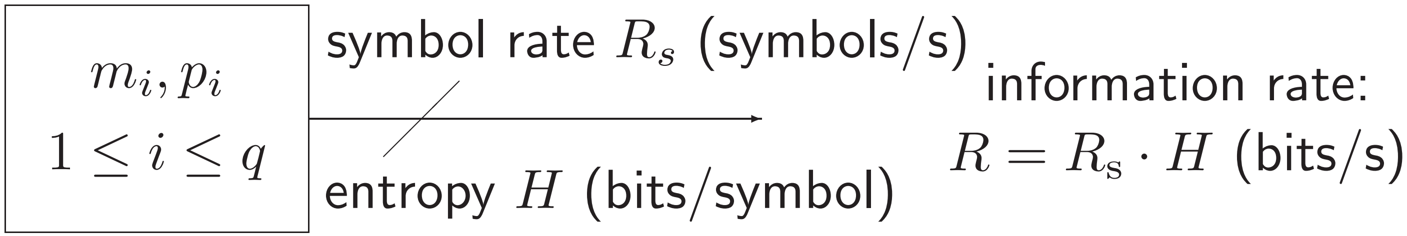
Revision of Lecture 2

Memoryless source with independent symbols



- Code each symbol by $\log_2 q$ bits (BCD), then data rate $R_{\mathrm{s}} \cdot \log_2 q > R$, unless source is equal probable $p_i = 1/q$, $1 \leq i \leq q$

- How to code symbols in an *efficient* way so that data rate as close as possible to R

- Two equivalent efficient encoding methods, achieving efficiency approximating 100%, by assigning codeword length (bits) according to symbol's information content

Remove memoryless assumption $\rightarrow$ this lecture

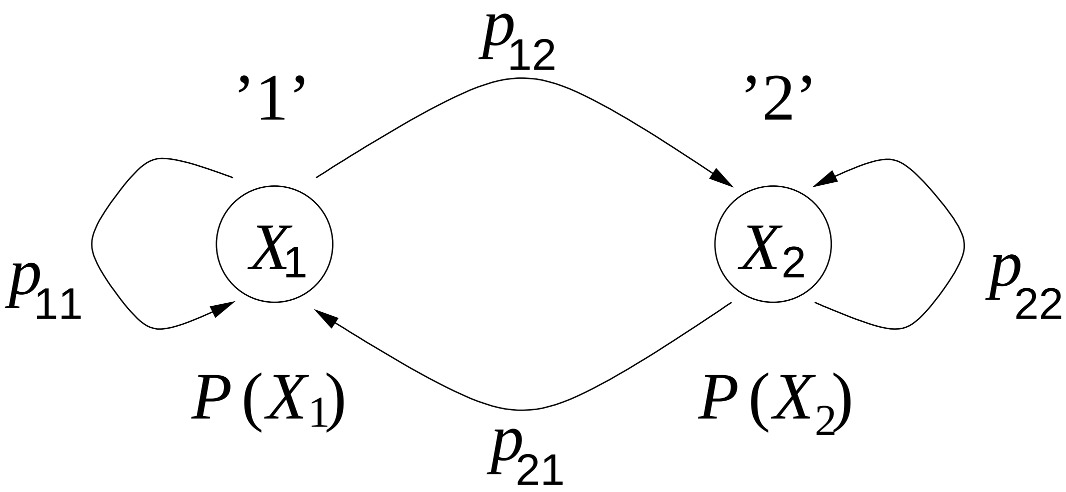
Information of Sources with Memory

- Most real world sources exhibit memory, resulting in *correlated source signals*; this property is retained during sampling and quantisation
  - This implies that the signal exhibits some form of *redundancy*, which should be exploited when the signal is coded
  - For example, samples of speech waveform are correlated; redundancy in samples is first removed, as it can be predicted; the resulting residuals, almost memoryless or uncorrelated, can then be coded with far fewer bits

- Here memory can be modelled by a **Markov process**
  - Consider source with memory that emits a sequence of symbols $\{S(k)\}$ with "time" index k
  - First order Markov process: the current symbol depends only on the previous symbol, $p(S(k)|S(k-1))$
  - N-th order Markov process: the current symbol depends on N previous symbols, $p(S(k)|S(k-1), S(k-2), \cdots, S(k-N))$

