GI12/4C59: Information Theory

Lectures 7–9

Massimiliano Pontil

About these lectures

Theme of lectures 7–9: We present the theory of data compression – finding a description of a discrete random variable which minimizes the average description length. We show the relation between the entropy of this random variable and the minimum description length and discuss several optimal and sub-optimal coding strategies.

Math required: Basic familiarly with probability and convexity.

Outline

1. Codes

2. Kraft inequality

3. Optimal codes and relation to entropy

4. Shannon codes

5. Huffman codes

6. Arithmetic codes

Codes

Let $\mathcal{X} = \{x_1, \ldots, x_m\}, m \in \mathbb{N}$ and $\mathcal{A}^*$ the set of finite sequences (strings) of symbols from a finite alphabet $\mathcal{A} = \{a_1, \ldots, a_D\}$. A (any!) function $C : \mathcal{X} \to \mathcal{A}^*$ is called a code for $\mathcal{X}$ with alphabet $\mathcal{A}$.

C is said *non-singular* if $C(x) \neq C(t)$ for every $x, t \in \mathcal{X}$, $x \neq t$.

- Without loss of generality we assume that $\mathcal{A} = \{0, 1, \ldots, D - 1\}$. The default choice is $\mathcal{A} = \{0, 1\}$.

- $C(x)$ is called the *codeword* corresponding to x.

- The length of the string $C(x)$ is denoted by $\ell(x)$.

Coding and data compression

We wish to find a code with has the shortest average code length for the random variable X (or source symbols in $\mathcal{X}$ drawn with probability p),

$$L = E[\ell(X)] = \sum_{x \in \mathcal{X}} p(x)\ell(x)$$

Minimizing L may be computationally intensive, so we are happy to just find a good approximation to the shortest description.

Minimal description length can be useful, for example, for document (text, image, sound) compression.

$N-$extension of a code

Often we need to code not just one element of $\mathcal{X}$ but a sequence of elements, $x^n = (x_1, x_2, \ldots, x_n)$, $x_i \in \mathcal{X},\ i = 1, \ldots, n$.

Can we use C to code x^n?

We introduce the $n-$extension of a code C to be the code $C^n : \mathcal{X}^n \to \mathcal{A}^*$ defined by

$$C^n(x^n) = C(x_1)C(x_2)\cdots C(x_n)$$

Even if C is nonsingular, its extension may be singular! (see example below). If C^n is non-singular for every $n > 0$ we say that C is a *uniquely decodable* code.

Instantaneous codes

Even if C is uniquely decodable, given a code sequence $v \in Range(C^n)$ we may have to look at the entire string v in order to find the first symbol in the decoded sequence.

A code is called an *instantaneous* or *prefix* free code if no codeword is the prefix of any other codeword.

An instantaneous code C can be decoded without reference to future codewords: the end of a codeword is *immediately recognizable* when we move left-to-right along a code sequence.

Example 1

Let $\mathcal{X} = \{x_1, x_2, x_3, x_4\}$, $\mathcal{A} = \{0, 1\}$, and consider the following non-singular codes.

Code	x_1	x_2	x_3	x_4
C_1	0	010	01	10
C_2	10	00	11	110
C_3	0	10	110	111

C_1 is not uniquely decodable: the string 010 could be decoded as x_2, (x_1, x_4) or (x_3, x_1).

C_2 is uniquely decodable (why?) but not instantaneous since the codeword 11 is a prefix of the codeword 110. In particular, to decode the string 11000 we need to count the number of "0"s after 11.

C_3 is instantaneous: no codeword is the prefix of any other codeword

Kraft inequality

We focus on instantaneous codes as they are easy to decode.

Theorem: If $C : \mathcal{X} \to \mathcal{A}^*$ is an instantaneous code and ℓ_i the length of $C(x_i)$, $i = 1, \ldots, m$, $m = |\mathcal{X}|$, then

$$\sum_{i=1}^{m} D^{-\ell_i} \leq 1 \qquad (1)$$

Conversely, if the sequence of numbers $\ell_1, \ldots, \ell_m$ satisfies (1) then there exists an instantaneous code with such codeword lengths.

