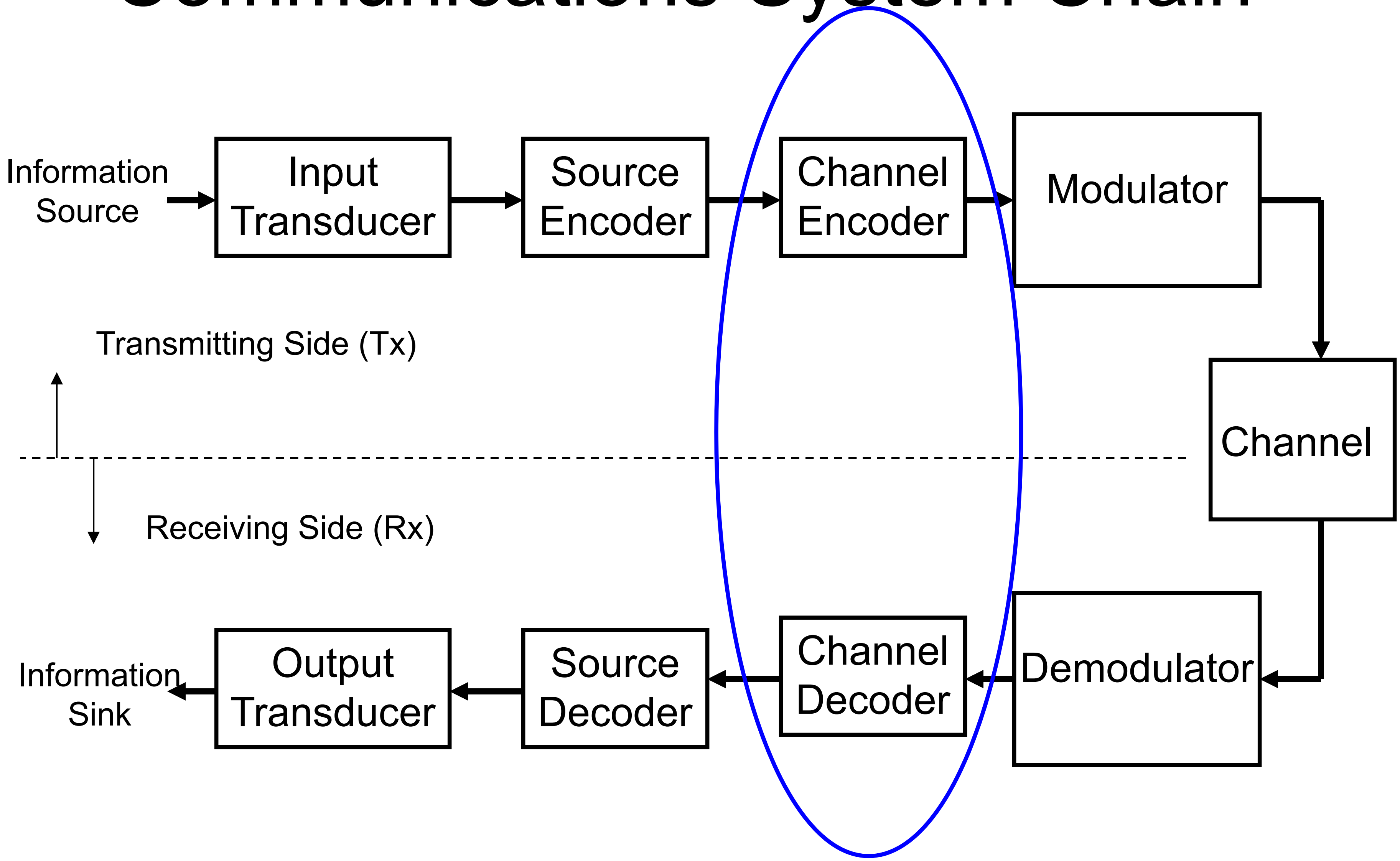
HMM Application in Convolutional Code Decoding

Gaurav Sharma

Objectives

- Illustrate a real-world application of HMMs
- Error correction coding using convolutional codes
- Communications context
- Will provide context (excluding design considerations, beyond scope)

Channel Coding in the Communications System Chain

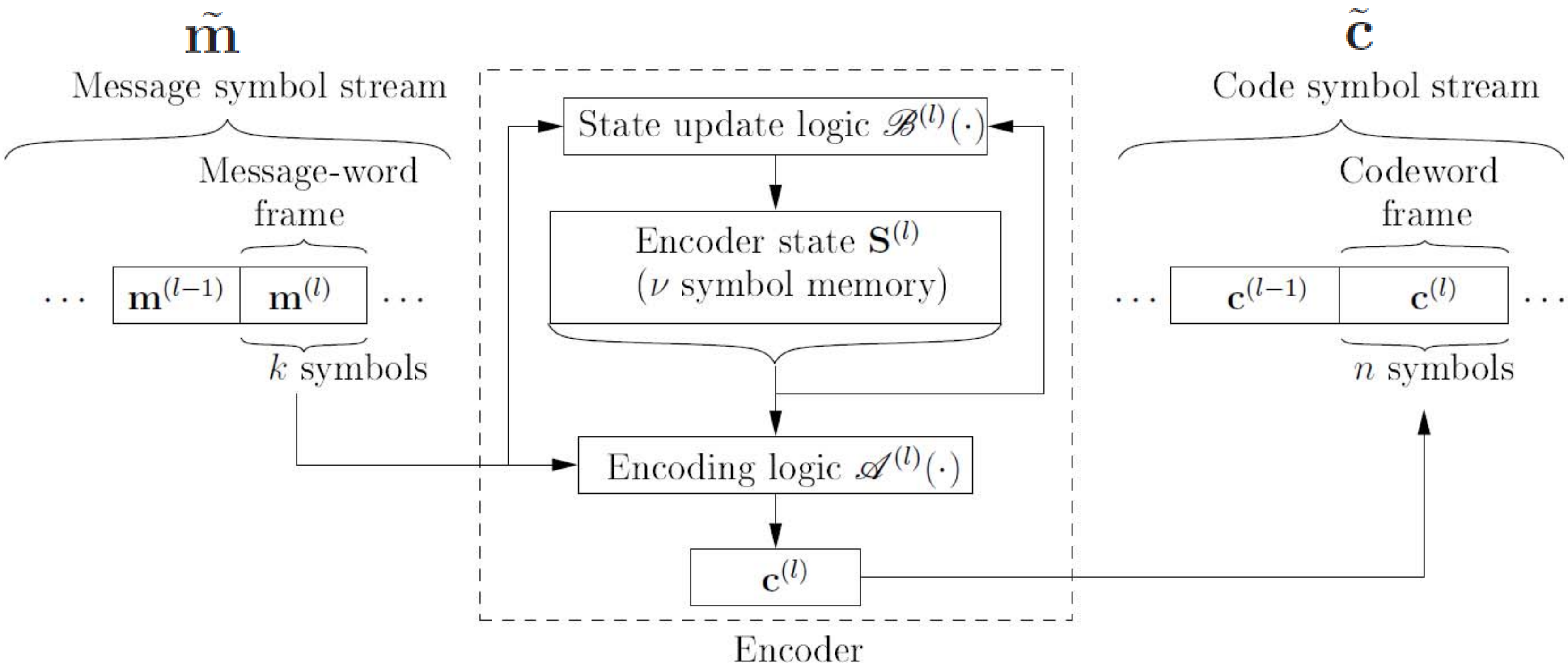


Trellis Codes

- Alternate form of encoding: stream-in stream out
- Output n symbols for every k input symbols
  - Encoding of symbols depends on past blocks of k symbols too
  - Typical k and n quite small as compared to block codes (for low complexity)
  - Typically k is 1 or 2, n values are 2, 3, 4

Trellis Codes

- Encoder = finite-state (causal) machine



Rate = k/n

Trellis Codes

- Encoder: A Finite State Machine
  - Time l, State $S^{(l)}$
  - Input: Message Word: $\mathbf{m}^{(l)} = [m_0^{(l)}, m_1^{(l)}, \ldots, m_{(k-1)}^{(l)}]$
  - Outputs:
    - Codeword: $\mathbf{c}^{(l)} = \mathscr{A}^{(l)}\left(\mathbf{m}^{(l)}, S^{(l)}\right)$
    - Next State: $S^{(l+1)} = \mathscr{B}^{(l)}\left(\mathbf{m}^{(l)}, S^{(l)}\right)$
  - Encoder Functions: $\mathscr{A}^{(l)}(\cdot)$ and $\mathscr{B}^{(l)}(\cdot)$ could be time varying in general
  - Finite state: finite set of possibilities for $S^{(l)}$
    - Complexity is dependent on size of state space
  - Causality is implicit in definition

