ECE 634: Digital Video Systems
Lossless compression: 2/7/17

Professor Amy Reibman
MSEE 356
reibman@purdue.edu

http://engineering.purdue.edu/~reibman/ece634/index.html

Compression Outline

- Overview (an eye to video coding)
- Lossless encoding
- Quantization and vector quantization
- Transform coding and wavelet coding
- Predictive coding
- Video coding (theory vs. practice)
- Standardization
- Video encoders

Think about compressing video

- Communicate what's in the video
  - What's most important to send?

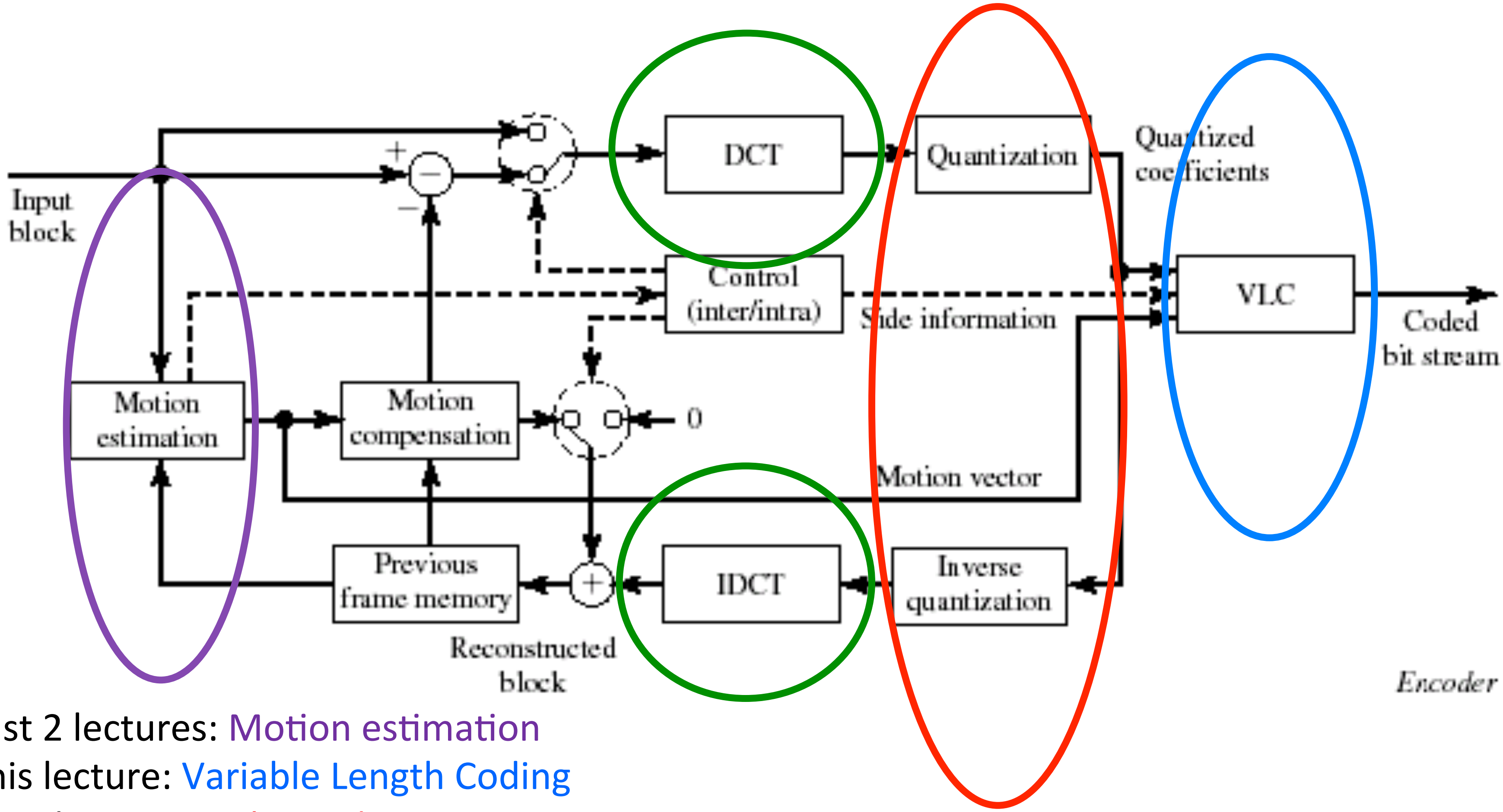
- Don't send things twice
  - What correlations are present in video?

Video Coding Techniques Based on Different Source Models

TABLE 8.1 COMPARISON OF SOURCE MODELS, PARAMETER SETS, AND CODING TECHNIQUES.

Source model	Encoded parameters	Coding technique
Statistically independent pels	Color of each pel	PCM
Statistically dependent pels	Color of each block	Transform coding, predictive coding, and vector quantization
Translationally moving blocks	Color and motion vector of each block	Block-based hybrid coding
		Waveform-based techniques
Moving unknown objects	Shape, motion, and color of each object	Analysis-synthesis coding
Moving known object	Shape, motion, and color of each known object	Knowledge-based coding
Moving known object with known behavior	Shape, color, and behavior of each object	Semantic coding
		Content-dependent-techniques

Encoder Block Diagram of a Typical Block-Based Video Coder
(Assuming No Intra Prediction)



