追随技术WithPassion

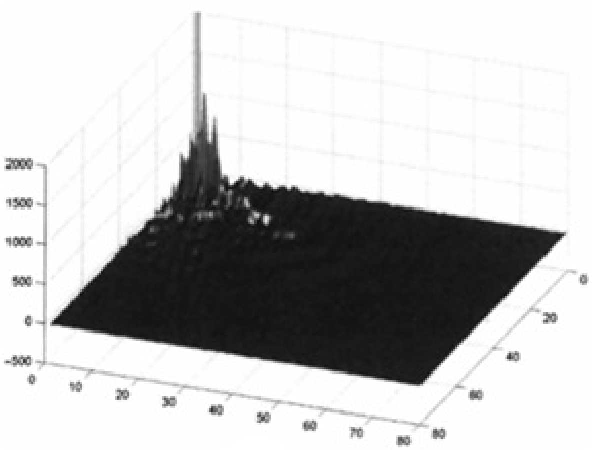
cnblogs.com/xkfz007/archive/2012/07/29/2614250.html

第6章 变换编码

1. 变换编码

- 变换编码的目的
    - 去除空间信号的相关性
    - 将空间信号的能力集中到频域的一小部分低频系数上
    - 能量小的系数可通过量化去除，而不会严重影响重构图像的质量
- 块变换和全局变换
    - 块变换：离散余弦变换(Discrete Cosine Transform，DCT)，4x4，8x8，16x16
    - 全局变换：小波变换(Wavelet)
- 变换的能量集中特性
    - DCT编码

原始图像



变换系数分布

2. 变换类型

- K-L变换
- 傅里叶变换
- 余弦变换
- 小波变换

3. KL变换

- 最优变换
- 基函数根据具体图像而确定
- 没有快速算法

- 实际中很少使用
    - 复杂度极高

- 给定N维随机变量：$\vec{x}=\begin{bmatrix}x_1\\x_2\\\vdots\\x_N\end{bmatrix},\quad \vec{x}\in R^n$
- 向量$\vec{x}$包含了N个随机变量，每个随机变量的数学期望表示为：$m_{\vec{x}}=E(\vec{x})=(m_1,\cdots,m_i,\cdots,m_N)^T$
- 其中 $m_i=E(x_i)\ i=1,2,\cdots,N$
- 利用向量$\vec{x}$的数学期望，可以得到$\vec{x}$向量的协方差矩阵

$$U=E((\vec{x}-m_{\vec{x}})(\vec{x}-m_{\vec{x}})^T)$$

- 协方差矩阵U的特征向量ϕ_k对应着其第k个特征值λ_k，则有 $U\phi_k=\lambda_k\phi_k\ (k=0,\cdots,N-1)$
- U是对称矩阵，所以其特征向量ϕ_k是正交的，即满足：

$$\phi_k^T\phi_l\begin{cases}=0,(k\neq l)\\ \neq 0,(k=l)\end{cases}$$

- 归一化可以得到单位正交矩阵$\Phi=(\phi_0,\phi_1,\cdots,\phi_{N-1})$，使其满足$\Phi^T\Phi=I$，则N个特征向量可以联合起来表示为：

$$U\Phi=(\lambda_0\phi_0,\lambda_1\phi_1,\cdots\lambda_{N-1}\phi_{N-1})$$

- 由于Φ是正交阵，因此在上式的两边分别左乘Φ^T可以得到$\Phi^TU\Phi=\Lambda$ 其中$\Lambda=diag(\lambda_0,\cdots,\lambda_{N-1})$
- 给定一维随机向量X，可以定义X的K-L变换为：

$$\begin{bmatrix}y_1\\y_2\\\vdots\\y_n\end{bmatrix}=\Phi^T\begin{bmatrix}x_1\\x_2\\\vdots\\x_n\end{bmatrix}$$

- 即$Y=\Phi^TX$，K-L变换就是将X的所有分量投影在ϕ_k得到频域映射y_k。Φ^T是K-L变换矩阵，显然它随着随机向量$\vec{x}$中每个成分的变化而改变。

- 变换后向量Y的均值为 $m_{\vec{y}} = E(\vec{y}) = E(\Phi^T X) = \Phi^T E(X) = \Phi^T m_{\vec{x}}$
- Y的协方差矩阵为

$$E((\vec{y}-m_{\vec{y}})(\vec{y}-m_{\vec{y}})^T) = E((\Phi^T\vec{x}-m_{\vec{y}})(\Phi^T\vec{x}-m_{\vec{y}})^T)$$

$$= E((\Phi^T\vec{x}-\Phi^T m_{\vec{x}})(\Phi^T\vec{x}-\Phi^T m_{\vec{x}})^T)$$

$$= E(\Phi^T(\vec{x}-m_{\vec{x}})(\vec{x}-m_{\vec{x}})^T\Phi) = \Phi^T E((\vec{x}-m_{\vec{x}})(\vec{x}-m_{\vec{x}})^T)\Phi$$

$$= \Phi^T U \Phi = \begin{pmatrix} \phi_0^T \\ \phi_1^T \\ \vdots \\ \phi_{N-1}^T \end{pmatrix} U(\phi_0, \phi_1, \cdots, \phi_{N-1}) = \begin{pmatrix} \phi_0^T \\ \phi_1^T \\ \vdots \\ \phi_{N-1}^T \end{pmatrix} (U\phi_0, U\phi_1, \cdots, U\phi_{N-1})$$

$$= \begin{pmatrix} \phi_0^T \\ \phi_1^T \\ \vdots \\ \phi_{N-1}^T \end{pmatrix} (U\phi_0, U\phi_1, \cdots, U\phi_{N-1}) = \begin{pmatrix} \phi_0^T \\ \phi_1^T \\ \vdots \\ \phi_{N-1}^T \end{pmatrix} (\lambda_0\phi_0, \lambda_1\phi_1, \cdots, \lambda_{N-1}\phi_{N-1}) = \begin{bmatrix} \lambda_0 & 0 & \cdots & 0 \\ 0 & \lambda_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \lambda_{N-1} \end{bmatrix}$$

- Y向量协方差矩阵是一个对角阵，说明Y向量之间的相关性最小，而X向量协方差矩阵非对角元素不为零，说明X向量有较强的相关性。

- 由于 $\Phi = (\Phi^T)^{-1}$，在K-L变换两边分别乘以 Φ 可以得到 X，即 $\Phi Y = \Phi\Phi^T X = X$ 这就是K-L反变换。

- K-L变换非常复杂度很高，不实用
  - 需要计算协方差矩阵 U
  - 需要计算特征向量
  - 需要发送 到解码器

4. 离散傅立叶变换

- 一维离散傅里叶变换：

$$F(u)=\frac{1}{N}\sum_{n=0}^{N-1}f(n)e^{-j\frac{2\pi nu}{N}}=\frac{1}{N}\sum_{n=0}^{N-1}f(n)W_N^{nu},u=0,\cdots,N-1$$

- 一维反离散傅里叶变换：

$$f(n)=\sum_{u=0}^{N-1}F(u)e^{j2\pi\frac{nu}{N}}=\sum_{u=0}^{N-1}F(u)W_N^{-nu},u=0,\cdots,N-1$$

- 二维离散傅里叶变换：

$$F(u,v)=\frac{1}{NM}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}f(n,m)e^{-j2\pi(\frac{nu}{N}+\frac{mv}{M})}$$

$$=\frac{1}{NM}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}f(n,m)W_{NM}^{\frac{nu}{N}+\frac{mv}{M}},u=0,\cdots,N-1,v=0,\cdots,M-1$$

- 二维反离散傅里叶变换：

$$f(n,m)=\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}F(u,v)e^{-j2\pi(\frac{nu}{N}+\frac{mv}{M})}$$

$$=\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}F(u,v)W_{MN}^{-(\frac{nu}{N}+\frac{mv}{M})},u=0,\cdots,N-1,v=0,\cdots,M-1$$

5. 离散傅立叶变换性质

- 线性： $DFT[af_1(n)+bf_2(n)]=aF_1(k)+bF_2(k)$
- 平移行：
  - 时间平移 $f(n-k)\Leftrightarrow F(u)\cdot W^{ku}$
  - 频率平移 $f(n)\cdot W^{-kn}\Leftrightarrow F(n-k)$
- 周期性： $f(n\pm rN)=f(n)$
- 奇偶性：
  - 实部是偶函数 $f_e(n)=f_e(-n)$
  - 虚部是奇函数 $f_o(n)=-f_o(-n)$

- 正交性：
- W_N可以表示为矩阵形式

$$W_N=[W^{mu}]=\begin{bmatrix} W^{00} & W^{01} & W^{02} & \cdots & W^{0(N-1)} \\ W^{10} & W^{11} & W^{12} & \cdots & W^{1(N-1)} \\ W^{20} & W^{21} & W^{22} & \cdots & W^{2(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ W^{(N-1)0} & W^{(N-1)1} & W^{(N-1)2} & \cdots & W^{(N-1)(N-1)} \end{bmatrix}$$

$$W_N^T W_N=\sum_{u=0}^{N-1} W^{mu}W^{-nu}=\sum_{u=0}^{N-1} W^{(m-n)u}=\begin{cases} N & m=n \\ 0 & m\neq n \end{cases}$$

6. 离散余弦变换

- 比K-L变换，傅里叶变换的复杂度更低
- 变换性能仅次于K-L变换
- 有快速算法可以加快变换速度
- 可以用整数变换进一步降低复杂度

- DCT的类型：
  - DCT-I：$F(u)=\frac{1}{2}(f(0)+(-1)^u f(N-1))+\sum_{n=1}^{N-2} f(n)\cos\frac{nu\pi}{N-1}, u=0,\cdots,N-1$
  - DCT-II：$F(u)=\sum_{n=0}^{N-1} f(n)\cos\frac{(2n+1)u\pi}{2N}, u=0,\cdots,N-1$
  - DCT-III：$F(u)=\frac{1}{2}f(0)+\sum_{n=1}^{N-1} f(n)\cos\frac{n(2u+1)\pi}{2N}, u=0,\cdots,N-1$
  - DCT-IV：$F(u)=\sum_{n=0}^{N-1} f(n)\cos\frac{(2n+1)(2u+1)\pi}{2N}, u=0,\cdots,N-1$

- 视频编码标准中长采用DCT-II
- DCT的反变换为：

$$f(n)=\sum_{u=0}^{N-1}c(u)F(u)\cos\frac{(2n+1)u\pi}{2N},n=0,\cdots,N-1 \qquad c(u)=\begin{cases}1/\sqrt{2} & u=0\\ 1 & 其它\end{cases}$$

- N点1-D DCT/IDCT的模(Norm)为$\sqrt{\frac{N}{2}}$，则归一化的1-D DCT/IDCT为：

$$F(u)=\sqrt{\frac{2}{N}}c(u)\sum_{n=0}^{N-1}f(n)\cos\frac{(2n+1)u\pi}{2N},u=0,\cdots,N-1$$

$$f(n)=\sqrt{\frac{2}{N}}\sum_{u=0}^{N-1}c(u)F(u)\cos\frac{(2n+1)u\pi}{2N},n=0,\cdots,N-1$$

- 常用的离散余弦变换为8点DCT
  - JPEG，MPEG-2，MPEG-4

$$F(u)=\frac{c(u)}{2}\sum_{n=0}^{7}f(n)\cos\frac{(2n+1)u\pi}{16},u=0,\cdots,7$$

$$f(n)=\sum_{u=0}^{7}\frac{c(u)}{2}F(u)\cos\frac{(2n+1)u\pi}{16},n=0,\cdots,7$$

- DCT的变换核可以用矩阵形式表示为：

$$C_8=\frac{1}{\sqrt{8}}\begin{bmatrix}1 & 1 & 1 & \cdots & 1\\ \sqrt{2}\cos\frac{\pi}{16} & \sqrt{2}\cos\frac{3\pi}{16} & \sqrt{2}\cos\frac{5\pi}{16} & \cdots & \sqrt{2}\cos\frac{15\pi}{16}\\ \vdots & \vdots & \vdots & \vdots & \vdots\\ \sqrt{2}\cos\frac{7\pi}{16} & \sqrt{2}\cos\frac{21\pi}{16} & \sqrt{2}\cos\frac{35\pi}{16} & \cdots & \sqrt{2}\cos\frac{105\pi}{16}\end{bmatrix}$$

- 2-D DCT

$$F(u,v)=\frac{2}{N}c(u)c(v)\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}f(n,m)\cos\frac{(2n+1)u\pi}{2N}\cos\frac{(2m+1)v\pi}{2M},u=0,\cdots,N-1,v=0,\cdots,M-1$$

- 2-D IDCT

$$f(n,m)=\frac{2}{N}\sum_{u=0}^{N-1}\sum_{m=0}^{M-1}c(u)c(v)F(u,v)\cos\frac{(2n+1)u\pi}{2N}\cos\frac{(2m+1)v\pi}{2M},n=0,\cdots,N-1,m=0,\cdots,M-1$$

- 8点2-D DCT

$$F(u,v)=\frac{c(u)}{2}\frac{c(v)}{2}\sum_{n=0}^{7}\sum_{m=0}^{7}f(n,m)\cos\frac{(2n+1)u\pi}{16}\cos\frac{(2m+1)v\pi}{16},u,v=0,\cdots,7$$

- 8点2-D IDCT

$$f(n,m)=\sum_{u=0}^{7}\frac{c(u)}{2}\sum_{m=0}^{7}\frac{c(v)}{2}F(u,v)\cos\frac{(2n+1)u\pi}{16}\cos\frac{(2m+1)v\pi}{16},n,m=0,\cdots,7$$

7. DCT与DFT的关系

- 将$f(n)$ $(0\le n\le N)$拓展成2N点DFT输入$f_{2N}(n)$，即

$$f_{2N}(n)=\begin{cases}f(n) & (0\le n\le N-1)\\ 0 & (N\le n\le 2N-1)\end{cases}$$

求$f_{2N}(n)$的2N点DFT

- 右项缩放$e^{-j\pi\frac{\mu}{2N}}$ 得到$F_{2N}(u)=\sum_{n=0}^{2N-1}f_{2N}(n)e^{-j2\pi\frac{\mu n}{2N}},u=0,\cdots,2N-1$

$$F_{2N}(u)=e^{-j\pi\frac{\mu}{2N}}\bullet\sum_{n=0}^{2N-1}f_{2N}(n)e^{-j2\pi\frac{\mu n}{2N}}=\sum_{n=0}^{2N-1}f_{2N}(n)e^{-j\pi\frac{\mu(2n+1)}{2N}}$$

$$=\sum_{n=0}^{2N-1}f_{2N}(n)(\cos(\frac{u\pi(2n+1)}{2N})-j\sin(\frac{u\pi(2n+1)}{2N})),u=0,\cdots,2N-1$$

- 取实部$\mathrm{Re}(F_{2N}(u))$可以得到

$$\mathrm{Re}(F_{2N}(u))=\sum_{n=0}^{2N-1}f_{2N}(n)\cos(\frac{u\pi(2n+1)}{2N})=\sum_{n=0}^{N-1}f(n)\cos(\frac{u\pi(2n+1)}{2N})$$

