计算机图像处理

COMPUTER IMAGE PROCESSING

信息理论基础与熵编码

离散信源的熵

设一个离散信源X： $(x_1, x_2, \cdots, x_N)$

其概率分布： $\{p_1, p_2, \cdots, p_N\}$ 满足 $\sum_{i=1}^{N} p_i = 1$

图像**X**

像素值 $(x_1, x_2, \cdots, x_N)$

直方图 $\{p_1, p_2, \cdots, p_N\}$

离散信源的熵

信源X，某个信源符号x_k，如果它出现的概率是p_k

x_k的自信息量

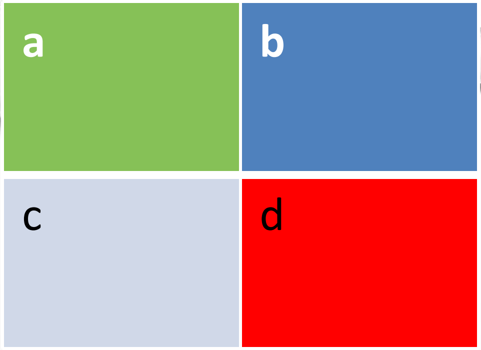
$$I(x_k) = \log_2 \frac{1}{p_k} = -\log_2 p_k$$

信源熵$H(X)$

$$H(X) = -\sum_{i=1}^{N} p_i \log_2 p_i$$

等长编码：对于一个离散信源中每一个符号，若采用相同长度的不同码字代表相应符号，就叫做等长编码，例如中国4位电报码。

变长编码：若对信源中的不同符号，用不同长度的码字表示就叫做不等长或变长编码。与定长编码相比，变长编码更复杂，除唯一可译码（也称为单义可译）的要求，还存在即时解码问题。



设 $X=\{a,b,c,d\}$

$$p(a)=p(b)=p(c)=p(d)=1/4$$

各信源符号自信息量：

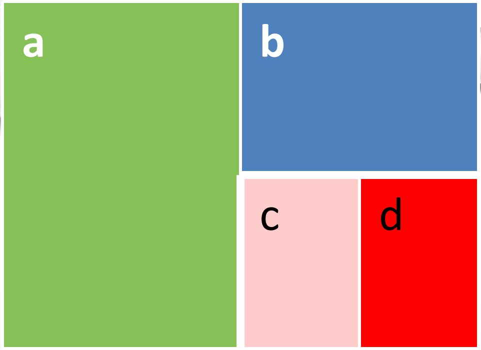
$$I(a)=I(b)=I(c)=I(d)=\log_2 4=2$$

信源熵

$$H(X)=1/4*2+1/4*2+1/4*2+1/4*2=2$$

符号	a	b	c	d	平均码长l_{avg}
码字	00	01	10	11	2

符号	a	b	c	d	平均码长l_{avg}
码字	0	10	110	111	2.25



设 $X=\{a,b,c,d\}$

$$p(a)=1/2, p(b)=1/4, p(c)=1/8, p(d)=1/8$$

各信源符号自信息量：

$$I(a)=\log_2 2=1, I(b)=\log_2 4=2, I(c)=I(d)=\log_2 8=3$$

信源熵

$$H(X)=1/2*1+1/4*2+1/8*3+1/8*3=1.75$$

符号	a	b	c	d	平均码长l_{avg}
码字	00	01	10	11	2

符号	a	b	c	d	平均码长l_{avg}
码字	0	10	110	111	1.75

a
b
c
d

设 $X=\{a,b,c,d\}$

$$p(a)=0.45, p(b)=0.25,$$
$$p(c)=0.18, p(d)=0.12$$

各信源符号自信息量：

$$I(a)=1.152, I(b)=2, I(c)=2.4739, I(d)=3.0589$$

信源熵

$$H(X)=0.45*1.152+0.25*2+0.18*2.4739+0.12*3.0589=1.8308$$

符号	a	b	c	d	平均码长l_{avg}
码字	00	01	10	11	2

符号	a	b	c	d	平均码长l_{avg}
码字	0	10	110	111	1.85

离散信源的熵

几点提示：

- 信源的平均码长$l_{avg}>=H(X)$；也就是说熵是无失真编码的下界。
- 如果所有$I(x_k)$都是整数，且$l(x_k)=I(x_k)$, 可以使平均码长等于熵。
- 对非等概率分布的信源，采用不等长编码其平均码长小于等长编码的平均码长。
- 如信源中各符号的概率相等，信源熵值达到最大，即最大离散熵定理。