Convolutional Codes

- Trellis code whose encoder is a linear and time invariant system
  - k-input, n-output: Multiple-input and multiple output LTI system
    - Necessary (though not sufficient) condition:

$$\alpha m^1_{t-ku} + \beta m^2_{t-ku} \rightarrow \quad \alpha c^1_{t-nu} + \beta c^2_{t-nu}$$

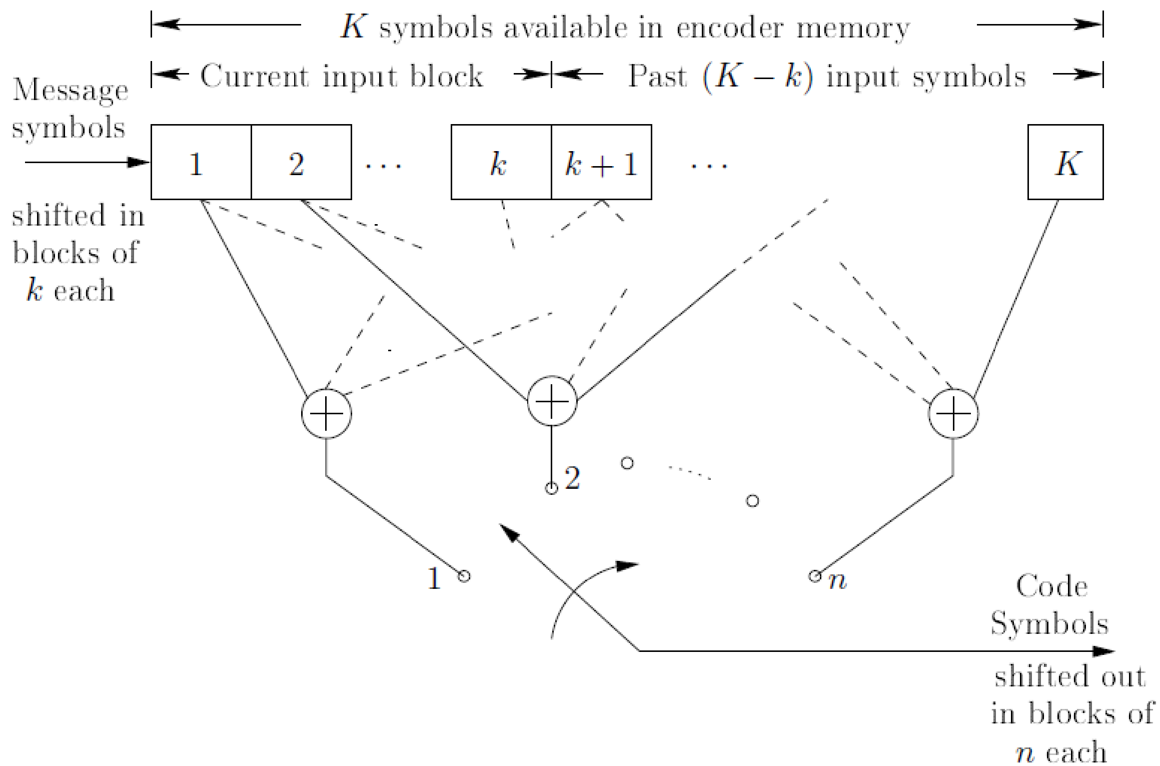
- Most widely studied class of trellis codes
  - Because of analysis, design, and decoding considerations (just like linear block codes)

Convolutional Codes Characterization

- Encoder: Finite State, Linear, Time Invariant, Causal System
  - The contribution of each input to each output can be represented as convolution with a rational impulse response
    - Rational = ratio or two polynomials
      - Recall FIR and IIR Filters and Rational trfr functions
  - Hence the name "Convolutional Code"

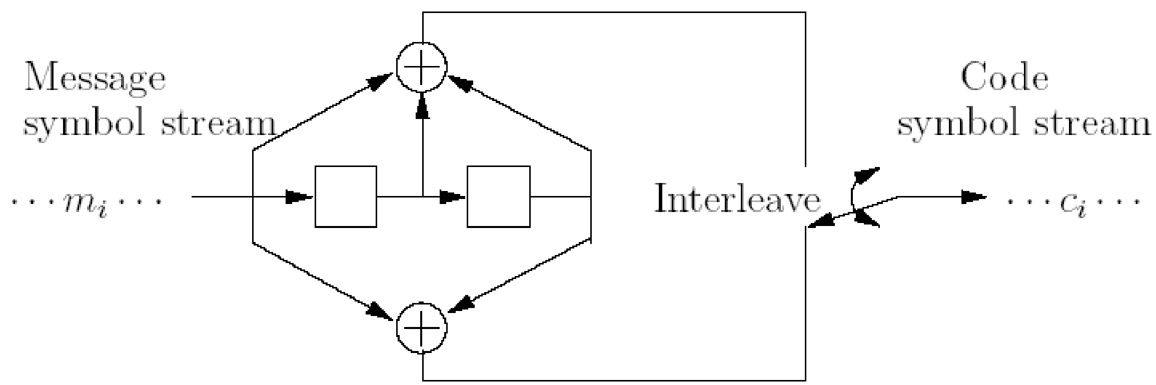
Feedforward Convolutional Encoder: An Obvious Realization

