

# Not Your Father's Coding Theory

G. David Forney, Jr.

MIT  
Cambridge MA 02139 USA

Andrew J. Viterbi Distinguished Lecture in Communications  
University of Southern California  
Los Angeles  
20 November 2003

*G. David Forney, Jr.*

# Synopsis

## 1948: Shannon's challenge

- There exist codes that can achieve arbitrarily reliable transmission at any rate  $R$  less than the channel capacity  $C$  (the **Shannon limit**)
  - Nonconstructive
  - No consideration of encoding and decoding complexity

## 1948-1993: a half-century of effort

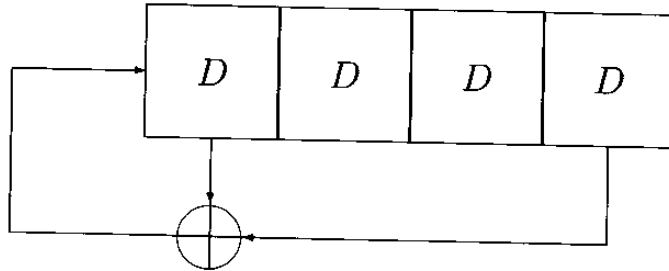
- What kinds of structure can codes have and still have ...
  - Feasible encoding and decoding complexity
  - Capacity-approaching performance
- Algebraic structure
  - Linear codes are fine (Elias)
  - Some excellent classes of codes (Reed-Muller, Reed-Solomon)
  - But ultimately not the way to get to the Shannon limit
- Dynamical structure
  - Convolutional codes are fine (Elias)
  - Often used in practice (with Viterbi algorithm)
  - But don't get to the Shannon limit either
- Concatenated codes (1965)
  - Convolutional inner code, algebraic outer code
  - By 1992, within 3 dB of Shannon limit on AWGN channel
- Wozencraft–Jacobs: “All codes are good, except those we know of”

## 1993-2003: a new generation of codes

- Turbo codes (1993)
  - “Parallel” concatenation of two convolutional codes
  - Iterative, message-passing decoding
  - Performance within 1 dB of Shannon limit on AWGN channel
- Rediscovery (1994-95) of low-density parity-check codes (1962)
  - Codes defined on sparse graphs
  - Iterative “*a posteriori* probability” decoding
  - Comparable performance to turbo codes
- Many variations (1995– )
  - Common structure: codes on graphs
  - Common decoding algorithm: iterative message-passing sum-product algorithm (belief propagation)