$$H(X)=-\sum_{i=1}^{N} p_i \log_2 p_i$$

香农信息保持编码定理

香农信息论已证明，信源熵是进行无失真编码的理论极限。低于此极限的无失真编码方法是不存在的，这是熵编码的理论基础。

熵编码：一类对无语义数据流利用数据的统计信息去冗余的无损编码

变长编码定理

若一个离散信源具有熵，并有个码元符号集，则总可以找到一种无失真信源编码，使其平均码长满足：

$$H(X) \le L < H(X) + 1$$

变长最佳编码定理

在变长编码中，对出现概率大的信息符号赋予短码字，而对于出现概率小的信息符号赋予长码字。如果码字长度严格按照所对应符号出现概率大小逆序排列，则编码结果平均码字长度一定小于任何其他排列形式.

如何编码？

- 两个符号 (x_1, x_2) $\{p_1, p_2\} = \{0.7, 0.3\}$

符号	x1	x2
码字	0	1

- 三个符号 (x_1, x_2, x_3) $\{p_1, p_2, p_3\} = \{0.3, 0.5, 0.2\}$

排序、合并、分码

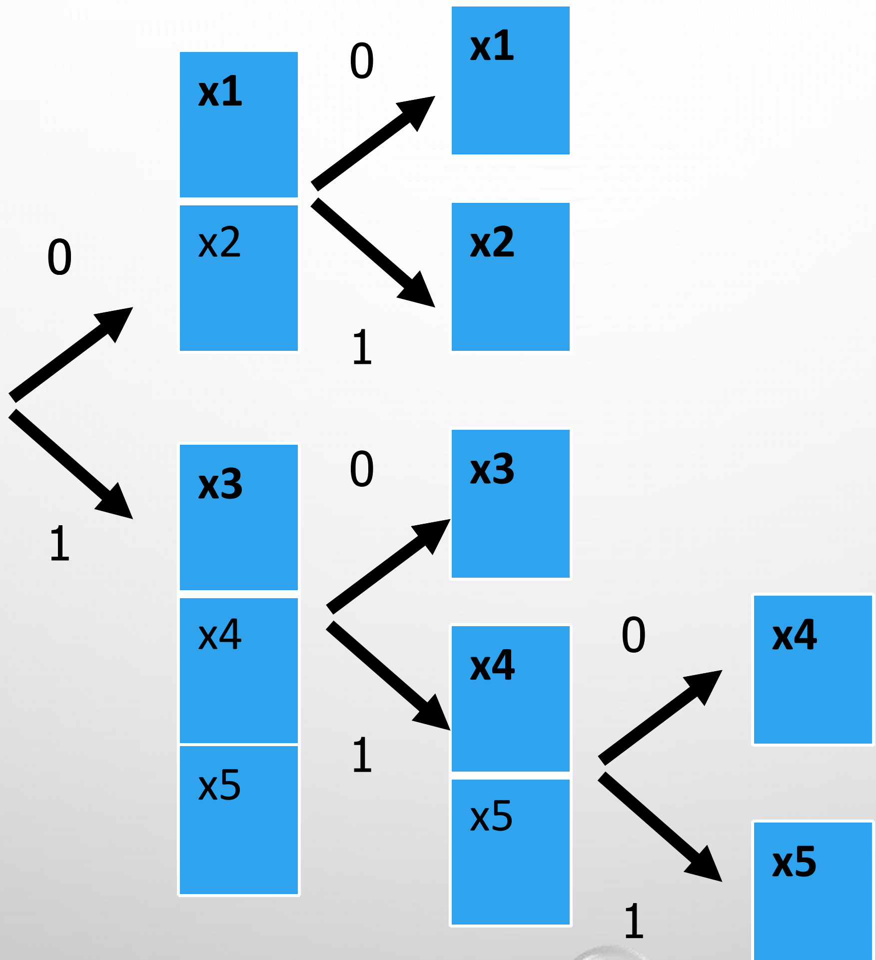
符号	x1	x2	x3
码字	10	0	11

- 四个符号？

- 方法一：
  - （1）将信源中符号X_I按其出现的概率，由大到小顺序排列。
  - （2）将信源分成两部分，使两个部分的概率和尽可能接近。重复第（2）步直至不可再分，即每一个叶子只对应一个符号。
  - （3）从左到右次序为这两部分标记0，1。
  - （4）将各个部分标记的0，1串接起来就得到各信源符号所对应的码字

