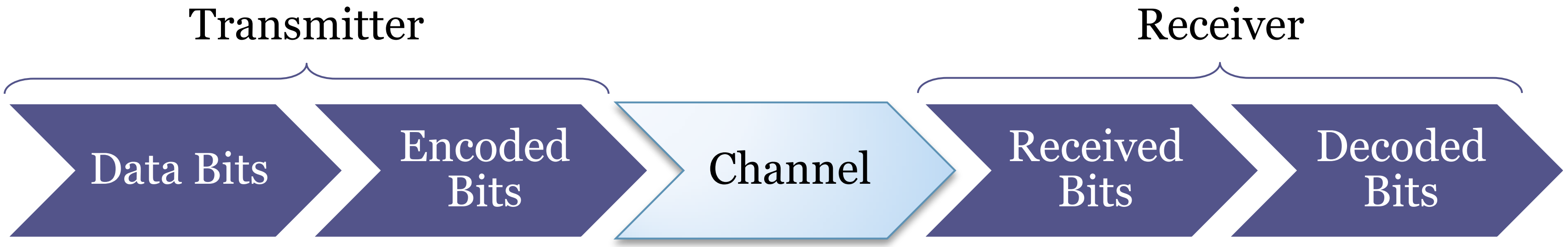
The Viterbi Algorithm

A Dynamic Programming Approach to Decoding Convolutional Codes

What is a Convolutional Code?

Error Correcting Codes

Want to send some data to a receiver, over a lossy Channel



Transmitter

Receiver

Data Bits

Encoded Bits

Channel

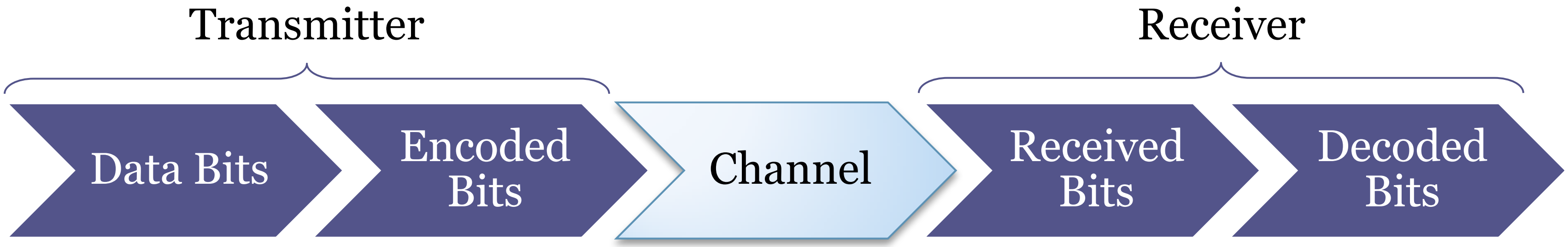
Received Bits

Decoded Bits

Encoder introduces redundant information in the data stream

Decoder uses that redundant information to correct any errors

A Simple Error Correcting Code



Transmitter

Receiver

Data Bits

Encoded Bits

Channel

Received Bits

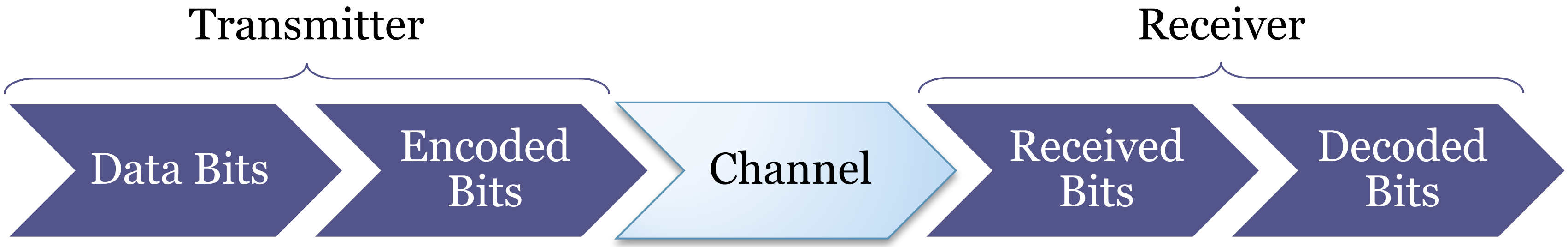
Decoded Bits

`1 0 1…` `111 000 111…` `110 000 011…` `1 0 1…`

Encoder sends each bit three times

Decoder takes majority vote from each triplet. Can correct up to one bit error in each

