Revision of Lecture Thirteen

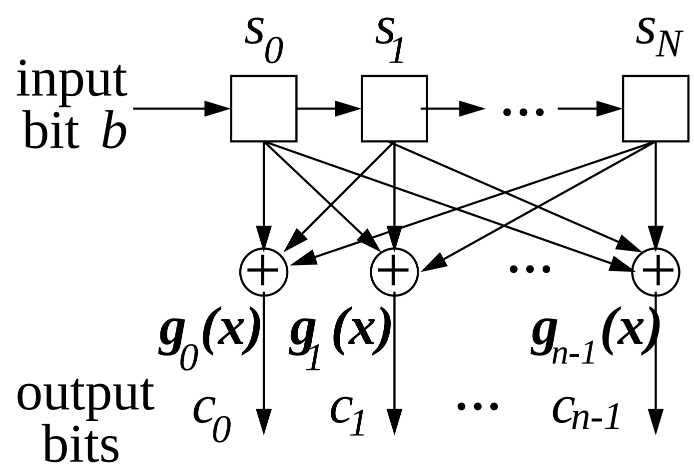
- Previous lecture is about channel coding introduction
  - Basic concepts introduced in this lecture are important and you should have a deep understand of them
  - These concepts are mainly based on hard-decision decoding, but there are more powerful soft-decision decoding which has higher error correction capability

- In the next two lectures, we will discuss two classes of practical channel coding schemes, namely,

1. Convolutional codes
2. A particular class of linear block codes known as BCH

Convolutional Coding

- **Convolutional codes** can be systematic or non-systematic, non-systematic ones are more powerful
- $CC(n, k, N)$: code rate $R = k/n$, N is **constraint length** (or **memory** $N + 1$ stages), usually n, k and N are small, and in particular, $k = 1$ is often used
- $CC(n, k = 1, N)$ encoder circuit:
  - During each bit interval, the register is shifted one stage: s_N is shifted out, $s_i \rightarrow s_{i+1}$, and the data bit enters s_0
  - After this shift, the values of $s_1 \cdots s_N$ define the current **state** of the register, which is used to produce code bits
  - Modulo-2 adders produce code bits c_i, $0 \le i \le n-1$, which are specified by the n polynomials



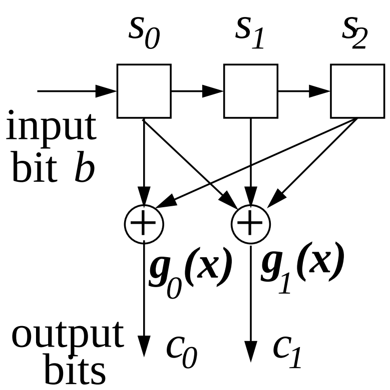
$$g_i(x) = g_{i,0} + g_{i,1}x + \cdots + g_{i,N}x^N, \quad 0 \le i \le n-1$$

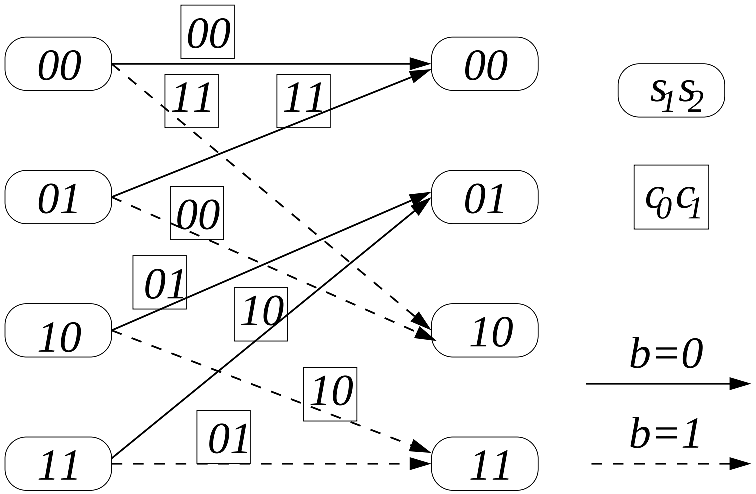
  - Systematic CC: $g_0(x) = 1$ and $c_0 = b$ (data bit), while non-systematic CC: some $g_{0,l} = 1$, $l > 0$
  - Output (code) bits depend on the state of $s_1 \cdots s_N$ after shift and the input bit

