} (1,0), & \text{if } S_0 = 0, \\ (0,0), & \text{otherwise.} \end{cases} \tag{14}$$

After the initialization, from trellis depth $t = 1$, compute

$$F_{S_t}(D) = \sum_{S_{t-1}\in\Lambda_{S_t}} F_{S_{t-1}}(D)\cdot G_{S_{t-1},S_t}(D) \tag{15}$$

for each state $S_t$, where $\Lambda_{S_t}$ is the set of states at trellis depth $t-1$ from which there exists a branch merging to $S_t$. At the end of the computation of (15) for all states at trellis depth $t$, increase $t$ by one and execute the evaluation for trellis depth $t+1$. Repeat this procedure until the trellis depth $t$ becomes $L_1 + L_2$. After the end of the procedure, the polynomial $F_{S_t=2^M}(D)$ at trellis depth $t = L_1 + L_2$ is the desired weight enumerator.

As a simple example of the modified trellis, consider a rate 1/2 convolutional code $C_1$ with four states in the encoder. The generator polynomial $G_1$ for the code is $G_1 = [x^2 + 1,\ x^2 + x + 1]$. Fig. 1 shows the modified trellis diagram of the convolutional code $C_1$. The gain of each branch is a polynomial with only one variable $D$ with the proposed coefficients.

From the modified trellis diagram in Fig. 1, the weight enumerator with one variable is computed as

$$\begin{aligned} W'(D) &= (5,5)D^5 + (7,14)D^6 + (8,24)D^7 \\ &+ (5,20)D^8 + (1,5)D^9. \end{aligned} \tag{16}$$

The conventional weight enumerator for the same configuration is

$$\begin{aligned} W(B,D) &= 5BD^5 + 7B^2D^6 + 8B^3D^7 \\ &+ 5B^4D^8 + B^5D^9. \end{aligned} \tag{17}$$

Comparing the two enumerators, it can be noticed that the number of terms is not reduced with the proposed enumerator. It is caused from the fact that all the codewords with a output Hamming weight are generated from information sequences with the same input Hamming weight for this specific example.

### C. Complexity

The storage needed for computing the proposed weight enumerator is dependent on the number of terms used for the performance upper bounds. To store one term of the proposed enumerator, three integer storages are necessary. Therefore, the total storage required for the algorithm is approximately $3\cdot N_t\cdot(N_{states}+1)$, where $N_t$ is the number of terms needed for the proposed weight enumerator and $N_{states}$ is the number of states in the trellis diagram.

It is necessary to store three integers for each term of the conventional weight enumerator. Therefore, the total required storage is approximately $3\cdot N_{t2}\cdot(N_{states}+1)$ for the conventional weight enumerator, where $N_{t2}$ is the number of terms needed for the conventional weight enumerator with two variables. Therefore, with the proposed weight enumerator, the

required storage reduces by $N_t/N_{t2}$ regardless of the coding rate.

Let us consider the computational complexity of the proposed weight enumerator. For a rate $k/n$ convolutional code, there are $2^k$ branches diverging from a state and merging into a state. At each branch, a polynomial $(1, k_1)D^{k_2}$ is multiplied to the tentative polynomial $F_{S_t}(D)$. An integer multiplication and two integer additions are needed for the multiplication of $(1, k_1)$ to another coefficient. The complexity of this coefficient multiplication is $X + 2$ integer additions, where $X$ is the complexity of integer multiplication converted to the number of integer addition operations. Therefore, total $N_t \cdot (X + 2)$ integer additions are necessary for each branch.

At each state node, addition of $2^k$ polynomials with $N_t$ terms is needed, which requires two integer additions for the terms with the same exponent. Total $(2^k - 1) \cdot 2 \cdot N_t$ integer additions are needed for each state node. Therefore, the total complexity to compute the proposed weight enumerator is approximately $(N_{states} + 1) \cdot (L_1 + L_2) \cdot \left[2^k \cdot (X + 2) + (2^k - 1) \cdot 2\right] \cdot N_t$ integer additions, since the number of terms in each state is less than or equal to $N_t$.