…But it’s not perfect

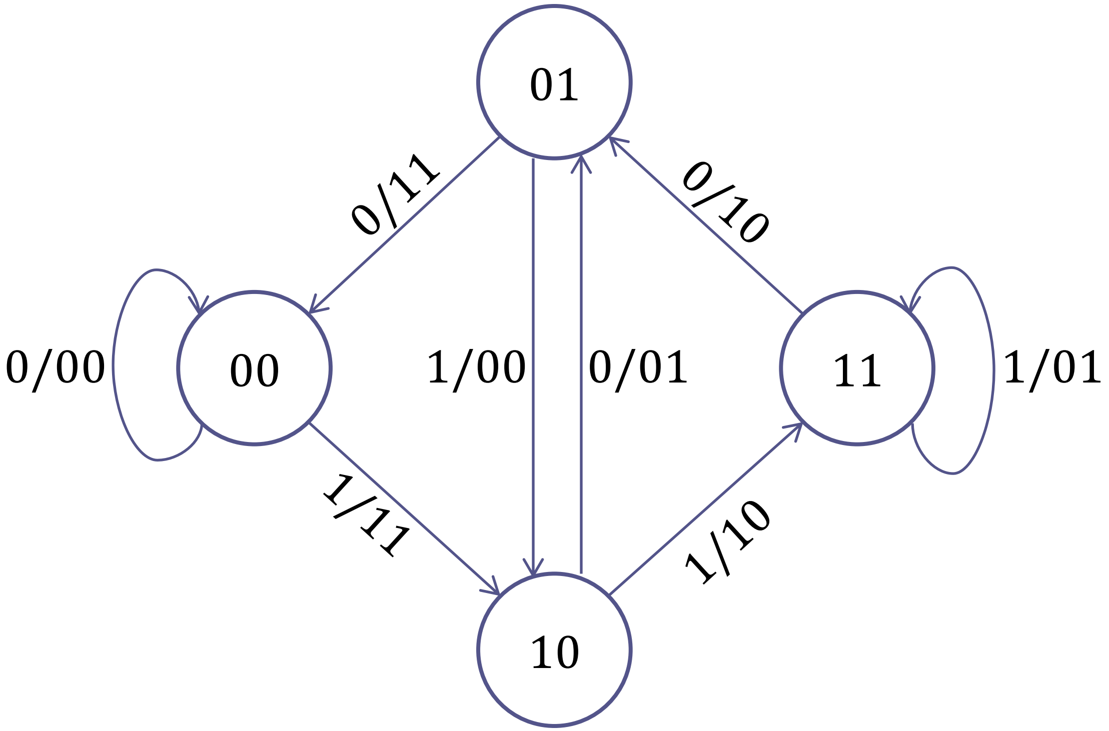


1 0 1… 111 000 111… 110 000 001… 1 0 0…

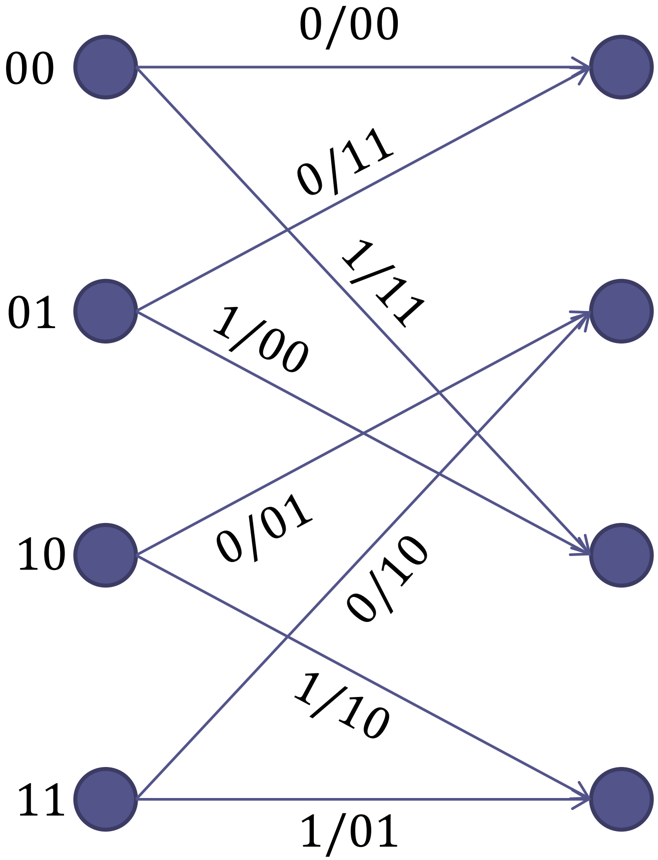
Too many errors in the last bit – decode fails. **How do we do better?**