CC Encoder: State-Transition Diagram

- State-transition diagram describes the encoder of a CC code, showing all the 2^N states and all the state transitions together with the output bits
- Example: half-rate constraint length $N = 2$, $CC(2, 1, 2)$, defined by $g_0(x) = 1 + x^2$ and $g_1(x) = 1 + x + x^2$, table of state transition, encoder circuit, state-transition diagram

input b	current $s_1 s_2$	next $s_1 s_2$	output $c_0 c_1$
0	0 0	0 0	0 0
0	0 1	0 0	1 1
0	1 0	0 1	0 1
0	1 1	0 1	1 0
1	0 0	1 0	1 1
1	0 1	1 0	0 0
1	1 0	1 1	1 0
1	1 1	1 1	0 1

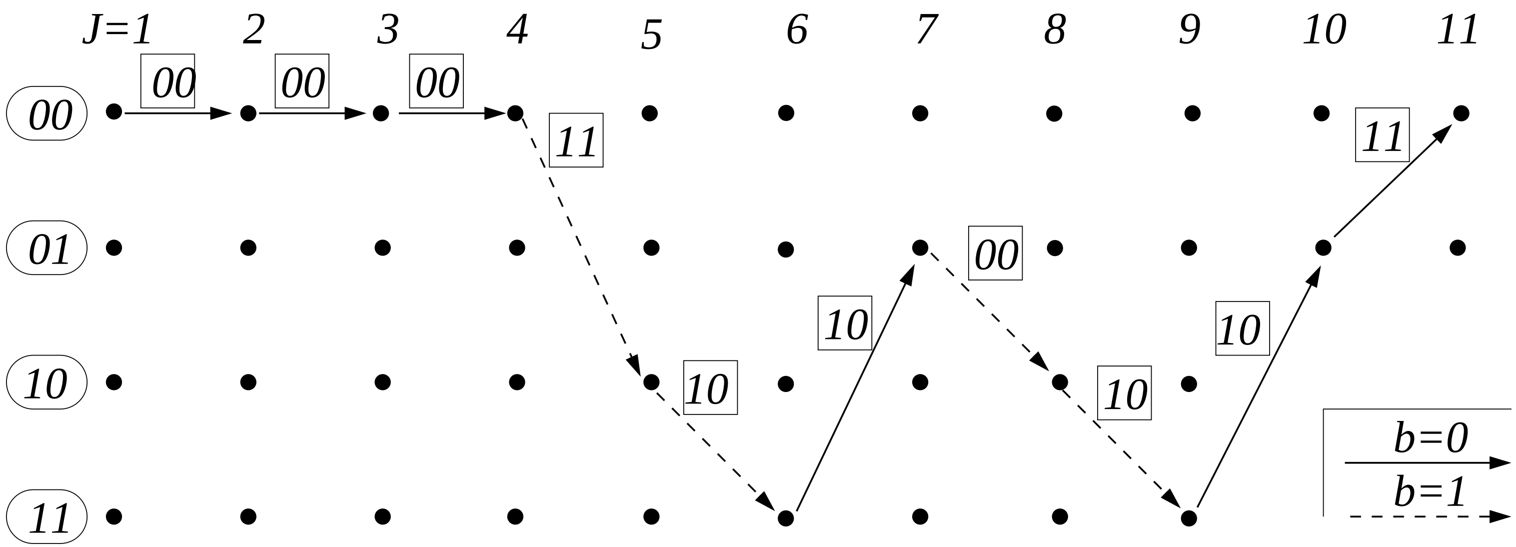




- Only **two legitimate state transitions** for each state, depending on the input bit b; similarly, each state has two merging paths; output bits are shown in the box

CC Encoder: Trellis Diagram

- An alternative way to describe a CC encoder is **trellis diagram**
- Same example, $CC(2,1,2)$, with $g_0(x) = 1 + x^2$ and $g_1 = 1 + x + x^2$
  - Information bit sequence $\cdots 0011011000$ (assume rightmost enters encoder first)
  - Recall the state-transition diagram of slide 183, we have **trellis diagram**:



- The state is initialised at zero and as the data bit sequence enters, trellis diagram shows the history of state transitions with output bits on it

CC Decoding

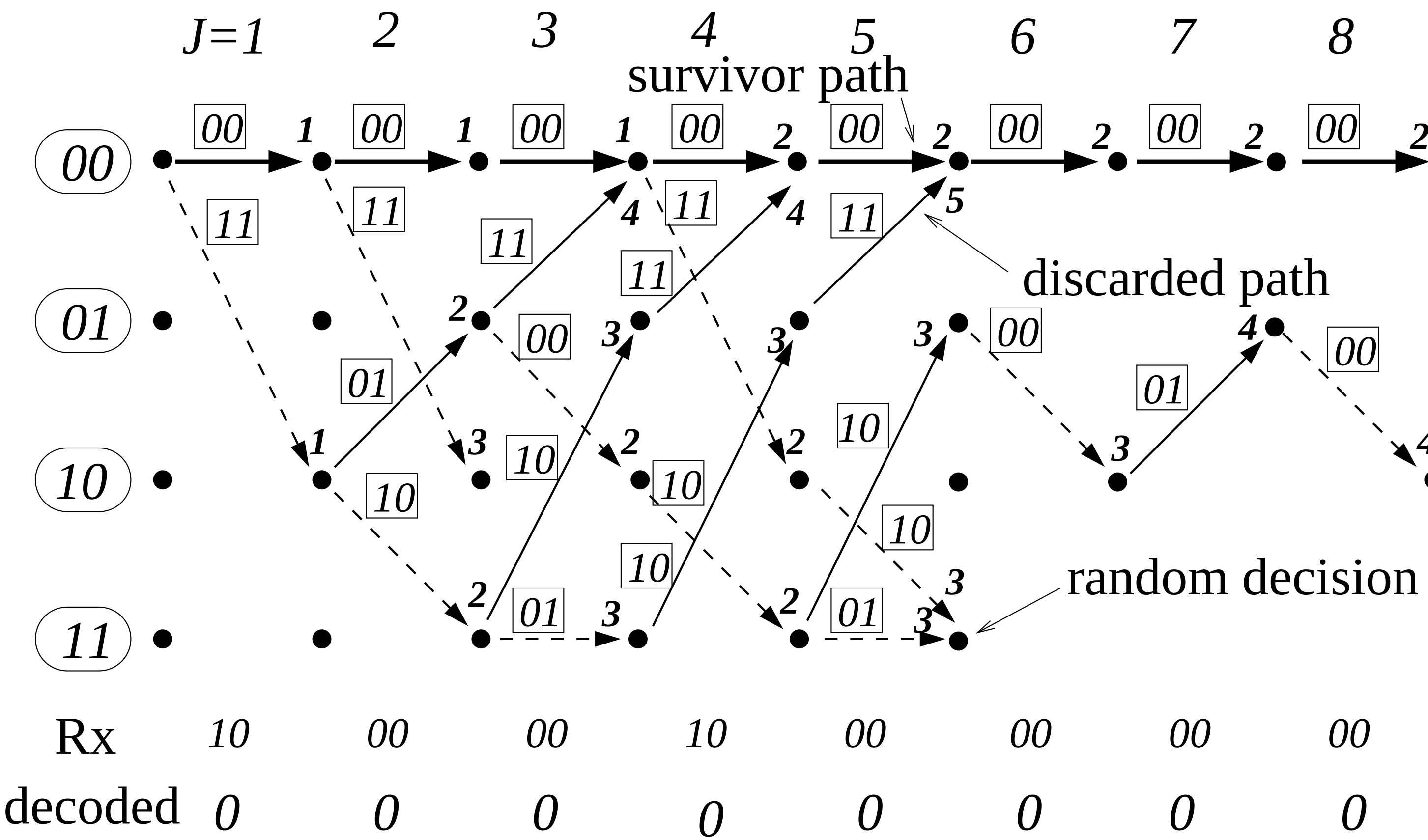
- In CC encoder: bits enter encoder register and has to travel through it
  - Thus the register sequence of state transitions is not arbitrary, and modulo-2 gates impose additional constraints on output bits
  - Result is only certain output bit sequences are **legitimate** and transmitted sequences are restricted to these legitimate sequences