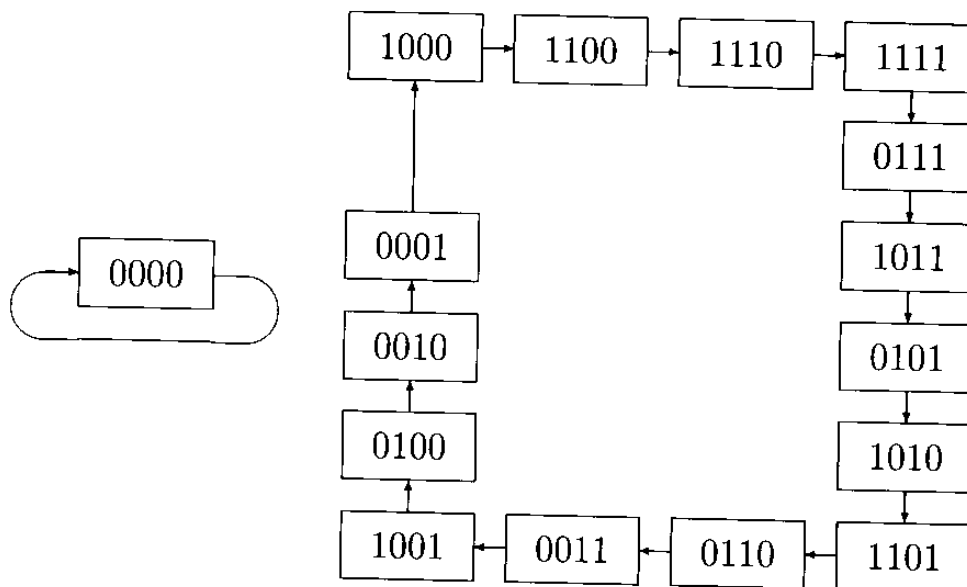
# Codes with dynamical structure

First example: Maximum-length shift-register codes



- Operation: load with a binary  $k$ -tuple, clock  $2^k - 1$  times
- Encoding complexity: a  $k$ -stage binary shift register
- Algebraic structure: a cyclic binary linear  $(2^k - 1, k, 2^{k-1})$  block code
- Geometric structure: maps to a  $2^k$ -simplex in Euclidean space
- Dynamical structure: a linear (over  $\mathbb{F}_2$ ) finite-state machine:

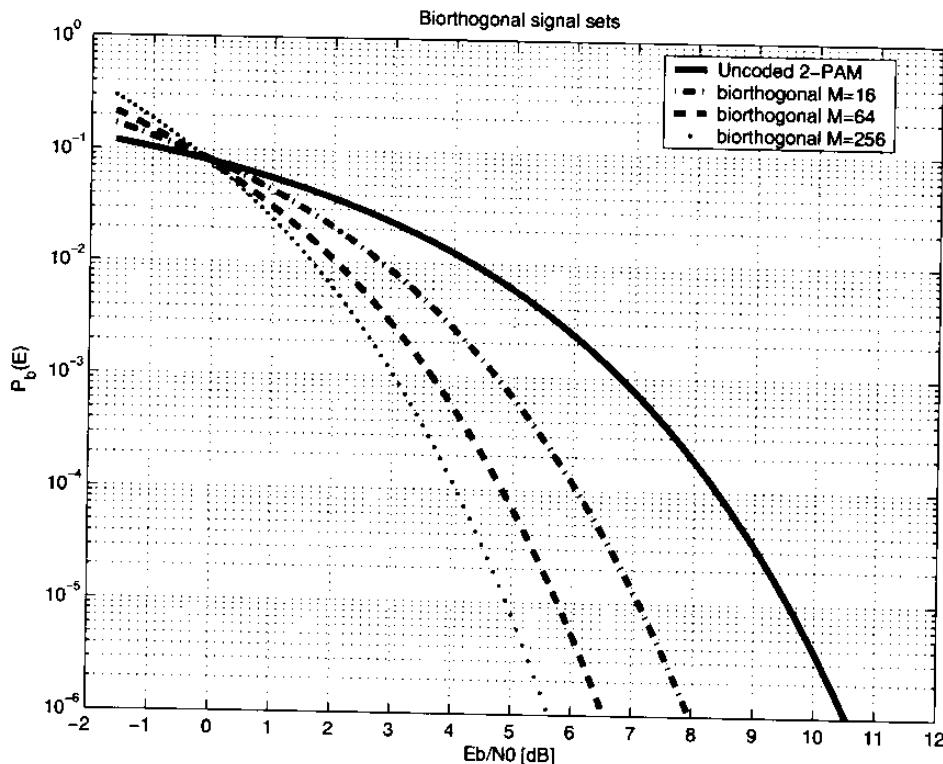
State transition diagram:



## Codes with dynamical structure (cont.)

### First example: Maximum-length shift-register codes (cont.)

- Decoding: fast Hadamard transform (“Green machine”)
  - “Soft decisions,” not “hard”
  - Maximum-likelihood (minimum-distance) decoding, not bounded-distance
  - Exponential complexity:  $O(k2^k)$
- Performance:
  - Probability of error decreases exponentially with  $k$
  - Approaches Shannon limit as  $k \rightarrow \infty$
  - Bandwidth  $\rightarrow \infty$