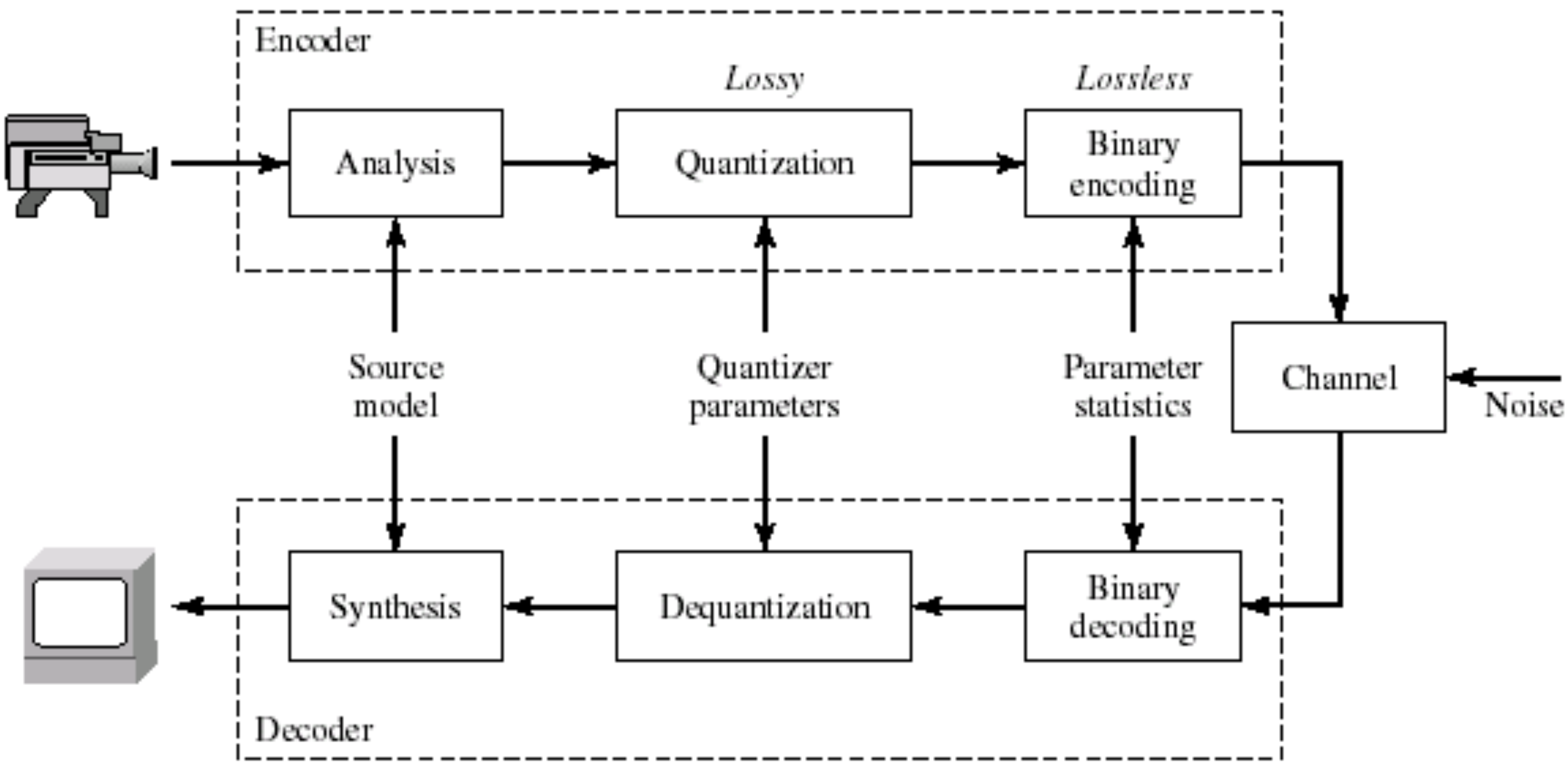
Last 2 lectures: Motion estimation

This lecture: Variable Length Coding

Next lecture: Scalar and Vector Quantization

And then: DCT, wavelet and predictive coding

Components in a Coding System



Lossless compression efficiency

- Compression efficiency depends on
  - Probability of the source
  - Algorithm (Huffman, Arithmetic coding)
  - Probability model (its accuracy)

- Lossless coding used in Video compression
  - Quantized transform coefficients
  - Motion vectors
  - Other side information

Probability and information theory review

- A given signal is a realization of a random process
- Efficiency of a source-coding technique: how fully are the source's statistics exploited?

- Characterize a source signal (aka random process):
  - Probability: marginal, joint, conditional
  - Entropy: joint, conditional
  - Mutual Information

Probability models and statistical characterization of random sources

- Source: a random sequence (discrete time random process),
  - Example: a video that follows a certain statistics
    - F_n represents the *possible value* of the n-th pixel of a video, $\mathbf{n}=(k,m,n)$
    - f_n represents the actual value taken in a realization

- Continuous source: F_n takes continuous values (analog image)
- Discrete source: F_n takes discrete values (digital image)

Statistical Characterization of Random Sources

- Stationary source:
  - Statistical distribution is invariant to time (or space) shift
  - Statistics of F**n** do not depend on the value of **n**
- Probability distribution
  - probability mass function (pmf) or probability density function (pdf): $p_{\mathcal{F}_n}(f)$ $p(f)$
  - Joint pmf or pdf:
    $p_{\mathcal{F}_{n+1},\mathcal{F}_{n+2},\ldots,\mathcal{F}_{n+N}}(f_1, f_2, \ldots, f_N)$ $p(f_1, f_2, \ldots f_N)$
  - Conditional pmf or pdf:
    $p_{\mathcal{F}_n|\mathcal{F}_{n-1},\mathcal{F}_{n-2},\ldots,\mathcal{F}_{n-M}}(f_{M+1} \mid f_M, f_{M-1}, \ldots, f_1)$ $p(f_{M+1} \mid f_M, f_{M-1}, \ldots, f_1)$

Recall: Probability relationships

- Bayes rule:

$$p(A \mid B)\,p(B) = p(B \mid A)\,p(A) = p(A,B)$$

- Theorem of total probability

$$p(A) = \sum_b p(A \mid B=b)\,p(B=b)$$

- Independent, identically distributed (iid; memoryless)

$$p(f_1, f_2, \ldots, f_M) = p(f_1)\,p(f_2)\ldots p(f_M)$$

$$p(f_{M+1} \mid f_M, f_{M-1}, \ldots, f_1) = p(f_{M+1})$$

Probability models, and three lossless coding options

- A single pixel (or “symbol”, ex: motion vector)
  - Single-symbol entropy
- Two adjacent pixels (or two dependent symbols)
  - Joint entropy, or conditional entropy
- A block of N pixels
- A block of N transform coefficients