Two-State First Order Markov Process

- Source $S(k)$ can only generate two symbols, $X_1 = 1$ and $X_2 = 2$; their probability explicitly depends on the previous state (i.e. $p(S(k)|S(k-1))$)



- Probabilities of occurrence (prior probabilities) for states X_1 and X_2: $P_1 = P(X_1)$ and $P_2 = P(X_2)$ (i.e. $p(S(0) = 1) = P(X_1)$ and $p(S(0) = 2) = P(X_2)$)

- Transition probabilities: transition probabilities from state X_1 are given by the conditional probabilities $p_{12} = P(X_2|X_1)$ and $p_{11} = P(X_1|X_1) = 1 - P(X_2|X_1)$, etc. (i.e. $p(S(k) = j|S(k-1) = i) = p_{ij}$)

Entropy for 2-State 1st Order Markov Source

- Entropy H_i for state X_i, $i = 1, 2$:

$$H_i = -\sum_{j=1}^{2} p_{ij} \cdot \log_2 p_{ij} = -p_{i1} \cdot \log_2 p_{i1} - p_{i2} \cdot \log_2 p_{i2} \quad \text{(bits/symbol)}$$

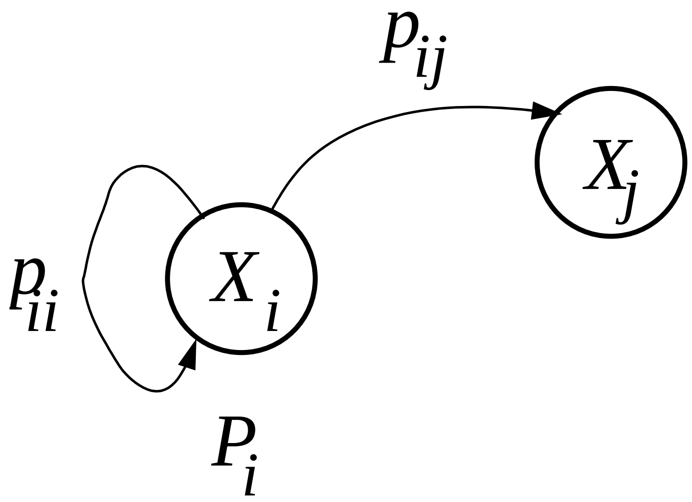
  This describes the average information carried by the symbols emitted in state X_i

- The overall entropy H includes the probabilities P_1, P_2 of the states X_1, X_2:

$$H = \sum_{i=1}^{2} P_i H_i = -\sum_{i=1}^{2} P_i \sum_{j=1}^{2} p_{ij} \cdot \log_2 p_{ij} \quad \text{(bits/symbol)}$$

- For a highly correlated source, it is likely to remain in a state rather than to change, and *entropy is decreasing with correlation*

Entropy for N-State 1st Order Markov Source



- A N-state (not N-th order) 1st-order Markov source can generate N symbols $X_i = i$, $1 \leq i \leq N$, and the symbol entropy H_i for state X_i:

$$H_i = -\sum_{j=1}^{N} p_{ij} \cdot \log_2 p_{ij} \quad \text{(bits/symbol)}$$

  where p_{ij} is transition probability from X_i to X_j

- The averaged, weighted symbol entropies give the source entropy

$$H = \sum_{i=1}^{N} P_i H_i = -\sum_{i=1}^{N} P_i \sum_{j=1}^{N} p_{ij} \cdot \log_2 p_{ij} \quad \text{(bits/symbol)}$$