Example 1 (cont.) the code C_3 in the above example satisfies Kraft inequality.

Note: Sometimes we write the Kraft inequality as $\sum_{x \in \mathcal{X}} D^{-\ell(x)}$

Preparing for the proof

Let ℓ_{max} be the maximum codeword length. The codewords can be seen as the leaves of a $D-$ary tree of depth ℓ_{max} (there are m leaves in such a tree.) The path from the root to one leaf traces out the sequence of symbols of the codeword. By construction, a codeword cannot be a prefix of any other codeword and, so, the code is instantaneous.

[Draw the tree for code C_3 above. Compare to C_2]

Proof

There are $D^{\ell_{max}}$ nodes at level ℓ_{max} in the tree.

Let $\mathcal{A}_i$ be the set of descendants of the $i-$th codeword. There are $D^{\ell_{max}-\ell_i}$ such descendants and, by construction, the sets $\mathcal{A}_i$ are disjoint. In addition, the total number of these descendants cannot be more than $D^{\ell_{\text{max}}}$, the maximum number of leaves in the tree. Thus, we have

$$\sum_{i=1}^{m} D^{\ell_{max}-\ell_i} \leq D^{\ell_{\text{max}}} \quad \Rightarrow \quad \sum_{i=1}^{m} D^{-\ell_i} \leq 1$$

Using the $D-$ary tree construction, the other part of the proof is immediate .

Kraft inequality for uniquely decodable codes

Kraft inequality also holds for uniquely decodable codes. This result is remarkable: the larger class of uniquely decodable codes does not provide any improvement on the minimal description length.

Proof: We need to show that

$$\sum_{x\in\mathcal{X}} D^{-\ell(x)} \leq 1.$$

We denote by $\ell(x^n)$ the length of the code sequence corresponding to $x^n = (x_1, \ldots, x_n) \in \mathcal{X}^n$. The extension code satisfies

$$\ell(x^n) = \sum_{i=1}^{n} \ell(x_i)$$

Proof

We have

$$\left(\sum_{x\in\mathcal{X}} D^{-\ell(x)}\right)^n = \sum_{x_1\in\mathcal{X}}\sum_{x_2\in\mathcal{X}}\cdots\sum_{x_n\in\mathcal{X}} D^{-\ell(x_1)}D^{-\ell(x_2)}\cdots D^{-\ell(x_n)} = \sum_{x^n\in\mathcal{X}^n} D^{-\ell(x^n)}$$

which we re-write as

$$\sum_{x^n\in\mathcal{X}^n} D^{-\ell(x^n)} = \sum_{k=1}^{n\ell_{max}} a(k)D^{-k}$$

where ℓ_{max} is the maximum codeword length and $a(k)$ the number of sequence x^n whose concatenation codeword has length k. Since the code is uniquely decodable, $a(k) \leq D^k$ which implies

$$\left(\sum_{x\in\mathcal{X}} D^{-\ell(x)}\right)^n = \sum_{k=1}^{n\ell_{max}} a(k)D^{-k} \leq \sum_{k=1}^{n\ell_{max}} D^k D^{-k} \leq \frac{(n\ell_{max}+1)n\ell_{max}}{2}$$

Proof (cont.)

We re-write last inequality as

$$\sum_{x\in\mathcal{X}} D^{-\ell(x)} \leq \left(\frac{(n\ell_{max}+1)n\ell_{max}}{2}\right)^{\frac{1}{n}}$$

But since this holds for every n, we have that, for $n \to \infty$

$$\sum_{x\in\mathcal{X}} D^{-\ell(x)} \leq \left(\frac{(n\ell_{max}+1)n\ell_{max}}{2}\right)^{\frac{1}{n}} < (n\ell_{max})^{\frac{2}{n}} \to 1.$$

from which

$$\sum_{x\in\mathcal{X}} D^{-\ell(x)} \leq 1.$$

The converse of the proof is like for prefix-free codes

Note: The Kraft inequality holds also for infinite set $\mathcal{X}$

Optimal codes

Problem: We wish to find an instantaneous code with has the shortest average code length $L = E[\ell(X)]$.