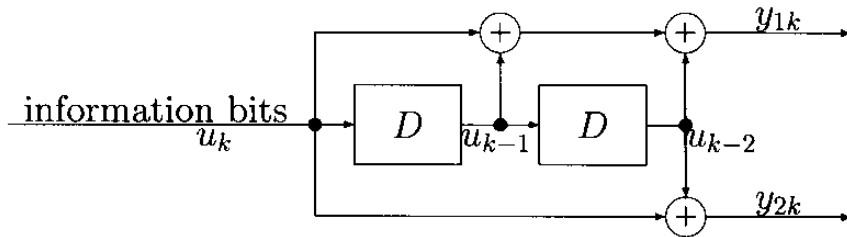


$P_b(E)$  vs.  $E_b/N_0$  for biorthogonal codes with  $2^k = 16, 64$  and  $256$ .  
 ( $E_b/N_0$  plotted on a log scale in dB. Shannon limit:  $-1.59$  dB)

# More codes with dynamical structure

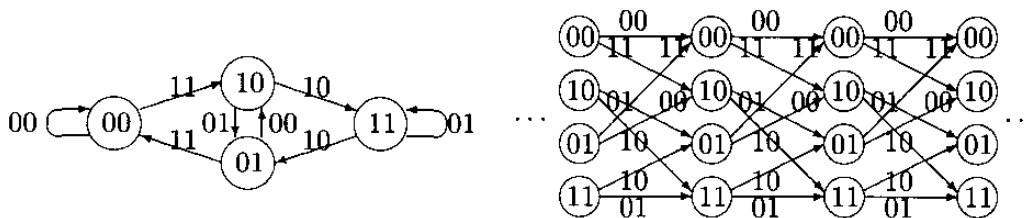
## Second example: Binary linear rate-1/n convolutional codes

4-state rate-1/2 binary linear convolutional encoder:



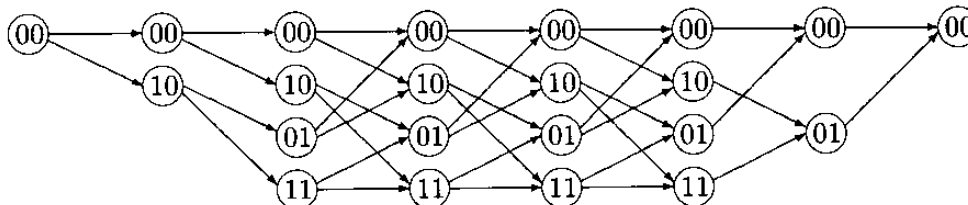
- Operation: At each time: 1 bit in, 2 bits out
- Encoding complexity: a  $\nu$ -stage binary shift register
- Algebraic structure: a binary linear code, but not a block code
- Dynamical structure:
  - A linear (over  $\mathbb{F}_2$ ) time-invariant system;
  - A finite( $2^\nu$ )-state machine

State transition diagram and trellis diagram:



## Terminated convolutional codes

Trellis diagram of a terminated 4-state rate-1/2 convolutional encoder:

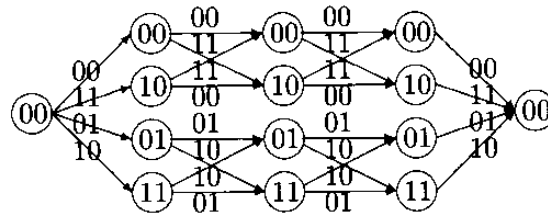


- A binary linear block code
- Decoding: the Viterbi algorithm
  - “Soft decisions,” not “hard”
  - Maximum-likelihood (minimum-distance) decoding, not bounded-distance
  - Exponential complexity:  $O(2^\nu)$  per unit time
  - Extends directly to unterminated convolutional codes
- Performance:
  - Probability of error decreases exponentially with  $\nu$
  - Approaches Shannon limit as  $\nu \rightarrow \infty$  in principle
  - 6-7 dB coding gain with 64-256 states in practice
  - Properly terminated convolutional codes are optimal block codes

