Lecture 18

$$\underbrace{H(A^{(n)})}_{\text{Joint Entropy}} = -\sum_{i_1=1}^{m}\sum_{i_2=1}^{m}\cdots\sum_{i_n=1}^{m} \underbrace{p(a_{i_1}, a_{i_2}, \cdots, a_{i_n})}_{\text{Joint PMF}} \log_2 p(a_{i_1}, a_{i_2}, \cdots, a_{i_n})$$

$$= n\,H(A) \quad ?$$

Use m = 2, n = 2, as an example.

$A = \{a_1, a_2\}$, and $p(a_1) + p(a_2) = 1$.

$$H(A^{(n)}) = -\sum_{i_1=1}^{m}\sum_{i_2=1}^{m}\cdots\sum_{i_n=1}^{m} p(a_{i_1}, a_{i_2}, \cdots, a_{i_n}) \log_2 p(a_{i_1}, a_{i_2}, \cdots, a_{i_n})$$

$$= -\sum_{i_1=1}^{2}\sum_{i_2=1}^{2} p(a_{i_1}, a_{i_2}) \log_2 p(a_{i_1}, a_{i_2}), \text{ where}$$

$$p(a_{i_1}, a_{i_2}) = p(a_{i_1}) \cdot p(a_{i_2})$$

$$= -p(a_1, a_1)\log_2 p(a_1, a_1) - p(a_1, a_2)\log_2 p(a_1, a_2)$$

$$- p(a_2, a_1)\log_2 p(a_2, a_1) - p(a_2, a_2)\log_2 p(a_2, a_2)$$

$$p(a_1, a_1)\log_2 p(a_1, a_1) = p(a_1)\cdot p(a_1)\left[\log_2 p(a_1) + \log_2 p(a_1)\right]$$

$$= p(a_1)\,p(a_1)\log_2 p(a_1) + p(a_1)\,p(a_1)\log_2 p(a_1)$$

$$p(a_1, a_2)\log_2 p(a_1, a_2) = p(a_1)\cdot p(a_2)\left[\log_2 p(a_1) + \log_2 p(a_2)\right]$$

$$= p(a_1)\,p(a_2)\log_2 p(a_1) + p(a_1)\,p(a_2)\log_2 p(a_2)$$

$\vdots$

$\vdots$

$$p(a_1)\,p(a_1)\log_2 p(a_1) + p(a_1)\,p(a_2)\log_2 p(a_1) = p(a_1)\,[\underbrace{p(a_1)+p(a_2)}_{1}]\log_2 p(a_1) = p(a_1)\log_2 p(a_1)$$

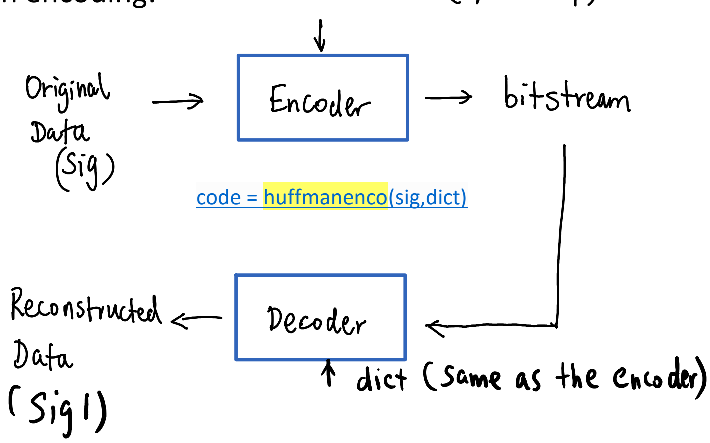
$$
\begin{aligned}
H(A^{(2)}) &= -P(a_1,a_1)\log_2 P(a_1,a_1) - P(a_1,a_2)\log_2 P(a_1,a_2) \\
&\quad - P(a_2,a_1)\log_2 P(a_2,a_1) - P(a_2,a_2)\log_2 P(a_2,a_2) \\
&= \left[-P(a_1)\log_2 P(a_1) - P(a_2)\log_2 P(a_2)\right] + \left[-P(a_1)\log_2 P(a_1) - P(a_2)\log_2 P(a_2)\right] \\
&= H(A) + H(A)
\end{aligned}
$$

$$H(A^{(2)}) = 2H(A)$$

- Golomb Codes

  Compared with Huffman Codes:

Huffman encoding: dict = huffmandict(symbols, p)



Sig1 = huffmandeco(code,dict);

```
>> symbols = 1:6;
p = [.5 .125 .125 .125 .0625 .0625];
>>[dict,avglen] = huffmandict(symbols,p);
>> Sig = randsrc(100,1,[symbols;p]);

>> code = huffmanenco(Sig,dict);
>> 224/100
ans =
   2.2400
```

Huffman codes:

(1) Code book has to be trained based on the source probability distribution => Hard to adapt to the changing statistics.

(2) Code book has to be stored as "side" information, resulting in loss of compression efficiency.

(3) Decoding is complex.

- Golomb Codes
  (1) Designed to compress non-negative integers.
  (2) Optimal for geometric sources with certain **parameters**.

