ECE 5654 Lecture 16
Convolutional Codes

Ravi Tandon

Virginia Tech

April 2, 2014

Learning Objectives

At the end of this lecture, the student should be able to:

- Describe the difference between a convolutional code and a block code
- Describe the four ways of representing a convolutional code
- Implement the Viterbi Algorithm for a convolutional code

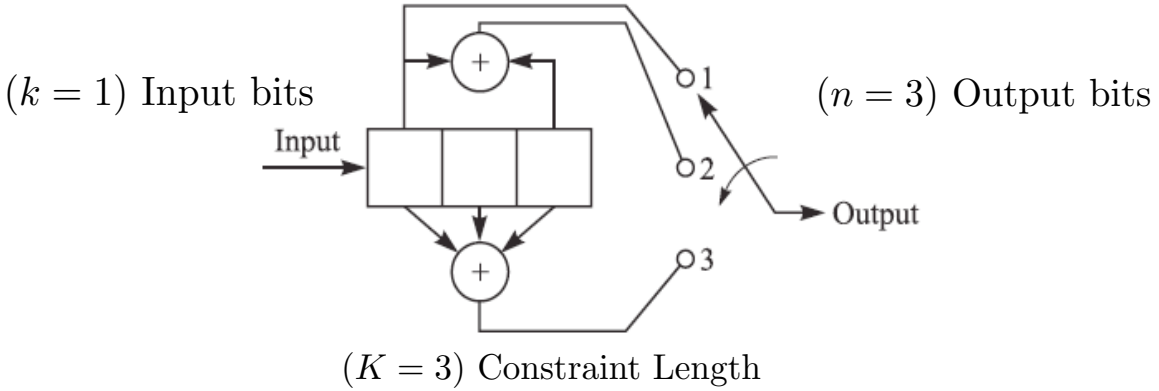
Block Vs Convolutional Codes

- Block codes take k input bits and produce n output bits, where k and n are large
  - There is no data dependency between blocks
  - Useful for data communications (i.e., delay tolerant)
- Convolutional codes take a small number of input bits and produce a small number of output bits each time period
  - data passes through convolutional coder in a continuous stream
  - useful for low-latency communications

Convolutional Codes

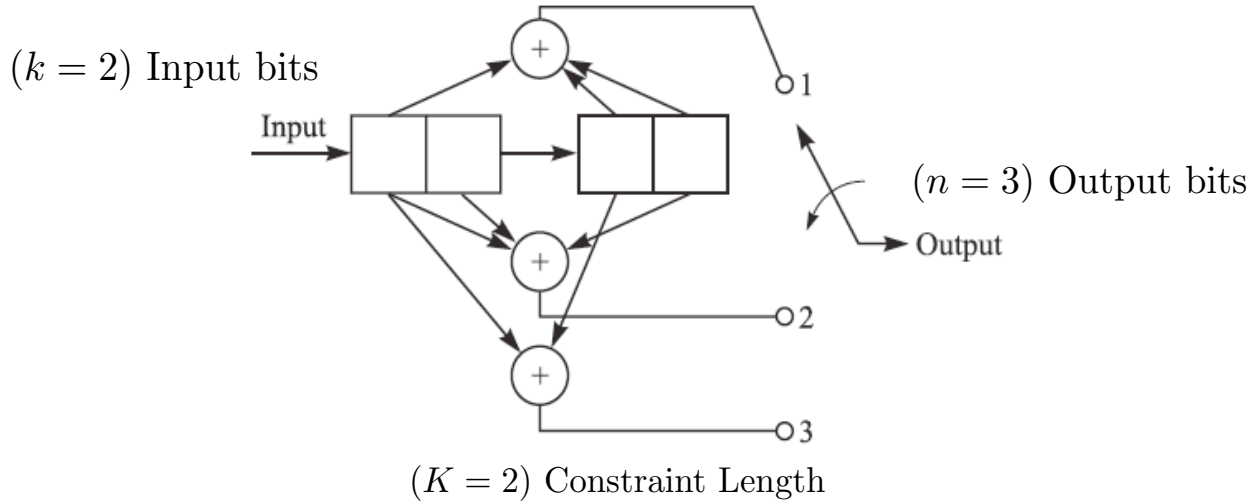
- k bits are input, n bits are output
- k and n are very small (usually k=1-3, n=2-6)
- Frequently, we will see that k=1
- However, the output bits depend not only on current set of k input bits, but also on past inputs
- The number of k-length bits on which output depends is called the "constraint length" K
- Can be thought of as the memory of the code
- Distance properties (and thus coding gain) increase with K