Entropy of a RV

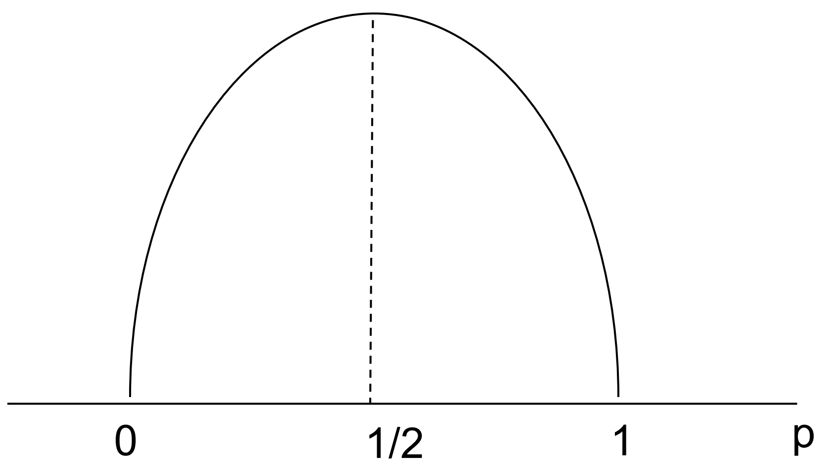
- Consider RV $F=\{f_1,f_2,\ldots,f_K\}$, with probability $p_k=Prob.\{F= f_K\}$
- Self-Information of one realization f_k : $H_k= -\log(p_k)$
  - $p_k=1$: always happen, no information
  - P_k near 0: seldom happen, its realization carries a lot of information
- Entropy = average information $H(\mathcal{F}) = -\sum_{f\in\mathcal{A}} p_{\mathcal{F}}(f)\log_2 p_{\mathcal{F}}(f).$
  - Entropy is a measure of uncertainty or information content, unit=bits – *if we use base 2 for the logarithm*
  - Very uncertain → high information content

Entropy

- Provides bounds on compression efficiency for lossless coding
- Larger when there's more uncertainty in the outcome of the random variable
- AVERAGE Information you obtain when you learn the value of the random variable. You obtain more information when you learn the random variable had taken on an unlikely value than had it taken on a likely value

Example: Two Possible Symbols

- Example: Two possible outcomes
  - Flip a coin, F={“head”,”tail”}: $p_1=p_2=1/2$: H=1 (highest uncertainty)
  - If the coin has a defect, so that $p_1=1$, $p_2=0$: H=0 (no uncertainty)
  - More generally: $p_1=p$, $p_2=1-p$,
    - H = - (p log p+ (1-p) log (1-p))



Example: English Letters

- 26 letters, each has a certain probability of occurrence
  - Some letters occur more often: “a”, “s”, “t”, …
  - Some letters occur less often: “q”, “z”, …
- Entropy ~= information you obtained after reading an article.
- We actually don’t get information at the alphabet level, but at the word level!
  - Some combination of letters occur more often: “it”, “qu”,…

A single pixel (or symbol)

- Entropy

$$H(\mathcal{F}) = -\sum_{f \in \mathcal{A}} p_{\mathcal{F}}(f) \log_2 p_{\mathcal{F}}(f).$$

- Scalar coding:
  - Assign one codeword to one symbol at a time
  - Difficulty: could differ from the entropy by up to 1 bit/symbol

$$H_1(\mathcal{F}) \leq \bar{R}_1(\mathcal{F}) \leq H_1(\mathcal{F}) + 1.$$

Two adjacent pixels (or two dependent symbols)

- Two approaches
  - Joint probability density function or probability mass function
  - Conditional probability

- Probability models accounting for relationship among pixels (next page)

Probability Models incorporating inter-symbol correlation

- 1st order Markov process
  - A sample only depends on its immediate predecessor

- M-th order Markov process
  - A sample depends only on its previous M samples
- Gaussian process
  - Any N samples form a N-dimensional Gaussian distribution
- Gauss-Markov process (in 1D) or Gauss-Markov Field (GMF) (in 2D)
  - 1st order GM: Covariance between two samples:

$$C(Fn, Fm) = \sigma^2 \rho^{|n-m|}$$

Joint entropy (high-level view)

- Entropy of more than one random variable, together

$$H(\mathcal{F}, \mathcal{G}) = -\sum_{f \in A_f} \sum_{g \in A_g} p_{\mathcal{F},\mathcal{G}}(f, g) \log_2 p_{\mathcal{F},\mathcal{G}}(f, g).$$

- Joint entropy is never bigger than the sum of the two individual entropy. Might do better if you code the symbols jointly

$$H(\mathcal{F}, \mathcal{G}) \le H(\mathcal{F}) + H(\mathcal{G})$$

Joint Entropy (more detail)

- Joint entropy of two RVs:
  - Uncertainty of two RVs together

$$H(\mathcal{F},\mathcal{G}) = -\sum_{f\in\mathcal{A}_f}\sum_{g\in\mathcal{A}_g} p_{\mathcal{F},\mathcal{G}}(f,g)\log_2 p_{\mathcal{F},\mathcal{G}}(f,g).$$

$$H(\mathcal{F},\mathcal{G}) \le H(\mathcal{F}) + H(\mathcal{G})$$

- N-th order entropy
  - Uncertainty of N RVs together

$$\begin{aligned} H_N(\mathcal{F}) &= H(\mathcal{F}_1,\mathcal{F}_2,\ldots,\mathcal{F}_N) \\ &= -\sum_{[f_1,f_2,\ldots,f_N]\in\mathcal{A}^N} p(f_1,f_2,\ldots,f_N)\log_2 p(f_1,f_2,\ldots,f_N). \end{aligned}$$

Vector coding of N symbols

- Vector coding:
  - Assign one codeword for each group of N symbols
  - Larger N → Lower Rate, but higher complexity
- Entropy rate (lossless coding bound)
  - As you take the limit of more and more random variables, and divide by the number N of rv's, the resulting *entropy rate* provides the lossless coding bound; it's the average uncertainty *per* random variable
  - Entropy rate is not bigger than the entropy of a single random variable; equal when samples independent

$$\overline{H}(\mathcal{F}) = \lim_{N \to \infty} \frac{1}{N} H_N(\mathcal{F})$$

Conditional Entropy

- Conditional entropy between *two* RVs:
  - Uncertainty of one RV *given* the other RV

$$H(\mathcal{F}\,|\,\mathcal{G}) = \sum_{g \in \mathcal{A}_g} p_{\mathcal{G}}(g) H(\mathcal{F}\,|\,g)$$
$$= -\sum_{g \in \mathcal{A}_g} p_{\mathcal{G}}(g) \sum_{f \in \mathcal{A}_f} p_{\mathcal{F}|\mathcal{G}}(f\,|\,g) \log_2 p_{\mathcal{F}|\mathcal{G}}(f\,|\,g).$$

$$H(\mathcal{F}) \geq H(\mathcal{F}\,|\,\mathcal{G})$$
$$H(\mathcal{F},\mathcal{G}) = H(\mathcal{G}) + H(\mathcal{F}\,|\,\mathcal{G})$$