This is equivalent to solve the constrained minimization problem:

$$\min \left\{ \sum_{k=1}^{m} p_k \ell_k : \sum_{k=1}^{m} D^{-\ell_k} \leq 1,\ \ell_k \in \mathbb{N}, k = 1, \ldots, m \right\} \qquad (P.1)$$

Solution

If we relax the integer constraints we can use the Lagrangian

$$J(\ell_1, \ldots, \ell_m) = \sum_{k=1}^{m} p_k \ell_k + \mu \left(\sum_{k=1}^{m} D^{-\ell_k} - 1 \right)$$

and

$$\frac{\partial J}{\partial \ell_k} = p_k + \mu \frac{-D^{-\ell_k}}{\log_D e}$$

If we set $\frac{\partial J}{\partial \ell_k} = 0$ and equal the kraft inequality we obtain the minimum

$$\bar{\ell}_k = -\log_D p_k$$

Since J is convex for fixed μ, this is the optimal real solution.

In general, $\bar{\ell}_k$ is not an integer and, so, it is not a minimum of (P.1). When ℓ_k is an integer we say that p a $D-$*adic* distribution, that is, p_k is a negative power of D for every $k = 1, \ldots, m$.

Lower bound

Note that, above, $\sum_{k=1}^{n} p_k \bar{\ell}_k = H_D(p)$, the entropy of p.

Lemma: Every instantaneous code satisfies $L \geq H_D(X)$.

Proof: We have

$$L - H_D(X) = \sum_k p_k(\ell_k - \bar{\ell}_k) = -\sum_k p_k \log_D D^{-\ell_k} + \sum_k p_k \log_D p_k.$$

If we set $C = \sum_k D^{-\ell_k} \leq 1$ and define the probability distribution $q_k = C^{-1} D^{-\ell_k}$,

$$\begin{aligned} L - H_D(X) &= -\sum_k p_k \log_D Cq_k + \sum_k p_k \log_D p_k \\ &= -\sum_k p_k \log_D \frac{q_k}{p_k} - p_k \log_D C = D(p||q) + \log_D \frac{1}{C} \geq 0 \end{aligned}$$

with equality if and only if p is $D-$adic with $p_k = D^{-\ell_k}$.

Upper bound

We exhibit a sub-optimal instantaneous code for which

$$H_D(X) \leq L < H_D(X) + 1$$

Proof: Let $\lceil x \rceil$ be the smallest integer which is greater than or equal to x, and choose

$$\ell_k = \left\lceil \log_D \frac{1}{p_k} \right\rceil$$

This choice of code lengths satisfies Kraft inequality and, since

$$\log_D \frac{1}{p_k} \leq \ell_k < \log_D \frac{1}{p_k} + 1$$

the result follows.

Note: This code is called the Shannon code.

Description length with a wrong distribution

We show that if we use q as the distribution of X but the true distribution is p, and use the codeword lengths

$$\ell(x) = \left\lceil \log \frac{1}{q(x)} \right\rceil .$$

Then the average optimal description of X satisfies the bound

$$H(p) + D(p \parallel q) \leq E_p[\ell(X)] < H(p) + D(p \parallel q) + 1$$

Description length with a wrong distribution (cont.)

Proof: We have

$$\begin{aligned} E[\ell(X)] &= \sum_{x \in \mathcal{X}} p(x) \left\lceil \log \frac{1}{q(x)} \right\rceil \\ &< \sum_{x \in \mathcal{X}} p(x) \left(\log \frac{1}{q(x)} + 1 \right) \\ &= \sum_{x \in \mathcal{X}} p(x) \left(\log \frac{p(x)}{q(x)} + \log \frac{1}{p(x)} + 1 \right) \\ &= D(p \parallel q) + H(p) + 1 \end{aligned}$$

The lower bound is derived similarly.