Shannon-Fano编码

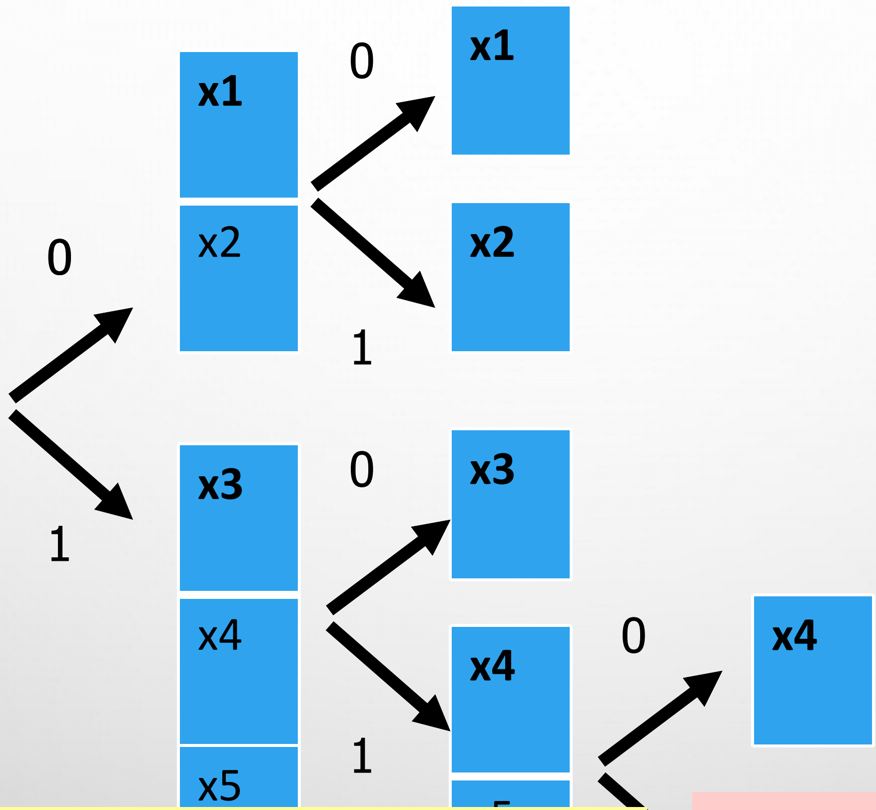
概率	灰度级
0.4	x1
0.175	x2
0.15	x3
0.15	x4
0.125	x5



灰度级	码字	码长
x1	00	2
x2	01	2
x3	10	2
x4	110	3
x5	111	3

Shannon-Fano编码

概率	灰度级
0.4	x1
0.175	x2
0.15	x3
0.15	x4
0.125	x5



灰度级	码字	码长
x1	00	2
x2	01	2
x3	10	2
x4	110	3
x5	111	3

$$H(X) = -\sum_{i=1}^{5} p(x_i)\log p(x_i) = 2.1649$$

$$L = \sum_{i=1}^{5} p(x_i) l_i = 2.275$$

- 方法二
  - （1）将信源符号X_i按其出现的概率，由大到小顺序排列。
  - （2）将两个最小的概率的信源符号组合相加，并重复这一步骤，始终将较大的概率分支放在上部，直到只剩下一个信源符号且概率达到1.0为止；
  - （3）对每对组合的上边一个指定为1，下边一个指定为0（或上边为0，下边为1）；
  - （4）画出由每个信源符号到概率1.0处的路径，记下沿路径的1和0；
  - （5）对于每个信源符号都写出1、0序列，则从右到左就得到非等长的码字。