Conditional Entropy

- M-th order conditional entropy

$$H_{C,M}(\mathcal{F}) = H(\mathcal{F}_{M+1} \mid \mathcal{F}_M, \mathcal{F}_{M-1}, \ldots, \mathcal{F}_1)$$

$$= \sum_{[f_1, f_2, \ldots, f_M] \in \mathcal{A}^M} p(f_1, f_2, \ldots, f_M) H(\mathcal{F}_{M+1} \mid f_M, f_{M-1}, \ldots, f_1)$$

$$H(\mathcal{F}_{M+1} \mid f_M, f_{M-1}, \ldots, f_1)$$

$$= -\sum_{f_{M+1} \in \mathcal{A}} p(f_{M+1} \mid f_M, f_{M-1}, \ldots, f_1) \log_2 p(f_{M+1} \mid f_M, f_{M-1}, \ldots, f_1).$$

Conditional coding (context adaptive coding)

- The codeword for the current symbol depends on the pattern formed by the previous M symbols (the context)
- Use a separate codebook for each possible context

Which is better?

- It depends

Entropy rate

- A lower bound on lossless compression
- To achieve it requires coding an infinite # of samples together

$$\bar{H}(\mathcal{F}) = \lim_{N\to\infty} \frac{1}{N} H_N(\mathcal{F}) = \lim_{N\to\infty} H_{C,N}(\mathcal{F}).$$

Lossless coding options (1 and 2)

- Scalar coding: $$H_1(\mathcal{F}) \leq \bar{R}_1(\mathcal{F}) \leq H_1(\mathcal{F}) + 1.$$
  - Assign one codeword to one symbol at a time
  - Difficulty: could differ from the entropy by up to 1 bit/symbol
- Vector coding:
  - Assign one codeword for each group of N symbols
  - Larger N → Lower Rate, but higher complexity

$$H_N(\mathcal{F}) \leq R^N(\mathcal{F}) \leq H_N(\mathcal{F}) + 1$$

$$H_N(\mathcal{F})/N \leq \bar{R}_N(\mathcal{F}) \leq H_N(\mathcal{F})/N + 1/N.$$

  - Bit-rate can be arbitrarily close to the source entropy rate IF we code many samples together

$$\lim_{N\to\infty} \bar{R}_N(\mathcal{F}) = \bar{H}(\mathcal{F}).$$

Lossless coding options (3)

- Conditional coding (aka, context-based coding)
  - The codeword for the current symbol depends on the pattern formed by the previous M symbols (the context)
  - Use a separate codebook for each possible context

$$H^{m}_{C,M}(\mathcal{F}) \le \bar{R}^{m}_{C,M}(\mathcal{F}) \le H^{m}_{C,M}(\mathcal{F}) + 1,$$

(entropy conditioned on context m)

$$H_{C,M}(\mathcal{F}) \le \bar{R}_{C,M}(\mathcal{F}) \le H_{C,M}(\mathcal{F}) + 1.$$

$$\bar{H}(\mathcal{F}) \le \lim_{M\to\infty} R_{C,M}(\mathcal{F}) \le \bar{H}(\mathcal{F}) + 1.$$

Which is better?

- It depends (on the source)

$$\bar{H}(\mathcal{F}) \leq H_{C,N-1}(\mathcal{F}) \leq \frac{1}{N} H_N(\mathcal{F}) \leq H_1(\mathcal{F}).$$

- Conditional and vector coding can both achieve a lower rate than scalar coding
- Conditional coding (for one symbol, conditioned on N-1 others) is at least as cheap as vector coding, but obviously conditional coding requires you to have sent all the N-1 others first

Example: 4-symbol source

- Four symbols: Alphabet {“a”,”b”,”c”,”d”}
- Probability mass function:

$$\mathbf{p}^T = [p(a), p(b), p(c), p(d)]$$
$$\mathbf{p}^T = [0.5000, 0.2143, 0.1703, 0.1154]$$

- Compute 1st order entropy:

$$H_1 = -\log(p(a))p(a) - \log(p(b))p(b) - \log(p(c))p(c) - \log(p(d))$$
$$= 1.7707$$

Example 8.1 from page 236 of Wang, Ostermann, Zhang

Example: 4-symbol Markov source

- Also: a 1st order conditional pmf: $q_{ij}=Prob(f_i|f_j)$

$$Q=\begin{bmatrix} p(a|a) & p(a|b) & p(a|c) & p(a|d) \\ p(b|a) & p(b|b) & p(b|c) & p(b|d) \\ p(c|a) & p(c|b) & p(c|c) & p(c|d) \\ p(d|a) & p(d|b) & p(d|c) & p(d|d) \end{bmatrix}$$

Row I, column j entry is q(i|j); probability that Fn=i and Fn-1=j

- 2nd order pmf:

$$p(f_{n-1}, f_n) = p(f_{n-1})q(f_n \mid f_{n-1}).$$

- Note that Qp=p; p is an eigenvector of Q; This is a stationary source

Example 8.2 and 8.3 from page 236-238 of Wang, Ostermann, Zhang

Example: 4-symbol Markov source

- Also: a 1st order conditional pmf: q_{ij}=*Prob*(f_i|f_j)

$$Q = \begin{bmatrix} 0.6250 & 0.3750 & 0.3750 & 0.3750 \\ 0.1875 & 0.3125 & 0.1875 & 0.1875 \\ 0.1250 & 0.1875 & 0.3125 & 0.1250 \\ 0.0625 & 0.1250 & 0.1250 & 0.3125 \end{bmatrix}$$

Row I, column j entry is q(i|j); probability that Fn=i and Fn-1=j

$$\mathbf{p}^T = [0.5000, 0.2143, 0.1703, 0.1154]$$

$$\text{Ex.} \quad p(\text{"}ab\text{"}) = p(\text{"}a\text{"})q(\text{"}b\text{"}/\text{"}a\text{"}) = 0.5 * 0.1875 = 0.0938$$