Because the value of $k_1$ is an integer between 0 and $k$, $X$ is not greater than $\lceil \log_2(k+1) \rceil - 1$ integer additions, where $\lceil x \rceil$ is the smallest integer not less than $x$. $\lceil \log_2(k+1) \rceil - 1$ is the worst case estimation of the complexity, since for some value of $k_1$, such as 2 and 4, the multiplication can be computed with much less complexity using a shift operation.

For the conventional weight enumerator, two integer additions are required for each term of the tentative polynomial at each branch. One integer addition is required for the terms with the same exponents at each state node. Since there are total $2^k$ branches at each state, the total complexity to compute the weight enumerator is approximately $(N_{states} + 1) \cdot (L_1 + L_2) \cdot 2^k \cdot 3 \cdot N_{t2}$ additions.

Compared with the conventional enumerator, the proposed enumerator reduces the complexity by about $N_t/N_{t2} \cdot (X + 4 - 2 \cdot 2^{-k})/3$. For the case of a rate $1/n$ convolutional code, the complexity reduction is about $N_t/N_{t2}$.

If $k \geq 2$, the complexity reduction is about $N_t/N_{t2} \cdot (X + 4)/3$, where $X$ is not greater than $\lceil \log_2(k + 1) \rceil - 1$. Therefore, for $k$ values between 2 and 15, the complexity reduction is about $2N_t/N_{t2}$. For a convolutional code with a small number of states, the complexity reduction can be larger than 1. However, generally $N_t/N_{t2}$ decreases rapidly as the number of states increases. Therefore, $N_t$ is only a fraction of $N_{t2}$ for convolutional codes with a large number of states, which is the numerically more demanding case. It is also important to note that $N_t/N_{t2} \cdot (X + 4)/3$ is the worst case complexity reduction for a rate $k/n$ convolutional codes. There have been research efforts to find the minimal complexity trellis for a rate $k/n$ convolutional code [9][10]. With the convolutional code with minimal complexity trellis, further complexity reduction can be achieved. A special case of the rate $k/n$ convolutional code with minimal complexity trellis is a punctured convolutional code, which is widely used in practical communication systems [11][12]. For the rate $k/n$ convolutional code obtained by puncturing a rate $1/n$ code, the complexity reduction is about $N_t/N_{t2}$, which is the same as that for a rate $1/n$ convolutional code.

## IV. Numerical Results

In this section, some examples are presented with the proposed weight enumerator for some convolutional codes used in practical systems.

Consider a rate 1/2 convolutional code $C_2$ of constraint length 9 used for an IS-95 system [13]. The generator polynomial $G_2$ for the code is $[x^8 + x^7 + x^5 + x^3 + x^2 + x + 1,\ x^8 + x^4 + x^3 + x^2 + 1]$. $L_1 = 184$ and $L_2 = 8$ are assumed. The conventional enumerator is computed as

$$\begin{aligned} W(B, D) = {} & (184B + 365B^2 + 902B^3 + 359B^4 + 177B^5)D^{12} \\ & + (532B^3 + 1594B^4 + 2291B^5 + 2448B^6 + 1040B^7 \\ & + 343B^8 + 349B^9 + 173B^{11})\ D^{14} + \cdots . \end{aligned} \tag{18}$$

The proposed weight enumerator is computed as

$$W'(D) = (1987, 5941)D^{12} + (8770, 49183)D^{14} + \cdots . \tag{19}$$

Comparing two enumerators, it can be observed that the numbers of terms required for up to $D^{14}$ are $N_{t2} = 13$ and $N_t = 2$, respectively. If the enumerators are computed up to $D^{20}$, the numbers of the required terms are $N_{t2} = 68$ and $N_t = 5$, respectively.