Example 1: $n = 3, k = 1, K = 3$ convolutional code



- The current output bits depend on $K \times k = 3$ bits
- One of the them is the current bit, remaining two are the past 2 bits
- Initial State of shift register: 000
  - First input bit $= 1 \Rightarrow$ Shift register: 100 $\Rightarrow$ Output bits: 111
  - Second input bit $= 0 \Rightarrow$ Shift register: 010 $\Rightarrow$ Output bits: 001
  - Third input bit $= 1 \Rightarrow$ Shift register: 101 $\Rightarrow$ Output bits: 100

Example 2: $n = 3, k = 2, K = 3$ convolutional code

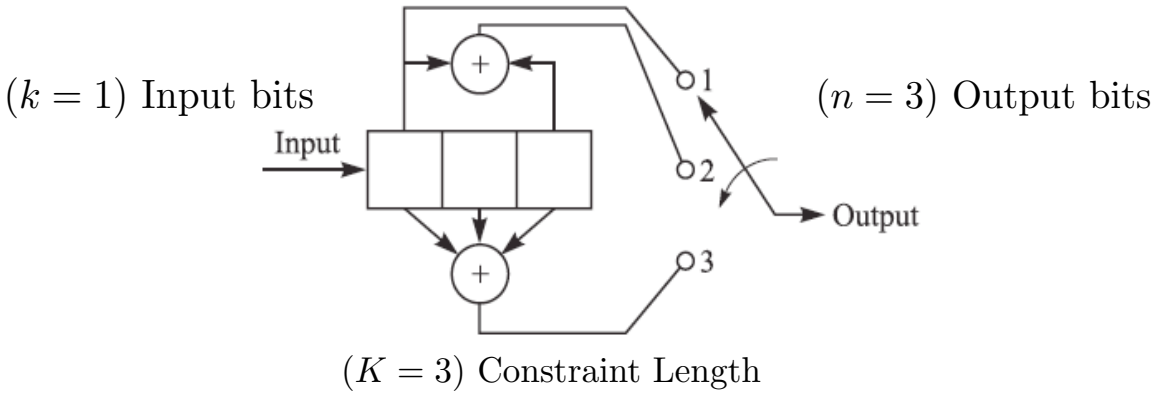


- The current output bits depend on $K \times k = 4$ bits
- Two of the them are the input bits, remaining two are the past 2 bits
- Initial State of shift register: 0000
  - First input $= 11 \Rightarrow$ Shift register: 1100 $\Rightarrow$ Output bits: 101
  - Second input $= 01 \Rightarrow$ Shift register: 0111 $\Rightarrow$ Output bits: 001
  - Third input $= 10 \Rightarrow$ Shift register: 1001 $\Rightarrow$ Output bits: 001