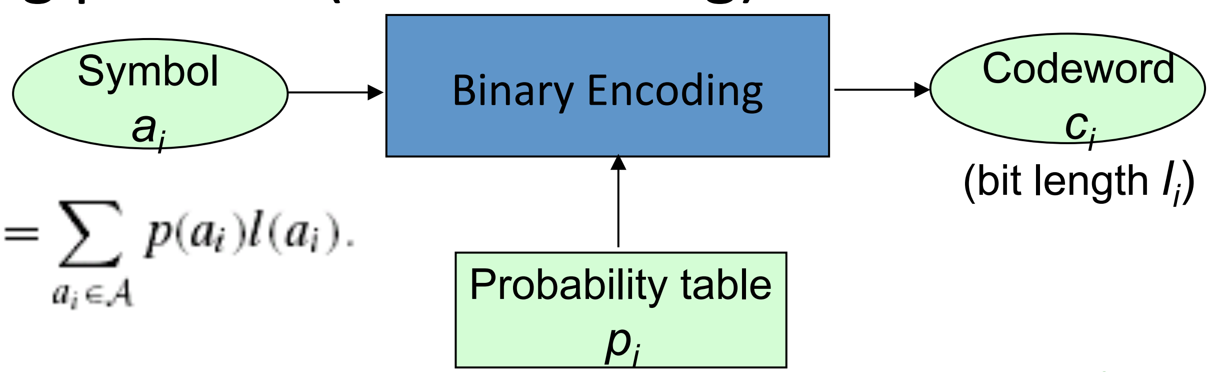
- $H_2/2$ = 1.7314 bits is the pair-wise joint entropy
- $H_{c,1}$ = 1.6922 bits is the average conditional entropy across the 4 contexts
- Conditional entropy for context “a” is 1.5016; for other contexts is 1.8829

Example 8.2 and 8.3 from page 236-238 of Wang, Ostermann, Zhang

Lossless encoding:
Symbols into binary

Lossless Coding (Binary Encoding)

- Binary encoding is a necessary step in any coding system
  - Applied to
    - original symbols (e.g. image pixels) in a discrete source,
    - converted symbols (e.g. quantized transformed coefficients) from a continuous or discrete source
- Binary encoding process (scalar coding)

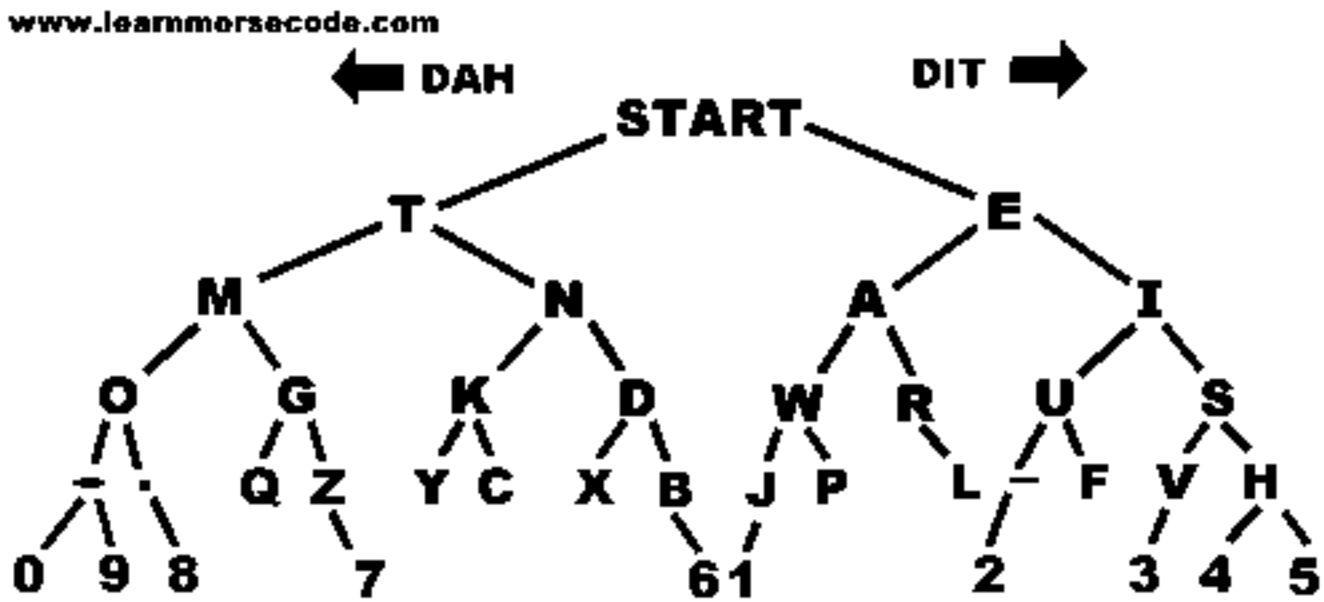


Bit rate (bit/symbol): $$R = \sum_{a_i \in \mathcal{A}} p(a_i) l(a_i).$$

Again

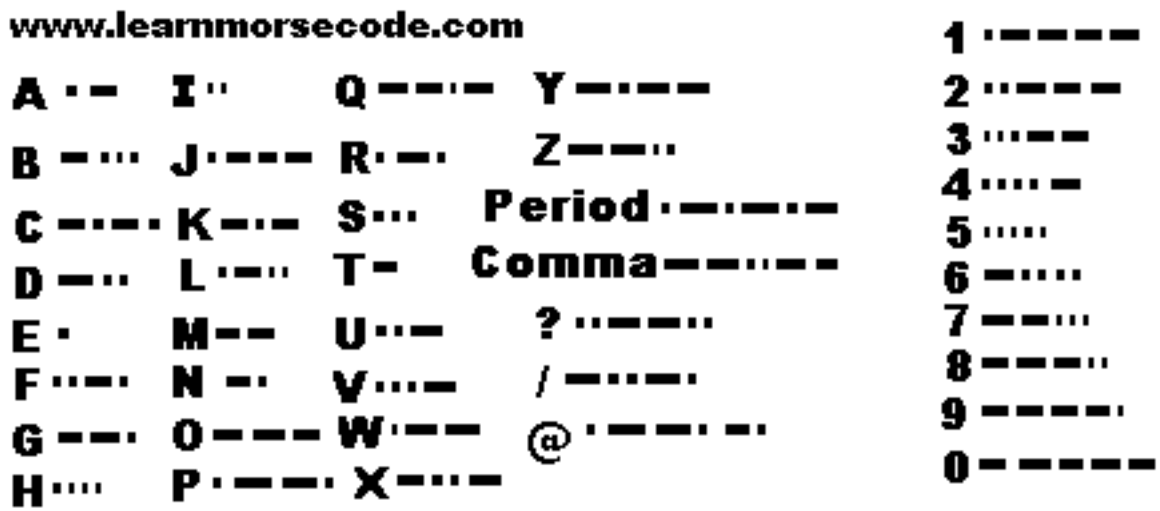
- Three options:
  - One symbol at a time
  - A clumping of N symbols
  - One symbol depending on what was sent earlier
- How-to
  - Huffman coding (Fixed # symbols, variable # bits)
  - Arithmetic coding (Variable # symbols, variable # bits)