8. 离散余弦变换的重要性质

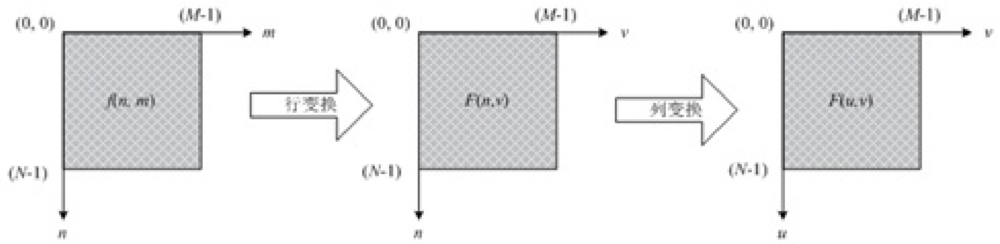
- 二维离散余弦变换的可分离性：
  - 二维函数$f(n,m)$的一维离散余弦变换为：

$$F(n,v)=\sum_{m=0}^{M-1} f(n,m)\cos\frac{(2m+1)v\pi}{2M}, v=0,\cdots,M-1$$

  - 对$F(n,v)$的一维离散余弦变换为：

$$F(u,v)=\frac{2}{N}c(u)c(v)\sum_{n=0}^{N-1} F(n,v)\cos\frac{(2n+1)u\pi}{2N}, u=0,\cdots,N-1, v=0,\cdots,M-1$$

$$F=[(Tf)T^T]=TfT^T$$



- 对称（反对称）性：

$$\cos\frac{(2(N-1-n)+1)u\pi}{2N}=\cos\frac{((2N-2-2n)+1)u\pi}{2N}=\cos\left(u\pi-\frac{(2n+1)u\pi}{2N}\right)$$

$$=\pm\cos\frac{(2n+1)u\pi}{2N}$$

$$\cos\frac{13\pi}{16}=-\cos\frac{3\pi}{16},\quad N=8, n=1, u=1 \qquad \cos\frac{22\pi}{16}=\cos\frac{10\pi}{16},\quad N=8, n=2, u=2$$

$$\cos\frac{45\pi}{16}=-\cos\frac{3\pi}{16},\quad N=8, n=0, u=3 \qquad \cos\frac{36\pi}{16}=\cos\frac{28\pi}{16},\quad N=8, n=3, u=4$$

$$\cos\frac{65\pi}{16}=-\cos\frac{15\pi}{16},\quad N=8, n=1, u=5 \qquad \cos\frac{42\pi}{16}=\cos\frac{54\pi}{16},\quad N=8, n=4, u=6$$

$$\cos\frac{105\pi}{16}=-\cos\frac{7\pi}{16},\quad N=8, n=0, u=7 \qquad \cos\frac{40\pi}{16}=\cos\frac{88\pi}{16},\quad N=8, n=5, u=8$$

- 正交性：$C^T C = I$

$$C_8 = \frac{1}{\sqrt{8}}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ \sqrt{2}\cos\frac{\pi}{16} & \sqrt{2}\cos\frac{3\pi}{16} & \sqrt{2}\cos\frac{5\pi}{16} & \cdots & \sqrt{2}\cos\frac{15\pi}{16} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \sqrt{2}\cos\frac{7\pi}{16} & \sqrt{2}\cos\frac{21\pi}{16} & \sqrt{2}\cos\frac{35\pi}{16} & \cdots & \sqrt{2}\cos\frac{105\pi}{16} \end{bmatrix}$$

- 三角函数公式：

$$\begin{aligned}
&\sin(-a) = -\sin(a) \quad \cos(-a) = \cos(a) \\
&\cos(\pi - a) = -\cos(a) \ \sin(\pi - a) = \sin(a) \\
&\cos(\pi + a) = -\cos(a) \ \sin(\pi + a) = -\sin(a) \\
&\cos(\pi/2 - a) = \sin(a) \ \sin(\pi/2 - a) = \cos(a) \\
&\cos(\pi/2 + a) = -\sin(a) \ \sin(\pi/2 + a) = \cos(a) \\
&\sin(a+b) = \sin(a)\cos(b) + \cos(a)\sin(b) \\
&\sin(a-b) = \sin(a)\cos(b) - \cos(a)\sin(b) \\
&\cos(a+b) = \cos(a)\cos(b) - \sin(a)\sin(b) \\
&\cos(a-b) = \cos(a)\cos(b) + \sin(a)\sin(b)
\end{aligned}$$

- 正交性：

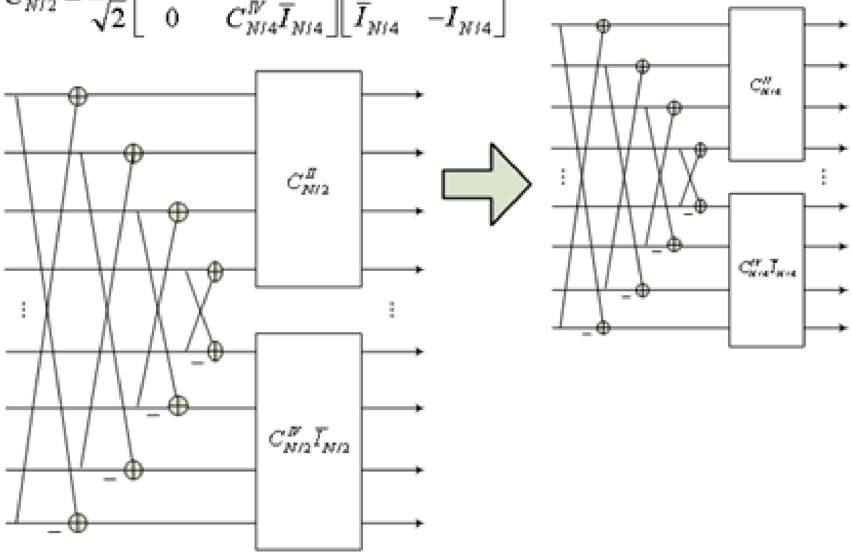
$$\begin{aligned}
&\frac{1}{4}(\cos\frac{\pi}{16}\cos\frac{7\pi}{16} + \cos\frac{3\pi}{16}\cos\frac{21\pi}{16} + \cos\frac{5\pi}{16}\cos\frac{35\pi}{16} + \cos\frac{7\pi}{16}\cos\frac{49\pi}{16} \\
&+\cos\frac{9\pi}{16}\cos\frac{63\pi}{16} + \cos\frac{11\pi}{16}\cos\frac{77\pi}{16} + \cos\frac{13\pi}{16}\cos\frac{91\pi}{16} + \cos\frac{15\pi}{16}\cos\frac{105\pi}{16}) \\
&= \frac{1}{4}\left[(\cos\frac{8\pi}{16} + \cos\frac{6\pi}{16}) + (\cos\frac{24\pi}{16} + \cos\frac{18\pi}{16}) + \ldots + (\cos\frac{104\pi}{16} + \cos\frac{78\pi}{16}) + (\cos\frac{120\pi}{16} + \cos\frac{90\pi}{16})\right] \\
&= \frac{1}{4}\left[\cos\frac{6\pi}{16} + \cos\frac{18\pi}{16} + \cos\frac{30\pi}{16} + \cos\frac{42\pi}{16} + \cos\frac{54\pi}{16} + \cos\frac{66\pi}{16} + \cos\frac{78\pi}{16} + \cos\frac{90\pi}{16}\right] \\
&= \frac{1}{4}[\cos\frac{6\pi}{16} - \cos\frac{2\pi}{16} + \cos\frac{2\pi}{16} + \cos\frac{10\pi}{16} - \cos\frac{6\pi}{16} + \cos\frac{2\pi}{16} - \cos\frac{2\pi}{16} - \cos\frac{10\pi}{16}] = 0
\end{aligned}$$

$$\begin{aligned}
&\frac{1}{4}[\cos(\frac{\pi}{16})^2 + \cos(\frac{3\pi}{16})^2 + \cos(\frac{5\pi}{16})^2 + \cos(\frac{7\pi}{16})^2 + \cos(\frac{9\pi}{16})^2 + \cos(\frac{11\pi}{16})^2 + \cos(\frac{13\pi}{16})^2 + \cos(\frac{15\pi}{16})^2] \\
&= \frac{1}{4}[\cos(\frac{\pi}{16})^2 + \cos(\frac{3\pi}{16})^2 + \cos(\frac{5\pi}{16})^2 + \cos(\frac{7\pi}{16})^2 + \cos(\frac{7\pi}{16})^2 + \cos(\frac{5\pi}{16})^2 + \cos(\frac{3\pi}{16})^2 + \cos(\frac{\pi}{16})^2] \\
&= \frac{1}{2}[\cos(\frac{\pi}{16})^2 + \cos(\frac{3\pi}{16})^2 + \cos(\frac{5\pi}{16})^2 + \cos(\frac{7\pi}{16})^2] \\
&= [\cos(\frac{\pi}{16})^2 + \cos(\frac{3\pi}{16})^2 + \sin(\frac{3\pi}{16})^2 + \sin(\frac{\pi}{16})^2] = 1
\end{aligned}$$

- 递归性 $C_N^{II}=\frac{1}{\sqrt{2}}\begin{bmatrix} C_{N/2}^{II} & 0 \\ 0 & C_{N/2}^{IV}\bar{I}_{N/4} \end{bmatrix}\begin{bmatrix} I_{N/2} & \bar{I}_{N/2} \\ \bar{I}_{N/2} & -I_{N/2} \end{bmatrix}$

$\vdots$

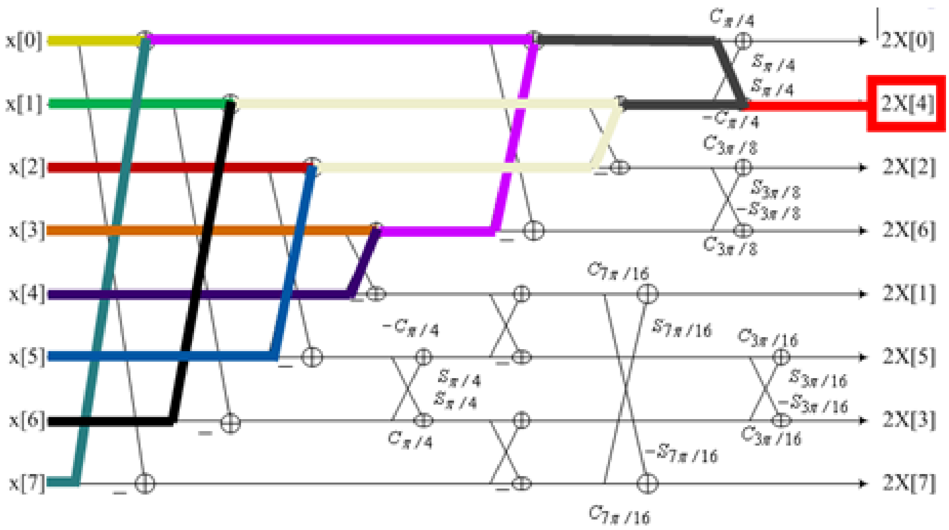
$$C_{N/2}^{II}=\frac{1}{\sqrt{2}}\begin{bmatrix} C_{N/4}^{II} & 0 \\ 0 & C_{N/4}^{IV}\bar{I}_{N/4} \end{bmatrix}\begin{bmatrix} I_{N/4} & \bar{I}_{N/4} \\ \bar{I}_{N/4} & -I_{N/4} \end{bmatrix}$$



9. 快速DCT变换

- 以一维8x8变换为例
- $$C_8=\frac{1}{\sqrt{8}}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ \sqrt{2}\cos\frac{\pi}{16} & \sqrt{2}\cos\frac{3\pi}{16} & \sqrt{2}\cos\frac{5\pi}{16} & \cdots & \sqrt{2}\cos\frac{15\pi}{16} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \sqrt{2}\cos\frac{7\pi}{16} & \sqrt{2}\cos\frac{21\pi}{16} & \sqrt{2}\cos\frac{35\pi}{16} & \cdots & \sqrt{2}\cos\frac{105\pi}{16} \end{bmatrix}$$
- 对于DCT系数X[4]:

$$X[4]=\frac{1}{2}\sum_{n=0}^{7}x[n]\cos\frac{(2n+1)4\pi}{16}$$
$$=\frac{1}{2}(x[0]\cos\frac{4\pi}{16}+x[1]\cos\frac{12\pi}{16}+x[2]\cos\frac{20\pi}{16}+x[3]\cos\frac{28\pi}{16}+x[4]\cos\frac{36\pi}{16}+x[5]\cos\frac{44\pi}{16}+x[6]\cos\frac{52\pi}{16}+x[7]\cos\frac{60\pi}{16})$$
$$=\frac{1}{2}\left[(x[0]+x[7]+x[3]+x[4])\cos\frac{\pi}{4}+(x[1]+x[6]+x[2]+x[5])(-\cos\frac{\pi}{4})\right]$$



$$X[4]=$$

$$\{(x[0]+x[7])+(x[3]+x[4])\}S_{\pi/4}+\{(x[1]+x[6])+(x[2]+x[5])\}C_{\pi/4}$$

下图是一个动态展示：

$$\cos\frac{3\pi}{16}=-\cos\frac{45\pi}{16}\quad \cos\frac{9\pi}{16}=-\cos\frac{39\pi}{16}$$
$$\cos\frac{15\pi}{16}=-\cos\frac{33\pi}{16}\quad \cos\frac{21\pi}{16}=-\cos\frac{27\pi}{16}$$
$$\cos\frac{21\pi}{16}=-\cos\frac{5\pi}{16}\quad \sin\frac{\pi}{4}=\cos\frac{\pi}{4}$$

- 对于DCT系数X[3]：

$$X[3]=\frac{1}{2}\sum_{n=0}^{7}x[n]\cos\frac{(2n+1)3\pi}{16}$$

$$=\frac{1}{2}\left(x[0]\cos\frac{3\pi}{16}+x[1]\cos\frac{9\pi}{16}+x[2]\cos\frac{15\pi}{16}+x[3]\cos\frac{21\pi}{16}+x[4]\cos\frac{27\pi}{16}+x[5]\cos\frac{33\pi}{16}+x[6]\cos\frac{39\pi}{16}+x[7]\cos\frac{45\pi}{16}\right)$$

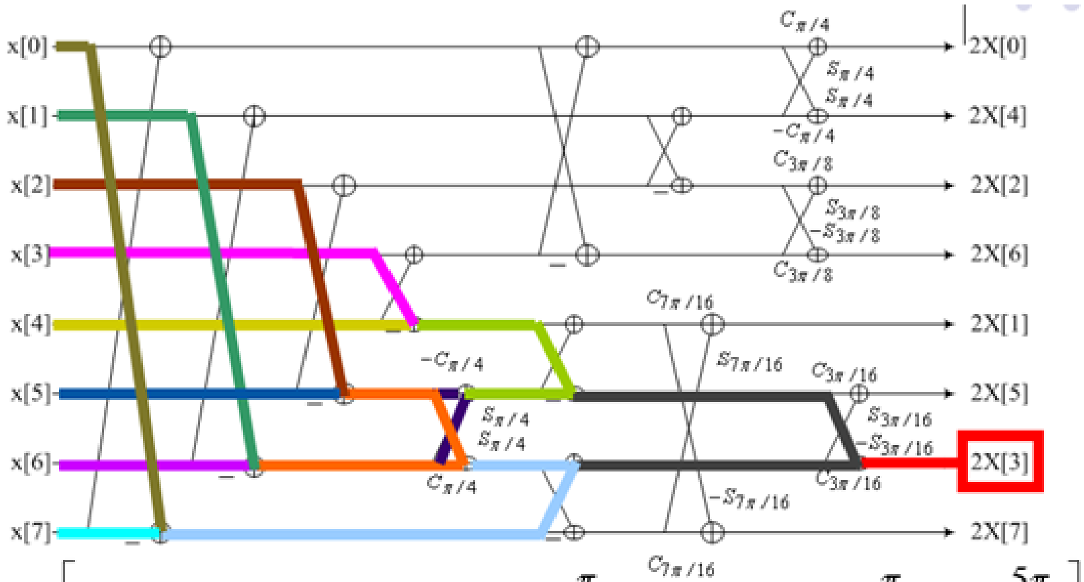
$$=\frac{1}{2}\left[(x[0]-x[7])\cos\frac{3\pi}{16}+(x[1]-x[6])\cos\frac{9\pi}{16}+(x[2]-x[5])\cos\frac{15\pi}{16}+(x[3]-x[4])\cos\frac{21\pi}{16}\right]$$

$$=\frac{1}{2}\left[(x[0]-x[7])\cos\frac{3\pi}{16}+(x[1]-x[6])\cos\left(\frac{5\pi}{16}+\frac{4\pi}{16}\right)+(x[2]-x[5])\cos\left(\frac{20\pi}{16}-\frac{5\pi}{16}\right)+(x[3]-x[4])\left(-\cos\frac{5\pi}{16}\right)\right]$$

$$=\frac{1}{2}\left[(x[0]-x[7])\cos\frac{3\pi}{16}+(x[1]-x[6])\left(\cos\frac{5\pi}{16}\cos\frac{4\pi}{16}-\sin\frac{5\pi}{16}\sin\frac{4\pi}{16}\right)+(x[2]-x[5])\left(\cos\frac{20\pi}{16}\cos\frac{5\pi}{16}+\sin\frac{20\pi}{16}\sin\frac{5\pi}{16}\right)+(x[3]-x[4])\left(-\cos\frac{5\pi}{16}\right)\right]$$

