Information Theory and Coding

EPFL, Fall 2022

Contents

Chapter 1

Lecture 1

1.1 Source Coding

Definition 1. Given an alphabet $\mathcal{U}$ with $|\mathcal{U}| < \infty$. A source code c is a mapping from the alphabet to the set of all finite binary strings represented as $\{0,1\}^* = \{null, 0, , 1, 00, 01, 10, 11, 000, \dots\}$ *i.e.* $c : \mathcal{U} \to \{0,1\}^*$.

Example. Consider $\mathcal{U} = \{a, b, c, d, e\}$. Then we can have a $c : \mathcal{U} \to \{0,1\}^*$ as $c(a) = 0$, $c(b) = 0$, $c(c) = null$, $c(d) = 1$, $c(e) = 010100$.

Definition 2. A code c is called singular if $\exists u, v \in \mathcal{U}$ s.t. $u \neq v$ but $c(u) = c(v)$.

Definition 3. A code c is called non-singular or injective if it is not singular.

Example. For the alphabet in the previous example, we can have an injective code as $c(a) = 0$, $c(b) = 00$, $c(c) = null$, $c(d) = 1$ and $c(e) = 010100$.

Definition 4. Given $c : \mathcal{U} \to \{0,1\}^*$, we write the concatenation of the codes of two letters $u_1, u_2 \in \mathcal{U}$ as $c(u_1)c(u_2)$. We further define

$$c^2 : \mathcal{U}^2 \to \{0,1\}^* \text{ to be } c^2(u_1, u_2) = c(u_1)c(u_2)$$

$$c^n : \mathcal{U}^n \to \{0,1\}^* \text{ to be } c^n(u_1, \dots, u_n) = c(u_1)c(u_2)\dots c(u_n)$$

$$c^* : \mathcal{U}^* \to \{0,1\}^* \text{ to be } c^*(u_1, \dots, u_n) = c(u_1)c(u_2)\dots c(u_n)$$

Definition 5. A code $c : \mathcal{U} \to \{0,1\}^*$ is called uniquely decodable if c^* is injective, *i.e.* if $u_1, \dots, u_n \neq v_1 \dots, v_k$, then $c(u_1)\dots c(u_n) \neq c(v_1)\dots c(v_k)$.

Example. With the same alphabet as before, we can have $c(a) = 0$, $c(b) = 10$, $c(c) = 110$, $c(d) = 1110$ and $c(e) = 1111$. This code is uniquely decodable and is called a prefix free code.

Definition 6. A code $c : \mathcal{U} \to \{0,1\}^*$ is called prefix free if $\forall u, v \in \mathcal{U}$ s.t $u \neq v$, $c(u)$ is not a prefix of $c(v)$.

Theorem 1. If a code c is prefix free (p.f.), then it is uniquely decodable (u.d.).

Proof. We prove the above theorem by contradiction. Consider there exists a code c which is p.f. but not u.d and we have $u_1, \dots, u_n, v_1, \dots, v_k \in \mathcal{U}$ such that $u_1, \dots, u_n \neq v_1, \dots, v_k$

and $c(u_1)\dots c(u_n) = c(v_1)\dots c(v_k)$. Traversing the code of the sequence of alphabets from the left, we immediately notice that if $\ell(u_1) > \ell(v_1)$ then $c(v_1)$ is a prefix of $c(u_1)$. Similarly if $\ell(u_1) < \ell(v_1)$, $c(u_1)$ is a prefix of $c(v_1)$. But since the code is prefix free, we arrive at a contradiction. □

Definition 7. Given a code $c : \mathcal{U} \to \{0,1\}^*$, we define the Kraft Sum

$$\mathcal{KS}(c) = \sum_{u \in \mathcal{U}} 2^{-\ell(u)}$$

where $\ell(u) = \text{length}(c(u))$. In case of confusion, we subscript the length by the code, i.e $\ell_c(u) \equiv \ell(u) = \text{length}(c(u))$

Example. Consider the code for the same alphabet as $c(a) = 0$, $c(b) = 1$, $c(c) = null$, $c(d) = 00$ and $c(e) = 010$. Then we have

$$\mathcal{KS}(c) = 2^{-1} + 2^{-1} + 2^0 + 2^{-2} + 2^{-2} = 2.375$$

Lemma 1. Suppose $c : \mathcal{U} \to \{0,1\}^*$ and $d : \mathcal{V} \to \{0,1\}^*$ are two codes, and denote $(c \times d) : \mathcal{U} \times \mathcal{V} \to \{0,1\}^*$ as $(c \times d)(u,v) = c(u)d(v)$. Then

$$\mathcal{KS}(c \times d) = \mathcal{KS}(c)\mathcal{KS}(d)$$

Proof. By definition, we have

$$\begin{aligned}\mathcal{KS}(c \times d) &= \sum_{u,v} 2^{-\ell_{c\times d}((u,v))} = \sum_{u,v} 2^{-[\ell_c(u)+\ell_d(v)]} \\ &= \sum_{u,v} 2^{-\ell_c(u)} \times 2^{-\ell_d(v)} = \sum_u 2^{-\ell_c(u)} \sum_v 2^{-\ell_d(v)} = \mathcal{KS}(c)\mathcal{KS}(d)\end{aligned}$$