Representation of a Convolutional Code

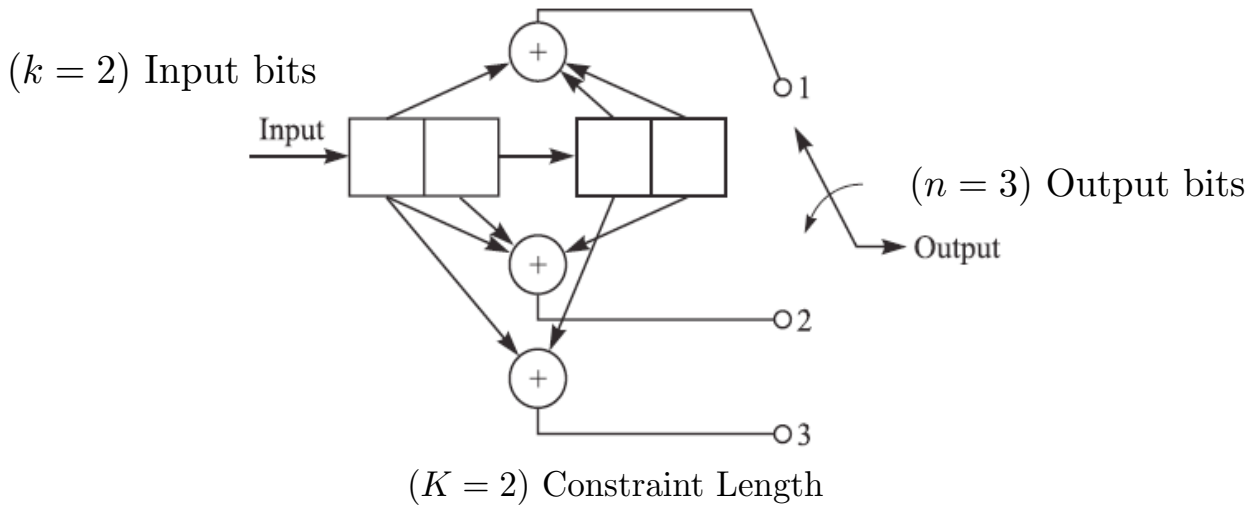
- There are in general four ways of representing a convolutional code
  - Encoder Block Diagram (two examples shown above)
  - Generator Representation (represents the relationship between input (and stored) bits and output bits)
  - State Diagram Representation
  - Trellis Representation

Generator Representation



- In general, we need to specify n vectors (one for each output bit)
- Each vector has $K \times k$ dimensions
- For the above example:
  - $g_1 = [100]$
  - $g_2 = [101]$
  - $g_3 = [111]$
- We usually represent these in octal form
- $(g_1, g_2, g_3)_8 = (4, 5, 7)$

Example 2: Generator Representation



- For the above example $n = 3, k = 2, K = 2$ convolutional code:
  - $g_1 = [1011]$
  - $g_2 = [1101]$
  - $g_3 = [1010]$
- In octal form
- $(g_1, g_2, g_3)_8 = (13, 15, 12)$

Transfer Domain Generator Matrix

- Let's focus on the $n = 3, k = 1, K = 3$ example
- For information sequence u, we get three output sequences
- In the above example, $c^{(1)} = u * g_1, c^{(2)} = u * g_2, c^{(3)} = u * g_3$
- $*$ denotes the convolution operation
- The code sequence c (which is transmitted over the channel) is the result of interleaving $c^{(1)}, c^{(2)}, c^{(3)}$ as

$$c = (c_1^{(1)}, c_1^{(2)}, c_1^{(3)}, c_2^{(1)}, c_2^{(2)}, c_2^{(3)}, \ldots)$$

- Convolution is equivalent to multiplication in transform domain
- We define the D-transform (or delay transform) of u as

$$u(D) = \sum_{i=0}^{\infty} u_i D^i$$

Transfer Domain Generator Matrix

- We also define the transfer function of the three impulse responses (g_1, g_2, g_3) as

$$g_1 = [100] \Rightarrow g_1(D) = 1$$
$$g_2 = [101] \Rightarrow g_2(D) = 1 + D^2$$
$$g_3 = [111] \Rightarrow g_3(D) = 1 + D + D^2$$

- Then, the output transforms are given by

$$c^{(1)}(D) = u(D)g_1(D)$$
$$c^{(2)}(D) = u(D)g_2(D)$$
$$c^{(3)}(D) = u(D)g_3(D)$$

- Transform of the encoder output c (by interleaving) is then

$$c(D) = c^{(1)}(D^3) + Dc^{(2)}(D^3) + D^2c^{(3)}(D^3)$$

Example

- Suppose $u = (100111)$ be the input sequence
- D-transform of u: $u(D) = 1 + D^3 + D^4 + D^5$
- Suppose that this sequence is transmitted over the $(n, k, K) = (3, 1, 3)$ convolutional code
- Then, the output transforms are:

$$c^{(1)}(D) = (1 + D^3 + D^4 + D^5)(1) = 1 + D^3 + D^4 + D^5$$
$$c^{(2)}(D) = (1 + D^3 + D^4 + D^5)(1 + D^2) = 1 + D^2 + D^3 + D^4 + D^6 + D^7$$
$$c^{(3)}(D) = (1 + D^3 + D^4 + D^5)(1 + D + D^2) = 1 + D + D^2 + D^3 + D^5 + D^7$$

and

$$\begin{aligned} c(D) &= c^{(1)}(D^3) + Dc^{(2)}(D^3) + D^2c^{(3)}(D^3) \\ &= 1 + D + D^2 + D^5 + D^7 + D^8 + D^9 + D^{10} + D^{11} + D^{12} + D^{13} + D^{15} \\ &\quad + D^{17} + D^{19} + D^{22} + D^{23} \end{aligned}$$