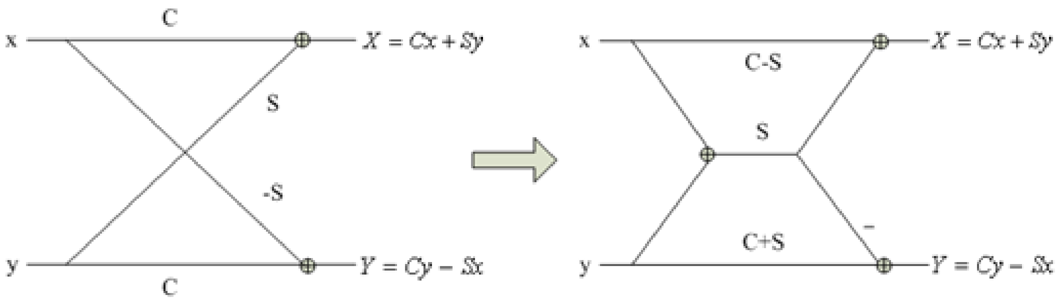
$$=\frac{1}{2}\left[(x[0]-x[7])\cos\frac{3\pi}{16}+(x[1]-x[6])\left(\cos\frac{5\pi}{16}\cos\frac{4\pi}{16}-\cos\frac{3\pi}{16}\cos\frac{4\pi}{16}\right)+(x[2]-x[5])\left(\cos\frac{4\pi}{16}\cos\frac{5\pi}{16}+\cos\frac{4\pi}{16}\cos\frac{3\pi}{16}\right)+(x[3]-x[4])\left(-\cos\frac{5\pi}{16}\right)\right]$$

$$=\frac{1}{2}\left[\begin{array}{l}\left[(x[0]-x[7])-(x[2]-x[5])\cos\frac{\pi}{4}-(x[1]-x[6])\cos\frac{\pi}{4}\right]\cos\frac{3\pi}{16}\\+\left[(x[3]-x[4])+(x[2]-x[5])\left(-\cos\frac{\pi}{4}\right)-(x[1]-x[6])\cos\frac{\pi}{4}\right]\left(-\cos\frac{5\pi}{16}\right)\end{array}\right]$$



$$X[3]=\frac{1}{2}\left[\begin{array}{l}[(x[3]-x[4])+(x[2]-x[5])(-\cos\frac{\pi}{4})-(x[1]-x[6])\cos\frac{\pi}{4}](-\cos\frac{5\pi}{16})\\+[(x[0]-x[7])-(x[2]-x[5])\cos\frac{\pi}{4}-(x[1]-x[6])\cos\frac{\pi}{4}]\cos\frac{3\pi}{16}\end{array}\right]$$

- 旋转平面



$$\begin{pmatrix}X\\Y\end{pmatrix}=\begin{pmatrix}C & S\\-S & C\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix}$$

$$X = Cx + Sy$$

$$Y = Cy - Sx$$

4个乘法2个加法

$$tmp = S(x+y)$$

$$X = (C-S)x + tmp$$

$$Y = (C+S)y - tmp$$

3个乘法3个加法

10. 整数离散余弦变换

- 离散余弦变换为浮点操作
  - 需要64位精度
  - 浮点计算复杂度高
  - 变换精度高

- 整数变换：离散余弦变换的整数近似
    - 需要更少的位宽
    - 整数计算复杂度低
    - 好的整数变换的变换精度接近浮点变换
- 浮点近似方法

$I = round(dR)$ 其中，$d=2^n$，n为位宽

$round(\bullet)$四舍五入操作

$$R \approx \frac{I}{d}$$

$$R = \cos\frac{\pi}{4} = 0.707107, d = 2^3 = 8$$

$$round(dR) = round(8\cos\frac{\pi}{4}) = round(5.656856) = 6$$

$$\cos\frac{\pi}{4} \approx \frac{6}{8} = 0.75$$

任何整数都可以表示为：

$$I = a_n 2^n + a_{n-1} 2^{n-1} + \cdots + a_0 2^0,\ a_i = 0 or 1$$

$$\frac{I}{d} = \frac{(a_n 2^n + a_{n-1} 2^{n-1} + \cdots + a_0 2^0)}{2^m}$$

$$= a_n 2^{n-m} + a_{n-1} 2^{n-1-m} + \cdots + a_0 2^{-m}$$

$$C\frac{I}{d} = Ca_n 2^{n-m} + Ca_{n-1} 2^{n-1-m} + \cdots + Ca_0 2^{-m}$$

$$C2^k \begin{cases} C << k, & k > 0 \\ C >> k & k < 0 \end{cases}$$

- 例： $\cos\frac{\pi}{4}\approx\frac{6}{8}=0.75$

$$f\cos\frac{\pi}{4}\approx\frac{6f}{8}=\frac{(4+2)}{8}f=(\frac{1}{2}+\frac{1}{4})f=(f>>1)+(f>>2)$$

浮点数的近似精度与位宽成正比

$$R=\cos\frac{\pi}{4}=0.707107,\ d=2^4=16$$

$$round(dR)=round(16\cos\frac{\pi}{4})=round(11.313712)=11$$

$$\cos\frac{\pi}{4}\approx\frac{11}{16}=0.6875$$

$$f\cos\frac{\pi}{4}\approx\frac{11f}{16}=\frac{(8+2+1)}{16}f=(\frac{1}{2}+\frac{1}{8}+\frac{1}{16})f=(f>>1)+(f>>3)+(f>>4)$$

11. H.264的4x4整数变换

- 一维**4x4**浮点变换矩阵

$$\begin{bmatrix} 1/2 & 1/2 & 1/2 & 1/2 \\ \sqrt{1/2}\cos(\pi/8) & \sqrt{1/2}\cos(3\pi/8) & -\sqrt{1/2}\cos(3\pi/8) & -\sqrt{1/2}\cos(\pi/8) \\ 1/2 & -1/2 & -1/2 & 1/2 \\ \sqrt{1/2}\cos(3\pi/8) & -\sqrt{1/2}\cos(\pi/8) & \sqrt{1/2}\cos(\pi/8) & \sqrt{1/2}\cos(3\pi/8) \end{bmatrix}$$

$$\begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & c \end{bmatrix} \quad a=1/2; b=\sqrt{1/2}\cos(\pi/8); c=\sqrt{1/2}\cos(3\pi/8)$$

- 整数近似

$$\begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & c \end{bmatrix} = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & d & -d & -1 \\ 1 & -1 & -1 & 1 \\ d & -1 & 1 & -d \end{bmatrix}$$

$d = c/b = \sqrt{2} - 1 = 0.414213\ldots$ 令d＝1/2

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & d & -d & -1 \\ 1 & -1 & -1 & 1 \\ d & -1 & 1 & -d \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1/2 & -1/2 & -1 \\ 1 & -1 & -1 & 1 \\ 1/2 & -1 & 1 & -1/2 \end{bmatrix}$$

由d＝1/2，$b^2+c^2=1$，d=c/b根据b和d的关系得到b值为：

$$b = \sqrt{\frac{1}{2(1+d^2)}} = \sqrt{2/5}$$

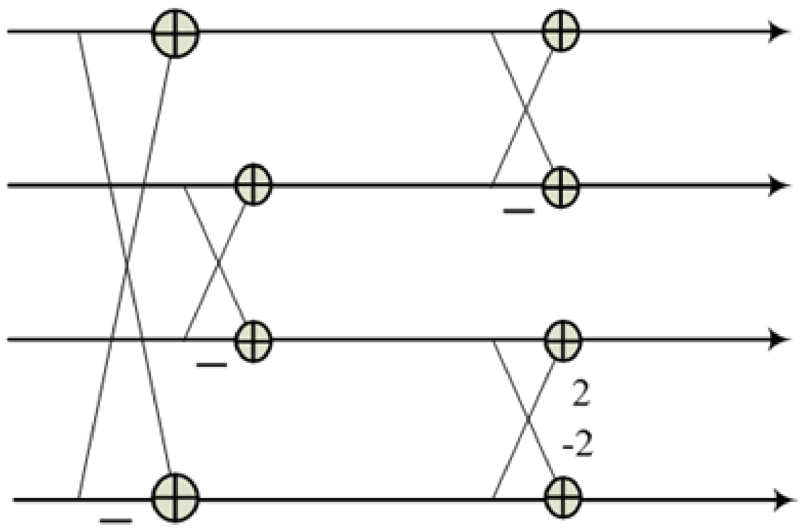
$$\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b \end{bmatrix} = \begin{bmatrix} 1/2 & 0 & 0 & 0 \\ 0 & \sqrt{2/5} & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & \sqrt{2/5} \end{bmatrix}$$

- 将变换核矩阵的1/2因子移到缩放因子矩阵中

$$\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1/2 & -1/2 & -1 \\ 1 & -1 & -1 & 1 \\ 1/2 & -1 & 1 & -1/2 \end{bmatrix} = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b/2 & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b/2 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}}_{\text{变换核}} \qquad \underbrace{\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b/2 & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b/2 \end{bmatrix}}_{\text{缩放因子}}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$



H.264 4x4整数变换的蝶形图

- 二维**4x4**整数变换

$$A = BC = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b/2 & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b/2 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

$$Y = AXA^T = (BC)X(BC)^T = B(CXC^T)B^T$$

$$= (CXC^T) \otimes \begin{bmatrix} a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \\ a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \end{bmatrix}$$

- 缩放因子矩阵与量化矩阵合并

$$\begin{bmatrix} q_{00} & q_{01} & q_{02} & q_{03} \\ q_{10} & q_{11} & q_{12} & q_{13} \\ q_{20} & q_{21} & q_{22} & q_{23} \\ q_{30} & q_{31} & q_{32} & q_{33} \end{bmatrix} \otimes \begin{bmatrix} a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \\ a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \end{bmatrix}$$

$$= \begin{bmatrix} q_{00}a^2 & q_{01}ab/2 & q_{02}a^2 & q_{03}ab/2 \\ q_{10}ab/2 & q_{11}b^2/4 & q_{12}ab/2 & q_{13}b^2/4 \\ q_{20}a^2 & q_{21}ab/2 & q_{22}a^2 & q_{23}ab/2 \\ q_{30}ab/2 & q_{31}b^2/4 & q_{32}ab/2 & q_{33}b^2/4 \end{bmatrix}$$

- 最简单的一种变换

$$H_1 = [1] \quad H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad H_4 = \begin{bmatrix} H_2 & H_2 \\ H_2 & -H_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

$$H_N = H_{2^n} = H_2 \otimes H_{2^{n-1}} = \begin{bmatrix} H_{2^{n-1}} & H_{2^{n-1}} \\ H_{2^{n-1}} & -H_{2^{n-1}} \end{bmatrix}$$

12. 小波变换

- 新的变换方法
- 避免由于块编码带来的块效应
- 更适合视频空间可分级编码

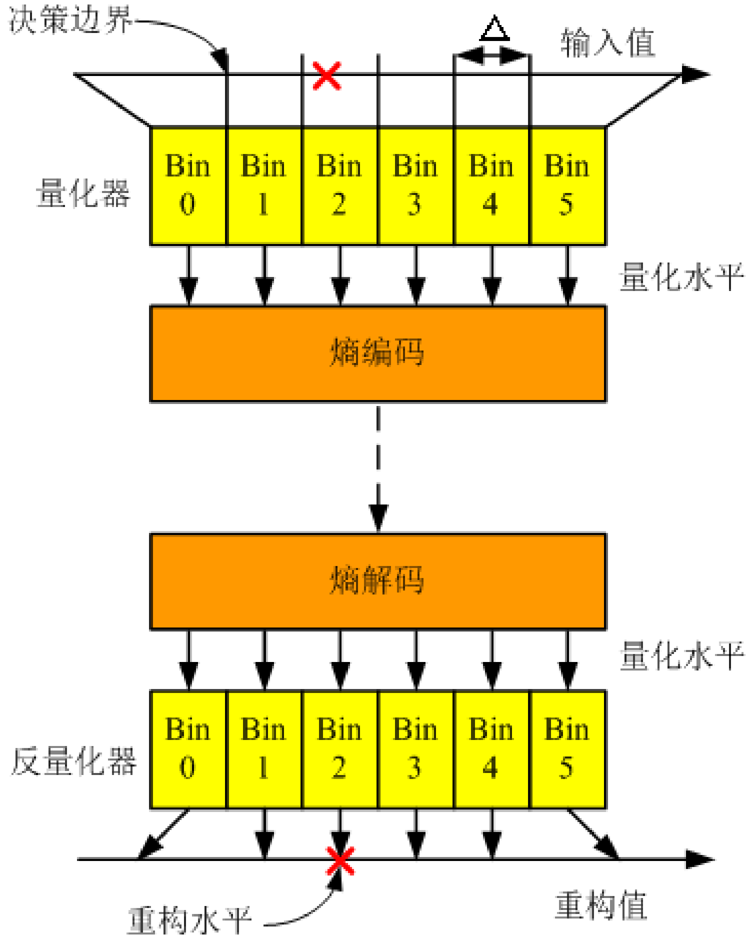
第7章 量化

1. 量化**Quantization**

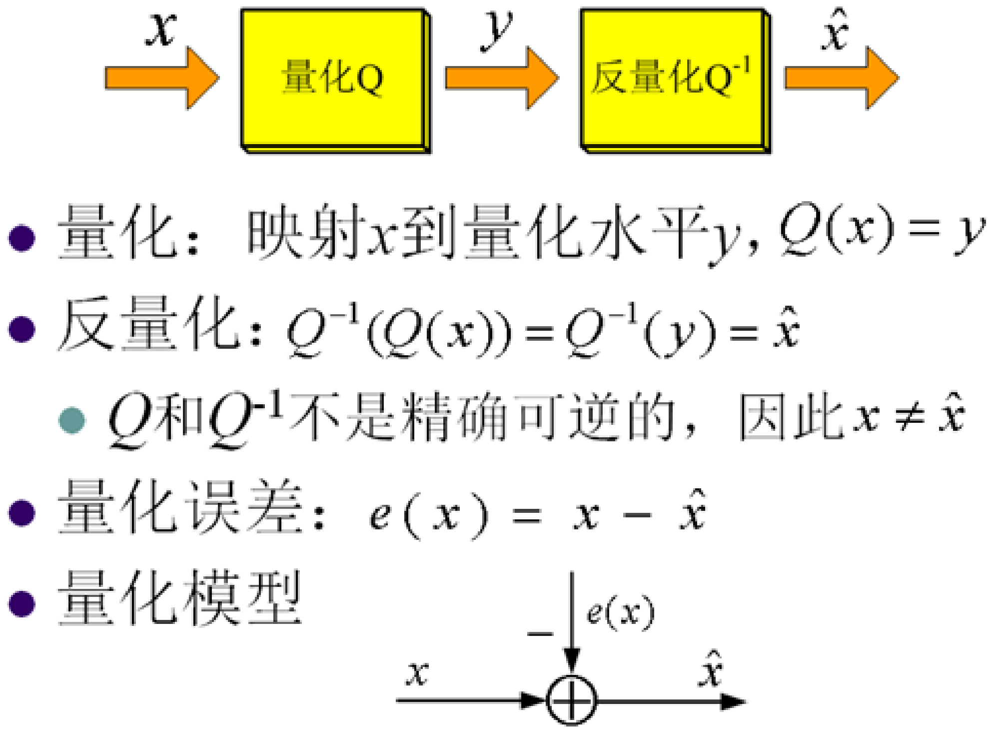
- 用更小的集合表示更大的集合的过程
    - 对信号源的有限近似
    - 有损过程
    - 应用
        - A/D转换
        - 压缩
    - 量化方法
        - 标量(Scalar)量化
        - 矢量(Vector)量化