Summary so far

- Instantaneous codes, a subset of uniquely decodable codes, allow efficient decoding of a code sequence.
- Any uniquely decodable (and, in particular, instantaneous) code satisfies Kraft inequality, $\sum_{x\in\mathcal{X}} D^{-\ell(x)} \leq 1$.
- For every uniquely decodable $D-$ary code, $E[\ell(X)] \geq H(X)$.
- A simple sub-optimal code is Shannon code, $\ell(x) = \left\lceil \log_D \frac{1}{p(x)} \right\rceil$. Its average length is less than $H(p)+1$. If p is a $D-$ary distribution, Shannon code is optimal and $E[\ell(X)] = H(X)$.
- For every optimal code, $E[\ell(X)] \in [H(p), H(p)+1)$.

A simple property of optimal codes

We first note that if C is an (any) optimal code and $p_j > p_k$, then $\ell_j \leq \ell_k$. In fact, consider the code C' which equals C except that the codewords C_j and C_k are swapped. We have

$$\ell'_i = \begin{cases} \ell_i & \text{if } i \neq j,k \\ \ell_j & \text{if } i = k \\ \ell_k & \text{if } i = j \end{cases}$$

and, so,

$$L(C') - L(C) = p_k\ell_j + p_j\ell_k - p_k\ell_k - p_j\ell_j = (p_j - p_k)(\ell_k - \ell_j)$$

Since $p_j - p_k > 0$, we must have $\ell_k - \ell_j \geq 0$, otherwise C is not optimal.

Characterization of optimal instantaneous codes

The following result holds for binary codes but its extension to $D-$ary codes is straightforward.

Lemma: For any probability distribution there exists an optimal instantaneous code such that the two longest codewords (a) have the same length, (b) are assigned to two of the least likely symbols, and (c) differ only in the last bit.

Sketch of the proof: (a) Suppose there is only one longest codeword. Then if we delete the last bit, the reduced codeword has still the prefix free property.
(b) This follows from the observation made in the previous slide.
(c) At least two codewords of maximal length must be siblings, otherwise we could delete the last symbol and still have an instantaneous code with smaller average length. We then make a permutation of these longest codewords so that two among the least likely ones are siblings.

Example 2

Let $\mathcal{X} = \{1, 2, 3, 4\}$ and $p(1) = .3$, $p(2) = p(3) = .25$, $p(4) = .2$. We have $H(p) = 1.9855$.

The following binary codes are instantaneous

Code	x_1	x_2	x_3	x_4	L
C_1	0	11	100	101	2.45
C_2	00	11	10	01	2
C_3	00	01	10	11	2

C_1 is not optimal. Both C_2 and C_3 are optimal. C_3 satisfies the lemma above and is obtained by C_2 by swapping $C_2(x_2)$ with $C_2(x_4)$.

Example 3

Assume now that X is distributed with q such that $q(1) = .6,\ q(2) = q(3) = .15,\ p(4) = .1$. In this case $H(p) = 1.5955$.

Code	x_1	x_2	x_3	x_4	L
C_1	0	11	100	101	1.65
C_2	00	11	10	01	2
C_3	00	01	10	11	2
C_4	0	10	01	111	1.5

Now C_2 and C_3 are not optimal anymore while C_1 is an optimal instantaneous code and satisfies the above lemma. C_4 has smaller description length than C_1 but in not uniquely decodable.

Huffman codes

Let C be an instantaneous code which satisfies the above lemma and assume, without loss of generality, that $p_1 \geq p_2 \geq \ldots \geq p_m$.

A code which satisfies the properties stated in the above lemma is called an Huffman code. It can be obtained by repeatedly "merging" the last two symbols, assigning to them the "last codeword minus the last bit", and reordering the symbols in order to have non-increasing probabilities.

Huffman codes (cont.)

More precisely, to find an Huffman code we repeat the following procedure until we end up with only two symbols.

1. Replace x_{m-1} and x_m by a new symbol t_{m-1} having probability $p_{m-1} + p_m$.
2. Assign to t_{m-1} the codeword obtained by removing the last bit in $C(x_{m-1})$ or C_m (which differ only in the last bit).
3. Reorder $x_1, x_2, \ldots, x_{m-2}, t_{m-1}$ according to non increasing probabilities.