Morse Code



tree from www.learnmorsecode.com

Morse Code



See also http://letterfrequency.org/ for some interesting orderings of English letters, including di-graphs and tri-graphs

chart from www.learnmorsecode.com

Designing binary codes

Represent each symbol as a sequence of bits, called a *codeword*

- A good code should be:
  - Uniquely decodable and a prefix code

Codebook 1
(a prefix code)

Symbol	Codeword
a_1	"0"
a_2	"10"
a_3	"110"
a_4	"111"

Codebook 2
(not a prefix code)

Symbol	Codeword
a_1	"0"
a_2	"01"
a_3	"100"
a_4	"011"

Bitstream: 0 0 1 1 0 1 0 1 1 0 1 0 0

Decoded string based on codebook 1: (can decode instantaneously)
0|0|1 1 0|1 0|1 1 0|1 0|0 $\rightarrow a_1\ a_1\ a_3\ a_2\ a_3\ a_2\ a_1$

Decoded string based on codebook 2: (must look ahead to decode)
0|0 1 1|0 1|0 1 1|0|1 0 0 $\rightarrow a_1\ a_4\ a_2\ a_4\ a_1\ a_3$

Huffman Coding

- Idea: more frequent symbols → shorter codewords
- Algorithm:

Step 1: Arrange the symbol probabilities $p(a_l)$, $l = 1, 2, \ldots, L$, in a decreasing order and consider them as leaf nodes of a tree.

Step 2: While there is more than one node:

(**a**) Find the two nodes with the smallest probability and arbitrarily assign 1 and 0 to these two nodes.

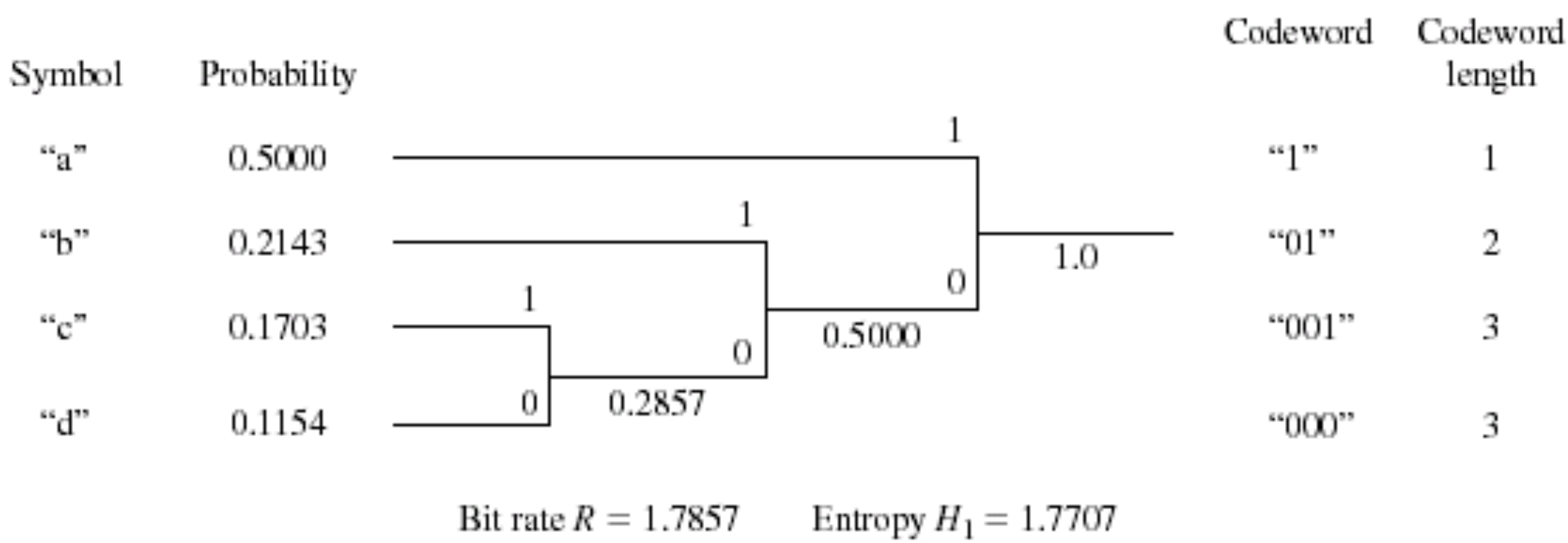
(**b**) Merge the two nodes to form a new node whose probability is the sum of the two merged nodes. Go back to Step 1.

Step 3: For each symbol, determine its codeword by tracing the assigned bits from the corresponding leaf node to the top of the tree. The bit at the leaf node is the last bit of the codeword.

- Huffman coding generates a prefix code
- Can be applied to one symbol at a time (scalar coding), or a group of symbols (vector coding), or one symbol conditioned on previous symbols (conditional coding)