2. 量化的基本思想

- 映射一个输入间隔到一个整数
- 减少信源编码的bit
- 一般情况重构值与输入值不同



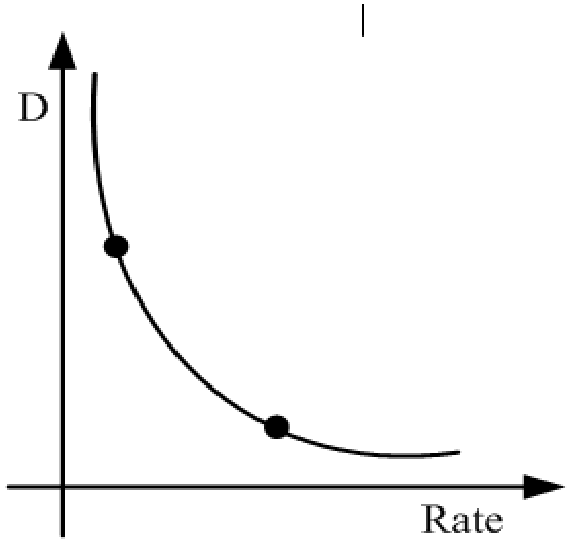
3. 量化模型



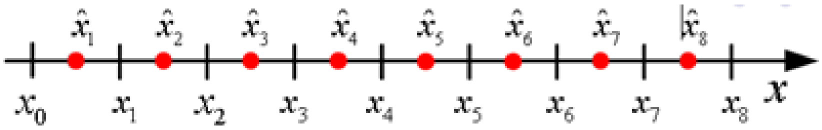
4. 量化的率失真优化

- 量化器设计问题
  - 量化水平的个数，即Bin的个数
  - 决策边界：Bin的边界
  - 重构水平
- 量化器设计是对率失真的优化
  - 为了减少码率的大小，需要减少Bin的个数
  - Bin的个数减少导致重构的误差增大，失真也就随着增大

5. 失真测量



- 量化的均方差



  - Bin的数目： M
  - 决策边界： x_i, $i=0, \ldots, M$
  - 重构水平： $\hat{x}_i$, $i=0, \ldots, M$
  - 如果 $x_{i-1} < x <= x_i$,重构水平为: $\hat{x}_i$
  - 量化误差:

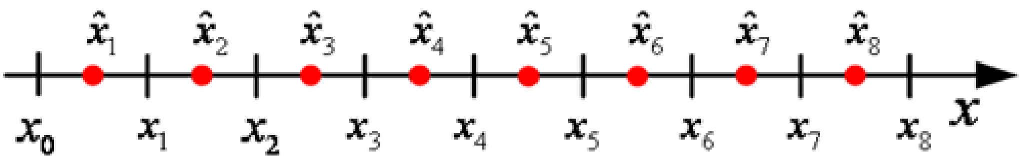
$$MSE = \int_{-\infty}^{\infty} e(x)^2 f_X(x)dx = \int_{-\infty}^{\infty} (x-\hat{x})^2 f_X(x)dx$$

$$= \int_{x_0}^{x_1} (x-\hat{x}_1)^2 f_X(x)dx + \int_{x_1}^{x_2} (x-\hat{x}_2)^2 f_X(x)dx + \cdots$$

$$= \sum_{i=1}^{M} \int_{x_{i-1}}^{x_i} (x-\hat{x}_i)^2 f_X(x)dx$$

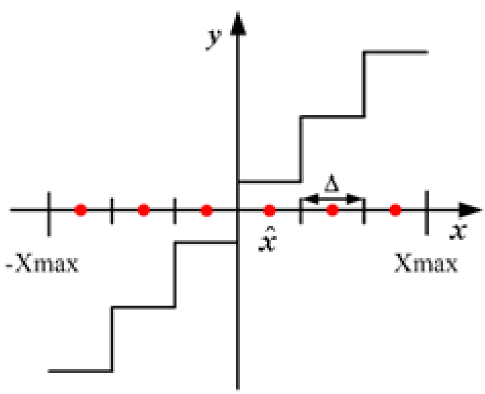
6. 量化器设计

- 量化器设计的两个方面
  - 给定量化水平数目M，找到决策边界x_i和重构水平 使MSE最小
  - 给定失真限制D，找到量化水平数目M，决策边界x_i和重构水平y_i使MSE<=D



7. **均匀量化（Uniform Quantization）**

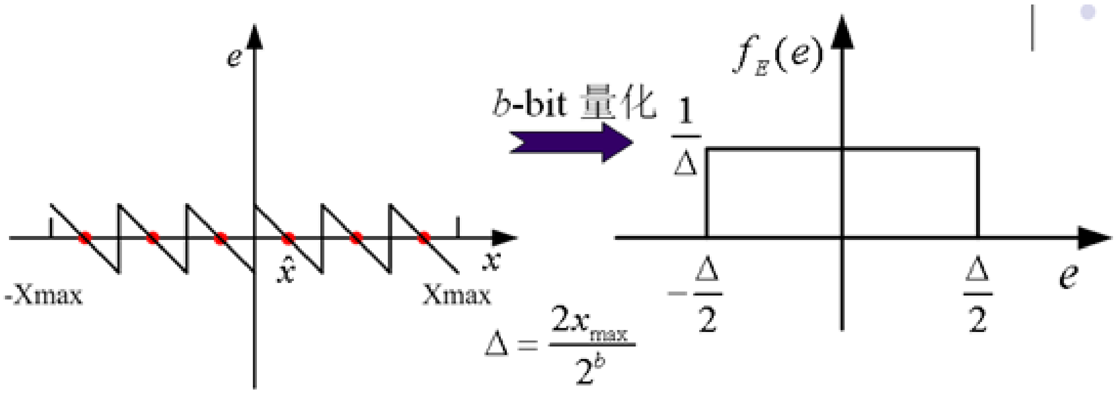
- 除了两端的间隔之外，所有的Bin有同样的量化步长Δ
  - x_i之间和$\hat{x}_i$之间间隔均匀
  - x_i之间和$\hat{x}_i$之间间隔都为固定值Δ



输入范围: $-x_{\max} \le x \le x_{\max}$

重构水平: $\hat{x}_i = \dfrac{x_i + x_{i-1}}{2}$

误差范围: $-\dfrac{\Delta}{2} \le e \le \dfrac{\Delta}{2}$



$$MSE=\int_{-\infty}^{\infty} e^2 f_E(e)de=\int_{-\Delta/2}^{\Delta/2} e^2\frac{1}{\Delta}de$$

$$=\frac{1}{3\Delta}e^3\Big|_{-\Delta/2}^{\Delta/2}=\frac{2}{3\Delta}\left(\frac{\Delta}{2}\right)^3=\frac{\Delta^2}{12}=\frac{1}{3}\frac{x_{\max}^2}{2^{2b}}$$

量化步长Δ减少一半，
量化水平数M增加一倍，
需要$(b+1)$-bit 量化

8. 量化与峰值信噪比

- 峰值信噪比：$PSNR_{dB}=10\log_{10}\frac{255^2}{MSE}$
- 量化噪声：

$$PSNR_{dB}=10\log_{10}\frac{12\times 255^2}{\Delta^2}=10\log_{10}12\times 255^2-20\log_{10}\Delta$$

- 量化水平编码比特增加1bit，量化水平数M增加一倍，量化步长减少一半$\Delta'=\frac{\Delta}{2}$

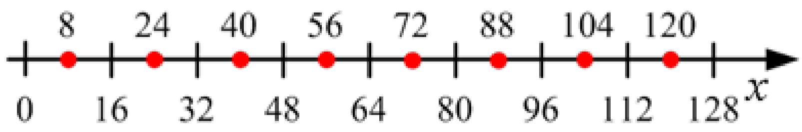
$$PSNR_{dB}(\Delta')=10\log_{10}\frac{12\times 255^2}{\Delta^2}=10\log_{10}12\times 255^2-20\log_{10}\Delta'$$

$$=10\log_{10}12\times 255^2-20\log_{10}\frac{\Delta}{2}$$

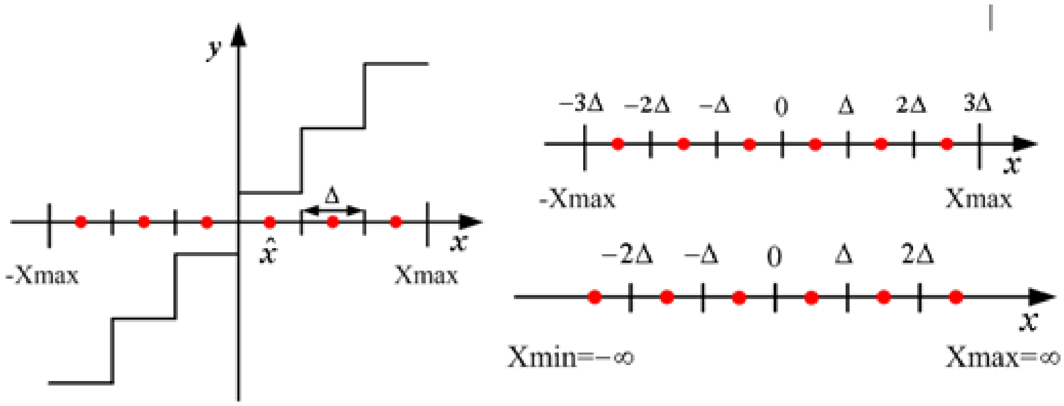
$$=10\log_{10}12\times 255^2-20\log_{10}\Delta+20\log_{10}2$$

$$\approx PSNR_{dB}(\Delta)+6$$

- 例：为取值范围在[0,128]的离散信号设计3-bit均匀量化器。
  - 最大可能量化水平数：$M = 2^3 = 8$
  - 量化步长：$\Delta = \frac{x_{\max} - x_{\min}}{M} = \frac{128}{8} = 16$
  - 量化水平：$y_i = \{0,1,2,3,4,5,6,7\}$
  - 重构水平：$\hat{x}_i = \{8,24,40,56,72,88,104,120\}$
  - 最大量化误差：$e_{max}=8$

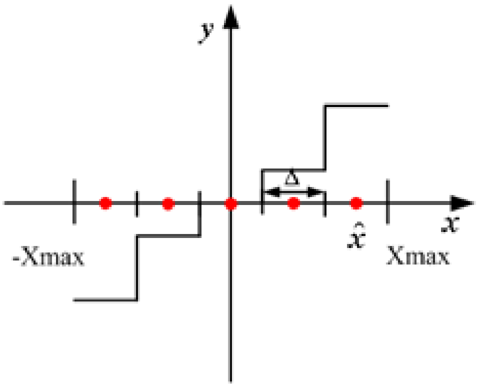


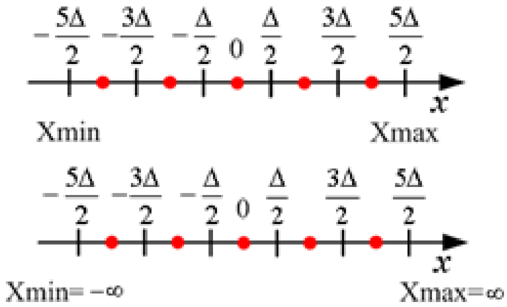
9. **中升量化器（Midrise Quantizer）**



- 偶数个量化水平，0不是重构水平值
- x_i之间和$\hat{x}_i$之间间隔均匀都为固定值$\Delta = \frac{2x_{\max}}{2^b}$
- 常用在A/D转换

10. **中平量化器（Midtread Quantizer）**

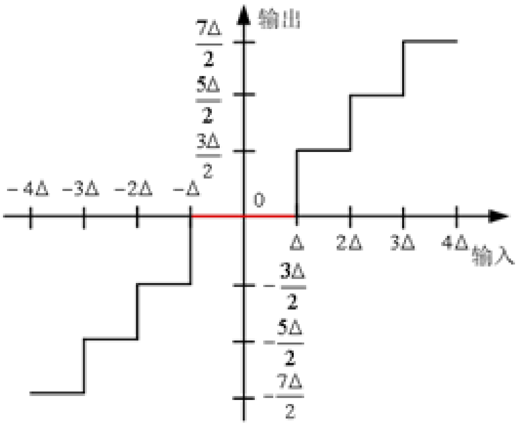




- 奇数个量化水平，0是重构水平值
- x_i之间和$\hat{x}_i$之间间隔均匀都为固定值$\Delta = \dfrac{2x_{max}}{2^b - 1}$
- 常用在视频压缩中

11. 死区量化器（Deadzone Quantizer）

- 0值所在区间的Bin的量化步长被加倍，$\Delta \Rightarrow 2\Delta$
- 其它的Bin的量化步长仍是均匀的，Δ
- 产生更多的重构值0



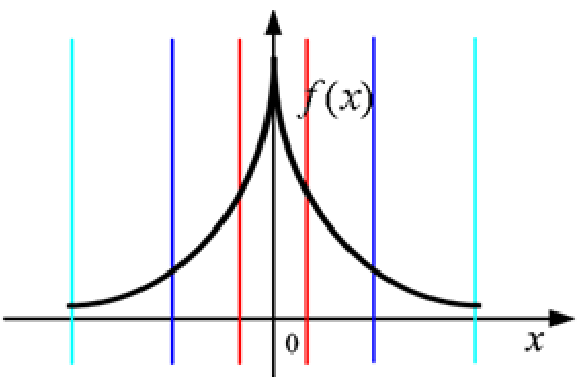
量化：$y = Q(x) = sign(x)\left\lfloor \dfrac{|x|}{\Delta} \right\rfloor$

反量化：

$$\hat{x} = Q^{-1}(y) = \begin{cases} 0, & y = 0 \\ sign(y)\left(|y| + \dfrac{1}{2}\right)\Delta, & y \neq 0 \end{cases}$$

12．非均匀量化（Non-uniform Quantization）

- 如果信源不是均匀分布的，采用均匀量化不是最优的
- 对于非均匀量化，为了减少MSE，当概率密度函数$f_X(x)$高时，使Bin的量化步长减小，当概率密度函数$f_X(x)$低时， 使Bin的量化步长增加。



$$MSE = \int_{-\infty}^{\infty} (x - \hat{x})^2 f_X(x)dx = \sum_{i=1}^{M} \int_{x_{i-1}}^{x_i} (x - \hat{x}_i)^2 f_X(x)dx$$

13. 最优的标量量化

- 给定解码器，优化编码器
- 已知$\hat{x}_i$，求x_i，使MSE最小，$MSE=\sum_{i=1}^{M}\int_{x_{i-1}}^{x_i}(x-\hat{x}_i)^2 f_X(x)dx$

求最小MSE，即对MSE求x_i导数

$$\left(\sum_{i=1}^{M}\int_{x_{i-1}}^{x_i}(x-\hat{x}_i)^2 f_X(x)dx\right)'$$

$$=\left(\int_{x_{i-1}}^{x_i}(x-\hat{x}_i)^2 f_X(x)dx+\int_{x_i}^{x_{i+1}}(x-\hat{x}_{i+1})^2 f_X(x)dx\right)'$$

$$=(x_i-\hat{x}_i)^2 f_X(x)-(x_i-\hat{x}_{i+1})^2 f_X(x)=0$$

则有 $(x_i-\hat{x}_{i-1})-(x_i-\hat{x}_i)=0$ 近邻法则

$$x_i=\frac{\hat{x}_i+\hat{x}_{i+1}}{2}$$

（Nearest Neighbor Rule)