□

Corollary. $\mathcal{KS}(c^n) = [\mathcal{KS}(c)]^n$

Theorem 2 (Kraft's Inequalities). Suppose $c : \mathcal{U} \to \{0,1\}^*$ is injective, then

$$\mathcal{KS}(c) \le \log_2(1 + |\mathcal{U}|)$$

Suppose c is uniquely decodable, then

$$\mathcal{KS}(c) \le 1$$

Suppose c is prefix free, then

$$\mathcal{KS}(c) \le 1$$

Proof. Suppose that c is injective, and let $|\mathcal{U}| = k$ and $\mathcal{U} = \{1, 2, \cdots, k\}$. To maximize the Kraft Sum, we put $c(1) = null$, $c(2) = 0$, $c(3) = 1$ and so on. Also, let $k = 1 + 2 + 4 + 8 + \cdots + 2^{m-1} + r$ where $0 \le r < 2^m$. Then, we can write

$$\mathcal{KS}(c) = 2 + 2 \times 2^{-1} + 2^2 \times 2^{-2} + \cdots + 2^{m-1} \times 2^{-(m-1)} + r \times 2^{-m} = m + r \cdot 2^{-m}$$

Now, consider $\log_2(1 + k) = \log_2(1 + 2^m - 1 + r) = \log_2(2^m + r)$. We can then write

$$\log_2(1 + k) \le \log_2(2^m(1 + r \cdot 2^{-m})) = m + \log_2(1 + r \cdot 2^{-m})$$

Notice that $r \cdot 2^{-m} \in [0, 1)$ and using the fact that $\log_2(1 + x) \ge x$ for $x \in [0, 1]$, we get

$$m + \log_2(1 + r2^{-m}) \ge m + r \cdot 2^{-m} = \mathcal{KS}(c)$$

Having proved for c being injective, we now show it for c being uniquely decodable. First observe that c is u.d. $\Leftrightarrow$ c^* is injective $\Rightarrow \forall n$, c^n is injective. Hence, we can write

$$[\mathcal{KS}(c)]^n = \mathcal{KS}(c^n) \leq \log_2(1 + k^n)$$

Now, notice that the LHS is exponentially growing in $\mathcal{KS}(c)$ while the RHS is linearly growing in $\mathcal{KS}(c)$. Since exponential growth cannot be dominated by linear growth, we notice that the term growing exponentially must be less than or equal to 1. Thus, we have $\mathcal{KS}(c) \leq 1$.
Finally, since every prefix free code is uniquely decodable, the last inequality follows immediately. □

Remark. Although the Kraft Sum for uniquely decodable codes is less than or equal to 1, the reverse doesn't hold true. Take the code $\mathcal{U} = \{a, b, c\}$ and the code c to be $c(a) = 0$, $c(b) = 00$ and $c(c) = 00$. It is clear that c is not uniquely decodable, but –

$$\mathcal{KS}(c) = 2^{-1} + 2^{-2} + 2^{-2} = 1 \leq 1$$

Chapter 2

Lecture 2

2.1 Prefix Free Codes

Example. Consider $\mathcal{U} = \{a, b, c\}$ and consider two codes

$$c: \quad c(a) = 0,\ c(b) = 10,\ c(c) = 11$$
$$\widetilde{c}: \quad \widetilde{c}(a) = 0,\ \widetilde{c}(b) = 01,\ \widetilde{c}(c) = 11$$

The first code is prefix free, and hence is uniquely decodable. The second is not prefix free, but we can still uniquely decode it! To do so, when we receive the entire sequence of bits, we reverse the sequence and apply decode it as if it was encoded using code c (note that $\widetilde{c}$ is the reverse of c).

A point to note is that in the second case, we might need to store an arbitrary length binary representation before we can decode the first letter and hence we require a Turing Machine! But in the first case, we need only a finite memory since we can decode the first letter when the required number of bits come in. For this reason, prefix free codes are also called **instantaneously decodable** codes. The above example also showed that although all prefix free codes are uniquely decodable, the reverse is not true. There exist uniquely decodable codes which are not prefix free.