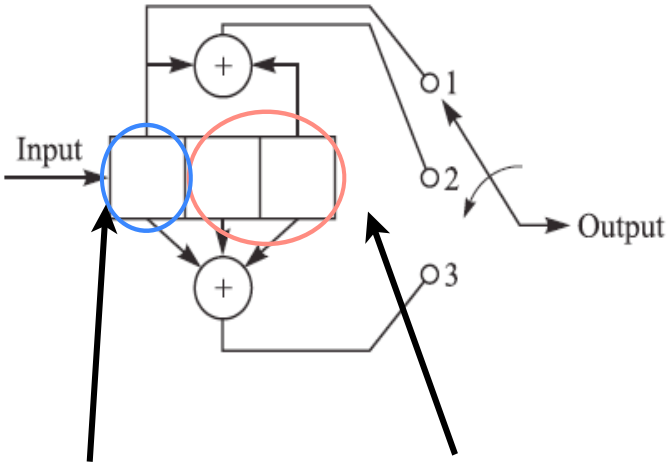
corresponding to the code sequence

$$\boldsymbol{c} = (11100101111111010101010011)$$

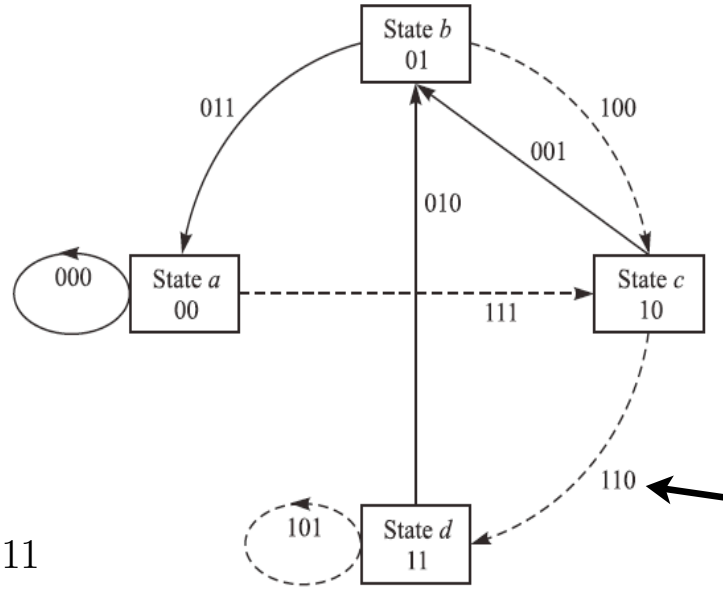
State Diagram Representation of Convolutional Codes

- Contents of shift registers make up the "state" of the code
  - Most recent input is most significant bit of state
  - Oldest input is least significant bit of state
- Arcs connecting states represent allowable transitions
- Arcs are labeled with output bits transmitted during transition

State Diagram Representation $(n, k, K) = (3, 1, 3)$



New input bits Contents of shift register



Four possible "states": $00, 01, 10, 11$

Suppose, we start at state 00

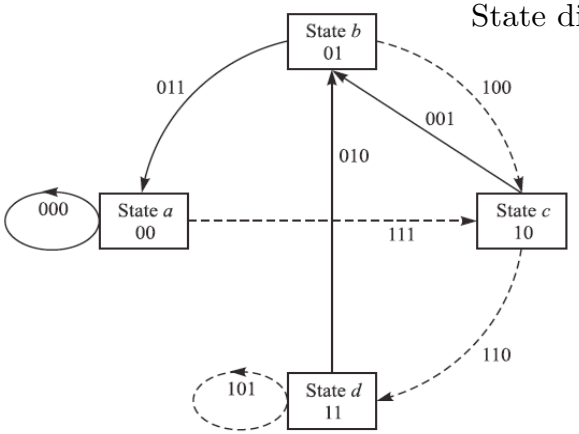
Input $= 0$	Output $= 000$	State $= 00$
Input $= 1$	Output $= 111$	State $= 10$
Input $= 0$	Output $= 001$	State $= 01$