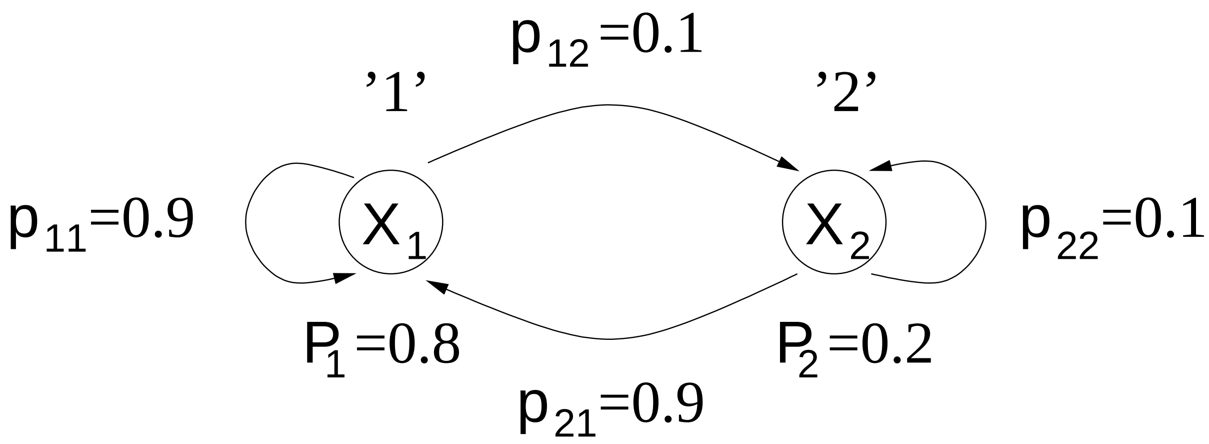
  where P_i is the probability of occurrence (prior probability) of the state X_i

- With a symbol rate R_s symbols/second, the average source information rate R is

$$R = R_s \cdot H \qquad \text{(bits/second)}$$

A 2-State 1st Order Markov Source – Problem

- Consider the following state diagram with associated probabilities:



- **Q1**: What is the source entropy?
- **Q2**: What is the average information content in message sequences of length 1, 2, and 3 symbols, respectively, constructed from a sequence of X_1 and X_2?

A 2-State 1st Order Markov Source – Solution

- **A1**: The source entropy is given by $H = -0.8 \cdot (0.9 \log_2 0.9 + 0.1 \log_2 0.1)$ $-0.2 \cdot (0.9 \log_2 0.9 + 0.1 \log_2 0.1) = 0.4690$ (bits/symbol)

- **A2** Average information for
  - 1-symbol sequence: $H^{(1)} = -0.8 \log_2 0.8 - 0.2 \log_2 0.2 = 0.7219$ (bits/symbol)
  - 2-symbols sequence: $P('11') = P_1 \cdot p_{11} = 0.72$; $P('12') = P_1 \cdot p_{12} = 0.08$; $P('21') = P_2 \cdot p_{21} = 0.18$; $P('22') = P_2 \cdot p_{22} = 0.02 \longrightarrow$ average 1.190924 bits for 2-symbol sequence, hence $H^{(2)} = 1.190924/2 = 0.5955$ (bits/symbol)
  - 3-symbols sequence: $P('111') = P('11') \cdot p_{11} = 0.648$; $P('112') = P('11') \cdot p_{12} = 0.072$; etc. $\longrightarrow H^{(3)} = 0.5533$ (bits/symbol)

- Consider sequence length of more symbols, which exhibits more memory dependency of the source, and therefore the average information or entropy *decreases*; e.g. $H^{(20)} = 0.4816$ bits/symbol

- In the limit: $H^{(k)} \longrightarrow H$ for message sequence length $k \longrightarrow \infty$

A2 Solution Explained

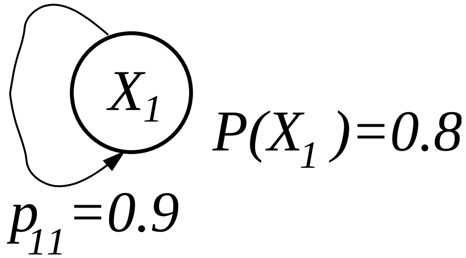
- One-symbol sequences: either "1" or "2" with $P(\text{“1”}) = 0.8$ and $P(\text{“2”}) = 0.2$