Huffman编码

概率	灰度级
0.4	x1
0.175	x2
0.15	x3
0.15	x4
0.125	x5

Huffman编码

概率	灰度级
0.4	x1
0.175	x2
0.15	x3
0.15	x4
0.125	x5

概率	灰度级
0.4	x1
0.175	x2
0.15	x3
0.275	x4+ x5

Huffman编码过程示意图

Huffman编码

概率	灰度级
0.4	x1
0.175	x2
0.15	x3
0.15	x4
0.125	x5

概率	灰度级
0.4	x1
0.275	x4+ x5
0.175	x2
0.15	x3

Huffman编码过程示意图

Huffman编码

概率	灰度级
0.4	x1
0.175	x2
0.15	x3
0.15	x4
0.125	x5

概率	灰度级
0.4	x1
0.275	x4+x5
0.175	x2
0.15	x3

概率	灰度级
0.4	x1
0.275	x4+x5
0.325	x2+x3

Huffman编码过程示意图

Huffman编码

概率	灰度级
0.4	x1
0.175	x2
0.15	x3
0.15	x4
0.125	x5

概率	灰度级
0.4	x1
0.275	x4+x5
0.175	x2
0.15	x3

概率	灰度级
0.4	x1
0.325	x2+x3
0.275	x4+x5

Huffman编码过程示意图

Huffman编码

概率	灰度级
0.4	x1
0.175	x2
0.15	x3
0.15	x4
0.125	x5

概率	灰度级
0.4	x1
0.275	x4+x5
0.175	x2
0.15	x3

概率	灰度级
0.4	x1
0.325	x2+x3
0.275	x4+x5

概率	灰度级
0.4	x1
0.6	x2+x3x4+x5

Huffman编码过程示意图

Huffman编码

概率	灰度级
0.4	x1
0.175	x2
0.15	x3
0.15	x4
0.125	x5

概率	灰度级
0.4	x1
0.275	x4+x5
0.175	x2
0.15	x3

概率	灰度级
0.4	x1
0.325	x2+x3
0.275	x4+x5

概率	灰度级
0.6	x2+x3x4+x5
0.4	x1

Huffman编码过程示意图

Huffman编码

概率	灰度级
0.4	x1
0.175	x2
0.15	x3
0.15	x4
0.125	x5

概率	灰度级
0.4	x1
0.275	x4+x5
0.175	x2
0.15	x3

概率	灰度级
0.4	x1
0.325	x2+x3
0.275	x4+x5

概率	灰度级
0.6	x2+x3x4+x5
0.4	x1

概率	灰度级
1	x2+x3x4+x5 x1

Huffman编码过程示意图

Huffman编码

概率	灰度级
0.4	x1
0.175	x2
0.15	x3
0.15	x4
0.125	x5

概率	灰度级
0.4	x1
0.275	x4+x5
0.175	x2
0.15	x3

概率	灰度级
0.4	x1
0.325	x2+x3
0.275	x4+x5

概率	灰度级
0.6	x2+x3x4+x5
0.4	x1

概率	灰度级
1	x2+x3x4+x5 x1

Huffman编码过程示意图

Huffman编码

概率	灰度级
0.4	x1
0.175	x2
0.15	x3
0.15	x4 1