Trellis Representation of Convolutional Code

- State diagram is "unfolded" as a function of time
- Time indicated by movement towards right
- Contents of shift registers make up "state" of code:
  - Most recent input is most significant bit of state.
  - Oldest input is least significant bit of state
- Allowable transitions are denoted by connections between states
- Transitions may be labeled with transmitted bits
- Allowable transitions determine which output bits are possible from a given state

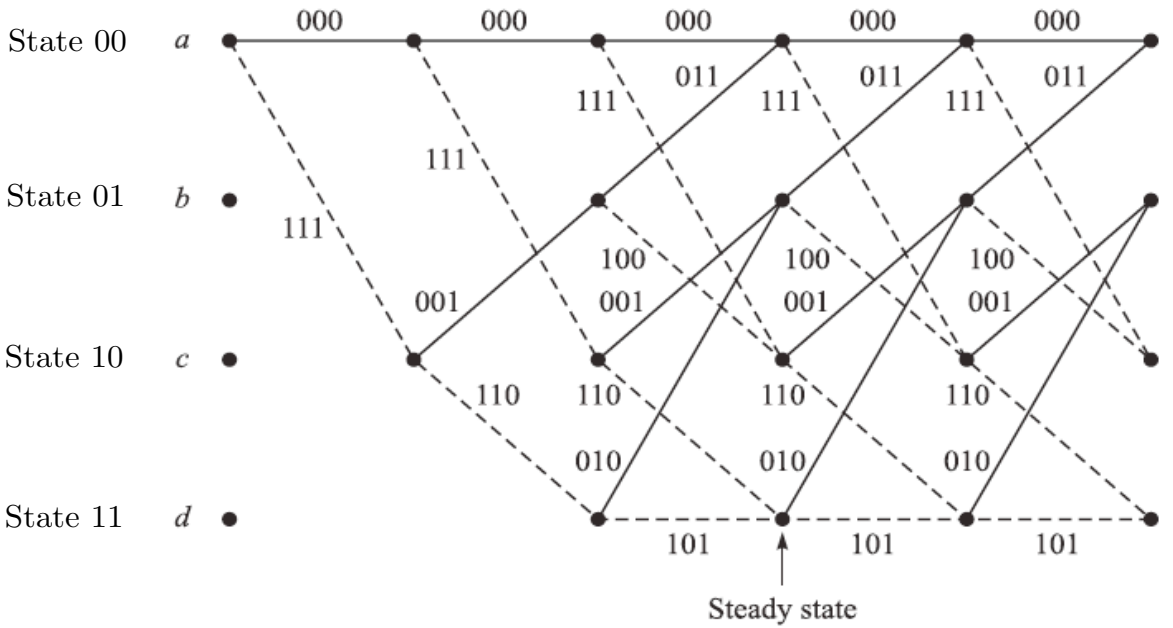
Trellis Representation; $(n, k, K) = (3, 1, 3)$

State diagram representation



Unfolding of state diagram in time

Trellis representation



Decoding of Convolutional Codes

- Recall that in decoding a block code, we computed the distances (Hamming for HDD, Euclidean for SDD) between received word and the 2^k possible transmitted codewords
- We then selected the codeword which was closest in distance to the received word
- Unlike a block code, which has a fixed length n, a convolutional encoder is a Finite State Machine
- Optimal decoder is a maximum-likelihood sequence estimator (MLSE)
- Optimum decoding of a convolutional code involves a search through the trellis for the most probable sequence
- Depending on HDD/SDD, the metric in the trellis search can be either Hamming/Euclidean

Viterbi Decoding for Convolutional Codes

- In the rest of this lecture, we will study the Viterbi Algorithm for decoding.
- Viterbi algorithm significantly reduces the decoding complexity
- Go to Board..