Consider a rate 3/4 convolutional code which is used in an IS-95B system [13]. The code is obtained by puncturing the rate 1/2 convolutional code $C_2$. The third and the fifth coded bits are punctured per every coded bit block of length six. $L_1 = 280$ and $L_2 = 8$ are assumed. This code has free distance of 6 and the conventional weight enumerator is computed as

$$\begin{aligned} W(B, D) = {} & (184B^3 + 183B^4 + 182B^5 + 179B^6 + 178B^8)D^6 \\ & + (187B + 186B^2 + 459B^3 + 456B^4 + 455B^5 + 632B^6 \\ & + 448B^7 + 798B^8 + 884B^9 + 523B^{10} + 87B^{11} \\ & + 515B^{12} + 429B^{13} + 85B^{14} + 256B^{15} + 334B^{16} \\ & + 80B^{23})D^7 + \cdots . \end{aligned} \tag{20}$$

The proposed weight enumerator is computed as

$$W'(D) = (906, 4692)D^6 + (6814, 57461)D^7 + \cdots . \tag{21}$$

Comparing (20) and (21), it can be noticed that the numbers of required terms for up to $D^7$ are $N_{t2} = 22$ and $N_t = 2$, respectively. If the enumerators are computed up to $D^{15}$ terms, then the numbers of the required terms are $N_{t2} = 441$ and $N_t = 10$, respectively. The more terms are required for computing the performance bounds, the more complexity reduction can be achieved with the proposed weight enumerator.

## V. Conclusion

In this letter, a numerically efficient algorithm is presented for the modified trellis approaches to compute the performance upper bounds of terminated convolutional codes. The conventional weight enumerator is a polynomial of two variables. To reduce the storage and computational complexity, a new weight enumerator using only one variable is presented with a new algebraic structure for the coefficients. Numerical results show that the proposed algorithm can save substantially the storages and computations needed to compute the performance bounds of terminated convolutional codes.

## REFERENCES


[1] A. J. Viterbi, "Convolutional codes and their performance in communication systems," *IEEE Trans. Commun. Tech.*, COM-19, pp. 751-772, Oct. 1971.

[2] R. J. McEliece, *The Theory of Information and Coding - A Mathematical Framework For Communication*. Boston, MA: Addison-Wesley, 1977.

[3] H. Moon, "Improved upper bound on bit error rate for truncated convolutional codes," *IEE Electron. Lett.*, vol. 34, no. 1, pp. 65-66, Jan. 1998.

[4] H. Moon and D. C. Cox, "Improved performance upper bound for terminated convolutional codes," *IEEE Commun. Lett.*, vol. 11, no. 6, pp. 519-521, June 2007.

[5] H. Moon and D. C. Cox, "Performance of undeaully punctured convolutional codes," *IEEE Trans. Wireless Commun.*, vol. 8, no. 8, pp. 3903-3909, Aug. 2009.

[6] G. Caire and E. Viterbo, "Upper bound on the frame error probability of terminated trellis codes," *IEEE Commun. Lett.*, vol. 1, no. 1, pp. 2-4, Jan. 1998.

[7] U. Hebisch, *Semirings: Algebraic Theory and Application*. Mountain View, CA: World Scientific Publishing Company, 1998.

[8] J. S. Golan, *Semirings and their Applications*. Berlin, Germany: Springer, 1999.

[9] R. J. McEliece and W. Lin, "The trellis complexity of convolutional codes," *IEEE Trans. Inf. Theory*, vol. 42, no. 6, pp. 1855-1864, Nov. 1996.

[10] B. F. Uchôa-Filho, R. D. Souza, C. Pimentel, and M. Jar, "Convolutional codes under a minimal trellis complexity measure," *IEEE Trans. Commun.*, vol. 57, no. 1, pp. 1-5, Jan. 2009.

[11] J. Cain, G. Clard, and J. Geister, "Punctured convolutional codes of rate $(n-1)/n$ and simplified maximum likelihood decoding," *IEEE Trans. Inf. Theory*, vol. 25, no. 1, pp. 97-100, Jan. 1979.

[12] Y. Yasuda, K. Kashiki, and Y. Hirata, "High rate punctured convolutional codes for soft decision viterbi decoding," *IEEE Trans. Commun.*, vol. 32, no. 3, pp. 315-319, Mar. 1984.

[13] TIA/EIA, IS-95B, Mobile station-base station compatibility standard for dual-mode spread spectrum systems, Oct. 1998.