- If a non-legitimate received sequence is encountered in the decoder, it must be due to channel errors, as such a sequence cannot be transmitted
  - In this case, the decoder can choose a legitimate sequence that is most resembling to the received sequence (e.g. in the sense of smallest Hamming distance)

- Such a decoding strategy is called **maximum likelihood** sequence decoding, as it finds the most likely transmitted sequence, given the received sequence
  - MLSD can be implemented efficiently using the **Viterbi algorithm**, which can be hard-input hard-output, soft-input hard-output, soft-input soft-output

Hard-Input Hard-Output Viterbi Algorithm

- **Hard-input** decoding is for hard-decision demodulation where demodulator demapper (recall slide 111) has made binary decision concerning received bits
- **Hard-output** decoding makes hard decision concerning decoded bits
  - $CC(2,1,2)$ with $g_0(x) = 1 + x^2$ and $g_1 = 1 + x + x^2$, all zero sequence is transmitted and received sequence is $10000010000000\cdots$ (leftmost bit at leftmost position of trellis)

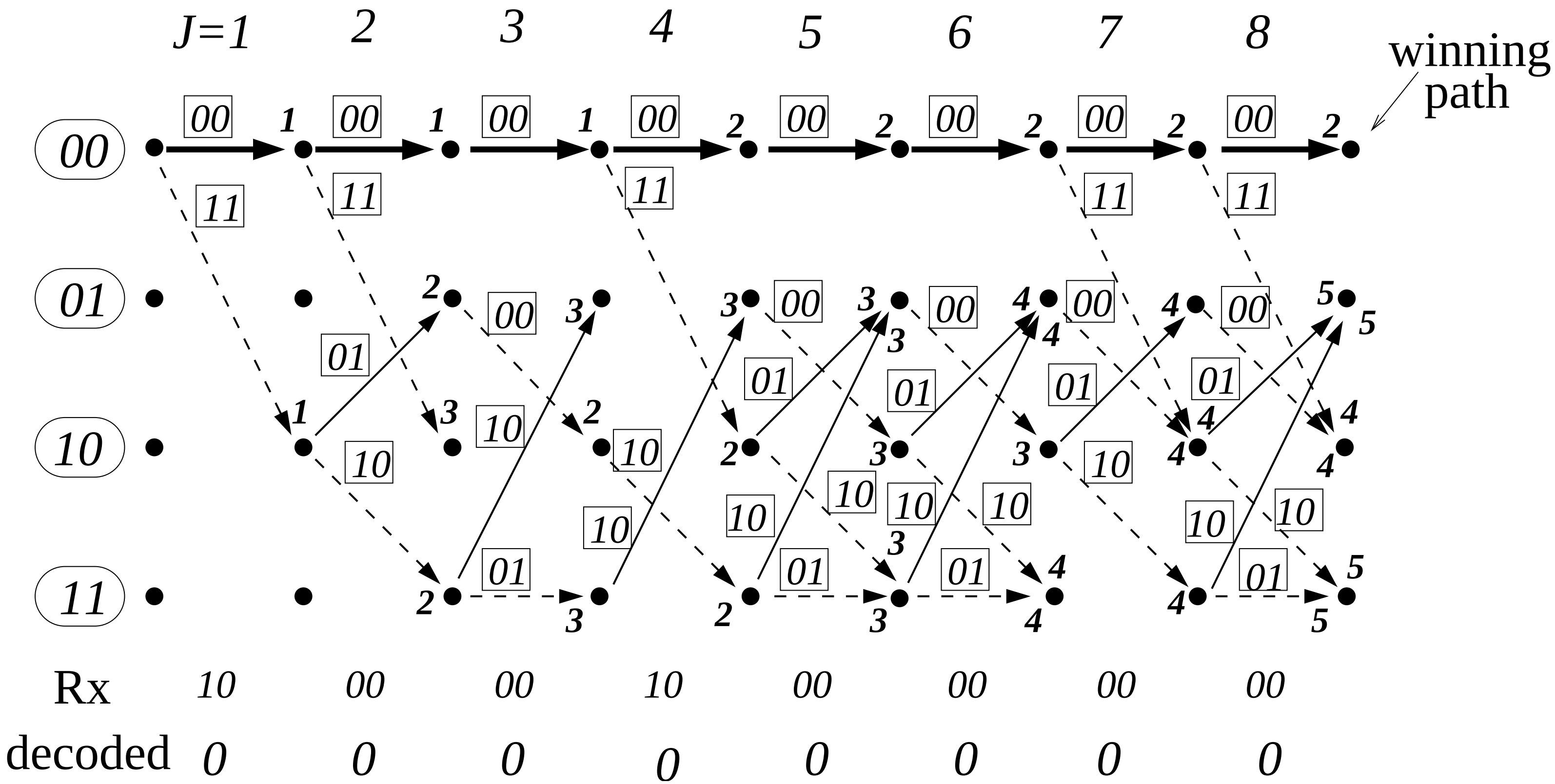


HIHO Viterbi Algorithm Decoding Rules