- Feedback free structure

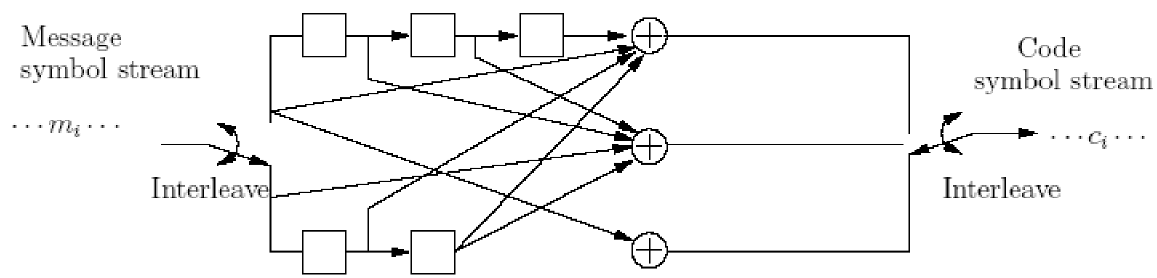


"Constraint length" K = total number of inputs available to encoder

Feedback Free Convolutional Encoders



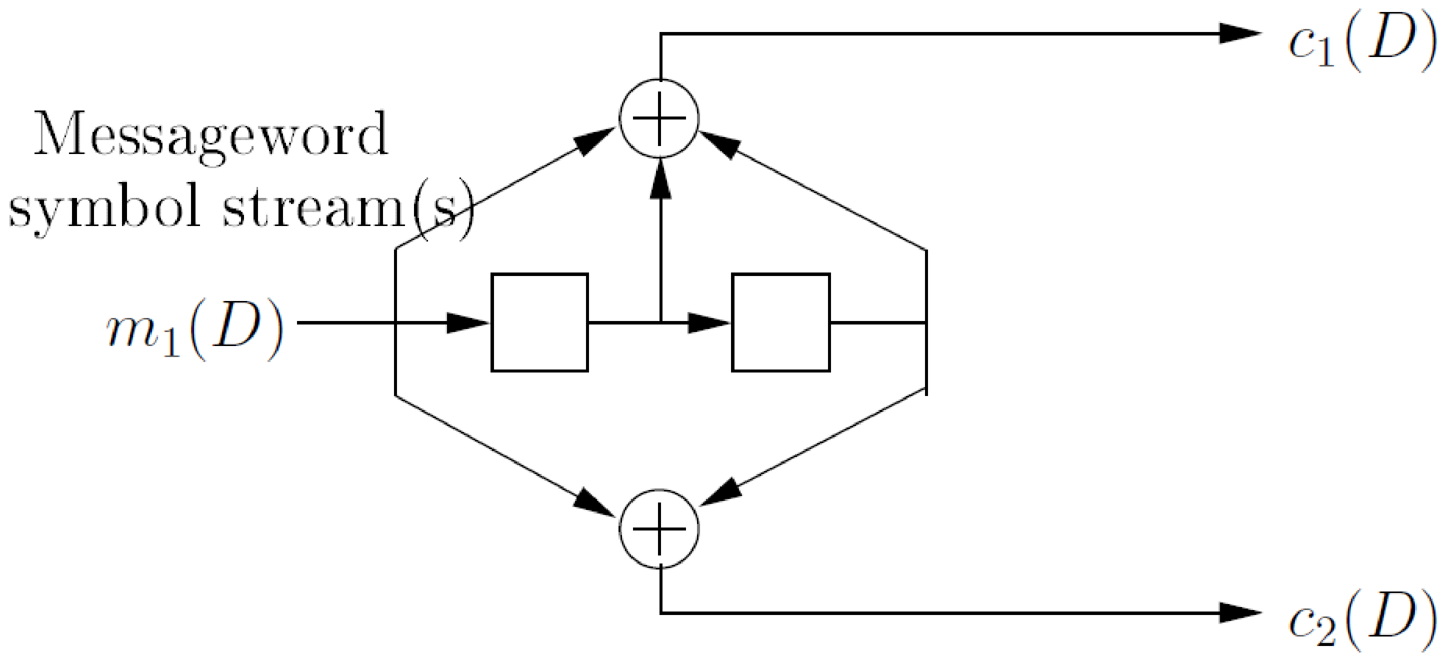
Rate 1/2 Encoder



Rate 2/3 Encoder

A Feedback Free Convolutional Encoder

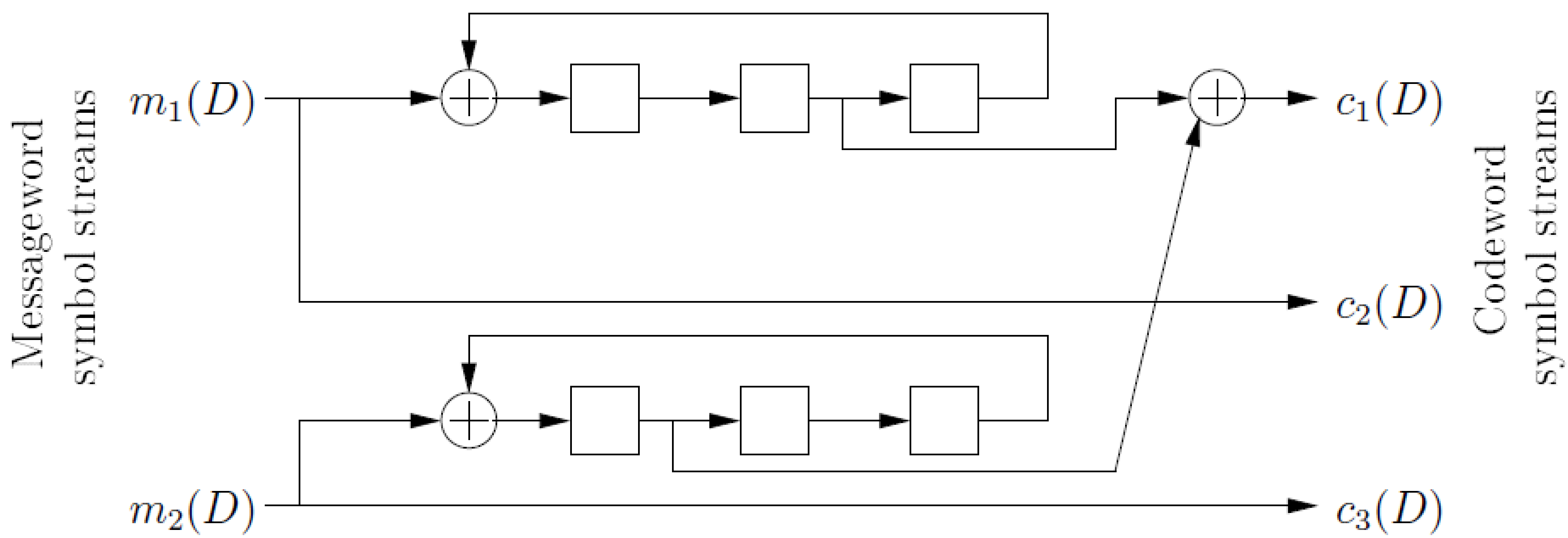
- A (1,2) Convolutional encoder



$$\mathbf{G} = \left[\begin{array}{cc} D^2 + D + 1 & D^2 + 1 \end{array} \right]$$

A Recursive Convolutional Encoder

- A (2,3) Convolutional Encoder



$$\mathbf{G} = \begin{bmatrix} \frac{D^2}{1+D^3} & 1 & 0 \\ \frac{D}{1+D^3} & 0 & 1 \end{bmatrix}$$

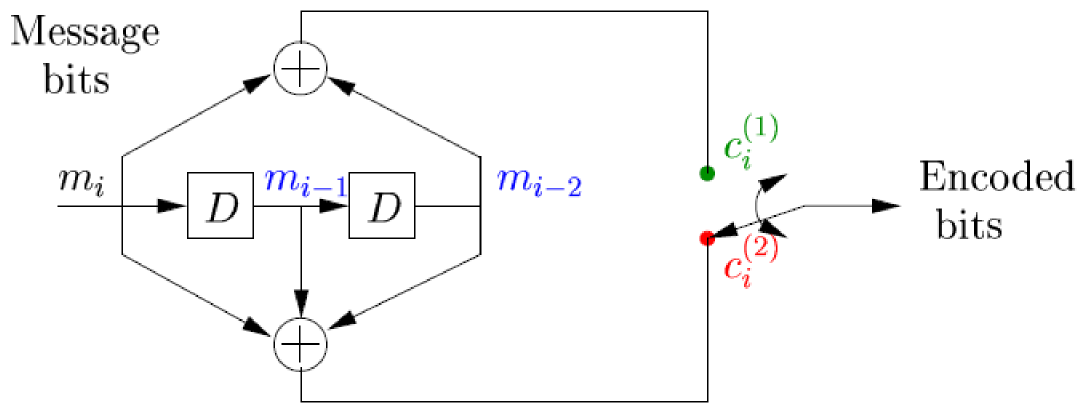
Decoding of Convolutional Codes

- Most naturally discussed in terms of state space realization of the encoder
- Presentation: Example driven to convey intuition
- Readily generalizes to other Convolutional codes and also Trellis codes
  - Nonlinearity and time variation is not a problem
  - Size of state space main issue: Decoding Complexity

G. Sharma

Decoding of Convolutional Codes: Example

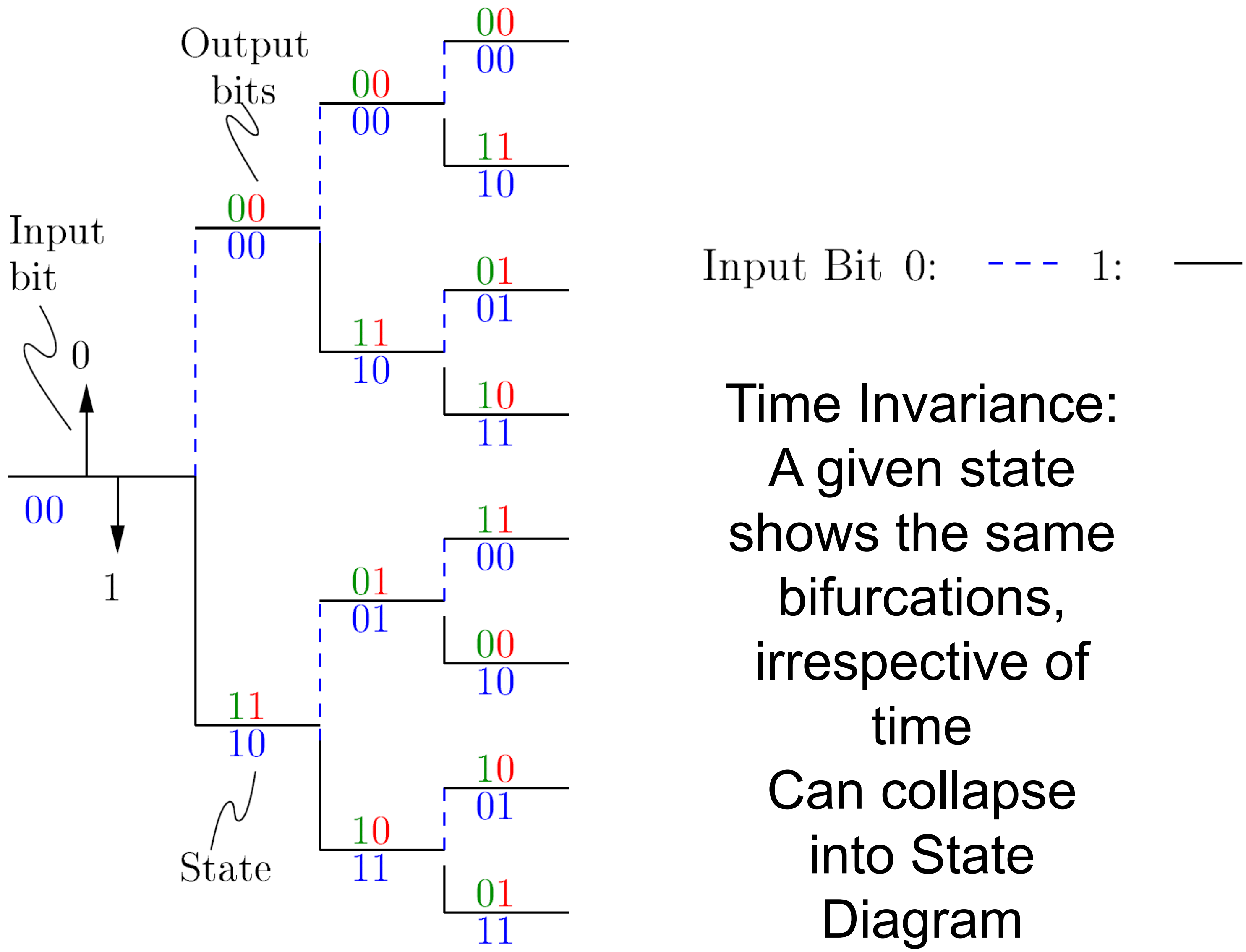
- Shift register encoder representation



  – State = value of stored elements $m_{i-1}m_{i-2}$
    - 4 possible states 00, 01, 10, 11
    - Linearity mandates: initial state=00