  Hence average information content (bits or bits/symbol as it is just one symbol)

$$-P(\text{“1”})\log_2 P(\text{“1”}) - P(\text{“2”})\log_2 P(\text{“2”})$$

$$= -0.8\log_2 0.8 - 0.2\log_2 0.2 = 0.7219 \text{ (bits/symbol)}$$

- Two-symbol sequences: "11", "12", "21" or "22"



  – Consider "11": $P(\text{“11”}) = 0.8 \times 0.9 = 0.72$

  – Average information contents (bits) for 2-symbol sequence:

$$-P(\text{“11”})\log_2 P(\text{“11”}) - P(\text{“12”})\log_2 P(\text{“12”}) - P(\text{“21”})\log_2 P(\text{“21”}) - P(\text{“22”})\log_2 P(\text{“22”})$$

$$= -0.72\log_2 0.72 - 0.08\log_2 0.08 - 0.18\log_2 0.18 - 0.02\log_2 0.02$$

$$= 0.3412304 + 0.2915084 + 0.4453076 + 0.1128771 = 1.1909235 \text{ (bits)}$$

A Few Comments on Markov Source Model

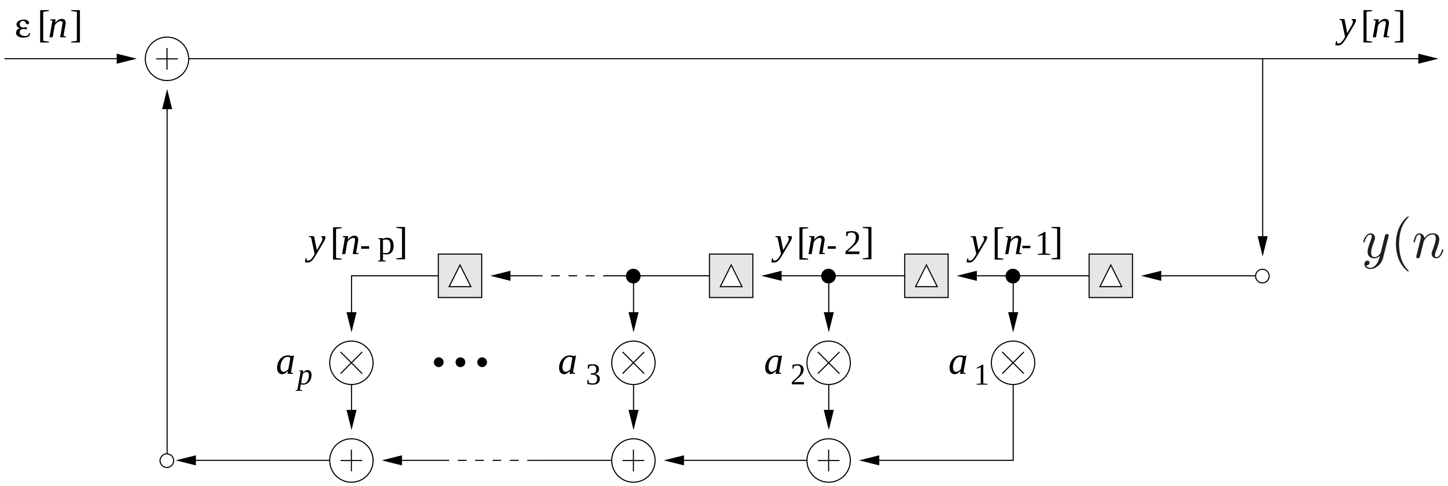
- Markov process is a most complete model to describe sources with memory; it is a *probabilistic* model

- Most widely used Markov process is 1st order Markov process, where
  - $P_i = P(X_i)$ is probability of occurrence of state X_i; image starting an experiment with time index t, at the beginning or $t = 0$, you can find that the process $S(0)$ starts from state X_i with probability P_i; hence P_i is *a priori* probability
  - Transition probability p_{ij} describes the probability of the process changing from state X_i to X_j, hence is *conditional* probability $p(S(t) = X_j|S(t-1) = X_i) = p_{ij}$

- To describe source with memory longer than 1, higher order Markov process is needed, but this is much more difficult to use
  - In practice, simplified parametric model is often used to describe source with higher-order memory, i.e.
  - Use **conditional mean** $E[s(t)|s(t-1), s(t-2), \cdots, s(t-N)]$ of **realisation** (observation) $s(t)$ to "replace" probabilities of stochastic process $S(t)$