Huffman Coding Example: Scalar Coding

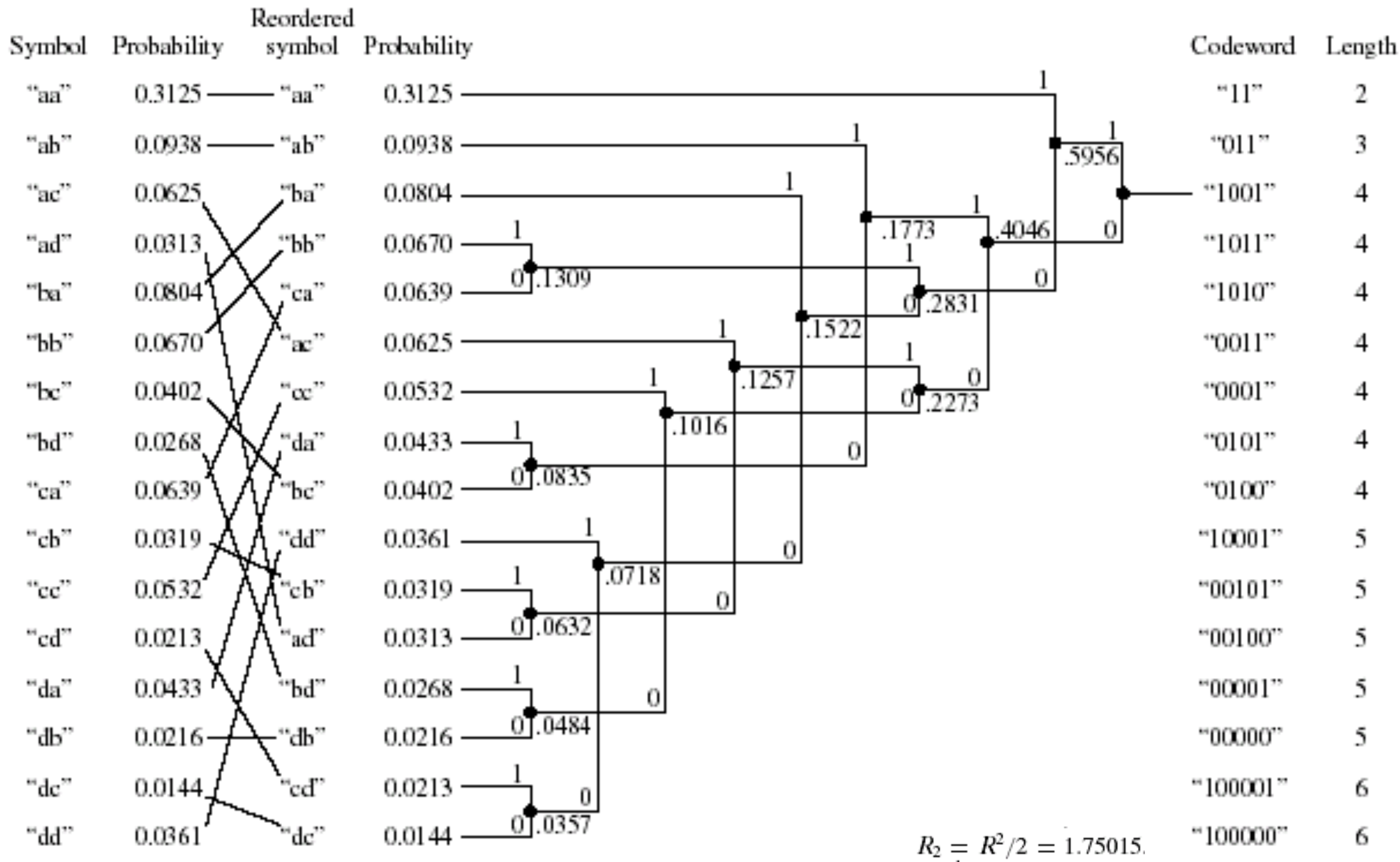
Example 8.1 from page 236 of Wang, Ostermann, Zhang



Huffman coding can NEVER use LESS than 1 bit per symbol

Huffman Coding Example: Vector Coding

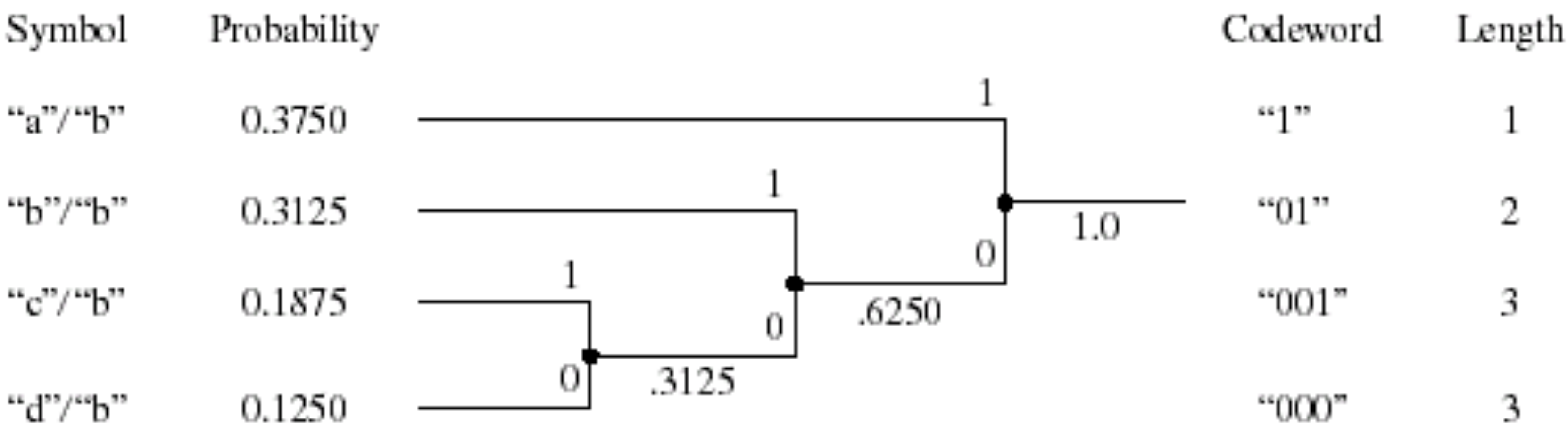
Example 8.2 from page 236-237 of Wang, Ostermann, Zhang



$R_2 = R^2/2 = 1.75015.$

$R^2 = 3.5003$ $H_2 = 3.4629$

Huffman Coding Example: Conditional Coding (context “b”)



$R_{C,\text{“b”}} = 1.9375 \qquad H_{C,\text{“b”}} = 1.8829$

$$R_{C,"a"} = 1.5625, R_{C,"b"} = R_{C,"c"} = R_{C,"d"} = 1.9375, R_{C,1} = 1.7500$$

$$H_{C,"a"} = 1.5016, H_{C,"b"} = H_{C,"c"} = H_{C,"d"} = 1.8829, H_{C,1} = 1.6922$$

Example 8.3 from page 238 of Wang, Ostermann, Zhang

Arithmetic Coding

- Another form of variable length coding: more frequent codewords should be shorter
- *Variable* number of symbols are mapped into a *variable* number of bits
- Any sequence of symbols is represented by an interval, [l,u), somewhere in the unit interval [0,1).
- More likely sequences are represented by longer intervals
- To represent any interval, we need only enough bits to get ANY value within that interval
- Longer intervals require fewer bits