Convolutional Code

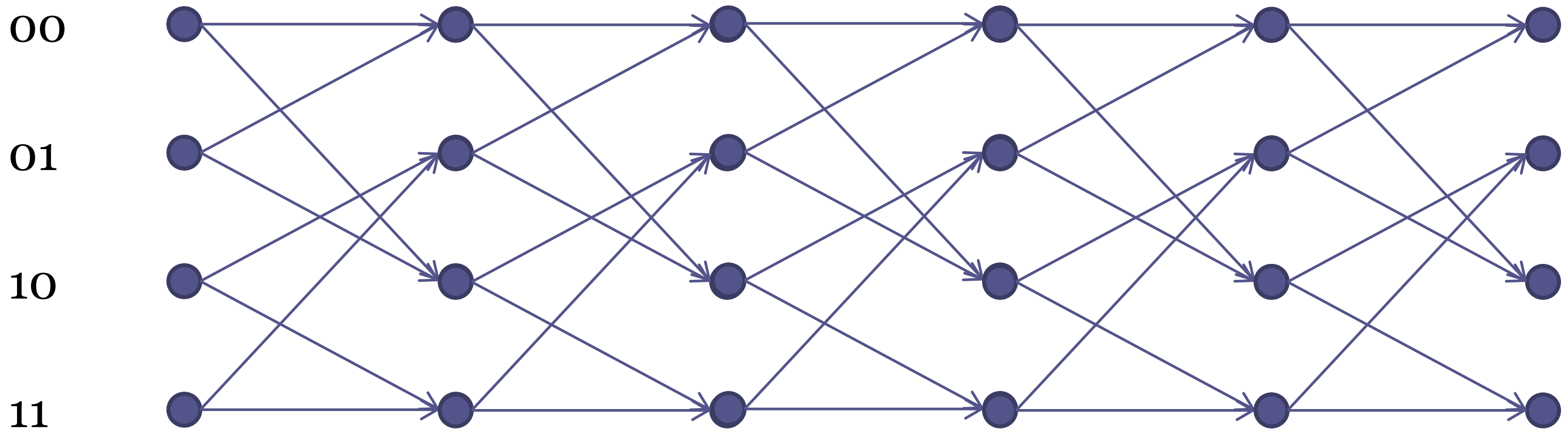
A much more advanced way of introducing complexity.
Relies on a Finite State Machine:



The Trellis Diagram

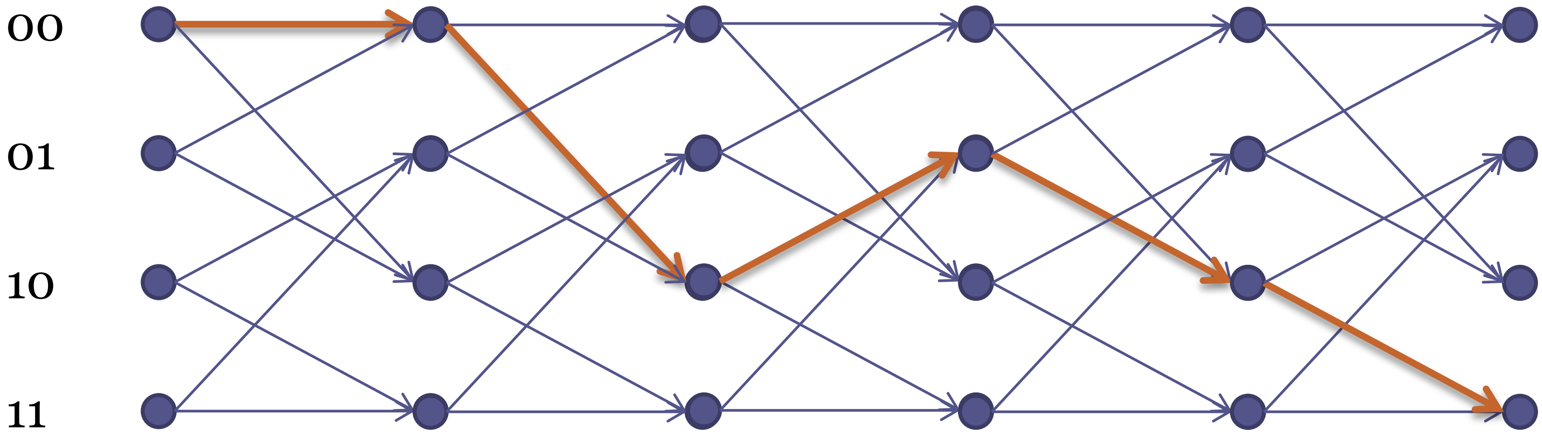


The Trellis Diagram



...So how do you use it?

Data Bits: 0 1 0 1 1



Code Bits: 00 11 01 00 10

Decoding Convolutional Codes

Brute Force – Recursive – Viterbi (Dynamic Programming)

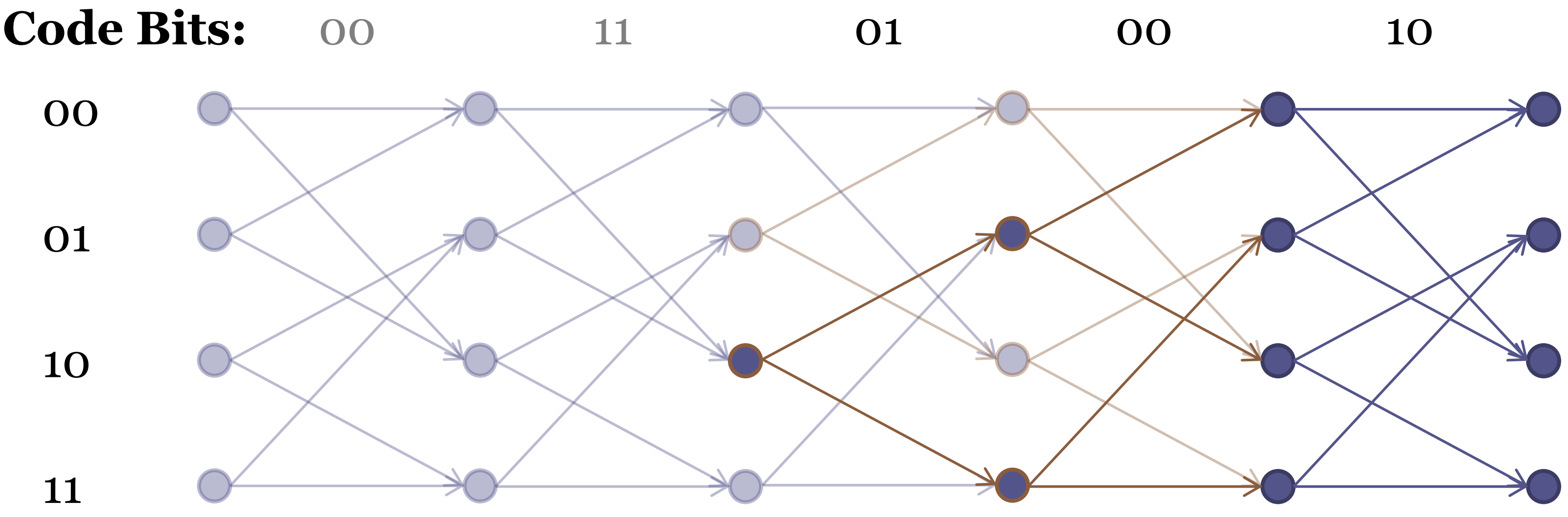
Brute Force

- Iterate through all possible data sequences
  - 2^n possible sequences
- Encode each sequence, measure the difference between the encoded sequence and the received bits
  - $O(n)$
- $O(n2^n)$