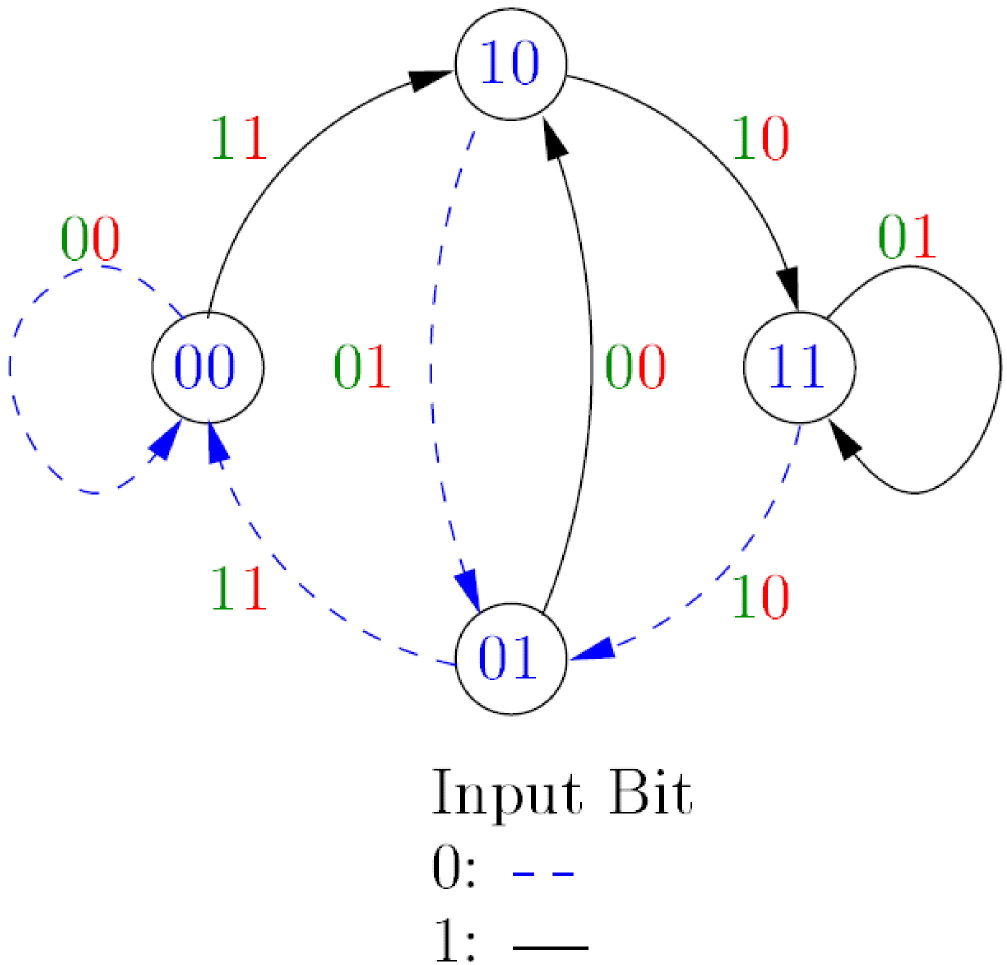
G. Sharma

Tree Diagram of Encoder



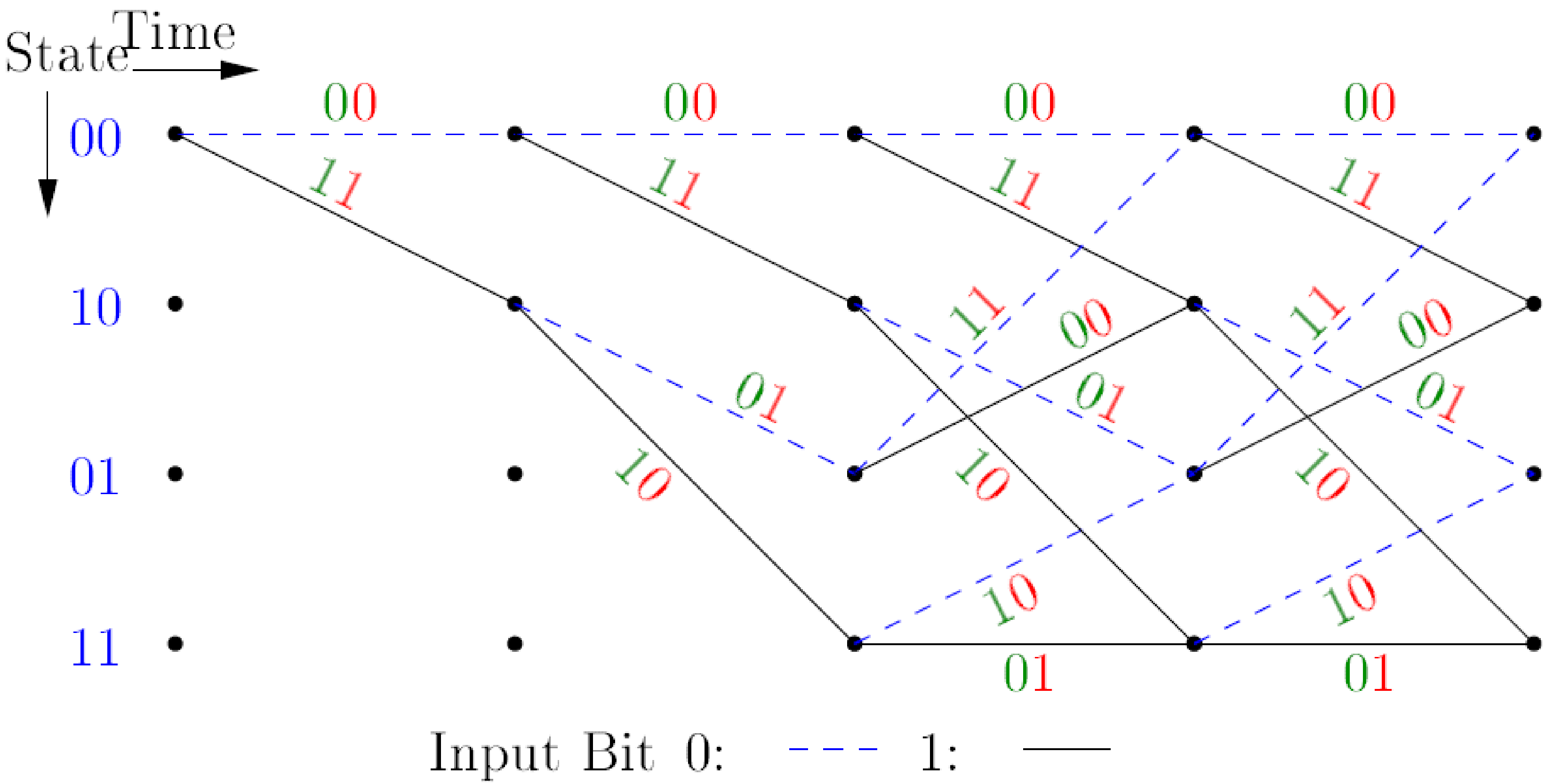
Time Invariance:
A given state shows the same bifurcations, irrespective of time
Can collapse into State Diagram

State Diagram of Encoder



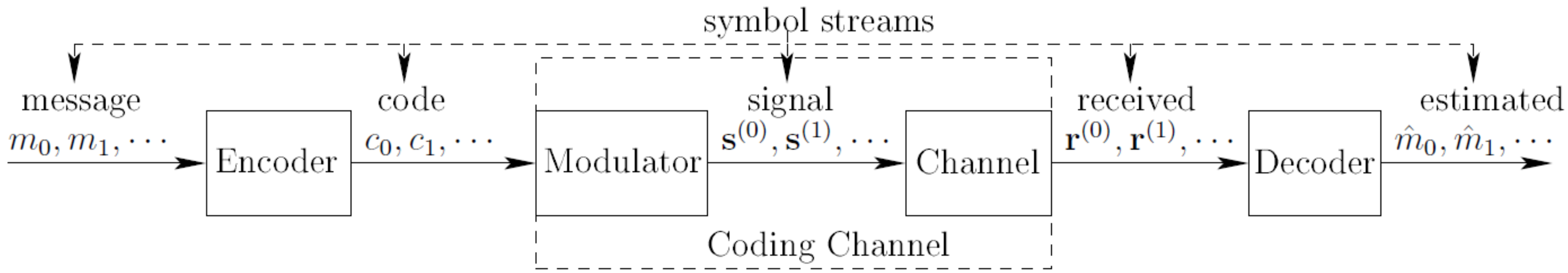
16

Trellis Diagram of Encoder

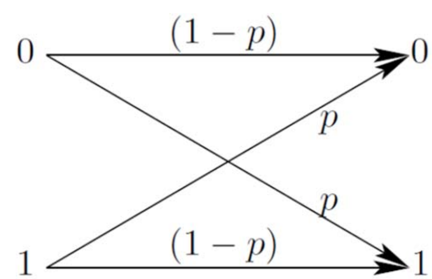


Communications Model

- Broader Model



- Channel introduces uncertainty
  - Several channel models
- Consider only simplest “hard detection” scenario with Binary Symmetric Channel
  - Memoryless channel (Bit errors IID with probability p)



Communications Model

- Received stream corresponds to a hidden Markov model
  - Why? What is hidden?
- What do our three questions of interest map to here and why are they of practical interest?