0.125	x5 0

概率	灰度级
0.4	x1
0.275	x4+x5
0.175	x2
0.15	x3

概率	灰度级
0.4	x1
0.325	x2+x3
0.275	x4+x5

概率	灰度级
0.6	x2+x3x4+x5
0.4	x1

概率	灰度级
1	x2+x3x4+x5 x1

Huffman编码过程示意图

Huffman编码

概率	灰度级
0.4	x1
0.175	x2
0.15	x3
0.15 1	x4
0.125 0	x5

概率	灰度级
0.4	x1
0.275	x4+x5
0.175 1	x2
0.15 0	x3

概率	灰度级
0.4	x1
0.325	x2+x3
0.275	x4+x5

概率	灰度级
0.6	x2+x3x4+x5
0.4	x1

概率	灰度级
1	x2+x3x4+x5 x1

Huffman编码过程示意图

Huffman编码

概率	灰度级
0.4	x1
0.175	x2
0.15	x3
0.15	1 x4
0.125	0 x5

概率	灰度级
0.4	x1
0.275	x4+ x5
0.175	1 x2
0.15	0 x3

概率	灰度级
0.4	x1
0.325	1 x2+ x3
0.275	0 x4+ x5

概率	灰度级
0.6	x2+x3x 4+x5
0.4	x1

概率	灰度级
1	x2+x3x 4+x5 x1

Huffman编码过程示意图

Huffman编码

概率	灰度级	
0.4	x1	
0.175	x2	
0.15	x3	
0.15	x4	1
0.125	x5	0

概率	灰度级	
0.4	x1	
0.275	x4+x5	
0.175	x2	1
0.15	x3	0

概率	灰度级	
0.4	x1	
0.325	x2+x3	1
0.275	x4+x5	0

概率	灰度级	
0.6	x2+x3x4+x5	1
0.4	x1	0

概率	灰度级
1	x2+x3x4+x5 x1

Huffman编码过程示意图

Huffman编码

概率	灰度级
0.4	x1
0.175	x2
0.15	x3
0.15	1 x4
0.125	0 x5

概率	灰度级
0.4	x1
0.275	x4+x5
0.175	1 x2
0.15	0 x3

概率	灰度级
0.4	x1
0.325	1 x2+x3
0.275	0 x4+x5

概率	灰度级
0.6	1 x2+x3x4+x5
0.4	0 x1

灰度级	灰度级
x1	0
x2	111
x3	110
x4	101
x5	100

Huffman编码过程示意图

Huffman编码

信源符号	出现概率	码字	码长
X1	0. 4	0	1
X2	0. 175	111	3
X3	0. 15	110	3
X4	0. 15	101	3
X5	0. 125	100	3

$$H(X) = -\sum_{i=1}^{5} p(x_i)\log p(x_i) = 2.164$$

$$L = \sum_{i=1}^{5} p(si)li = 2.2$$

Huffman编码

特点：

（1）Huffman编码不唯一

两个概率分配码字“0”和“1”是任意选择的（大概率为“0”，小概率为“1”，或者反之）

在排序过程中两个概率相等，谁前谁后随机

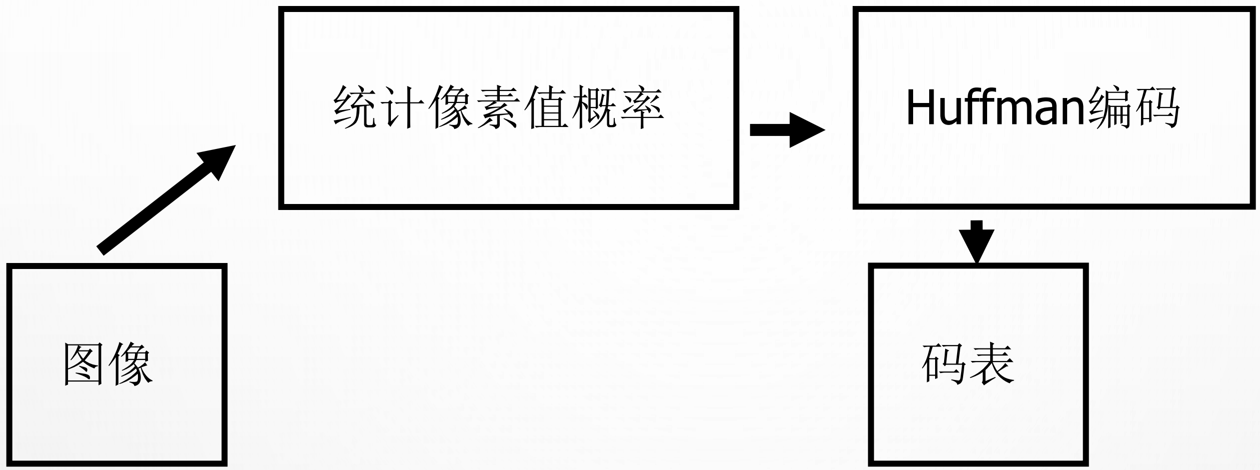
（2）变长码，平均码字短，效率高，但实时硬件实现很复杂（特别是译码），抗误码能力差