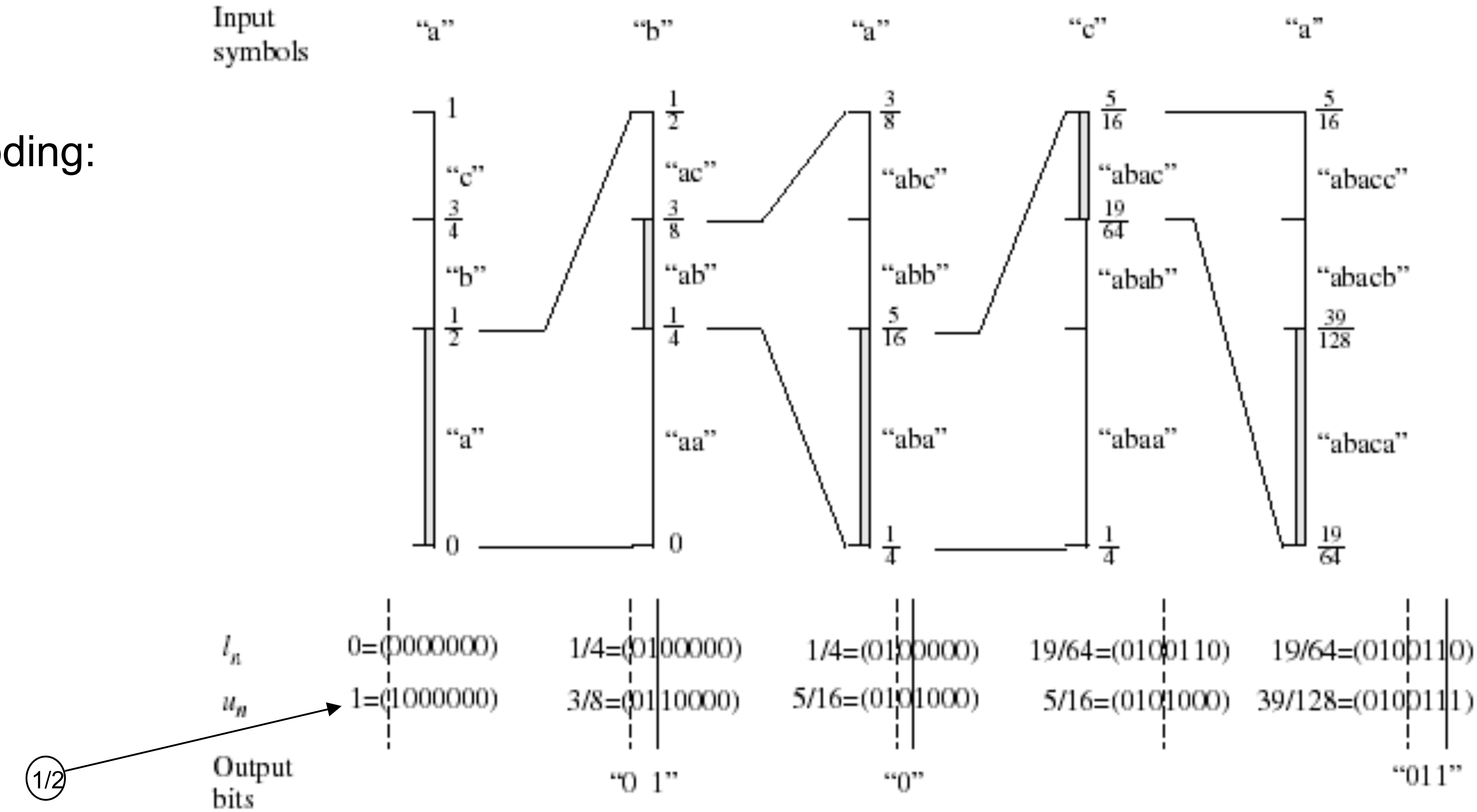
Arithmetic Coding (2)

- Calculate the interval sequentially
  - Keep track of the binary representations of both *l* and *u*
  - For each new symbol, the interval is further divided
  - When the MSBs of both *l* and *u* match, this bit is transmitted

$$d_n = d_{n-1} * p_l; \quad l_n = l_{n-1} + d_{n-1} * q_{l-1}; \quad u_n = l_n + d_n.$$

P(a)=1/2
P(b)=1/4
P(c)=1/4

Encoding:



(a)

Decoding:

Received bits	Interval	Decoded symbol
“0”	[0,1/2)	“a”
“01”	[1/4,1/2)	—
“010”	[1/4,3/8)	“b”
“0100”	[1/4,5/16)	“a”
“01001”	[9/32,5/16)	—
“010011”	[19/64,5/16)	“c”
…	…	…

(b)

Implementation of Arithmetic Coding

- Previous example is meant to illustrate the algorithm in a conceptual level
  - Require infinite precision arithmetic
  - Can be implemented with finite precision or integer precision only
- For more details on implementation, see
  - Witten, Neal and Cleary, "Arithmetic coding for data compression", J. ACM (1987), 30:520-40
  - Sayood, *Introduction to Data Compression*, Morgan Kaufmann, 1996

Huffman vs. Arithmetic Coding

- Huffman coding
  - Convert a fixed number of symbols into a variable length codeword
  - Efficiency: $$H_N(\mathcal{F})/N \leq \bar{R}_N(\mathcal{F}) \leq H_N(\mathcal{F})/N + 1/N.$$
  - To approach entropy rate, must code a large number of symbols together
  - Used in all image and video coding standards

- Arithmetic coding
  - Convert a variable number of symbols into a variable length codeword
  - Efficiency: $H_N(\mathcal{F})/N \leq R \leq H_N(\mathcal{F})/N + 2/N,$ *N* is sequence length
  - Can approach the entropy rate by processing one symbol at a time
  - Easy to adapt to changes in source statistics or to adapt to a context
  - Used as advanced options in earlier image and video coding standards
  - Becoming standard options in newer standards (JPEG2000,H.264)
  - Noticeable improvements in H.264 vs. Huffman coding; now heavily used

Summary

- Coding system:
  - original data → model parameters→quantization→ binary encoding
  - Waveform-based vs. content-dependent coding
- Characterization of information content by entropy
  - Entropy, Joint entropy, conditional entropy
  - Mutual information
- Lossless coding
  - Bit rate bounded by entropy rate of the source
  - Huffman coding:
    - Scalar, vector, conditional coding
    - can achieve the bound only if a large number of symbols are coded together
    - Huffman coding generates prefix code (instantaneously decodable)
  - Arithmetic coding
    - Can achieve the bound by processing one symbol at a time
    - More complicated than scalar or short vector Huffman coding