Convolutional Decoding as an HMM Estimation Algo

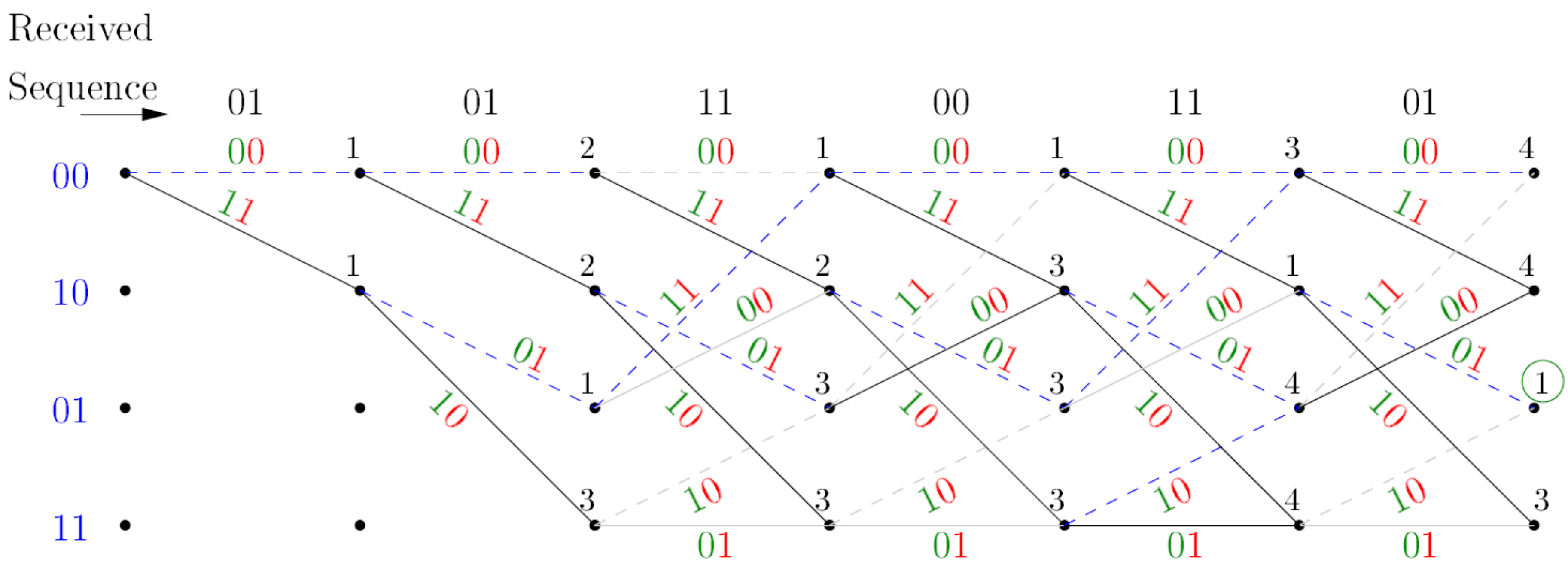
- What are the transition probabilities?
- What are the emission probabilities?
- What are the probabilities of the initial states?

ML Decoding of Convolutional (Trellis) Codes

- Hard Detection Scenario Considered
- ML Decoding ≡ Min Hamming Distance
- Key Idea: Conditioned on state, past is independent of future
  - Trellis allows efficient search for the Min Hamming Distance Codeword
    - Traceback to determine corresponding message

ML Decoding of Convolutional Code on the Trellis

- Retain most likely path to current time for each state: recurse

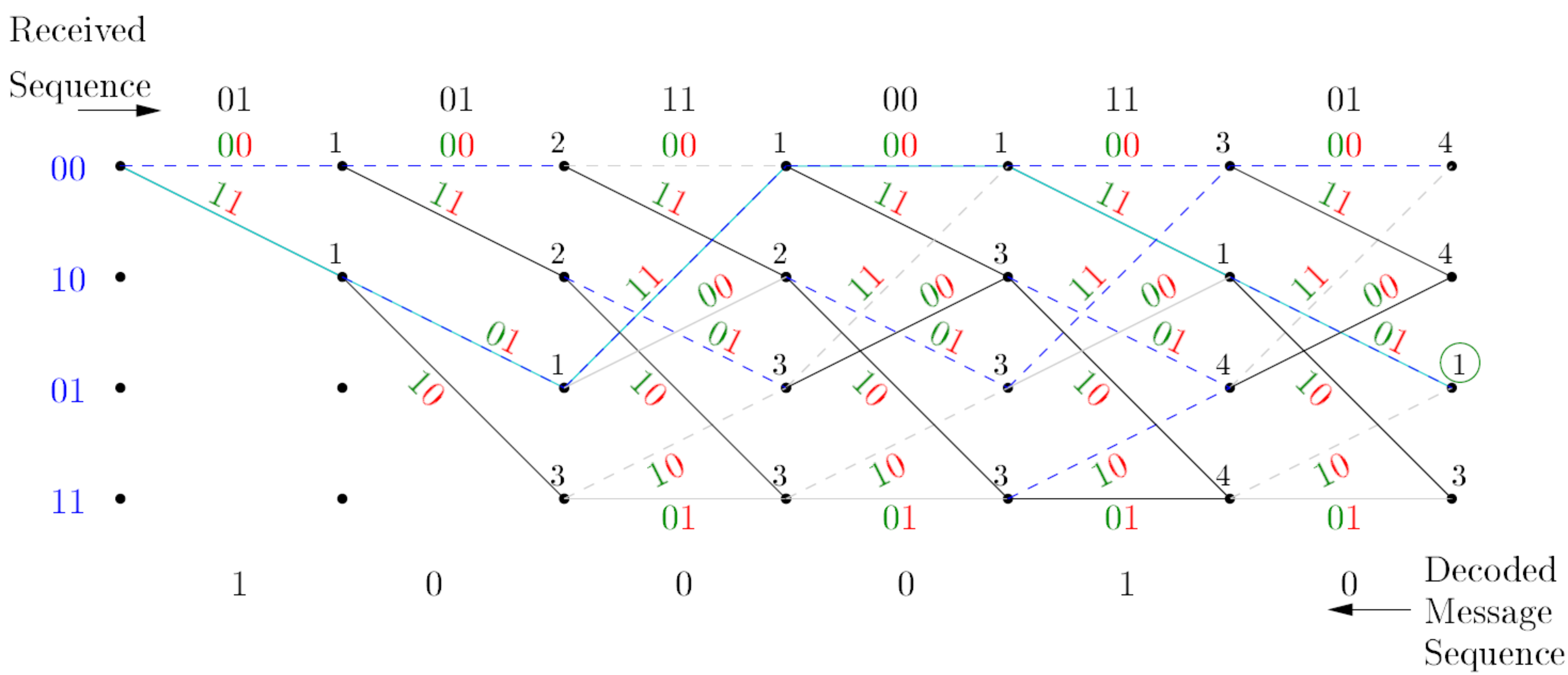


max-sum algorithm

G. Sharma

ML Decoding of Convolutional Code on the Trellis

- Traceback for ML decoded message



G. Sharma

ML Decoding of Trellis Codes: Mathematatcal Formulation

- ML Decoding Rule

$$\tilde{\mathbf{m}}^* = \arg\max_{\mathbf{m}} p\left(\tilde{\mathbf{r}} \mid \tilde{\mathbf{m}}\right)$$