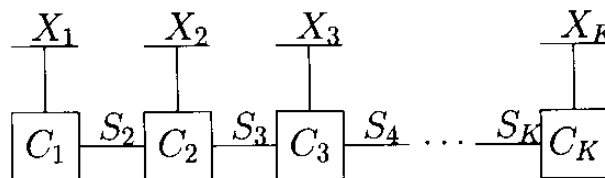
# Trellis realizations of block codes

## Trellis realizations of codes

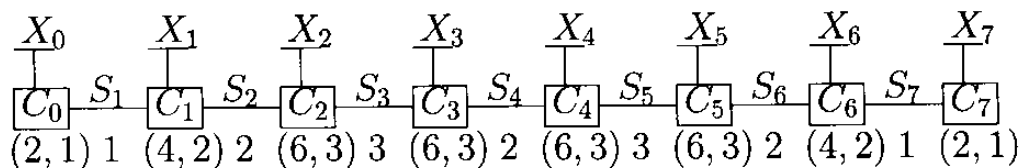
- Natural for convolutional codes (generated by state machines)
- 1988-1998: Trellis (state-space) realizations of block codes
- Example: Four-section 4-state trellis for (8, 4, 4) Reed-Muller code



- General trellis realizations
  - Finite time axis
  - Define a state space  $S_k$  at each time  $k$  satisfying Markov property
  - Define a constraint code  $C_k$  at each time  $k$ :  
 {possible combos of (state  $S_k$ , next state  $S_{k+1}$ , symbol  $X_k$ )}
- Generic graph

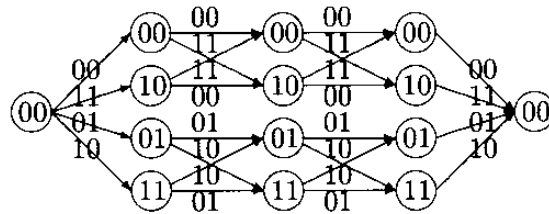


- Given a coordinate ordering,  $\exists$  unique minimal trellis realization
- Example: Graph of minimal 8-state trellis for (8, 4, 4) code



- Sectionalization (clustering) can reduce apparent state complexity, but not branch (constraint) complexity

## Trellis realizations of block codes (cont.)

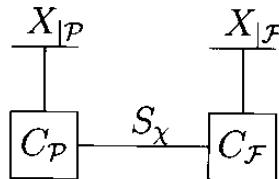


- Decoding: the Viterbi algorithm
  - Maximum-likelihood (minimum-distance) decoding
  - Complexity: proportional to number of states/branches
  - Reed-Muller codes have efficient trellis realizations
  - Average dimension bound:  
for any  $(n, k, d)$  binary linear block code,  
maximum branch space dimension  $\geq (k/n)d$ ;  
*i.e.*, complexity is exponential in minimum distance  $d$
- Performance:
  - ML decoding performance
  - Approaches Shannon limit in principle
  - 5–6 dB coding gain with 64–256 states in practice
  - For a given complexity, convolutional codes about 1 dB better

# Code realizations on general graphs

## The cut-set bound

- High-level view of a graphical realization with a cut set  $\chi$ :

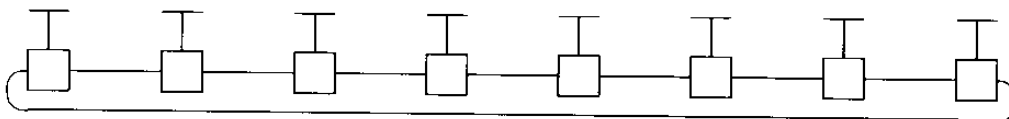


- Expresses a conditional independence (Markov) property.
- Cut-set bound: the size  $|S_\chi|$  of the “state space”  $S_\chi$  is not less than the size of the corresponding state space in a trellis realization with the same partition between “past”  $X_{|P}$  and “future”  $X_{|F}$
- Corollary: realizations of codes on cycle-free graphs cannot be very much simpler than trellis realizations (because every edge is a cut set)
  - Too bad, because a simple message-passing decoding algorithm (belief propagation, generalized Viterbi algorithm) can perform maximum-likelihood decoding on any cycle-free graph
- Need to consider realizations on graphs with cycles