We then assign codewords $0, 1$ to the last two symbols and "propagate back" these codewords.

Note that this procedure is a greedy algorithm.

Huffman codes (cont.)

More precisely, note that in the above procedure

$$\begin{aligned} L(C_m) - L(C_{m_1}) &= \sum_{i=1}^{m} p_i \ell_i - \sum_{i=1}^{m-2} p_i \ell_i - (p_{m-1} + p_m)(\ell_m - 1) \\ &= p_{m-1} + p_m \end{aligned}$$

Since the difference between we average length of code C_m and C_{m-1} does no depend on C_{m-1}, if C_{m-1} is optimal so is C_m and we can iterate. We conclude that the above procedure is optimal.

The next examples illustrate this algorithm.

Example 4 (Huffman code)

x	$C(x)$	p	w	p^1	w^1	p^2	w^2
1	0	.6	w_1	.6	w_1^1	.6	$w_1^2 = 0$
2	11	.15	w_2	.25	$w_2^1 = w_2^2 0$	.4	$w_2^2 = 1$
3	100	.15	$w_3 = w_2^1 0$	.15	$w_3^1 = w_2^2 1$		
4	101	.1	$w_4 = w_2^1 1$				

Example 5 (Huffman code)

x	$C(x)$	p	w	p^1	w^1	p^2	w^2	p^3	w^3
1	01	.25	w_1	.3	w_1^1	.45	w_1^2	.55	$w_1^3 = 0$
2	10	.25	w_2	.25	w_2^1	.3	$w_2^2 = w_1^3 0$	.45	$w_2^3 = 1$
3	11	.2	w_3	.25	$w_3^1 = w_1^2 0$	.25	$w_3^2 = w_1^3 1$		
4	000	.15	$w_4 = w_1^1 0$	.2	$w_4^1 = w_1^2 1$				
5	001	.15	$w_4 = w_1^1 1$						

Example 6 (Huffman code)

x	$C(x)$	p	w	p^1	w^1
1	1	.25	w_1	.5	$w_1^1 = 0$
2	2	.25	w_2	.25	$w_2^1 = 1$
3	00	.2	$w_3 = w_2^1 0$	.25	$w_3^1 = 2$
4	01	.15	$w_4 = w_2^1 1$		
5	02	.15	$w_5 = w_2^1 2$		

Example 6 (Huffman code)

The total number of symbols must equal to $1 + k(D-1)$ where k is the depth of the tree. We can always match this by adding dummy symbols

x	$C(x)$	p	w	p^1	w^1	p^2	w^2
1	1	.25	w_1	.25	w_1^1	.5	$w_1^2 = 0$
2	2	.25	w_2	.25	w_2^1	.25	$w_2^2 = 1$
3	00	.2	w_3	.2	$w_3^1 = w_1^2 0$	.25	$w_3^2 = 2$
4	02	.1	w_4	.2	$w_4^1 = w_1^2 1$		
5	010	.1	$w_5 = w_4^1 0$	.1	$w_5^1 = w_1^2 2$		
6	011	.1	$w_6 = w_4^1 1$				
$-$	012	.0	$w_7 = w_4^1 2$				

Source coding and 20 questions

In the game of 20 questions we wish to find the most efficient series of yes/no questions to determine an object from a class of objects.

Each question is of the type “Is $x \in \mathcal{A}$?” for some $\mathcal{A} \subset \mathcal{X}$ and, in general, depends on the answers to the questions before it.

Also, any code gives rise to the series of questions “Is the $k-$th bit of the code equal to 1?”, for $k = 1, \ldots, \ell_{max}$, and the average number of required questions equals the average length of that code. So, an optimal series of questions are those determined by an Huffman code!

Source coding and 20 questions (cont.)

example 5 (cont.) $p_1 = p_2 = .25$, $p_3 = .2$, $p_4 = p_5 = .15$ and we found the Huffman code: $01, 10, 11, 000, 001$.