- Consider likelihood upto time l $p\left(\tilde{\mathbf{r}}^{(l)} \mid \tilde{\mathbf{m}}^{(l)}\right)$
- Sequential encoding proc.+Memoryless Chl=>

$$p\left(\tilde{\mathbf{r}}^{(l)} \mid \tilde{\mathbf{m}}^{(l)}\right) = p\left(\tilde{\mathbf{r}}^{(l-1)} \mid \tilde{\mathbf{m}}^{(l-1)}\right) p\left(\mathbf{r}^{(l)} \mid S^{(l)}, \mathbf{m}^{(l)}\right)$$

- Allows ML Decoding by dynamic programming

$$\max_{\tilde{\mathbf{m}}^{(l)}} \log\left(p\left(\tilde{\mathbf{r}}^{(l)} \mid \tilde{\mathbf{m}}^{(l)}\right)\right) =$$

$$\max_{S^{(l)},\mathbf{m}^{(l)}} \left(\max_{\{\tilde{\mathbf{m}}^{(l-1)}:S^{(l)}\}} \log\left(p\left(\tilde{\mathbf{r}}^{(l)} \mid \tilde{\mathbf{m}}^{(l-1)}\right)\right) + \log\left(p\left(\mathbf{r}^{(l)} \mid S^{(l)}, \mathbf{m}^{(l)}\right)\right)\right)$$

Describes Trellis Based Decoding Procedure

G. Sharma

ML Decoding of Trellis Codes: Mathematatcal Formulation

- Start with initial state 00
- L successive messageword frames

$$\tilde{\mathbf{m}}^{(l)} = \mathbf{m}^{(0)}, \mathbf{m}^{(1)}, \ldots, \mathbf{m}^{(L-1)}$$

- Corresponding codeword frames

$$= \mathbf{c}^{(0)}, \mathbf{c}^{(1)}, \ldots, \mathbf{c}^{(L-1)}$$

- and receivedword frames

$$\tilde{\mathbf{r}}_{(0)}^{(L-1)} = \mathbf{r}^{(0)}, \mathbf{r}^{(1)}, \ldots, \mathbf{r}^{(L-1)}$$

$$\tilde{\mathbf{m}}^{*} = \arg\max_{\mathbf{m}} p\left(\tilde{\mathbf{r}} \mid \tilde{\mathbf{m}}\right)$$

ML Decoding of Trellis Codes: AWGN and Hard Decision Chl

- Binary (q-ary) symmetric Channel
  - ML Decoding ≡ Min Hamming Distance
    - Process outlined in example
    - Find path* through Trellis with min Hamming distance to the received sequence
- AWGN Channel
  - ML Decoding ≡ Min Euclidean Distance
    - Minor modification of process outlined in example
    - Find path* through Trellis with min Euclidean distance to the received sequence

*Assumes typical case where path defines message

ML Decoding: Complexity

- Brute Force Search for $\tilde{\mathbf{m}}^* = \arg\max_{\mathbf{m}} p(\tilde{\mathbf{r}} \mid \tilde{\mathbf{m}})$
  - Upto time l=T: 2^T options for $\tilde{\mathbf{m}}^{(l)}$
  - Grows very rapidly with l: $2^{64} \approx 10^{19}$
- Viterbi algorithm (VA):dynamic programming
  - Reduces complexity to linear in T:
    - Number of states x 2^k x T
    - State = binary vector with K entries: 2^K x 2^k x T
  - Still exponential in k and K
    - Need small k and small state size

MAP Decoding of Trellis Codes: Mathematatcal Formulation

- Start with initial state 00
- L successive messageword frames

$$= \mathbf{m}^{(0)}, \mathbf{m}^{(1)}, \ldots, \mathbf{m}^{(L-1)}$$

- Corresponding codeword frames

$$= \mathbf{c}^{(0)}, \mathbf{c}^{(1)}, \ldots, \mathbf{c}^{(L-1)}$$

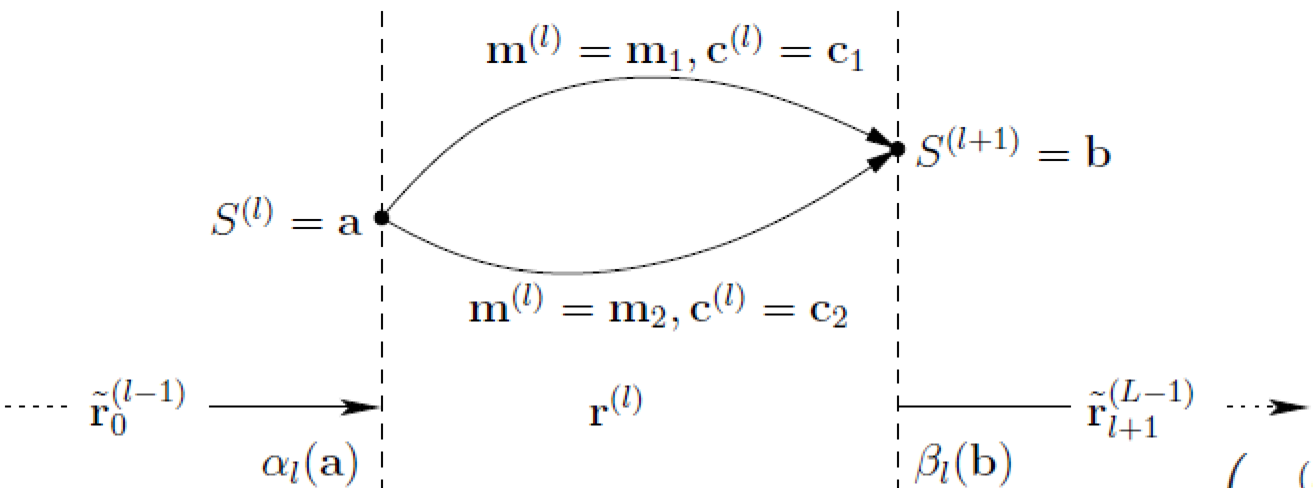
- and receivedword frames

$$\tilde{\mathbf{r}}_{(0)}^{(L-1)} = \mathbf{r}^{(0)}, \mathbf{r}^{(1)}, \ldots, \mathbf{r}^{(L-1)}$$

$$\hat{\mathbf{m}}^{(l)} = \arg\max_{\mathbf{m}} p\left(\mathbf{m}^{(l)} = \mathbf{m} \mid \tilde{\mathbf{r}}_{(0)}^{(L-1)}\right)$$

MAP Decoding

- Dynamic programming analogous to ML decoding: forward-backward/BCJR Algo



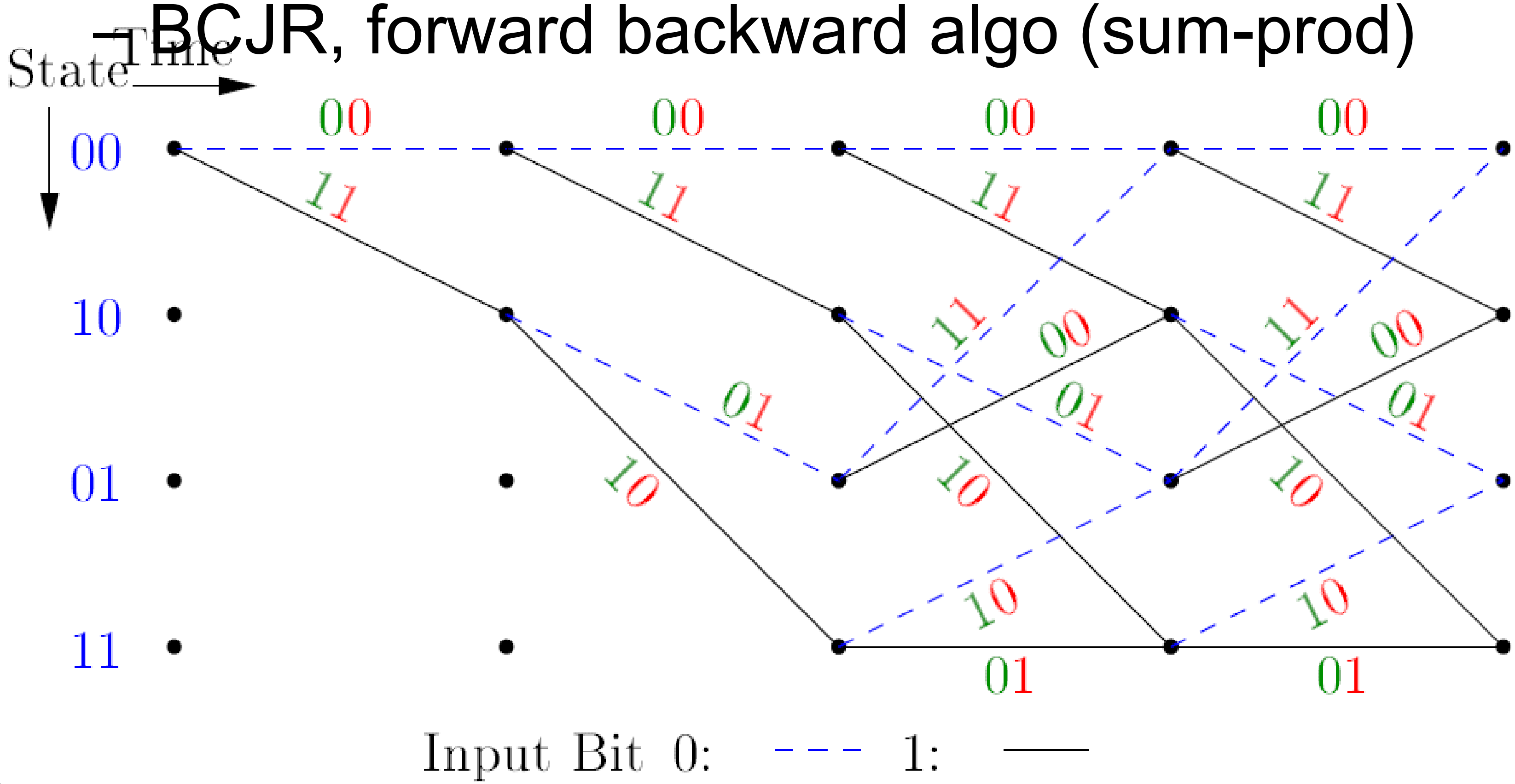
$$\alpha_l(\mathbf{a}) \stackrel{\text{def}}{=} p\left(\tilde{\mathbf{r}}_{(0)}^{(l-1)}, S^{(l)} = \mathbf{a}\right)$$