- 给定编码器，优化解码器
- 已知x_i，求$\hat{x}_i$，使MSE最小，$MSE=\sum_{i=1}^{M}\int_{x_{i-1}}^{x_i}(x-\hat{x}_i)^2 f_X(x)dx$

求最小MSE，即对MSE求$\hat{x}_i$导数

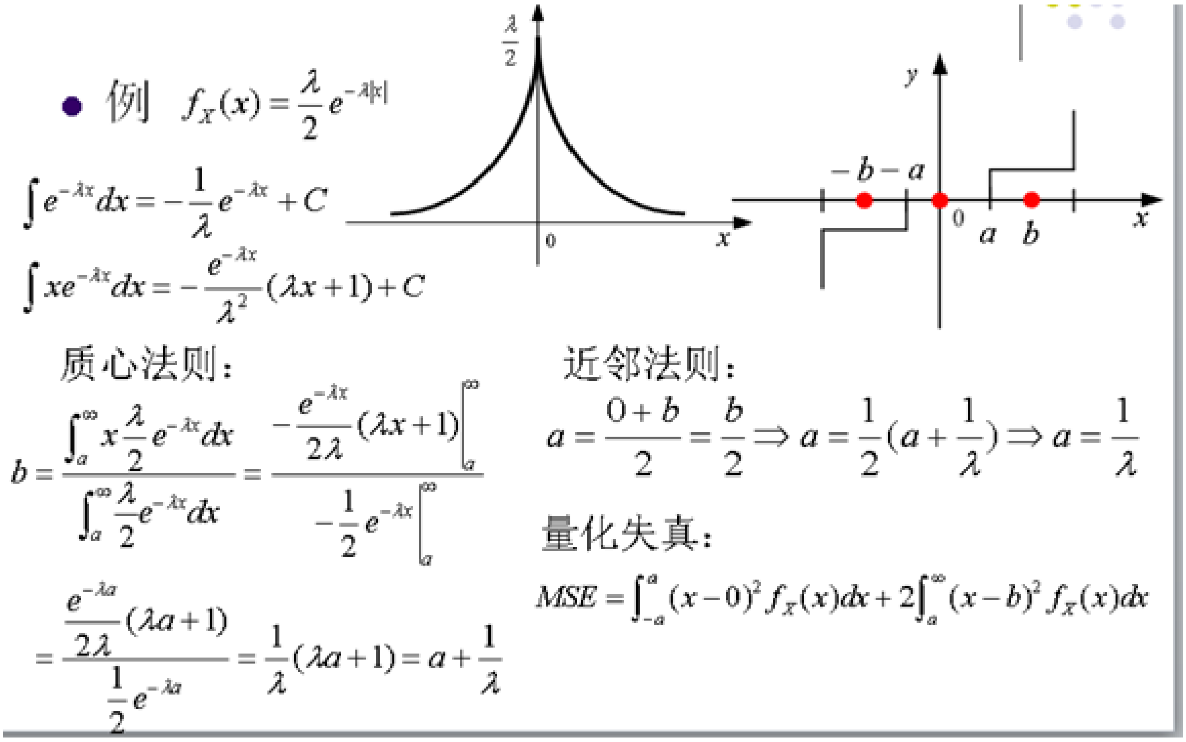
$$\left(\sum_{i=1}^{M}\int_{x_{i-1}}^{x_i}(x-\hat{x}_i)^2 f_X(x)dx\right)'_{\hat{x}_i}=\left(\int_{x_i}^{x_{i+1}}(x-\hat{x}_i)^2 f_X(x)dx\right)'_{\hat{x}_i}$$

$$=\int_{x_i}^{x_{i+1}}-2(x-\hat{x}_i)f_X(x)dx=0$$

则有 $\int_{x_i}^{x_{i+1}}xf_X(x)dx-\int_{x_i}^{x_{i+1}}\hat{x}_i f_X(x)dx=0$ 质心法则

（Centroid Rules）

$$\hat{x}_i=\frac{\int_{x_i}^{x_{i+1}}xf_X(x)dx}{\int_{x_i}^{x_{i+1}}f_X(x)dx}$$



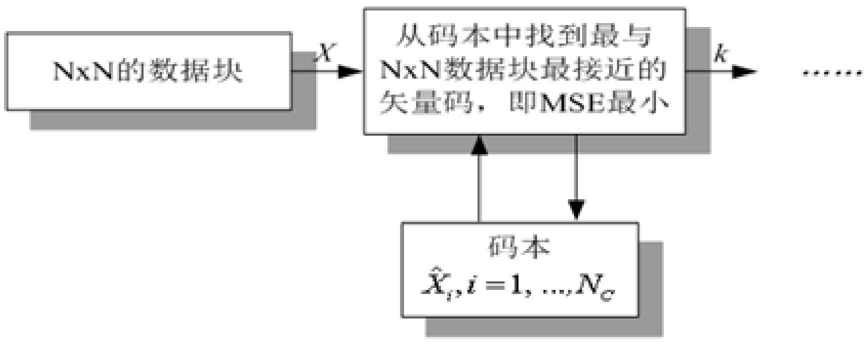
14. 量化编码

- 定长编码量化水平
  - 使用等长的码字编码每个量化水平，码字长为：$\lceil \log_2 M \rceil$
- 熵编码量化水平
  - 根据量化水平的概率分布情况，用变长的码字编码每个量化水平
  - 平均码字长 $\le \lceil \log_2 M \rceil$
  - 比定长编码量化水平效率高
  - 广泛应用在图像和视频编码中

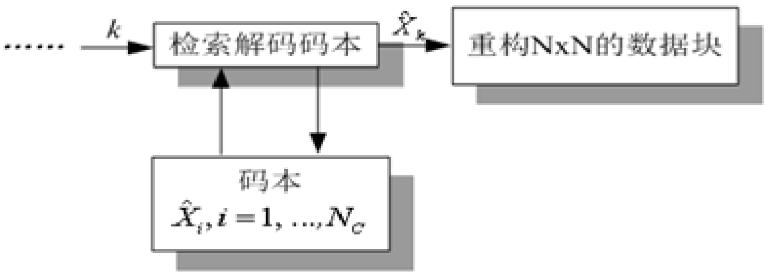
15. 矢量量化

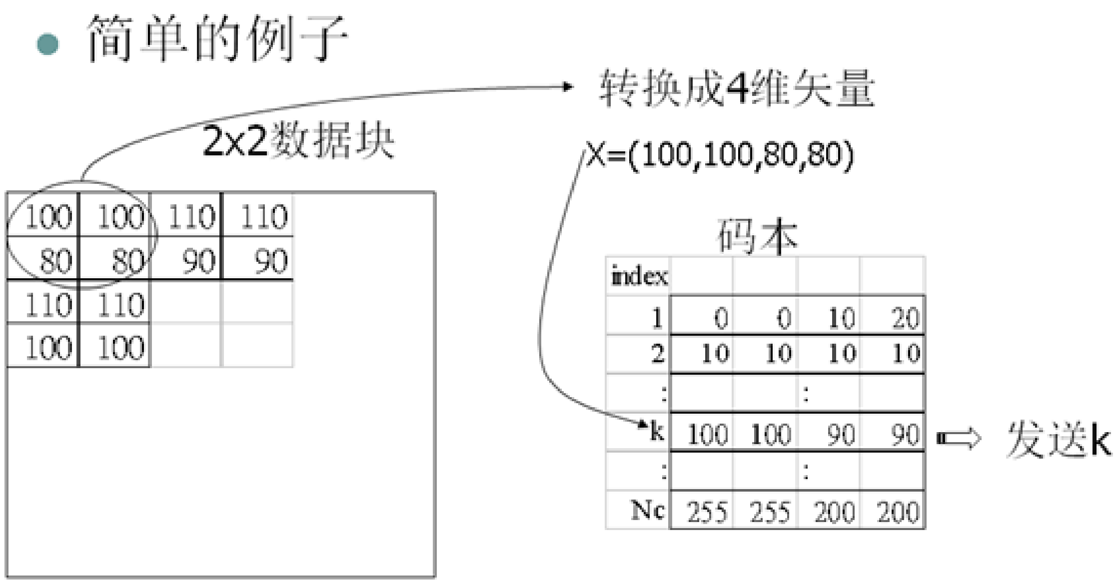
- 标量量化：对数据一个一个的进行量化，称为标量量化。
- 矢量量化：将数据分组，每组K个数据构成K维矢量，再以矢量为处理单元进行量化。
  - 矢量量化是标量量化的多维扩展
  - 标量量化是矢量量化的特殊情况
- 矢量量化工作过程

编码端



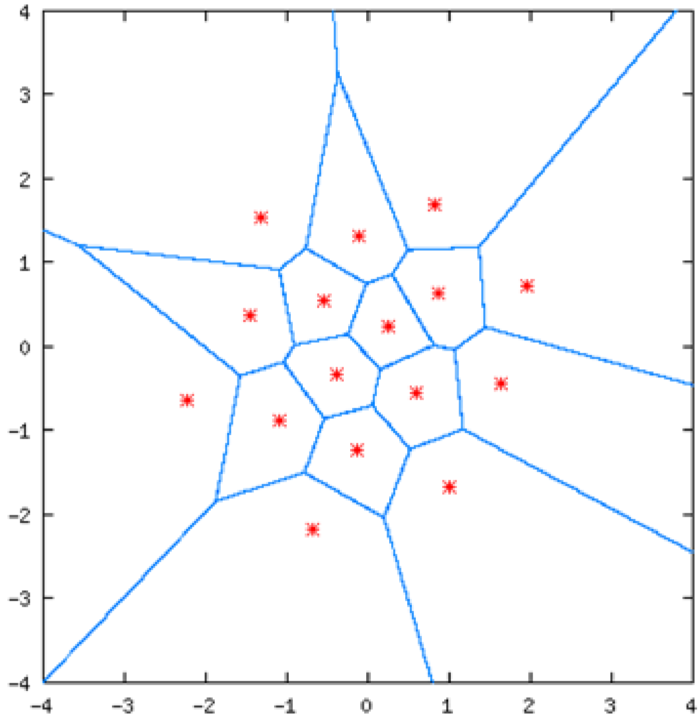
解码端





- 二维矢量量化

- 矢量量化优点
  - 只传码字的下标，编码效率高
  - 在相同码率下，比标量量化失真小
  - 在相同失真下，比标量量化码率低
- 矢量量化缺点：复杂度随着维数的增加呈指数增加



第8章 熵编码

1. 熵编码

- 熵（Entropy）：信源的平均信息量，更精确的描述为表示信源所有符号包含信息的平均比特数
  - 信源编码要尽可能的减少信源的冗余，使之接近熵
  - 用更少的比特传递更多的信源信息
- 熵编码：数据压缩中根据信源消息的概率模型使消息的熵最小化
  - 无损压缩
  - 变长编码

2. 熵

- 信息量：

$$I(x_i)=\log_2\frac{1}{p(x_i)}=-\log_2 p(x_i)$$

单位：比特

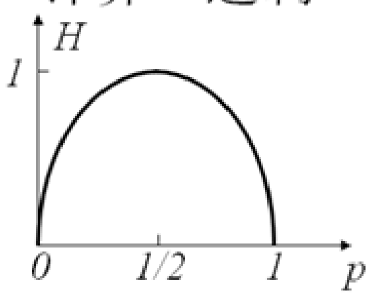
- 熵：

$$H(X)=E[-\log_2 p(x_i)]=-\sum_{i=0}^{N-1}p(x_i)\log_2 p(x_i)$$

单位：比特/符号

- 例：二进制编码中，符号“1”的发生概率是*p*，符号“0”的发生概率是1-*p*，计算二进制编码的熵。

$$H(X)=-\sum_{i=0}^{1}p(x_i)\log_2 p(x_i)$$
$$=-p(0)\log_2 p(0)-p(1)\log_2 p(1)$$
$$=-(1-p)\log_2(1-p)-p\log_2 p$$



$$p=\frac{1}{2}\Rightarrow H=-\frac{1}{2}\log_2\frac{1}{2}-\frac{1}{2}\log_2\frac{1}{2}=\frac{1}{2}+\frac{1}{2}=1\text{比特/符号}$$

$$p=\frac{1}{4}\Rightarrow H=-\frac{3}{4}\log_2\frac{3}{4}-\frac{1}{4}\log_2\frac{1}{4}=0.8113\text{比特/符号}$$

$$p=1\Rightarrow H=-0\log_2 0-\log_2 1=0\text{比特/符号}$$

- 例：求四个符号的信源$X=\{A,B,C,D\}$的熵，

  1)四个符号发生概率相同，都为1/4

$$H=\left(-\frac{1}{4}\log_2\frac{1}{4}\right)\times 4=\log_2 4=2\ \text{比特/符号}$$

  2) $p(A)=1/2$, $p(B)=1/4$, $p(C)=1/8$, $p(D)=1/8$

$$H=-\frac{1}{2}\log_2\frac{1}{2}-\frac{1}{4}\log_2\frac{1}{4}-\frac{1}{8}\log_2\frac{1}{8}-\frac{1}{8}\log_2\frac{1}{8}$$
$$=\frac{1}{2}+\frac{1}{2}+\frac{3}{8}+\frac{3}{8}=\frac{7}{4}\text{比特/符号}<2\ \text{比特/符号}$$

3. 定长编码

A → 00

B → 01

C → 10

D → 11

- 信源$X=\{A,B,C,D\}$

  1)等概率

$$H=\left(-\frac{1}{4}\log_2\frac{1}{4}\right)\times 4=\log_2 4=2\ \text{比特/符号}$$

  2) $p(A)=1/2$, $p(B)=1/4$, $p(C)=1/8$, $p(D)=1/8$

$$H=-\frac{1}{2}\log_2\frac{1}{2}-\frac{1}{4}\log_2\frac{1}{4}-\frac{1}{8}\log_2\frac{1}{8}-\frac{1}{8}\log_2\frac{1}{8}$$
$$=\frac{1}{2}+\frac{1}{2}+\frac{3}{8}+\frac{3}{8}=\frac{7}{4}\text{比特/符号}<2\ \text{比特/符号}$$