- **Branch metric** is the **Hamming distance** between the legitimate encoded (output) bits of a trellis branch at a stage and the received bits at the same stage
- **Path metric** is the accumulated branch metrics of a path (bold number shown on trellis diagram)
  - A path presents a legitimate sequence
- For two merging paths at a stage, the one with a larger path metric is discarded and the other, called **survivor path**, is kept
  - If two metrics are equal, a random decision is made to keep one path
- A final decision is made after a sufficiently large number of stages (8 for this example) to choose a winning path with the smallest path metric
  - This is MLSD, as winning path is most resembling to received sequence
- If decoding decision is correct, the winning path metric is the number of transmission errors inflicted by the channel

Full Trellis for Previous VA decoding Example

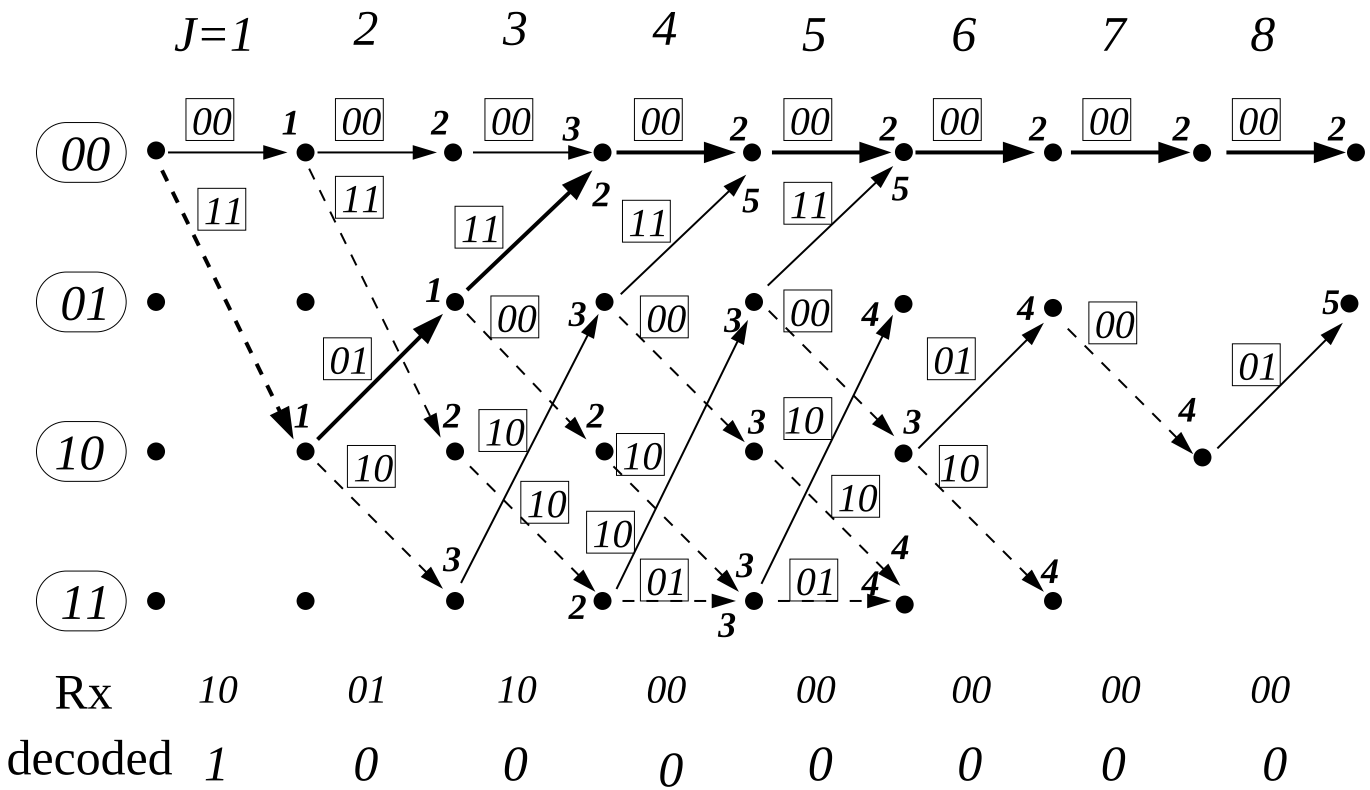
- $CC(2,1,2)$ with $g_0(x) = 1 + x^2$ and $g_1 = 1 + x + x^2$, all zero sequence is transmitted and received sequence is $10000010000000 \cdots$ (the leftmost bit at leftmost position of trellis)
  - Using VA decoding rules of previous slide and state-transition diagram of slide 183, we obtain:



- Since this example has 4 states, from stage 3 onward, there are 4 survivor paths (only survivor paths are kept but random decisions were not made in the diagram)

Another Example of Hard-Input Hard-Output VA

- Example: $CC(2,1,2)$ with $g_0(x) = 1 + x^2$ and $g_1 = 1 + x + x^2$, all zero sequence is transmitted but received sequence is $10011000000000 \cdots$ (the leftmost bit at leftmost position of trellis)
  - Note a burst error of two bits in demodulated sequence



  - Winning path is most resembling to received sequence
- But this is erroneous decoding. If decoding decision is incorrect, the winning path metric is not the number of transmission errors caused by the channel

Soft-Input Hard-Output Viterbi Algorithm

- **Soft-input** decoding is for soft-decision demodulation where demapper does not make hard binary decision but gives confidence measure concerning probability of a binary bit being 1 or 0
  - As the decoder has more information, it does better than hard-input decoding
  - Recall slide 112, this confidence measure or soft decision is given as **log likelihood ratio**
  - Here we consider a quantised **confidence measure** alternative
- Assume we use $\pm 4, \pm 3, \pm 2, \pm 1$ 8-level (or 3-bit quantisation) for confidence measure with interpretation:
  - $+4$: extremely like to be 1; $+3$: strongly like to be 1; $+2$: like to be 1; $+1$: weakly like to be 1
  - -4: extremely like to be 0; -3: strongly like to be 0; -2: like to be 0; -1: weakly like to be 0
- Hard-output decoder outputs hard decoded bits, but **soft-output decoder** outputs sequence of log likelihood ratios or confidence measures and it is for iterative decoding
- Example: $CC(2,1,2)$ with $g_0(x) = 1 + x^2$ and $g_1 = 1 + x + x^2$
  - An all zero sequence is transmitted, but the received soft decision sequence is