An optimal series of questions to determine X are:

Q1 = “is X equal to 2 or 3?” = “Is the first bit = 1?”.

If the answer is Yes we could ask: Q2 = “is $X = 3$?” Or we can directly ask: Q2 = “Is $X = 1$ or 3?” = “Is the second bit = 1?”.

Q3 = “is X equal to 4” = “Is the third bit = 1?”..

The expected number of questions N in this optimal scheme satisfies

$$H(X) \leq N < H(X) + 1$$

20 questions and slice codes

Problem: Suppose $p_1 \geq p_2 \geq \cdots \geq p_m$. Is there an optimal sequence of questions of the type "is $X > a$?", $a \in \mathbb{N}$? ("slice" questions)

In general an Huffman code does not provide slice questions (see, e.g., the above example) but we can always reorder its codewords to obtain another Huffman code (called *alphabetic code* or "slice" code) which provides such questions.

Example 5 (cont.) The above Huffman code, $01, 10, 11, 000, 001$ tells us the optimal codelengths are $2, 2, 2, 3, 3$. Using these codelengths and assigning the symbols to the first available node in the binary tree for the code we obtain the alphabetic code $C(1) = 00$, $C(2) = 01$, $C(3) = 10$, $C(4) = 110$, $C(5) = 111$

Shannon-Fano-Elias coding

We assume that $p(x) > 0$ for every $x \in \mathcal{X}$ and describe a code based on the cumulative distribution function of X, $F(x) = \sum_{t \leq x} p(t)$.

We introduce a slight modification of F,

$$\bar{F}(x) = \sum_{t<x} p(t) + \frac{1}{2} p(x)$$

F consists of steps of size $p(x)$ at x and $\bar{F}(x)$ is the midpoint of that step.

Since $p(x) > 0$ for every x, $\bar{F}(x)$ can be use to code x. But this is in general a real number (infinite bits).

A finite length code can be obtained by rounding off $\bar{F}(x)$ to $\ell(x)$ bits, that is, the codeword of x is

$$C(x) = \lfloor \bar{F}(x) \rfloor_{\ell(x)}$$

Shannon-Fano-Elias coding (cont.)

Example 5 (cont)

x	$p(x)$	$F(x)$	$\bar{F}(x)$	$\ell(x) = \lceil \log \frac{1}{p(x)} \rceil + 1$	$\lfloor \bar{F}(x) \rfloor_{\ell(x)}$	codeword
1	.25	.25	.125	3	.001	001
2	.25	.5	.375	3	.011	011
3	.2	.7	.6	4	$.1\overline{0011}$	1001
4	.15	.85	.775	4	$.110\overline{0011}$	1100
5	.15	1.0	.925	4	$.111\overline{0110}$	1110

Note that, since $p(x)$ is not 2−adic, the binary representation of $F(x)$ may have infinite number of bits.

Note that this code is instantaneous. It is on average 1.2bits longer the Huffman code derived above.

Shannon-Fano-Elias coding (cont.)

How many bits do we need to make C instantaneous? We show that it is sufficient to choose

$$\ell(x) = \left\lceil \log \frac{1}{p(x)} \right\rceil + 1$$

In fact, by definition, $\bar{F}(x) - \lfloor \bar{F}(x) \rfloor_{\ell(x)} < 2^{-\ell(x)}$. We also have that

$$2^{-\ell(x)} = 2^{-\lceil \log \frac{1}{p(x)} \rceil - 1} < \frac{p(x)}{2} \Rightarrow \bar{F}(x) - \lfloor \bar{F}(x) \rfloor_{\ell(x)} < \frac{p(x)}{2}$$

This proves that if the roundoff of $\bar{F}(x)$ uses $\lceil \log \frac{1}{p(x)} \rceil + 1$ bit, the corresponding code is nonsingular.

Shannon-Fano-Elias coding (cont.)