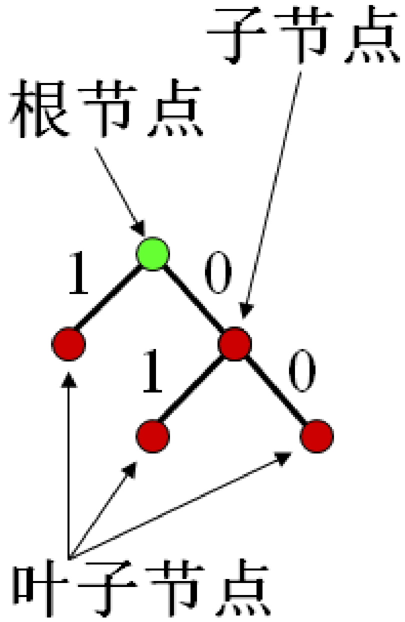
  定长编码存在编码冗余

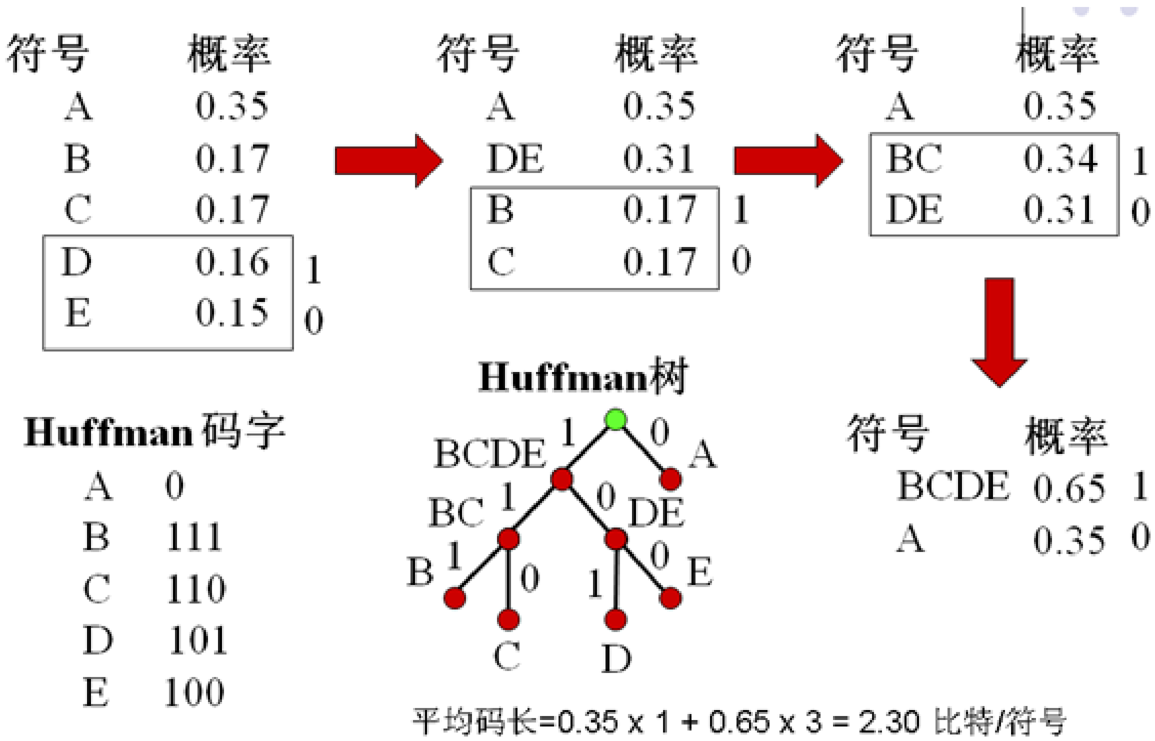
4. 变长编码

- 变长编码：用不同的比特数表示每一个符号
  - 为频繁发生的符号分配短码字
  - 为很少发生的符号分配长码字
  - 比定长编码有更高的效率
- 常用的变长编码
  - Huffman编码
  - 算术编码

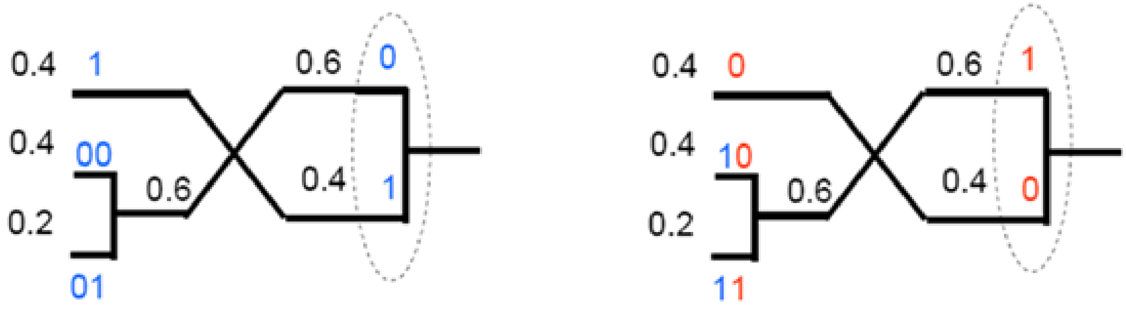
5. Huffman编码

- 前缀码：任何码字不是其它码字的前缀
    - 如果011为一个有效码字，则0，1，01，11必不是有效码字
    - 不会引起解码歧义
- Huffman：
    - 二叉树
    - 树节点:表示符号或符号组合
    - 分支:两个分支一个表示"0"，另一个表示"1"

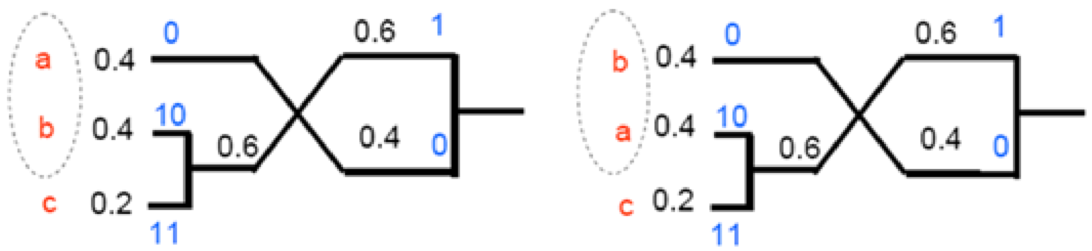




- Huffman的不唯一性
  - 每次分支有两种选择：0，1

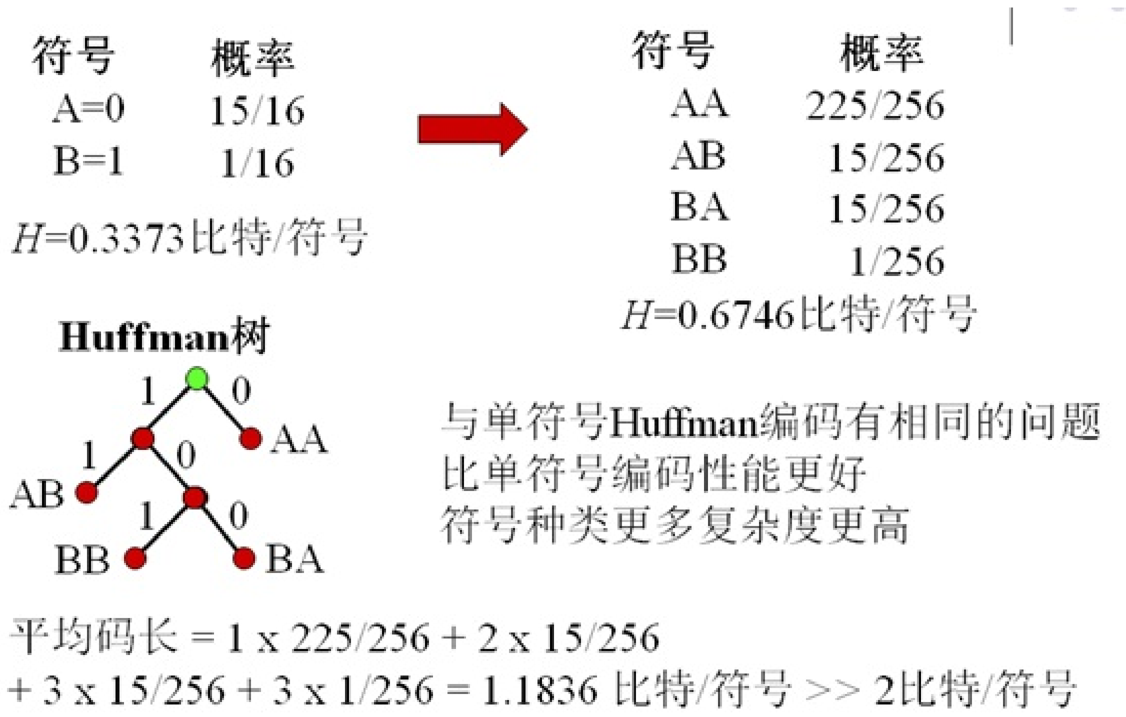


  - 相同的概率产生不同的组合



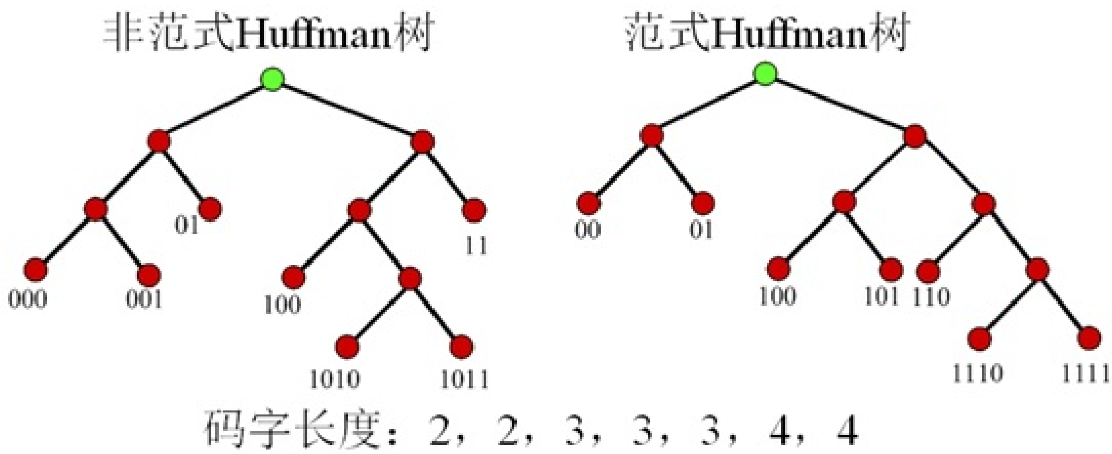
- 缺点：
  - 数据的概率变化难于实时统计
  - Huffman树需要编码传输给解码器
  - 只有在$p(x_i)=1/2^{ki}$时是最优编码
  - 最小码字长度为1比特/符号
- 如果有二值信源，其两个符号的概率相差很大
  - 例如：$p(1)=0.0625$，$p(0)=0.9375$则$H=0.3373$比特/符号，Huffman编码平均码长=1比特/符号
  - 两个符号联合编码有更高效率

6. 扩展Huffman编码

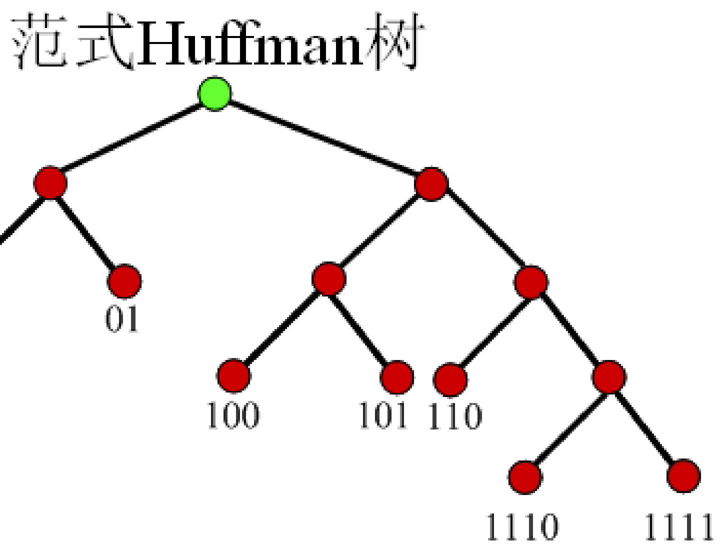


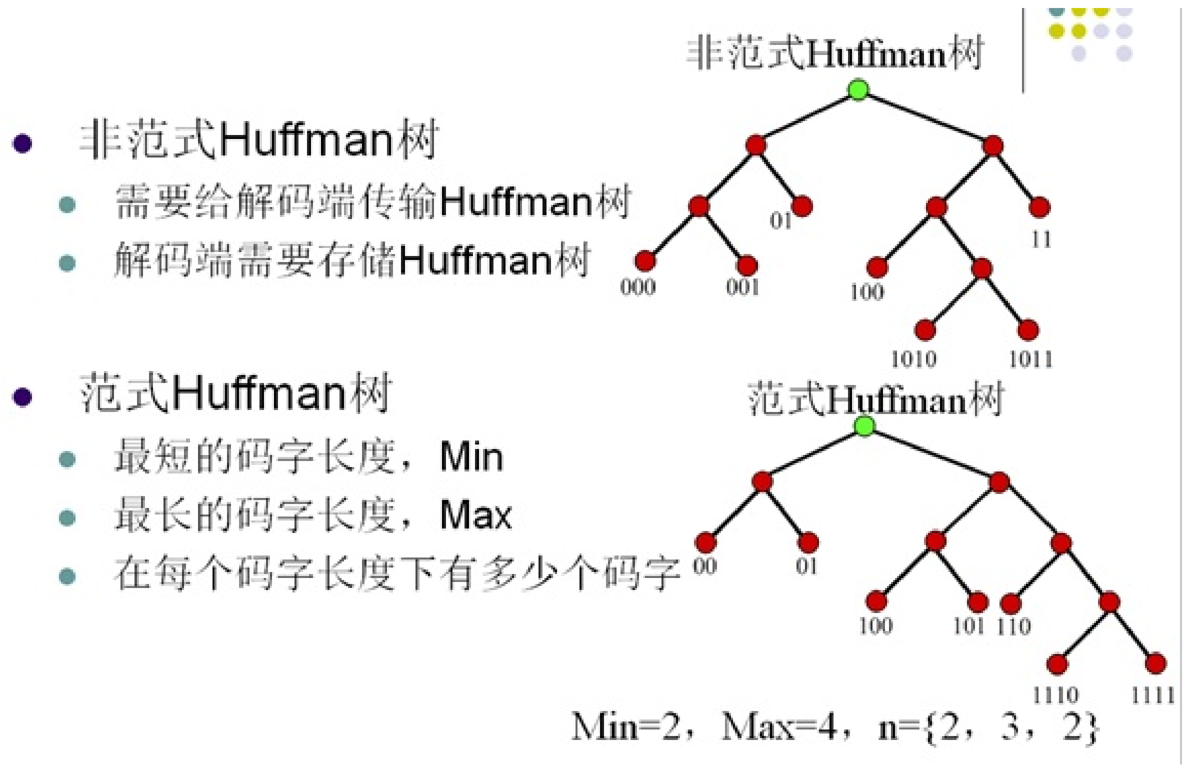
7. 范式Huffman编码

- 范式Huffman树的建立规则
    - 节点左支设为0，右支设为1
    - 树的深度从左至右增加
    - 每个符号被放在最先满足的叶子节点



- 特性
  - 第一个码字是一串0
  - 相同长度的码字的值是连续的
  - 如果所有的码字通过在低位补0的方式，使所有码字的长度相同则有0000<0100<1000<1010<1100<1110<1111
  - 从码字长度n到n+1有如下关系
    - C(n+1,1)=(C(n,last)+1)<<1
  - 从码字长度n到n+2有如下关系
    - C(n+2,1)=(C(n,last)+1)<<2





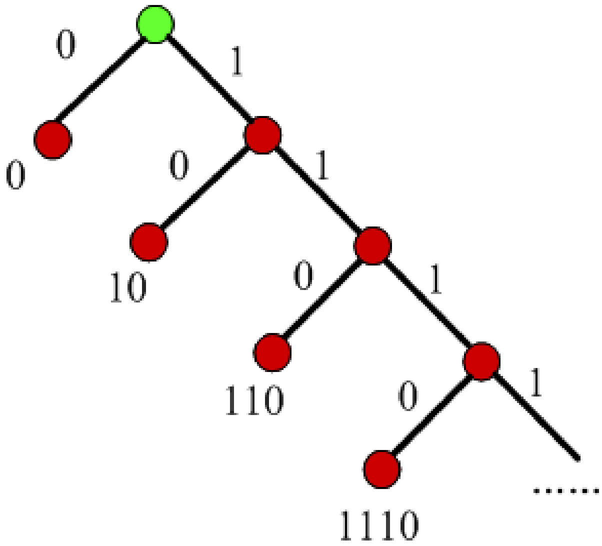
8. 一元码

- 编码一个非负整数n为n个1和一个0

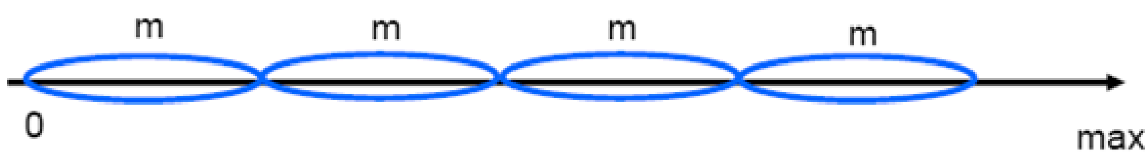
- 不需要存储码表
- 可以用Huffman树表示
- 码长增长太快：n=100，码长101

9. 哥伦布编码

- 将信源符号等分成几组，每组有相应的编号
- 编号小的分配码字短，编号大分配码字长
- 同组的符号有等长的码字，比一元码的码字长度增长慢



n	码字
0	0
1	10
2	110
3	1110
4	11110
5	111110
6	1111110
	....