$$\beta_l(\mathbf{a}) \stackrel{\text{def}}{=} p\left(\tilde{\mathbf{r}}_{(l)}^{(L-1)} \mid S^{(l)} = \mathbf{a}\right)$$

$$p\left(\mathbf{m}^{(l)} = \mathbf{m}, \tilde{\mathbf{r}}_{(0)}^{(L)}\right) = \sum_{\mathbf{a},\mathbf{b} \in \mathcal{S}} \alpha_l(\mathbf{a}) p\left(\mathbf{r}^{(l)}, \mathbf{m}^{(l)} = \mathbf{m}, S^{(l+1)} = \mathbf{b} \mid S^{(l)} = \mathbf{a}\right) \beta_{l+1}(\mathbf{b})$$

MAP Decoding of Trellis Codes

- Propagate probabilities on trellis to obtain symbol MAP probabilities
  - BCJR, forward backward algo (sum-prod)



MAP Decoding

- Forward-backward/BCJR Algo
  Recursions

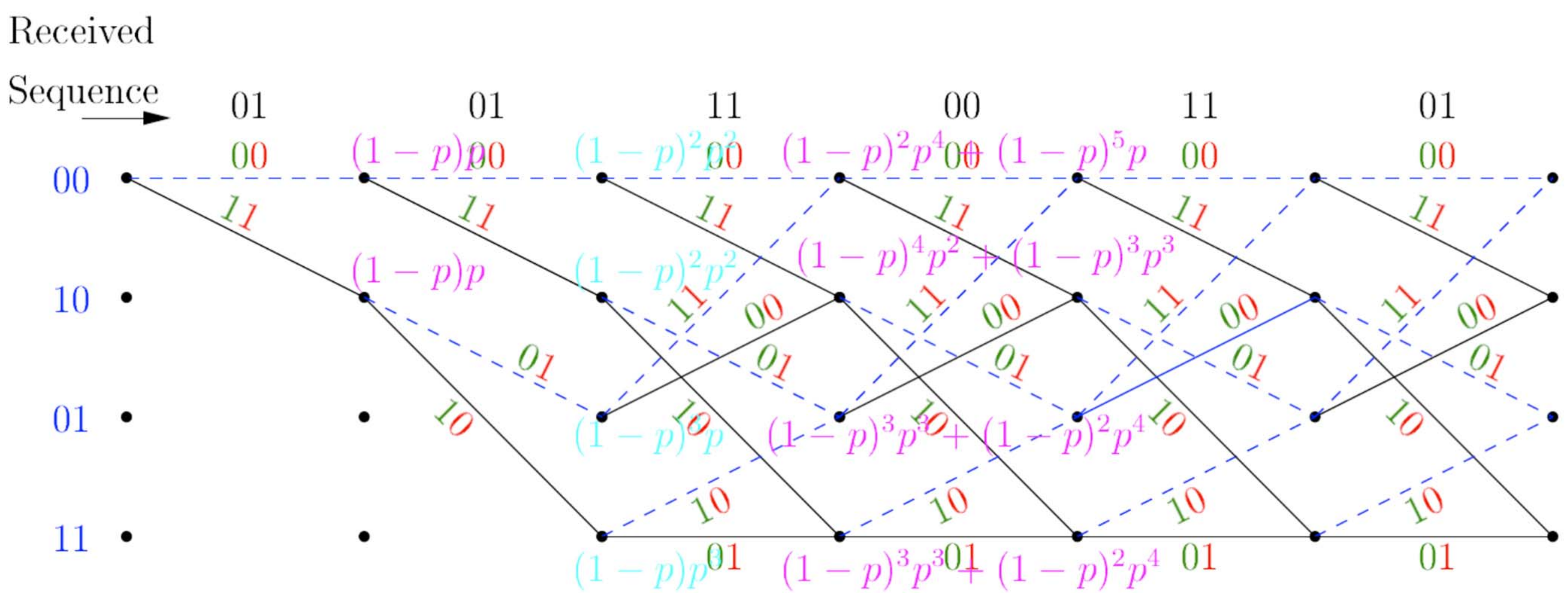
$$
\begin{aligned}
\alpha_l(i) &= p\left(\tilde{\mathbf{r}}_{(0)}^{(l-1)}, S^{(l)} = i\right) \\
&= \sum_{j \in \mathcal{S}} \alpha_{l-1}(j) p\left(\mathbf{r}^{(l-1)}, S^{(l)} = i \mid S^{(l-1)} = j\right) \\
&= \sum_{j \in \mathcal{S}} \alpha_{l-1}(j) \gamma_{l-1}(i, j) \\
\beta_l(i) &= p\left(\tilde{\mathbf{r}}_{(l)}^{(L-1)} \mid S^{(l)} = i\right) \\
&= \sum_{j \in \mathcal{S}} p\left(\mathbf{r}^{(l)}, S^{(l+1)} = j \mid S^{(l)} = i\right) p\left(\tilde{\mathbf{r}}_{(l+1)}^{(L-1)} \mid S^{(l+1)} = j\right) \\
&= \sum_{j \in \mathcal{S}} \gamma_l(j, i) \beta_{l+1}(j).
\end{aligned}
$$

MAP Decoding of Trellis Codes: Example

- Channel: BSC(p)
- Forward Recursion

$$\boxed{\alpha_l(\mathbf{a}) \quad \stackrel{\text{def}}{=} \quad p\left(\tilde{\mathbf{r}}_{(0)}^{(l-1)}, S^{(l)} = \mathbf{a}\right)}$$

$$\beta_l(\mathbf{a}) \quad \stackrel{\text{def}}{=} \quad p\left(\tilde{\mathbf{r}}_{(l)}^{(L-1)} \mid S^{(l)} = \mathbf{a}\right)$$