（3）信源概率是2的负幂时，效率达100%

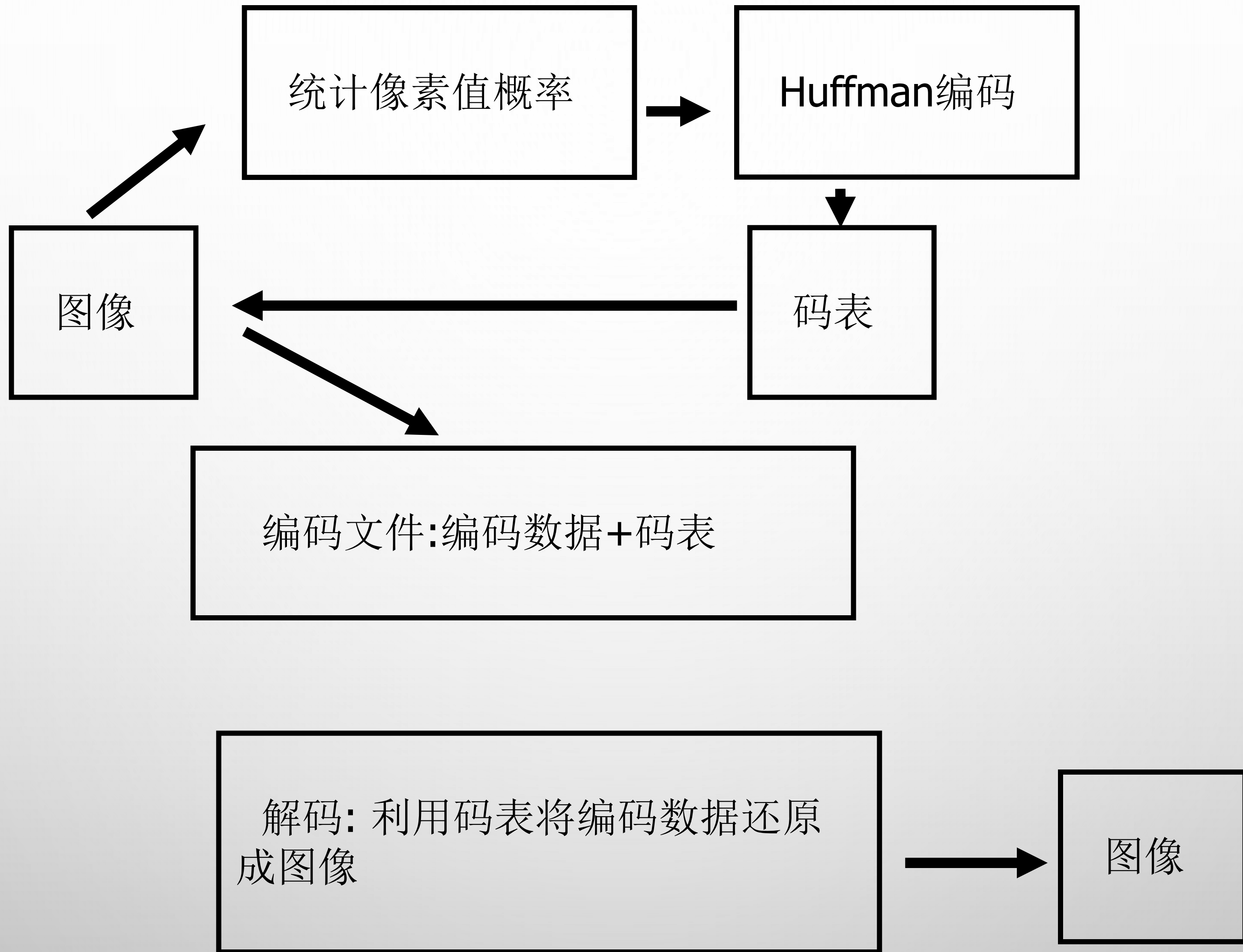
（4）Huffman编码只能用近似的整数位来表示单个符号，而不是理想的小数

这是Huffman编码无法达到最理想的压缩效果的原因

编码框架



编码框架



程序？！