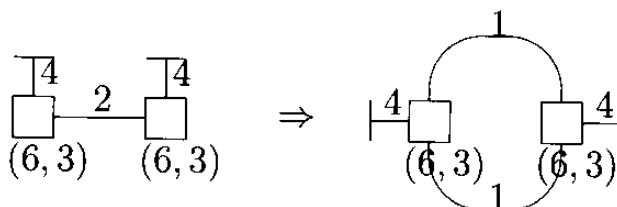
# Realizations on graphs with cycles

## Examples of graphical realizations with cycles

- “Tail-biting” realizations are the simplest realizations with cycles:



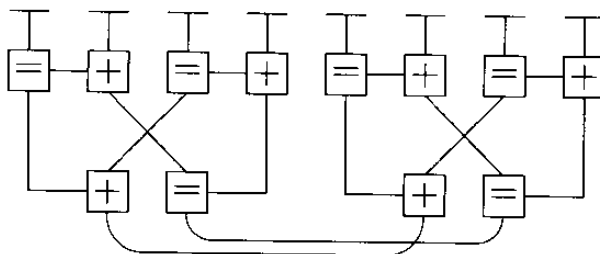
- A graph with a single cycle
- Cut sets involve two edges  $\Rightarrow$  Cut-set bound  $\rightarrow$  Square-root bound
- Example: tail-biting realization of  $(8, 4, 4)$  code:



- Best example: 16-state TB realization of  $(24, 12, 8)$  Golay code (*vs.* optimal 256-state trellis realization)

- Hadamard transform realizations of Reed-Muller codes

- Example: Reduced HT realization of  $(8, 4, 4)$  code



- All variables binary
- Six  $(3, 2, 2)$  constraints and six  $(3, 1, 3)$  constraints
- Short cycles (girth 6)  $\Rightarrow$  suboptimal decoding

# Parity-check realizations and Tanner graphs

Gallager (1961), Tanner (1981)

- Parity-check (kernel) realizations of binary linear block codes

$$\mathcal{C} = \{\mathbf{x} \in \mathbb{F}^n \mid \mathbf{x}H^T = \mathbf{0}\}$$

- Example:  $(8, 4, 4)$  extended Hamming code over  $\mathbb{F}_2$

$$H = \begin{bmatrix} 11110000 \\ 01011010 \\ 00111100 \\ 00001111 \end{bmatrix}.$$

- In detail:

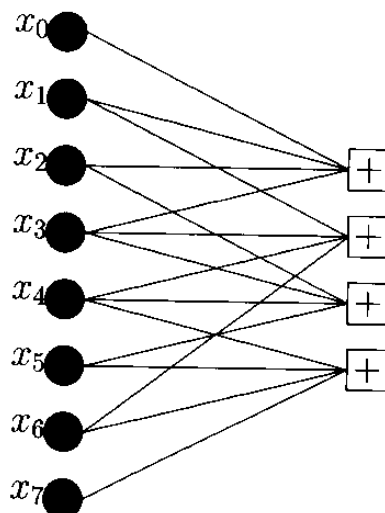
$$x_0 + x_1 + x_2 + x_3 = 0;$$

$$x_1 + x_3 + x_4 + x_6 = 0;$$

$$x_2 + x_3 + x_4 + x_5 = 0;$$

$$x_4 + x_5 + x_6 + x_7 = 0.$$

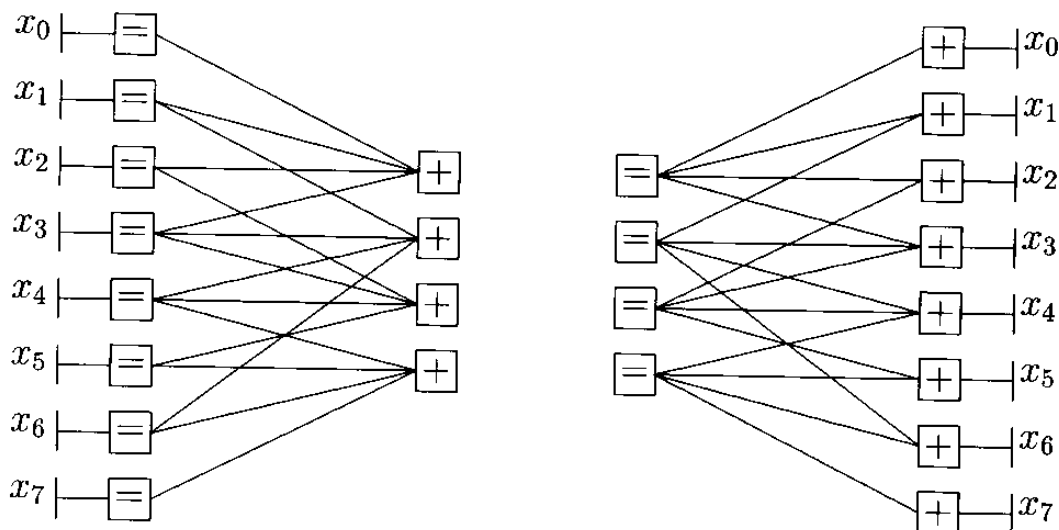
- Tanner graph:



# General graphical models of codes

## Progressive generalizations of Tanner graphs

- Constraints could be arbitrary codes (Tanner, 1981)
- Variables could include hidden (state) variables (Wiberg *et al.*, 1996)
- Variable alphabets could be arbitrary
- Normalize graphs (Forney, 2001)
  - Nodes = constraints, edges = variables
- Examples: parity-check and generator realizations of  $(8, 4, 4)$  code

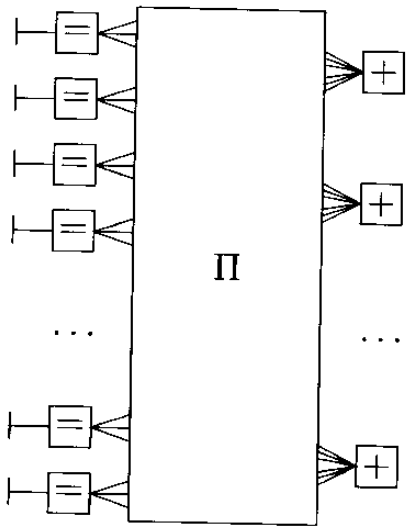


- “Inputs” are hidden in the generator (image) realization
- An instance of a beautiful general duality theorem:

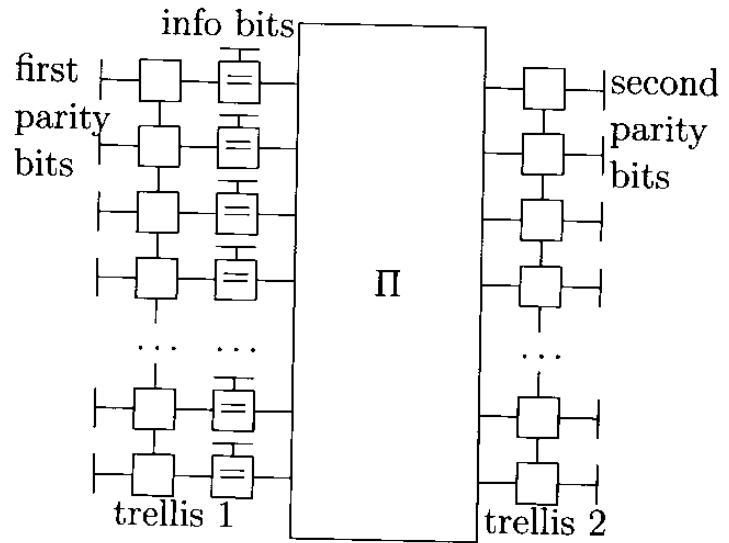
Dual normal graphs generate dual codes.