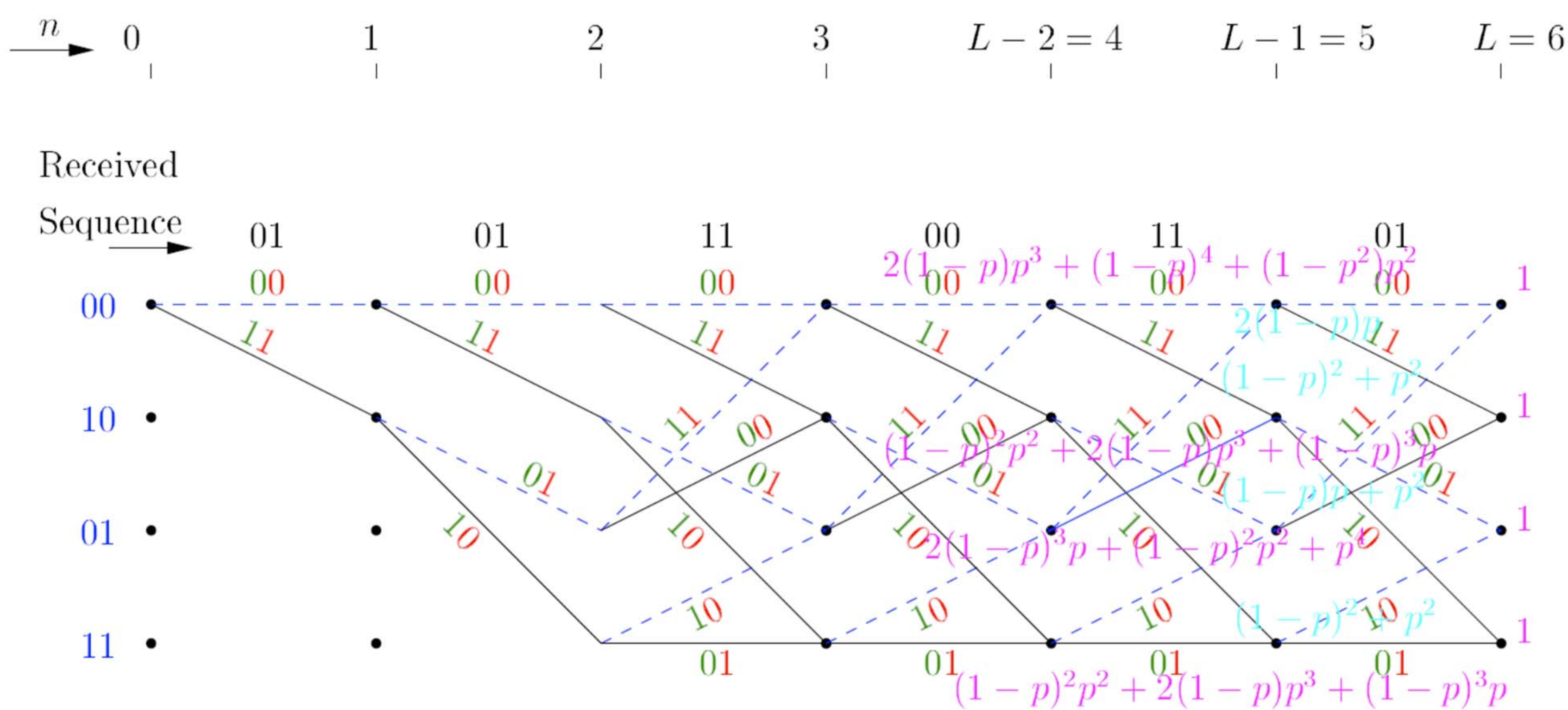


MAP Decoding of Trellis Codes: Example

- Channel: BSC(p)
- Backward Recursion

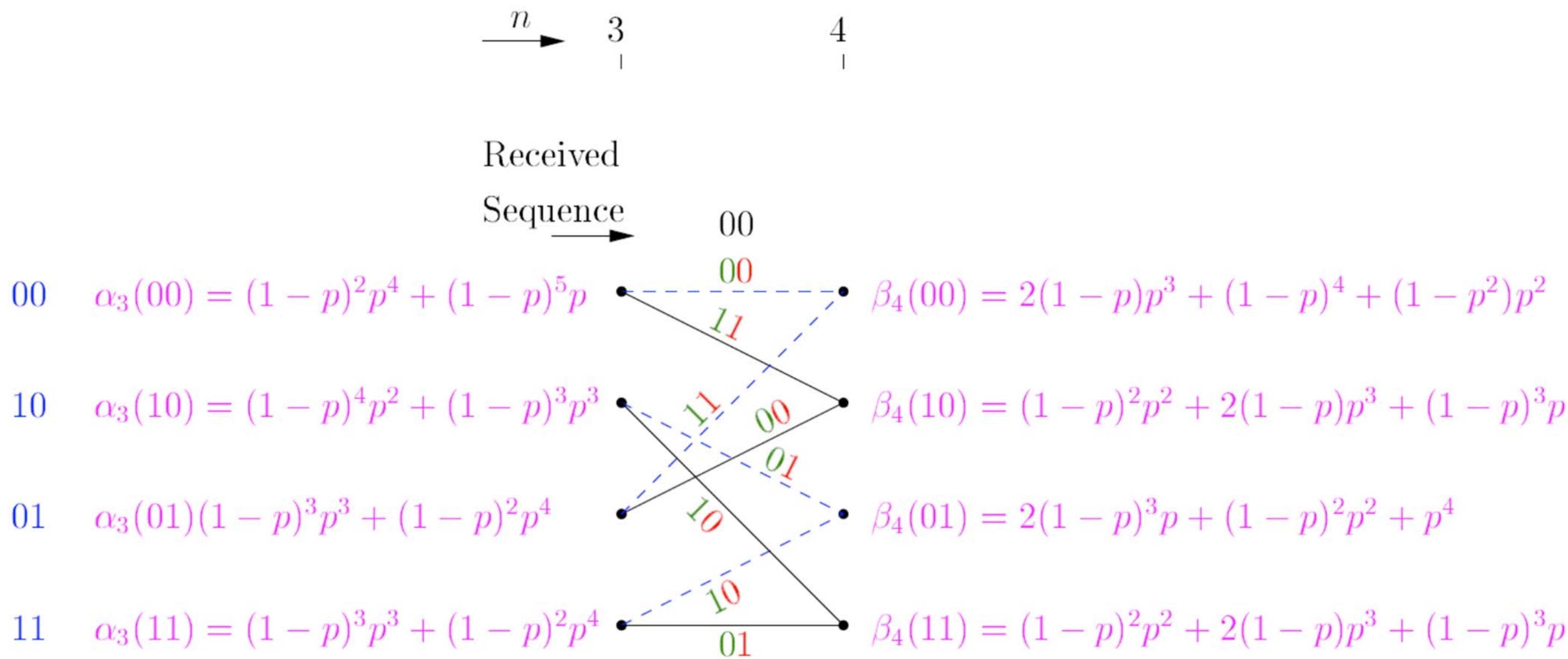
$$\alpha_l(\mathbf{a}) \stackrel{\text{def}}{=} p\left(\tilde{\mathbf{r}}_{(0)}^{(l-1)}, S^{(l)} = \mathbf{a}\right)$$

$$\boxed{\beta_l(\mathbf{a}) \stackrel{\text{def}}{=} p\left(\tilde{\mathbf{r}}_{(l)}^{(L-1)} \mid S^{(l)} = \mathbf{a}\right)}$$



MAP Decoding of Trellis Codes: Example

- Bit-wise Posterior Probability Computation



MAP Decoding of Trellis Codes: Example

- Bit-wise Posterior Probability Computation
  - Sum over 0 (dashed) transitions

$$
\begin{aligned}
& p\left(\mathbf{m}^{(3)}=0, \tilde{\mathbf{r}}_{(0)}^{(L)}=010111000001\right) \\
= & \sum_{\mathbf{a}, \mathbf{b} \in \mathcal{S}} \alpha_3(\mathbf{a}) p\left(\mathbf{r}^{(l)}=00, \mathbf{m}^{(l)}=0, S^{(l+1)}=\mathbf{b} \mid S^{(l)}=\mathbf{a}\right) \beta_4(\mathbf{b}) \\
= & \alpha_3(00)(1-p)^2 \beta_4(00)+\alpha_3(10)(1-p) p \beta_4(01) \\
& +\alpha_3(01) p^2 \beta_4(00)+\alpha_3(11) p(1-p) \beta_4(01)
\end{aligned}
$$

MAP Decoding of Trellis Codes: Example

- Bit-wise Posterior Probability Computation
  - Sum over 1 (solid) transitions

$$p\left(\mathbf{m}^{(3)} = 1, \tilde{\mathbf{r}}_{(0)}^{(L)} = 010111000001\right)$$

$$= \sum_{\mathbf{a},\mathbf{b}\in\mathcal{S}} \alpha_3(\mathbf{a}) p\left(\mathbf{r}^{(l)} = 00, \mathbf{m}^{(l)} = 1, S^{(l+1)} = \mathbf{b} \mid S^{(l)} = \mathbf{a}\right) \beta_4(\mathbf{b})$$

$$= \alpha_3(00) p^2 \beta_4(10) \alpha_3(10) p(1-p) \beta_4(11)$$

$$+ \alpha_3(01)(1-p)^2 \beta_4(10) + \alpha_3(11)(1-p) p \beta_4(11)$$

MAP Decoding of Covolutional Code

- Simplification for binary convolutional codes
  - Only logs of posterior probability ratios tracked (1/2 as many computations)
  - Decision Rule

$$log\left(\frac{p\left(\mathbf{m}^{(l)}=0,\tilde{\mathbf{r}}_{(0)}^{(L)}\right)}{p\left(\mathbf{m}^{(l)}=1,\tilde{\mathbf{r}}_{(0)}^{(L)}\right)}\right)\ \underset{1}{\overset{0}{\gtrless}}\ 0$$

G. Sharma

MAP Decoding of Convolutional Codes

- Notes
  - MAP Decoding approx. 3x as much computation as ML decoding
  - Error performance similar
  - Largely ignored for several decades, until ..

- Turbo decoding builds on MAP decoding of convolutional codes
  - Near capacity achieving performance
  - An instance of belief propagation

Concepts

- Received data from transmitting a convolutional encoded stream overa a memoryless channel is exactly described by an HMM
- Decoding for a convolutional code can be formulated in terms of HMM Viterbi and Forward-Backward Algorithms
- Symbol MAP decoding forms basis of Turbo-Decoding