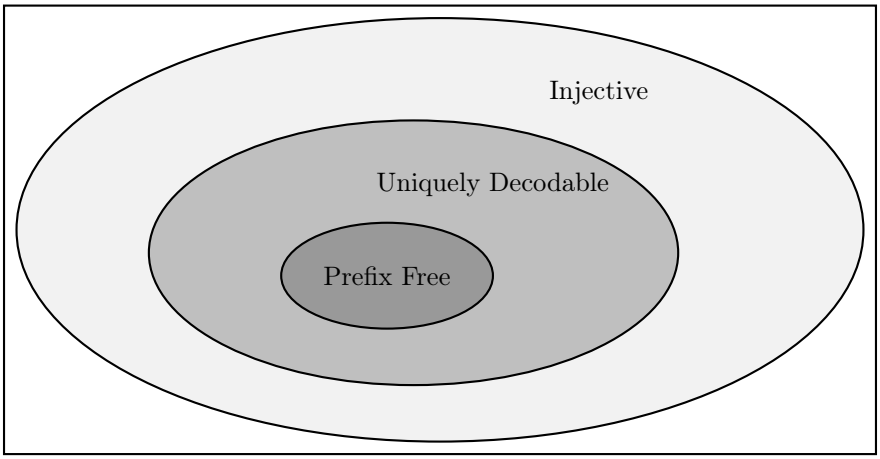


Figure 2.1: The box represents all codes possible, prefix free codes are a small category of it.

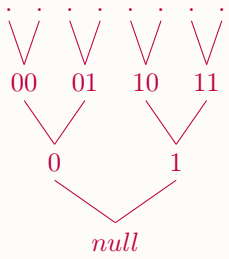
Remark. Although prefix free codes are a subset of uniquely decodable codes, there is no inherent advantage of using non prefix free codes which are uniquely decodable.

Theorem 3. Suppose $c : \mathcal{U} \to \{0,1\}^*$ for which $\mathcal{KS}(c) \leq 1$. Then $\exists$ a code $\widetilde{c} : \mathcal{U} \to \{0,1\}^*$ s.t.

- $\widetilde{c}$ is prefix free

- $\ell_{\widetilde{c}}(u) = \ell_c(u) \; \forall \; u$

Proof. Let $\mathcal{U} = \{1, 2, \ldots, k\}$ and let $\ell_1 = \ell_c(u_1), \ldots \ell_k = \ell_c(u_k)$ s.t. $\ell_1 \leq \ell_2 \leq \cdots \leq \ell_k$. It is also known that $\sum_i 2^{-\ell_i} \leq 1$.
Consider the complete binary tree on the right. It is constructed starting from *null* and each time you grow each level one bit at a time. Each parent is a prefix of its children. Now we consider the following algorithm:



1. Start with the complete binary tree of depth ℓ_k as shown in the figure. In the beginning, mark every node as available
2. For $i = 1, \ldots, k$:
   - Find a node available at depth ℓ_i (say it is n_i)
   - Set $\widetilde{c}(i) = n_i$
   - Mark the node n_i and all if its ascendants as unavailable

Although the algorithm does find such a $\widetilde{c}$, we need to rigorously show that we can always find a node at depth ℓ_i. To show this, notice that at the beginning of the algorithm, there were 2^{ℓ_i} nodes at depth ℓ_i. At every iteration until $i-1$, some nodes at depth ℓ_i get marked unavailable. Hence, at the i^{th} iteration, we have the number of available nodes as

$$\#\text{ available nodes} = 2^{\ell_i} - \sum_{j=1}^{i-1} 2^{\ell_i - \ell_j} = 2^{\ell_i}\left[1 - \sum_{j=1}^{i-1} 2^{-\ell_j}\right]$$

Since the Kraft Sum is at most 1, we have $\sum_{j=1}^{i-1} 2^{-\ell_j} < 1$ strictly. Hence the RHS of the above equation is always positive. But since it is just a difference of integers, the RHS must be at least 1. Hence, the algorithm executes successfully and we can construct such a $\widetilde{c}$. □

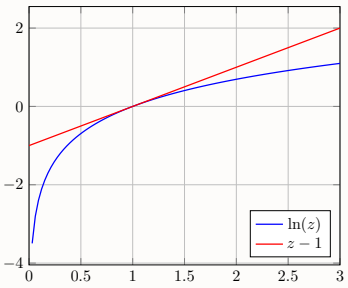
2.2 The Logarithm

Lemma 2. $\ln(z) \leq z - 1$ and equality only occurs at $z = 1$

Proof. We consider the finite Taylor's expansion of $\ln(z)$ around $z = 1$.
Recall that, the Taylor series expansion of $f(z)$ around z_0, for α between z and z_0 is given as

$$f(z) = f(z_0) + (z - z_0)f'(z_0) + \frac{1}{2}(z - z_0)^2 f''(\alpha)$$

$$\Rightarrow \ln(z) = \ln(1) + (z-1)\cdot 1 + \frac{1}{2}(z-1)^2 \cdot \left(-\frac{1}{2\alpha^2}\right)$$
$$= (z-1) - \frac{1}{2}\left(\frac{z-1}{\alpha}\right)^2 \leq z - 1$$



□

Corollary. Suppose $p_1, \ldots, p_k \geq 0$ are such that $\sum_i p_i = 1$. Suppose $z_1, \ldots, z_k > 0$, then we have

$$\sum_i^k p_i \ln(z_i) \leq \ln\left(\sum_{i=1}^k p_i z_i\right)$$

with equality only when $z_1 = z_2 = \cdots = z_k$.