- Graphical realizations generalize state-space realizations
  - Time axis generalizes to a general graph
  - Graph elements:
    - visible variables, hidden (state) variables, and constraints
  - Cut sets  $\leftrightarrow$  conditional independence (Markov) properties

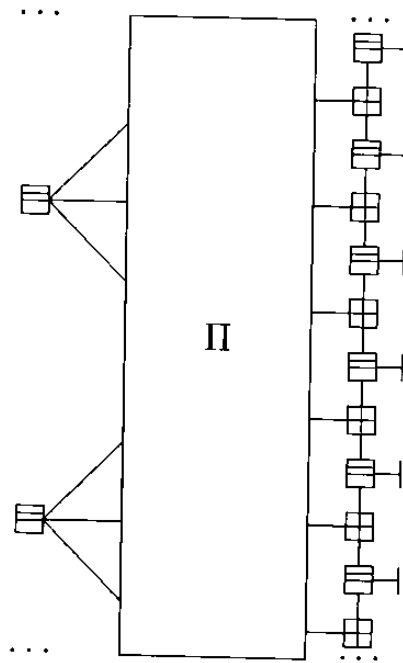
# Capacity-approaching codes



LDPC code  
Gallager, 1961



Turbo code  
Berrou *et al.*, 1993



Repeat-accumulate code  
Divsalar, McEliece, Jin, 1998

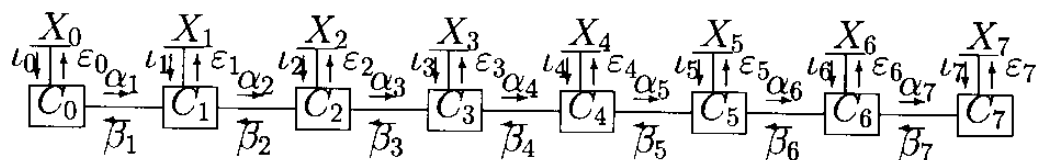
# Exact decoding on cycle-free graphs

## The sum-product algorithm

- Problem: given observed likelihoods  $\{p(y_i | x_i), \forall x_i\}$ , compute the *a posteriori* probability (APP) of each variable:

$$p(X_i = x_i | \mathbf{y}) \propto \sum_{\mathbf{x} \in \mathcal{C}: \mathbf{x}_i = x_i} \left( \prod_{i' \in I} p(y_{i'} | x_{i'}) \right)$$

- Solution (on cycle-free graphs): the sum-product algorithm, *aka*
  - APP decoding (Gallager, 1961)
  - BCJR algorithm (Bahl, Cocke, Jelinek, Raviv, 1974)
  - Algorithm B (Tanner, 1981)
  - Belief propagation (Pearl, 1988)
- A forward-backward message-passing algorithm in which each message summarizes all information in its “past.”
- Local computations, complexity proportional to constraint code size
- Finite and exact on cycle-free graphs.
- Flow of messages and computations when the sum-product (BCJR) algorithm is applied to a trellis:



## Exact decoding on cycle-free graphs (cont.)

### The max-product algorithm

- Problem: given observed likelihoods  $\{p(y_i | x_i), \forall x_i\}$ , find the maximum-likelihood codeword  $\hat{\mathbf{x}}$ :

$$\hat{\mathbf{x}} = \arg \max_{\mathbf{x} \in \mathcal{C}} \prod_{i \in I} p(y_i | x_i)$$