The above choice is also sufficient to guarantee the prefix free property. In fact, since $2^{-\ell(x)} < \frac{p(x)}{2}$ the sets

$$\left[\lfloor \bar{F}(x) \rfloor_{\ell(x)}, \lfloor \bar{F}(x) \rfloor_{\ell(x)} + \frac{1}{2^{\ell(x)}}\right]$$

are disjoint and the code is prefix-free if and only if the interval corresponding to codewords are disjoints.

Finally note that

$$L = \sum_{x \in \mathcal{X}} p(x)\ell(x) = \sum_{x \in \mathcal{X}} p(x)\left(\left\lceil \log \frac{1}{p(x)} \right\rceil + 1\right) < H(X) + 2$$

Arithmetic coding

Let $\mathcal{X} = \{0, 1\}$. If we have a block (or sequence) of symbols from $\mathcal{X}$ and code each symbol separately we may lose one bit per symbol). We are better off coding directly block symbols from $\mathcal{X}$!

Huffman code is not efficient to do this as it requires the construction of the code for each fixed block length!

Let us use the above idea. If $x^n = (x_1, \ldots, x_n) \in \mathcal{X}^n$, we need to calculate

$$\bar{F}(x^n), \quad \text{and} \quad \ell(x^n) = \lceil \log \frac{1}{p(x^n)} \rceil + 1$$

How can we do this efficiently?

Computing $F(x)$

We represent all block symbols of length n as the leaves of a binary tree T^n of depth n where in each node the symbol 1 (0) brings us to the right (left) branch on the tree. Thus, if $x^n > y^n$ the leaf corresponding to x^n is to the right of the leaf corresponding to y^n.

Let $\mathcal{T}$ be the space of all subtrees of T^n. We can compute $F(x^n)$ efficiently as

$$F(x^n) = \sum_{y^n \leq \ x^n} p(y^n) = \sum_{T \prec \ x^n} p(\mathcal{T})$$

A subtree T is represented by the path from the root of the binary tree T^n to the root of T. So, if T is at the left of $x^n = (x_1, \ldots, x_n)$ – we use the notation $T \prec x^n$ – we have $T = x_1 x_2 \cdots x_{k-1} 0$ for some $k \leq n-2$ and, so,

$$p(T) = p(x_1, x_2, \ldots, x_{k-1}, 0)$$

Sequential encoding

Note that if we have computed $F(x^n)$ for every x^n, if is easy to compute $F(x^{n+1})$ for every x^{n+1}. In fact

$$F(x^{n+1}) = F(x^n, x_{n+1}) = \begin{cases} F(x^n) + p(x^n, 0) & \text{if } x_{n+1} = 1 \\ F(x^n) & \text{if } x_{n+1} = 0 \end{cases}$$

Thus encoding can be done sequentially.

The above procedure is efficient provided that $p(x^n)$ can be efficiently computed. This is true for example in the case of $i.i.d$ or Markov sources.

Decoding

Decoding can also be done sequentially by using the above tree T^n as a decision tree.

Given the codeword $\lfloor F(x^n) \rfloor_{\ell(x^n)}$, we first check at the root node in the tree whether $\lfloor F(x^n) \rfloor_{\ell(x^n)} > p(0)$. If yes, then $x_1 = 1$ (the subtree starting with 0 is to the left of x^n), otherwise $x_1 = 0$. We then iterate this process.

Example of arithmetic coding

Suppose $X_1, \ldots, X_n$ are i.i.d. Bernoulli random variables with $P(\{X = 1\}) = p$. Compute $F(01110)$.

We have

$$F(01110) = p(00) + p(010) + p(0110) = (1-p)^2 + p(1-p)^2 + p^2(1-p)^2$$

This can be calculated sequentially by the above formula, that is

$$F(x^{n+1}) = F(x^n, x_{n+1}) = \begin{cases} F(x^n) + p(x^n, 0) & \text{if } x_{n+1} = 1 \\ F(x^n) & \text{if } x_{n+1} = 0 \end{cases}$$

Bibliography

See Chapter 5 of
T.M. Cover and J.A. Thomas, *The elements of information theory*, Wiley, 1991.