- 码字分配

- 码字解析 $n = qm + r = \left\lfloor \frac{n}{m} \right\rfloor m + r$

q	码字
0	0
1	10
2	110
3	1110
4	11110
5	111110
6	1111110
	………

q：组编号

r：组内符号，定长编码

m：组内符号数

当$m=2^k$时采用定长编码

当$m\neq 2^k$时可以采用定长编码也可以采用变长编码

r值小的采用$\lfloor \log_2 m \rfloor$

r值大的采用$\lceil \log_2 m \rceil$

m=5，00，01，10，110，111

- 哥伦布编码 m=5

n	q	r	code
0	0	0	000
1	0	1	001
2	0	2	010
3	0	3	0110
4	0	4	0111

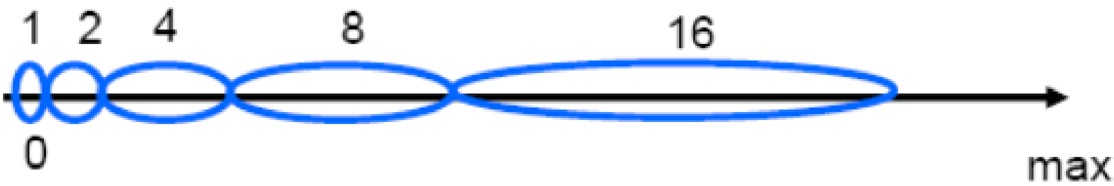
n	q	r	code
5	1	0	1000
6	1	1	1001
7	1	2	1010
8	1	3	10110
9	1	4	10111

n	q	r	code
10	2	0	11000
11	2	1	11001
12	2	2	11010
13	2	3	110110
14	2	4	110111

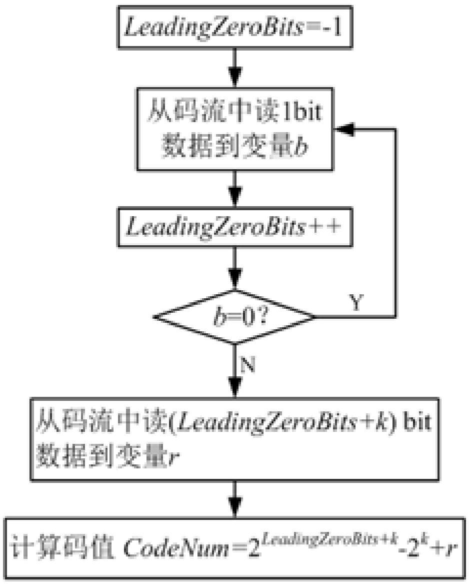
- 哥伦布码是范式Huffman编码

10. 指数哥伦布编码

- 哥伦布码对信源符号的分组大小相同
- 指数哥伦布码对信源符号的分组大小按照指数增长
- 指数哥伦布码依然是一元码加定长码的形式
- 指数哥伦布码的指数*k*=0，1，2，…



*K*阶指数哥伦布码解码



n	码字	组号
0	1	0
1	010	1
2	011	
3	00100	2
4	00101	
5	00110	
6	00111	
7	0001000	3
8	0001001	
9	0001010	
10	0001011	
11	0001100	
12	0001101	
13	0001110	
14	0001111	
15	000010000	4

阶数	码字结构	CodeNum取值范围
k = 0	1	0
	0 1 x_0	1～2
	0 0 1 x_1 x_0	3～6
	0 0 0 1 x_2 x_1 x_0	7～14
	......	......
k = 1	1 x_0	0～1
	0 1 x_1 x_0	2～5
	0 0 1 x_2 x_1 x_0	6～13
	0 0 0 1 x_3 x_2 x_1 x_0	14～29
	......	......
k = 2	1 x_1 x_0	0～3
	0 1 x_2 x_1 x_0	4～11
	0 0 1 x_3 x_2 x_1 x_0	12～27
	0 0 0 1 x_4 x_3 x_2 x_1 x_0	28～59
	......	......
k = 3	1 x_2 x_1 x_0	0～7
	0 1 x_3 x_2 x_1 x_0	8～23
	0 0 1 x_4 x_3 x_2 x_1 x_0	24～55
	0 0 0 1 x_5 x_4 x_3 x_2 x_1 x_0	56～119
	......	......

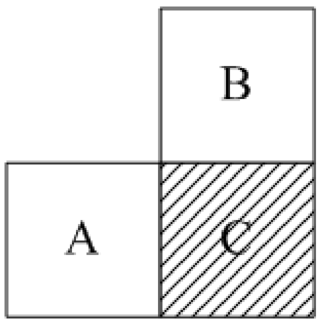
11. **CAVLC （ Context-Based Adaptive Variable Length Code）**

- 当前块的系数分布和其邻块的系数分布情况相关
  - N_X为块X的非零系数个数，当前块C的第一个系数的编码码表由N_C决定， N_C=(N_A+ N_B)/2
- 当前待编码系数和前面编码系数有相关性
  - 当前块C的其它系数的编码码表由前一个系数的幅值决定cof_{N-1}=>GolombTab_x，用GolombTab_x编码cof_N

$$Nc = \begin{cases} 0, & GolombTab0 \\ 1, & GolombTab1 \\ \vdots & \vdots \end{cases}$$

12. 算术编码

- 信息量=>符号编码比特数
- Huffman编码为每个符号分配一个码字，这说明Huffman编码的压缩上限是1比特/符号
- 算术编码若干个符号可编码成1bit
- 算术编码是把信源表示为实数轴上[0,1]区间，信源中每个符号都用来缩短这个区间
  - 输出[0,1]区间的一个实数表示一串编码符号
  - 比Huffman编码更有效



- 编码思想
  - 编码器用熵编码算法编码一串符号产生一个[0,1]区间的实数，将实数的一个二进制表示传给解码器
  - 解码器用熵解码算法解码得到一串符号

$$\log_2\frac{1}{2}=1bit\ \ \log_2\frac{1}{4}=2bit\ \ \log_2\frac{1}{8}=3bit$$

- 小数的二进制表示

$$0.75_{10}=0.5+0.25=2^{-1}+2^{-2}=0.11_2$$
$$0.384765625_{10}=2^{-2}+2^{-3}+2^{-7}+2^{-9}=0.011000101_2$$

- 信源符号概率分布

Symbol	Probability	Huffman Code
X1	0.05	10101
X2	0.2	01
X3	0.1	100
X4	0.05	10100
X5	0.3	11
X6	0.2	00
X7	0.1	1011

- 字符串：X2 X2 X3 X3 X6 X5 X7
- Huffman编码，01 01 100 100 00 11 1011，18bit

- 字符串：X2 X2 X3 X3 X6 X5 X7
- 算术编码过程

1）确定区间：

range = high – low

new_high = low + range x subinterval_high

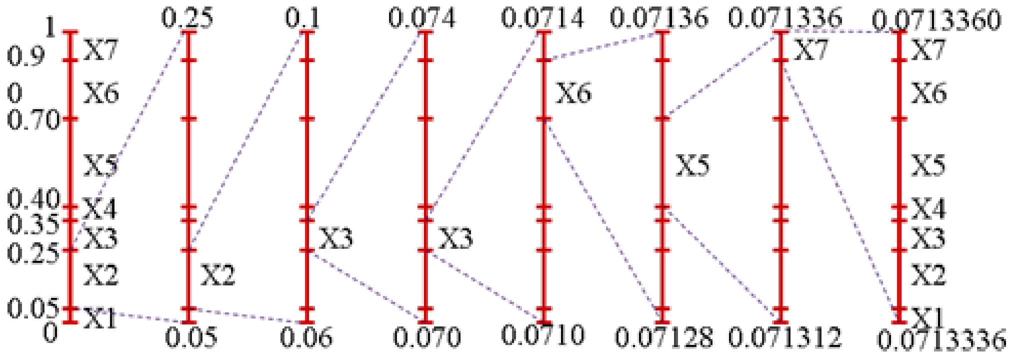
new_low = low+range x subinterval_low

编码结束后区间：[0.0713336,0.0713360)

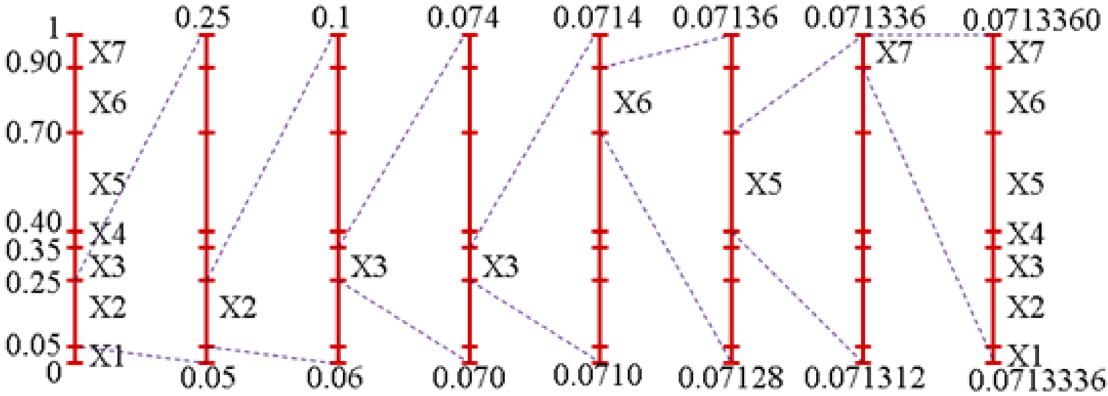
2）发送最终区间中的任意一值：0.07133483886719

$$2^{-4}+2^{-7}+2^{-10}+2^{-15}+2^{-16}=0.0001001001000011_2$$

Symbol	Probability	Interval
X1	0.05	[0,0.05)
X2	0.2	[0,05.0.25)
X3	0.1	[0,25.0.35)
X4	0.05	[0,35.0.4)
X5	0.3	[0,4.0.7)
X6	0.2	[0,7.0.9)
X7	0.1	[0,9.1)



- 算术解码



$$low=0; high=1; range=0$$

$$0.05 \le \frac{0.071334...-0}{1} \le 0.25 \Rightarrow X_2$$

$$high=0+1\times 0.25=0.25$$

$$low=0+1\times 0.05=0.05$$

$$range=high-low=0.2$$

$$0.05 \le \frac{0.071334...-0.05}{0.20}=0.1067 \le 0.25 \Rightarrow X_2$$

$$high=0.05+0.2\times 0.25=0.1$$

$$low=0.05+0.2\times 0.05=0.06$$

$$range=high-low=0.04$$

$$0.25 \le \frac{0.071334...-0.06}{0.04}=0.2834 \le 0.35 \Rightarrow X_3...$$

- 码字的二进制表示精度
  - 用8位精度实现概率区间划分
  - 得到的子区间也应为8位精度
  - 最后码字也是一个8位精度

符号	概率（使用分数表示）	减到8位精度的间隔（用分数表示）	减到8位精度的间隔（用二进制表示）	二进制范围
A	1/3	[0, 85/256)	[0.00000000, 0.01010101)	00000000 - 01010100
B	1/3	[85/256, 171/256)	[0.01010101, 0.10101011)	01010101 - 10101010
C	1/3	[171/256, 1)	[0.10101011, 1.00000000)	10101011 - 11111111

- 算术编码更接近熵
- 有限精度算术编码是次优（Near-optimal）编码，发送整数比特给解码端
- 算术编码到最后一个字符编码结束才输出码字
- 编码复杂度也比较高

13. 二值算术编码

- 算术编码计算复杂度高
  - 需要一些列乘法和判断操作

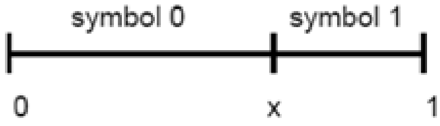
$$0.05 \le \frac{0.071334\ldots - 0.05}{0.20} = 0.1067 \le 0.25 \Rightarrow X_2$$

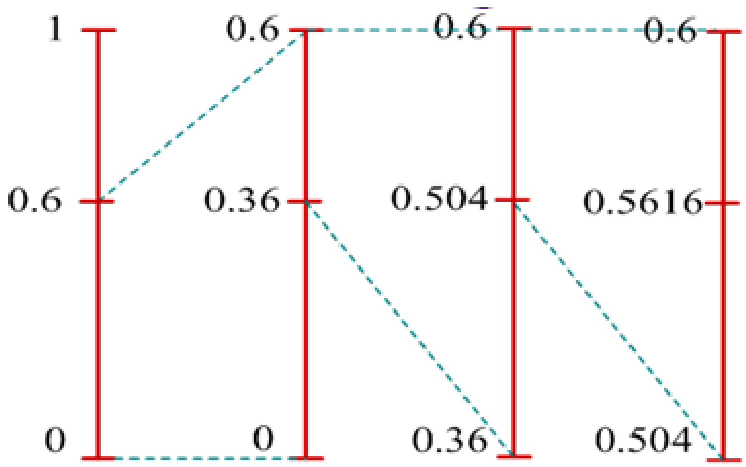
$$high = 0.05 + 0.2 \times 0.25 = 0.1$$

$$low = 0.05 + 0.2 \times 0.05 = 0.06$$

$$range = high - low = 0.04$$

- 如果只用两个符号0和1构成信源，复杂度将大大减小
  - 只有两个间隔[0,x)，[x,1)。





- $p(0)$=0.6，字符串：0110
- 0：[0,0.6)，1：[0.6,1)

  1)low=0, high=0.6

  2)low=0.36, high=0.6

  3)low=0.504, high=0.6

  4)low=0.504, high=0.5616
- 只需要更新low和high

二值算术解码

只需要一个条件检查

```
if (value < low+range x subinterval_low)
{
    n=1;
}
else
{
    n=0;
}
```

一般算术解码

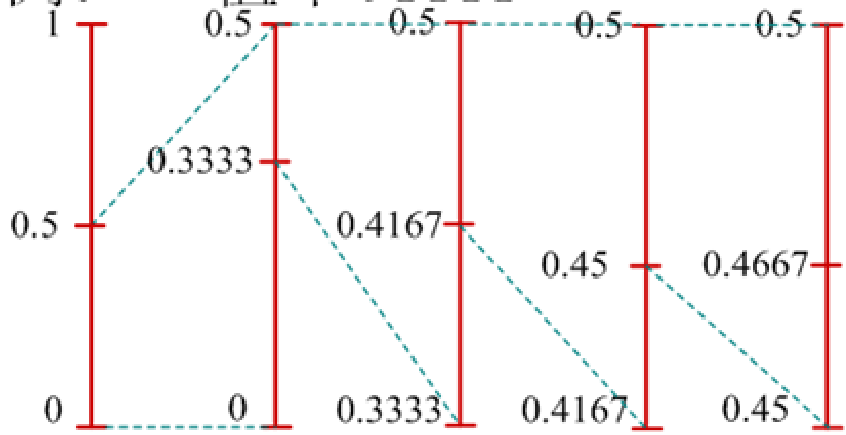
循环检查

```
while (value < low+range x subinterval_low)
{
    n++;
}
```