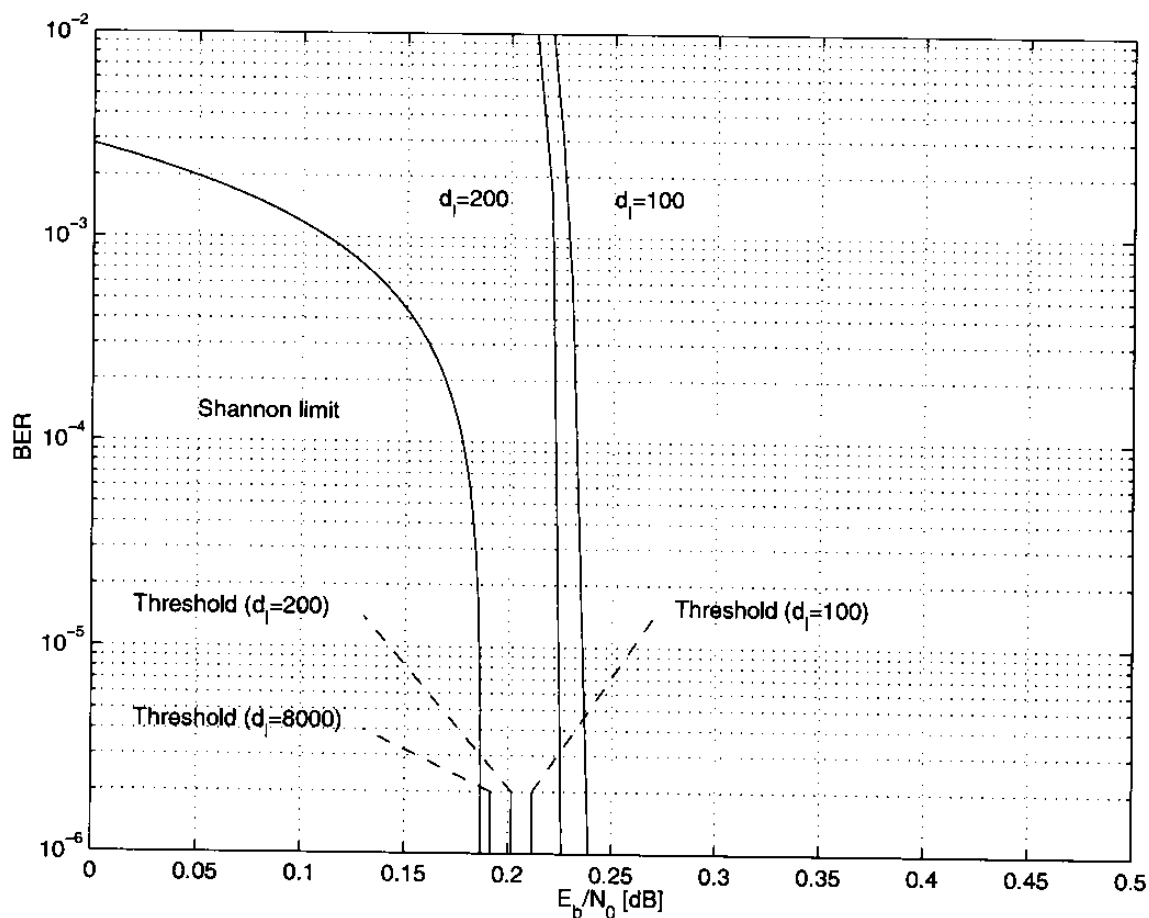
- Solution (on cycle-free graphs): the max-product algorithm, *aka* the Viterbi algorithm (1968), dynamic programming
- Can be obtained by replacing “sum” by “max” in the sum-product algorithm
- Can be simplified to a one-way (forward-only) message-passing algorithm
- Finite and exact on cycle-free graphs.

## Decoding on graphs with cycles

**Try the sum-product algorithm and hope for the best**

- All operations in the sum-product algorithm are local
- Need initialization, schedule, stopping rule
- Result: an iterative, approximate message-passing algorithm
  - Convergence not guaranteed
  - Convergence rate = ?
  - Fixed points of algorithm: fixed points of a non-convex, approximate target function (Bethe free energy)
- Nonetheless, in coding applications this algorithm works very well
  - Turbo codes, low-density parity-check (LDPC) codes now approach the Shannon limit within tenths of a dB

# Progress report



Bit error rate *vs.*  $E_b/N_0$  in dB for optimized irregular rate-1/2 binary LDPC codes with maximum left degree  $d_l$ . (From Chung *et al.*, 2001.)

- Threshold: theoretical limit as block length  $\rightarrow \infty$ .
- Solid curves: simulation results for block length =  $10^7$ .
- Shannon limit: binary codes,  $R = 1/2$ .

# Summary

## Codes/systems on graphs

- Linear/group codes/systems on **cycle-free graphs**
  - Generalize conventional state-space realizations
  - General theory of minimal realizations
  - Efficient exact decoding/inference algorithms
    - *e.g.*, generalized Kalman filter
- Linear/group codes/systems on **graphs with cycles**
  - For example, two-dimensional linear systems
  - Efficient realizations, but no general minimality theory
  - Efficient approximate iterative decoding/inference algorithms
    - *e.g.*, generalized belief propagation
  - Duality theorems
  - Connections to statistical physics