Recursive

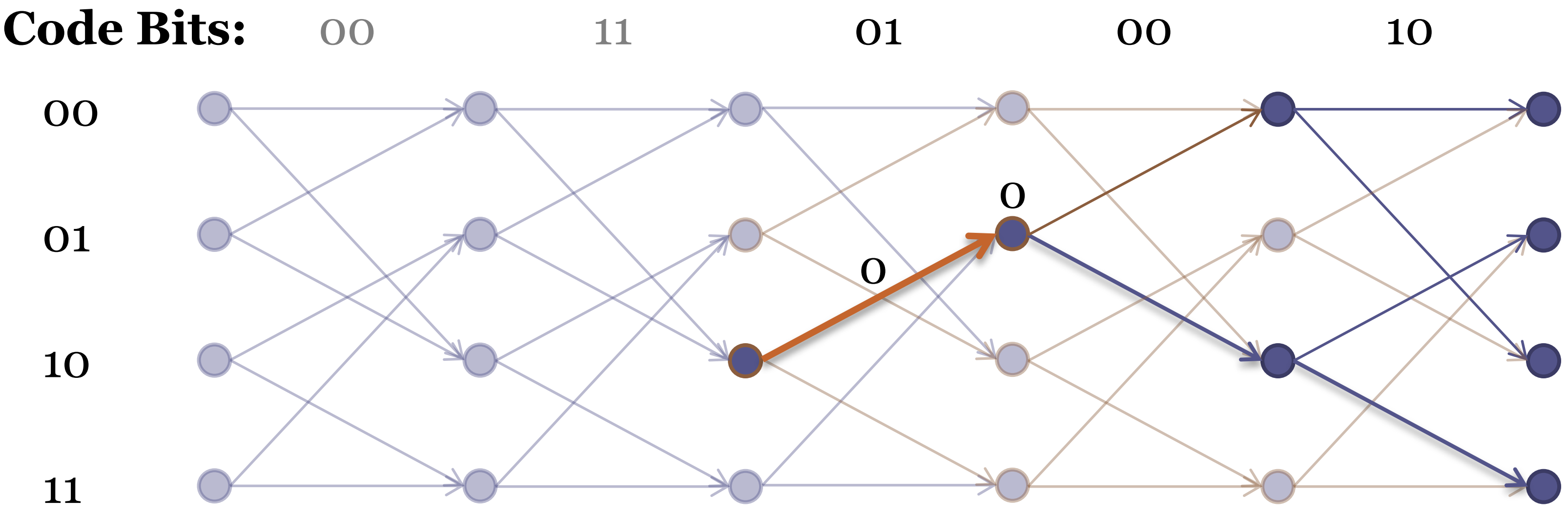
- Idea is to find the path that the encoder took through the trellis
- From any state in the trellis – For each possible path:
  - Repeat the process for the next state – produces a path metric
  - Add this path's metric to the result
- Choose the path which yields the lower path metric

Recursive – Example



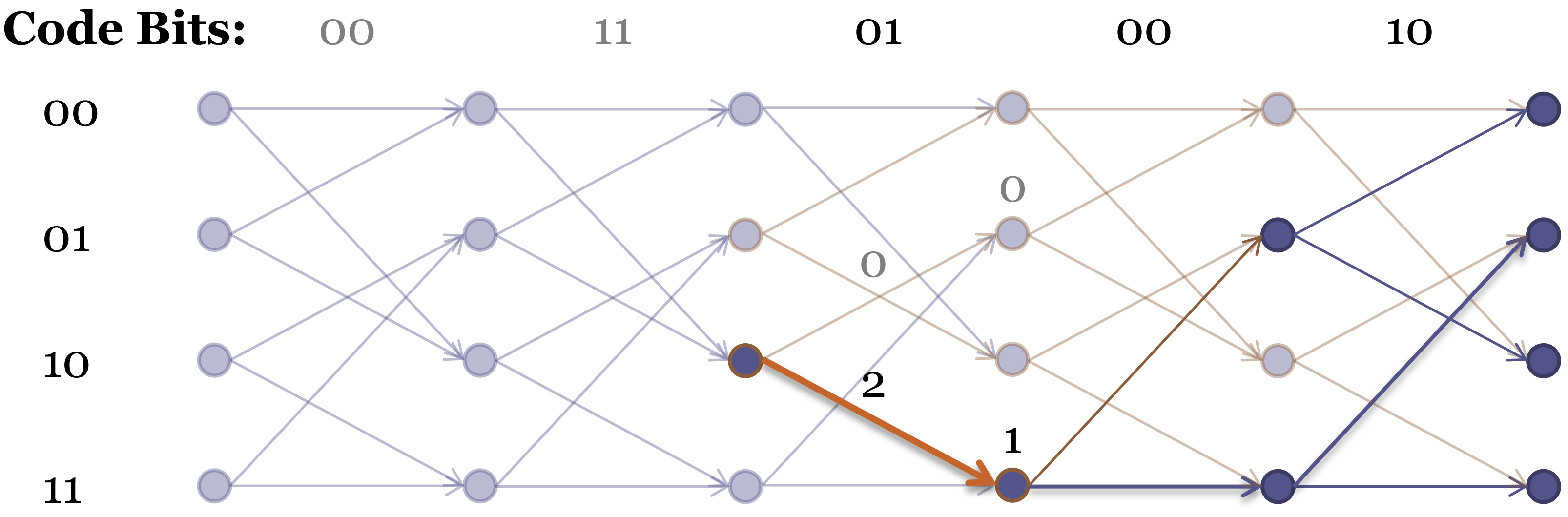
Brute Force – **Recursive** – Viterbi (Dynamic Programming)