13. 自适应二值算术编码

- 由于信源0和1出现的概率是在不断变化的，因此0和1的概率区间也应该不断改变
- 自适应二值算术编码每编码一个0或1都重新统计0和1的概率并重新划分[0,1)区间
- 编解码端的概率统计模型一致，能够得到同样的[0,1)区间划分

- 例：二值串01111



1）初始化0和1数目计数器

C(0)=1, C(1)=1 => p(0)=1/2, p(1)=1/2

2)编码0, C(0)=2, C(1)=1 => p(0)=2/3, p(1)=1/3

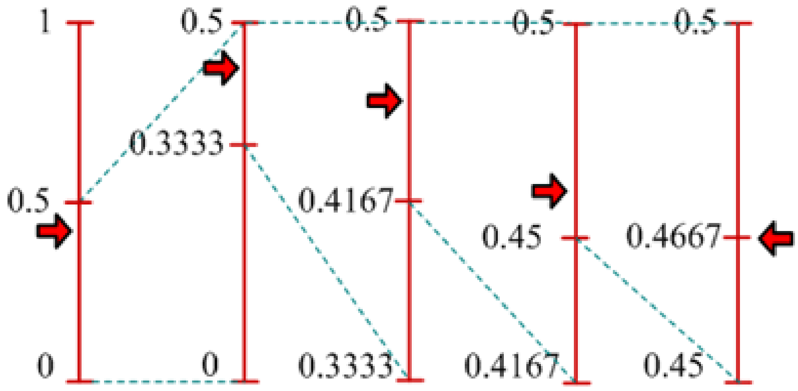
3)编码01, C(0)=2, C(1)=2 => p(0)=1/2, p(1)=1/2

4)编码011, C(0)=2, C(1)=3 => p(0)=2/5, p(1)=3/5

5)编码0111, C(0)=2, C(1)=4 => p(0)=1/3, p(1)=2/3

6)编码01111,计算得到0.4667并传给解码端

- 例：解码0.4667



1）初始化0和1数目计数器

C(0)=1, C(1)=1 => p(0)=1/2, p(1)=1/2

2)解码0, C(0)=2, C(1)=1 => p(0)=2/3, p(1)=1/3

3)解码01, C(0)=2, C(1)=2 => p(0)=1/2, p(1)=1/2

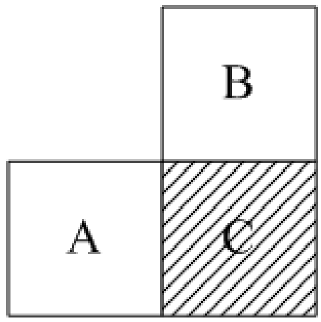
4)解码011, C(0)=2, C(1)=3 => p(0)=2/5, p(1)=3/5

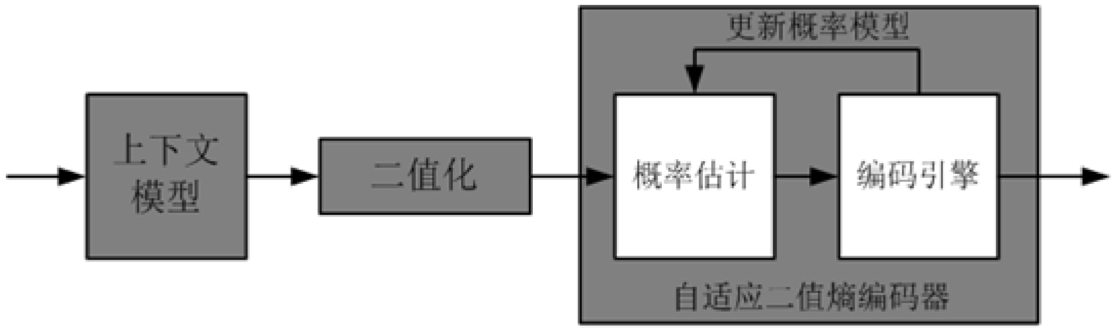
5)解码0111, C(0)=2, C(1)=4 => p(0)=1/3, p(1)=2/3

6)解码得到01111

14. **CABAC（Context-Based Adaptive Binary Arithmetic Coding）**

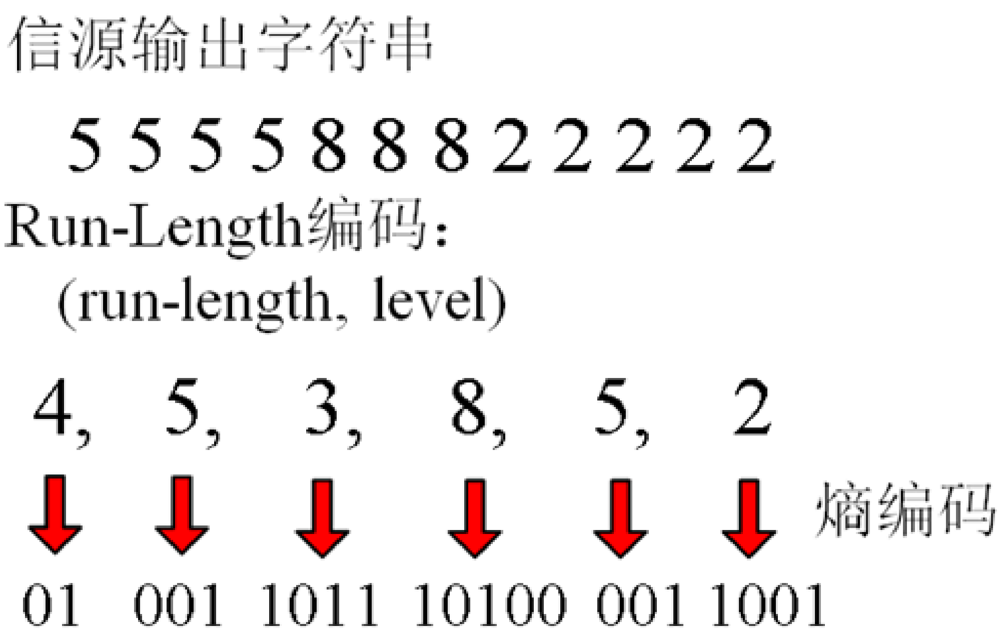
- 当前块的语法元素概率分布和其邻块的语法元素概率分布情况相关
  - 当前块C的邻块A和B的语法元素S_A与S_B可以为编码C块的语法元素S_C选择概率模型
- 二值化
  - 将语法元素值转换成二进制值串
- 概率模型更新
  - 根据已经编码的比特，重新估计二进制值串的概率并更新概率模型，用新的概率模型编码下一个比特

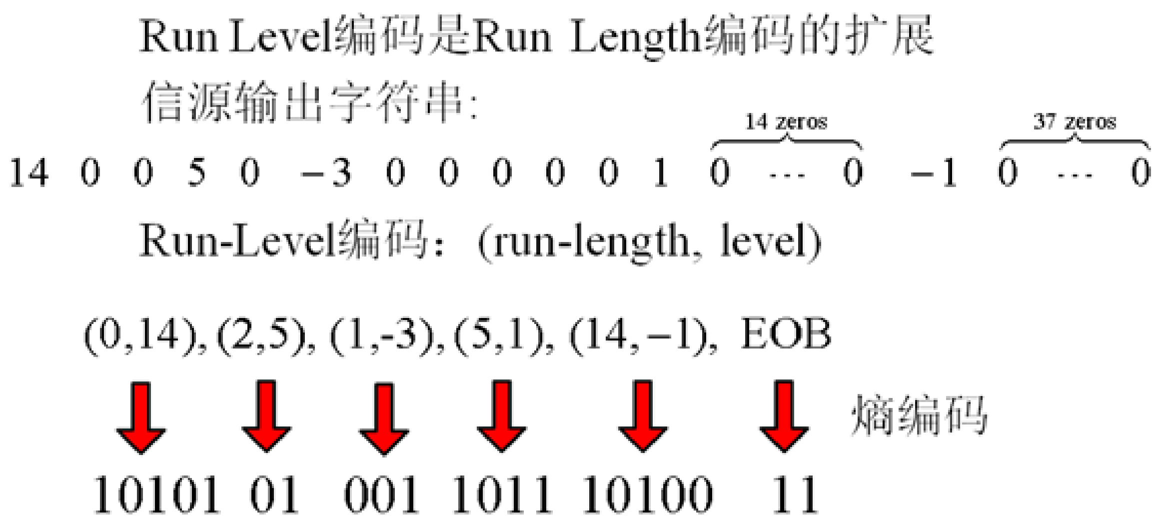




15. Run Length 编码

- 利用信源字符的重复性来编码的技术
- 对有很长，很多重复字符的信源编码非常有效
- 重复字符称为run，重复的字符个数称为run length
- Run-length编码能够和其它熵编码一起来压缩数据



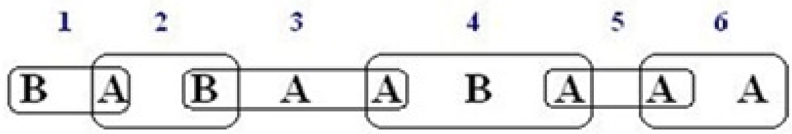


16. 字典编码

- 字典编码：根据信源符号（消息）的组合特点，建立包含各种符号组合的字典，编码这些符号组合的索引
    - LZ78=>Winzip
    - LZW=> GIF
- 适合一般意义上的数据压缩，去除数据的统计冗余

17. LZW

- 将信源输出的字符串中，每个第一次出现的字符或者字符串用索引来表示，并将字符或字符串对应的索引编码到码流中
- 解码端根据从码流中解码的字符，在线的建立和编码器完全一样的字典，并恢复出信源输出的字符串
- 适用于字符串中有大量的子字符串多次重复出现，重复次数越多，压缩效果越好
- **单个符号被分配为0-255之间的值**
- **初始码表，使其包含值为0-255的256个符号，值大于255的符号为空**
- **编码器将根据编码的符号情况确定字符组合为从 256 到 4095 之间的值**
- **编码时，编码器识别新的字符组合，并将他们增加到码表中**
- **编码器用码表中的符号组合所对应的值编码**

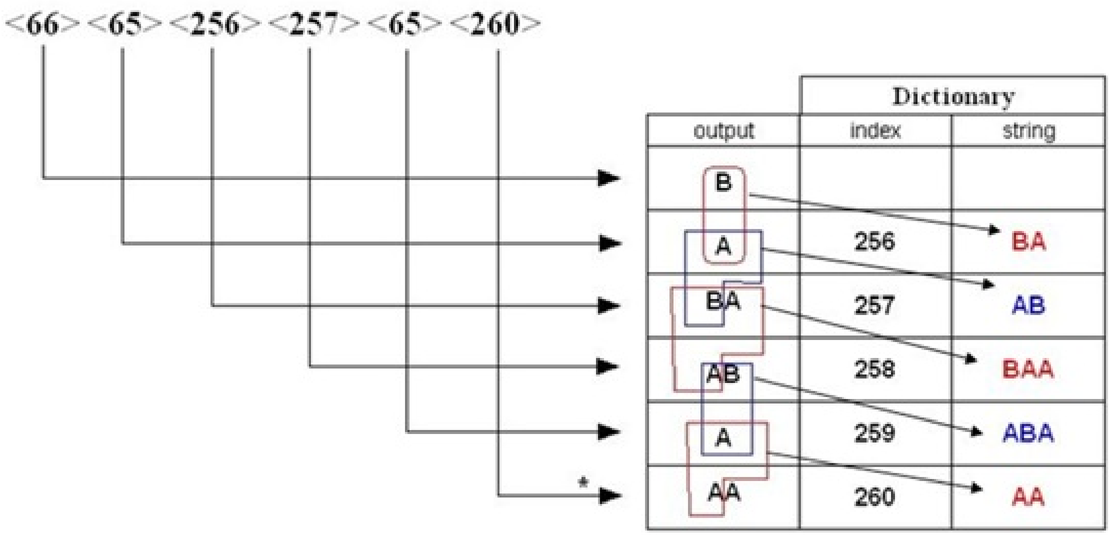


output	Dictionary	
	Code Word	String
66	256	BA
65	257	AB
256	258	BAA
257	259	ABA
65	260	AA
260		

编码结果： **<66><65><256><257><65><260>**

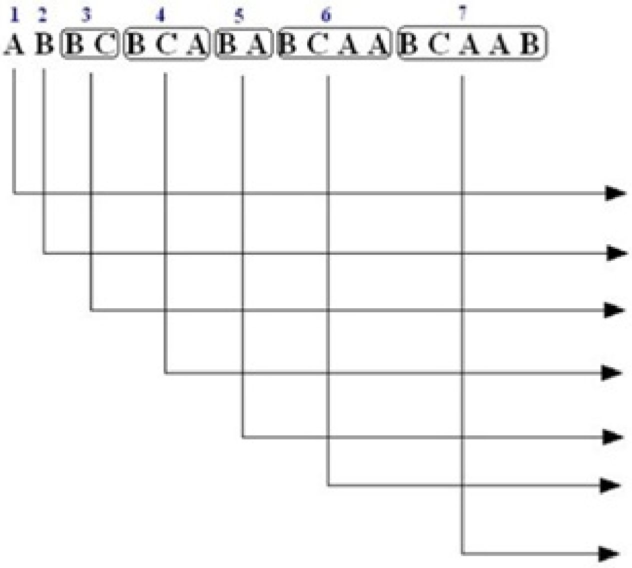
- **解压时LZW解码器能够产生和编码器完全一样的码表**
- **和编码器一样先初始化所有的单字符，将0-255之间的值分配给它们**
- **除了解码第一个字符外，解码其它字符时都要更新码表**
- **通过读码字并根据码表中的值将它们解码为对应的字符或字符组合**

输入：<66> <65> <256> <257> <65> <260>



18. LZ78

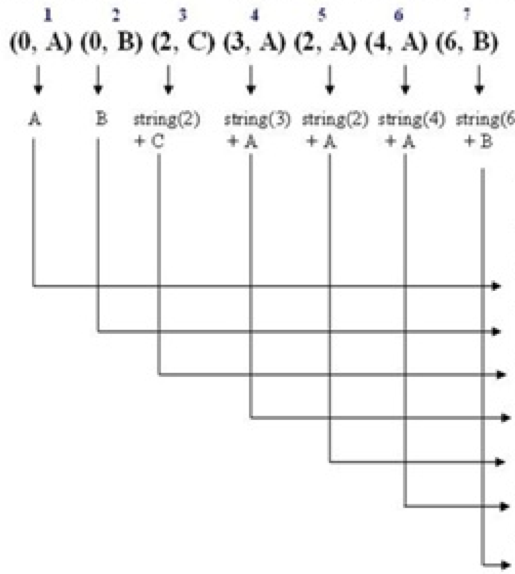
- 用LZ78算法编码字符串：ABBCBCABABCAABCAAB



	Dictionary	
output	index	string
(0, A)	1	A
(0, B)	2	B
(2, C)	3	BC
(3, A)	4	BCA
(2, A)	5	BA
(4, A)	6	BCAA
(6, B)	7	BCAAB

- 输出：(0,A)(0,B)(2,C)(3,A)(2,A)(4,A)(6,B)

- 用LZ78算法解码：(0,A)(0,B)(2,C)(3,A)(2,A)(4,A)(6,B)



	Dictionary	
output	index	string
A	1	A
B	2	B
BC	3	BC
BCA	4	BCA
BA	5	BA
BCAA	6	BCAA
BCAAB	7	BCAAB

- 输出：ABBCBCABABCAABCAAB

参考资料：

- **LZW**：http://www.cs.usyd.edu.au/~loki/cs2csys/gif-info/lzw.html
- **LZ78**：http://www.cs.usyd.edu.au/~loki/cs2csys/gif-info/lz78.html