Autoregressive Models

- An Nth order Markov process can be represented (simplified) as an Nth order autoregressive (AR) model:



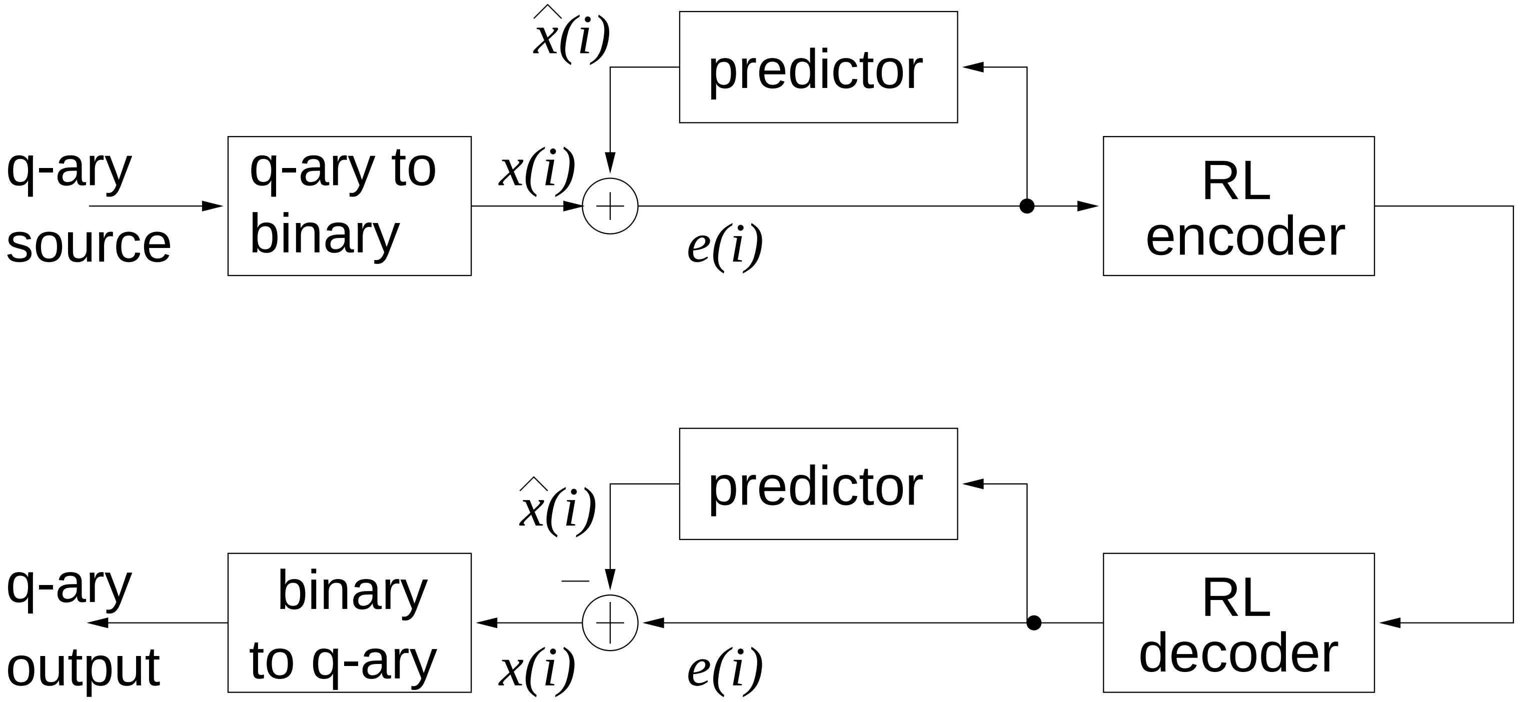
$$y(n) = \sum_{k=1}^{N} a_k y(n-k) + \epsilon(n)$$

- The input process $\epsilon(n)$ is uncorrelated, zero-mean; the output process $y(n)$ is the symbol sequence emitted by the source described by the Nth order Markov process (with appropriate parameters a_k); n is a time index for the symbol sequence