Recursive - Example



Brute Force – **Recursive** – Viterbi (Dynamic Programming)

Recursive - Example



Code Bits: 00 11 01 00 10

00

01

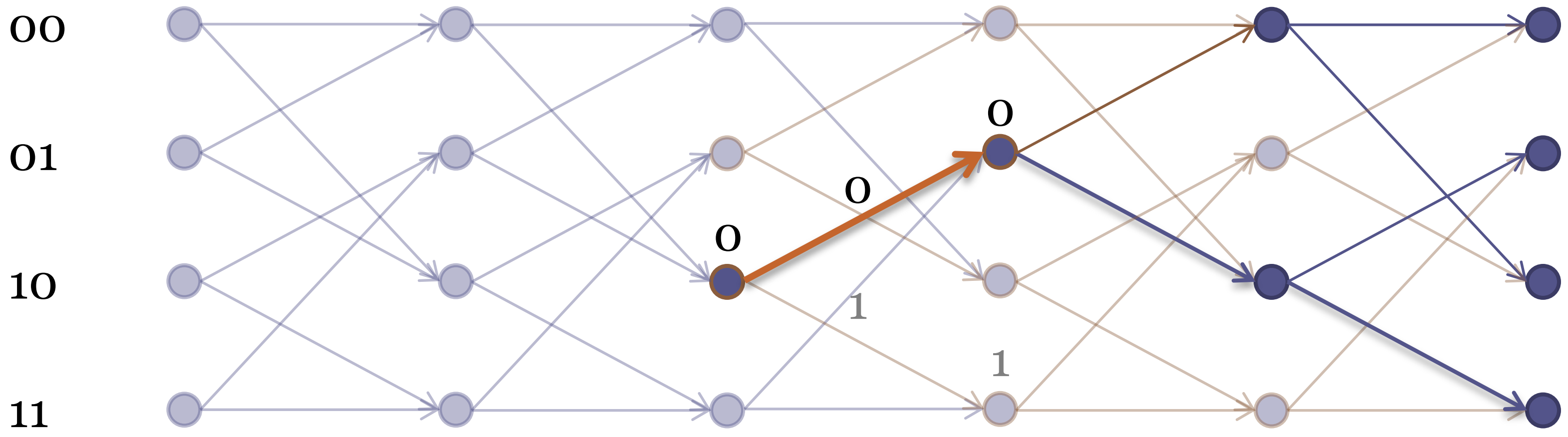
10

11

Brute Force – **Recursive** – Viterbi (Dynamic Programming)

Recursive - Example

Code Bits: 00 11 01 00 10



Recursive - Performance

- To decode – run recursive algorithm starting in state 00
- Runtime is defined recursively:

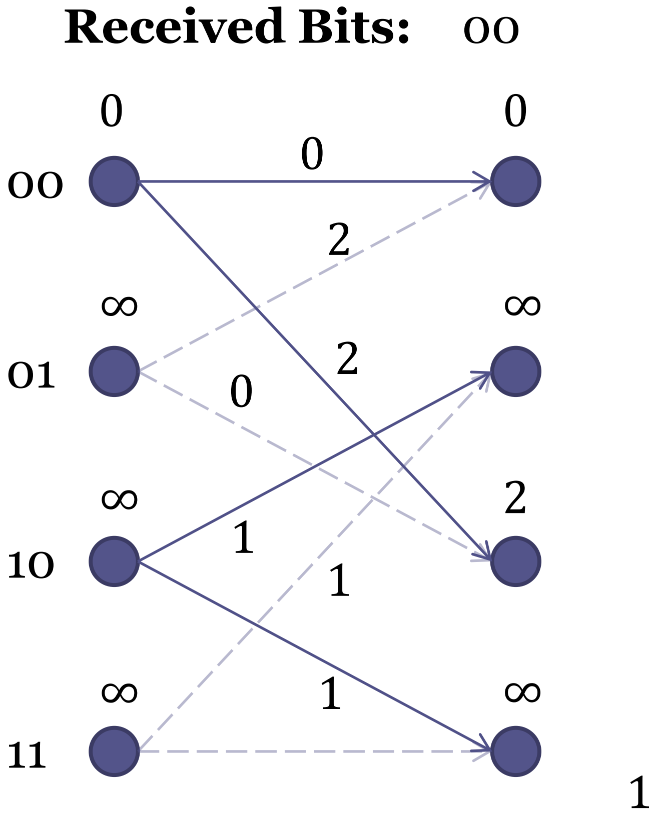
$$T(n) = 2T(n-2) + O(1)$$

- Yields a tree with $n/2$ levels, 2^L nodes on each level
- $O(2^n)$

Viterbi – Basic Idea

- Go through the trellis from left to right
- Select branch “winners” and compute path metrics as we go
  - “Step 3” of a DP algorithm
- Once we reach the end, traceback through the winners to find the optimal path
  - “Step 4” of a DP algorithm

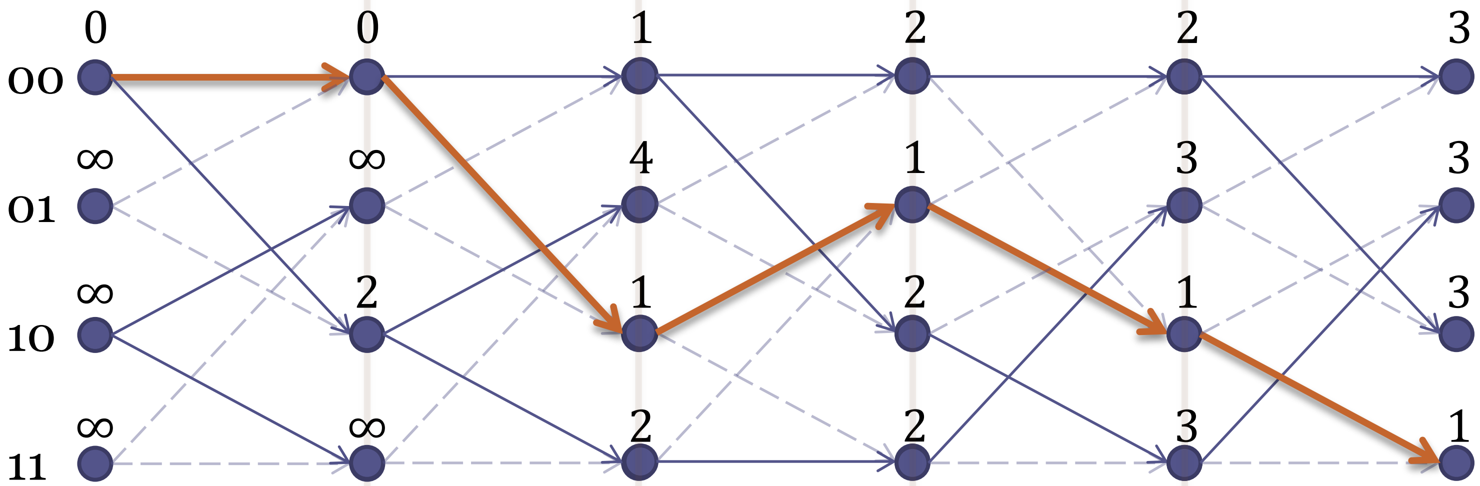
Viterbi – Computing Path Metrics



Brute Force – Recursive – Viterbi (Dynamic Programming)

Viterbi – The traceback

Data:	0	1	0	1	1
Code:	00	11	01	00	10
Received:	00	10	01	00	10



Decoded:	0	1	0	1	1

Viterbi - Performance

- Simply have to iterate through all n trellis steps, and compute path metrics for each state: $O(n)$
- Traceback also just iterates through n trellis steps: $O(n)$
- Overall Performance: $O(n)$

Viterbi – Other Parameters

- Runtime for computing path metrics is also proportional to the number of states
- Other convolutional codes can have more or less states, depending on the constraint length, K
  - Number of states = 2^{K-1}
- Thus Viterbi runtime with respect to constraint length is $O(2^K)$

Demo