(3) Variable-length codes

Example: RV with **Geometric** Distribution

X: takes non-negative integer values (n)

$$G(n) = P(X=n) = p^n(1-p), \quad \text{where } 0<p<1 : \text{given parameter}$$

Assume that the probability of a symbol '0' occurring is: $p \Rightarrow P('1') = 1-p$ for binary source

$$P(\underbrace{0\,0 \cdots 0}_{\text{run length} = n}\,1) = p^n \cdot (1-p)$$

Golomb code: $\boxed{p^m = \frac{1}{2}}$.

Golomb Coding Scheme

Code the non-negative integers n = mq + r, where m is a coding parameter (positive integer).

Split the integer n into two parts:

(1) Code q with unary code. Here q is the quotient of (n/m).

Unary code: q 1's, followed by one '0'. Codeword length of this unary code: (q + 1) bits

(2) Code r using binary code. Here r is the remainder of (n/m). Binary code has $(\log_2 m)$ bits

$(\log_2 m)$ bits for binary representation of r.

If m is not power of two, **discuss later** …

Assumption: the integer n's follow the geometric distribution:

$$G(n) = p^n(1-p), \quad \text{where } p^m = \frac{1}{2}.$$

Examples:

n	$G(n)$ $m=1$	Codeword
0	1/2	0
1	1/4	10
2	1/8	110
3	1/16	1110
4	1/32	11110
5	1/64	111110
6	1/128	1111110
7	1/256	11111110
8	1/512	111111110
9	1/1024	1111111110
10	1/2048	11111111110

$m = 1 \Rightarrow p^m = p^1 = p = \frac{1}{2}$.

$$G(n) = \left(\frac{1}{2}\right)^n \cdot \left(1 - \frac{1}{2}\right) = \left(\frac{1}{2}\right)^{n+1}$$

$$\frac{n}{m} = \frac{n}{1} \Rightarrow \begin{cases} q = n \Rightarrow n \text{ 1's, followed by '0'} \\ r = 0 \Rightarrow \underbrace{\text{no binary code}}_{\text{null}} \end{cases}$$

$\log_2 m = \log_2 1 = 0$ bit

$m = 16$			
n	Codeword	n	Codeword
0	00000	24	101000
1	00001	25	101001
2	00010	26	101010
3	00011	27	101011
4	00100	28	101100
5	00101	29	101101
6	00110	30	101110
7	00111	31	101111
8	01000		
9	01001	32	1100000
10	01010	33	1100001
11	01011	34	1100010
12	01100	35	1100011
13	01101	36	1100100
14	01110	37	1100101
15	01111	38	1100110
		39	1100111
16	100000	40	1101000
17	100001	41	1101001
18	100010	42	1101010
19	100011	43	1101011
20	100100	44	1101100
21	100101	45	1101101
22	100110	46	1101110
23	100111	47	1101111

$p^m = \frac{1}{2} \Rightarrow p^{16} = \frac{1}{2} \Rightarrow p = \left(\frac{1}{2}\right)^{\frac{1}{16}}$

$\Downarrow$

$m \log_2 p = -1$

$\Rightarrow m = -\frac{1}{\log_2 p}$

```
>> (1/2)^(1/16)
ans =
   0.9576
```

$G(n) = p^n(1-p)$, where $p^m = \frac{1}{2}$.

$\frac{n}{m} = \frac{n}{16} = \begin{cases} \text{unary code} \\ \text{binary code } (\log_2 m = \log_2 16 = 4 \text{ bits}) \end{cases}$

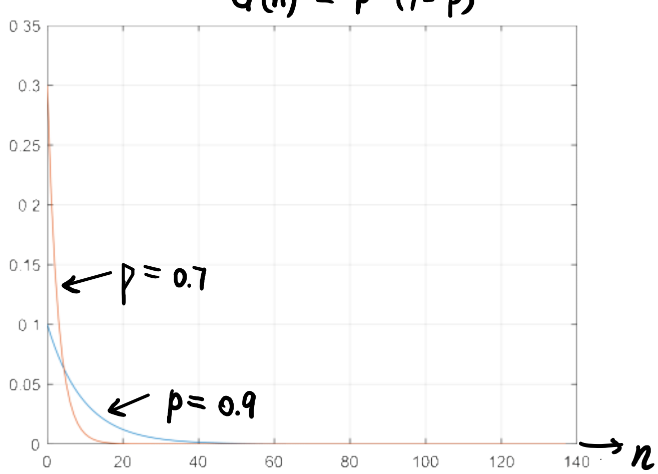
If $n = 27$

$\frac{n}{m} = \frac{27}{16} = \begin{cases} q = 1 \rightarrow \text{'10'} \\ r = 11 \rightarrow \text{'}\underbrace{1011}_{4 \text{ bit}}\text{'} \end{cases}$

$n = 29$

$\frac{n}{m} = \frac{29}{16} = \begin{cases} q = 1 \rightarrow \text{'10'} \\ r = 13 \rightarrow \text{'}\underbrace{1101}_{4 \text{ bit}}\text{'} \end{cases}$

$G(n) = p^n(1-p)$




```
>> p = 0.9;
n = 0: 137;
G = p.^n*(1 - p);
figure; plot(n, G); grid

>> p = 0.7;
>> G = p.^n*(1 - p);
>> hold on;
>> plot(n,G)
>> p = 0.9576;
>> -1/log2(p)
ans =
   15.9987
```

$p^m = \frac{1}{2} \Rightarrow$

$\Downarrow$

$m \log_2 p = -1$

$\Rightarrow m = -\frac{1}{\log_2 p}$

```
>> p = 0.7;
>> -1/log2(p)
ans =
   1.9434
```