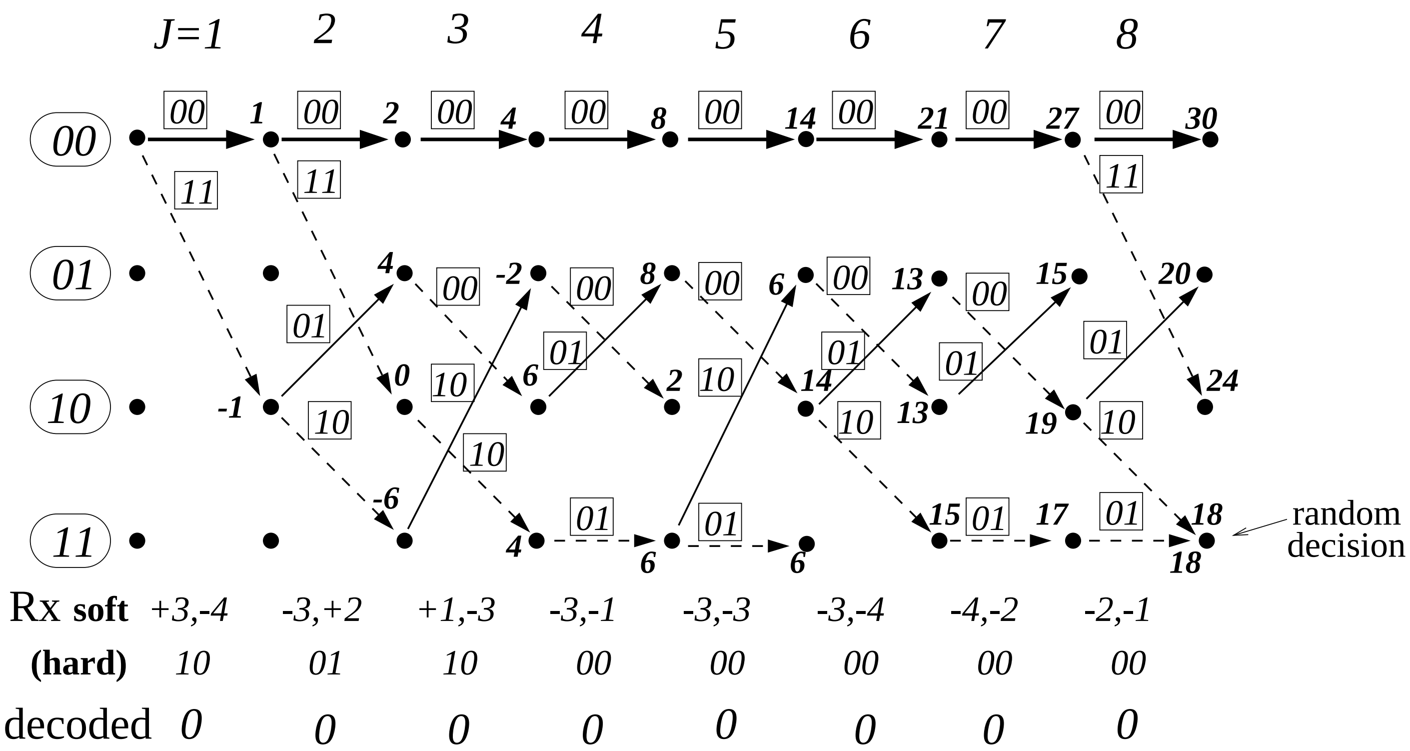
$$+3, -4, -3, +2, +1, -3, -3, -1, -3, -3, -3, -4, -4, -2, -2, -1, \cdots$$

  - Hard demapper would produce the hard decisions $10011000000000 \cdots$ (with the leftmost bit at leftmost position of trellis)
  - As previous slide shows, HIHO VA chooses a legitimate sequence that is most resembling to $10011000000000 \cdots$, but it is erroneous decoding

SIHO Viterbi Algorithm (continue)

- Trellis of SIHO VA decoding:

  Previously, HIHO VA gave erroneous decoding but SIHO VA produces correct decoding



- In quantised confidence measure based soft-input decoding, the meaning of metric has changed
  - If trellis branch output bits are 01 and receive soft decisions are $+3$,$+1$, it has a penalty -3 for the 1st bit and a credit $+1$ for the 2nd bit, so that branch metric is $(-3) + (+1) = -2$
  - If trellis branch output bits are 00 and receive soft decisions are $+3, -4$, it has a penalty -3 for the 1st bit and a credit $+4$ for the 2nd bit, so that branch metric is $(-3) + (+4) = +1$
- The winning path is the survivor path with the largest path metric (credit)
- Straightforward to generalise to log likelihood ratio based soft-input decoding

Summary

- Convolutional code $CC(n, k, N)$:
  - Code rate $R = k/n$, constraint length N (memory length $N + 1$)
  - Concepts of encoder states, state transitions, and generator polynomials

- $CC(n, k = 1, N)$ encoding:
  - Encoder circuit, table of state transitions and output bits
  - State-transition diagram, and trellis diagram

- $CC(n, k = 1, N)$ decoding:
  - Maximum likelihood sequence decoding, and trellis diagram based Viterbi decoding
  - Hard-input and hard-output decoding, soft-input and hard-output decoding