- This parametric model is widely used, for example, in speech source coding (transmit a_k and $\epsilon(n)$ instead of $y(n)$) – **Why doing this?**

Predictive Run-Length Coding (RLC)

- Since "memory" of the source makes the source signal partially predictable, this can be exploited in the following scheme:



- If prediction is successful, the signal $e(i)$ will mostly contain zeros, and this property is exploited in RLC

Run Length Coding Table

- Code words with fixed length of n bits are formed from a bit stream (encoder input pattern) of upto $l \leq N - 1 = 2^n - 2$ successive zeros followed by a one or zero:

length of 0-run l	encoder input pattern (length $= \min\{N, l+1\}$)	encoder output codeword (fixed n bits)
0	1	$00 \cdots 000$
1	01	$00 \cdots 001$
2	001	$00 \cdots 010$
3	0001	$00 \cdots 011$
$\vdots$	$\vdots$	$\vdots$
$N-2$	$0 \cdots 01$	$11 \cdots 101$
$N-1$	$00 \cdots 01$	$11 \cdots 110$
$N = 2^n - 1$	$00 \cdots 00$	$11 \cdots 111$

- Assumption is input bit stream contains mostly “0”s, i.e. $p = P($“0”$)$ is very high

- Thus encoder on average reduces the word length

RLC Efficiency

- Code word length after run length coding: n bits;
- Average code word length d before coding with $N = 2^n - 1$:

$$d = \sum_{l=0}^{N-1} (l+1) \cdot p^l \cdot (1-p) + N \cdot p^N = 1 + p + p^2 + \cdots p^{N-1} = \frac{1-p^N}{1-p}$$

where p is the probability of a bit is '0'

- Therefore compression ratio $C = d/n$
- A numerical example: $p = 0.95$, $n = 5$ $(N = 31)$

$$C = \frac{d}{n} = \frac{1-p^N}{n(1-p)} \approx \frac{15.92}{5} \approx 3.18$$

RLC Re-exam Again

- RLC is widely used in various applications, so let us exam RLC more closely

  Input patterns have variable lengths, $2^n - 1$ bits to just 1 bit, depending on length of "**0**" runs before "**1**"; while output codewords have fixed length of n bits

Input pattern

00...0000	**00...0001**	**00...001**	**...**	**01**	**1**
2^n−1+0 bits	2^n−2+1 bits	2^n−3+1 bits		1+1 bits	0+1 bits

$\Downarrow$ RLC

Output codeword

11...111	**11...110**	**11...101**	**...**	**00...001**	**00...000**
n bits	n bits	n bits		n bits	n bits

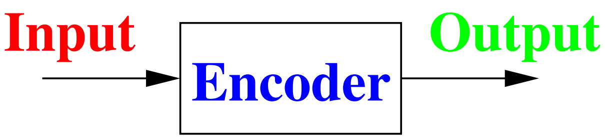
- Shannon-Fano and Huffman: inputs have fixed length while outputs variable lengths

  RLC appears very different from Shannon-Fano and Huffman or is it?

- RLC, Shannon-Fano and Huffman encodings are **lossless** or **entropy** encodings

Lossless Encodings Comparison

- Lossless or entropy encodings



Shannon–Fano
Huffman
RLC (for binary data most 0s)

- Same principle:

rare input pattern/message/symbol coded with large output codeword
large probability coded with small codeword

- Shannon-Fano and Huffman: input fixed length $\longrightarrow$ output variable length

  RLC: input variable length $\longrightarrow$ output fixed length

- It is the ratio

$$\textbf{ratio} = \frac{\textbf{output length}}{\textbf{input length}}$$

small **probability** $\longrightarrow$ large **ratio** large **probability** $\longrightarrow$ small **ratio**

Summary

- First-order Markov process model for sources with 1st-order memory

  Entropy and information rate of first-order Markov source

- Autoregressive models of N-th order for sources with N-th order memory

- The **need** to remove **redundancy** to make it **memoryless**

- Run-length encoding

  Comparison with Shannon-Fano and Huffman lossless or entropy encodings