Proof. Let $\mu = \sum_i p_i z_i$ and we need to show that

$$\sum_i p_i \ln(z_i) \leq \ln \mu \Rightarrow \sum_i p_i \left[\ln \left(\frac{z_i}{\mu} \right) \right] \leq 0$$

Using the previous lemma, we have

$$\sum_i p_i \left[\ln \left(\frac{z_i}{\mu} \right) \right] \leq \sum_i p_i \left[\frac{z_i}{\mu} - 1 \right] = \frac{1}{\mu} \sum_i p_i z_i - \sum p_i = 0$$

□

Definition 8. Suppose p is a probability distribution on $\mathcal{U}$ and suppose $q : \mathcal{U} \to [0, \infty)$. Then we define the Kullbeck-Liebler Divergence as

$$D(p \parallel q) \triangleq \sum_u p(u) \log \left(\frac{p(u)}{q(u)} \right)$$

Remark. If for any u, $p(u) = 0$, we consider the value of the term to be 0, and if for any u, $q(u) = 0$, then we say that $D(p \parallel q) \to \infty$.

Lemma 3. $D(p \parallel q) \geq -\log \left(\sum_u q(u) \right)$

Proof.

$$\begin{aligned} -D(p \parallel q) &= \sum_u p(u) \log \left(\frac{p(u)}{q(u)} \right) \\ &\leq \log \left[\sum_u p(u) \frac{q(u)}{p(u)} \right] = \log \left(\sum_u q(u) \right) \\ \Rightarrow D(p \parallel q) &\geq -\log \left(\sum_u q(u) \right) \end{aligned}$$

□

Remark. In particular when q is also a probability distribution on $\mathcal{U}$, we have $D(p \parallel q) \geq 0$.

Suppose we have an alphabet $\mathcal{U}$, a distribution p defined on $\mathcal{U}$ and a code $c : \mathcal{U} \to \{0,1\}^*$. If we choose a letter U randomly from the alphabet using the probability distribution, we can say that $\mathbb{E}[\ell(U)]$ denotes the expected number of bits representing a letter.

Example. Consider $\mathcal{U} = \{a, b, c, d, e\}$, let $p = (0.3, 0.1, 0.15, 0.2, 0.25)$ and $c = (00, 01, 10, 110, 111)$. Then we have $\mathbb{E}[\ell(u)] = 0.3 \times 2 + 0.1 \times 2 + 0.15 \times 2 + 0.2 \times 3 + 0.15 \times 3 = 2.45$

Theorem 4. For any code $c : \mathcal{U} \to \{0,1\}^*$ and a probability distribution p defined on $\mathcal{U}$, we have

$$\mathbb{E}[\ell(u)] \geq \sum_u p(u) \log_2 \frac{1}{p(u)} - \log_2[\mathcal{KS}(c)]$$

In particular, when c is uniquely decodable, we have

$$\mathbb{E}[\ell(u)] \geq \sum_u p(u) \log_2 \frac{1}{p(u)}$$

Proof. We define q on $\mathcal{U}$ as $q(u) = 2^{-\ell(u)} \Rightarrow \ell(u) = \log_2 \frac{1}{q(u)}$. Noticing the correspondence, we

can write $\mathbb{E}[\ell(u)] = \sum_u p(u) \log_2 \frac{1}{q(u)} = D(p \parallel q) + \sum_u p(u) \log \frac{1}{p(u)}$. Now, we have

$$\begin{aligned}
\mathbb{E}[\ell(u)] &= \sum_u p(u) \log \frac{1}{p(u)} + D(p \parallel q) \\
&\geq \sum_u p(u) \log \frac{1}{p(u)} - \log(\sum q(u)) \\
&= \sum_u p(u) \log \frac{1}{p(u)} - \log \mathcal{KS}(c)
\end{aligned}$$

□

Theorem 5. Given $\mathcal{U}, p$, $\exists$ a prefix free code c s.t.

$$\mathbb{E}[\ell(u)] \leq \sum p(u) \log \frac{1}{p(u)} + 1$$

Proof. We set $\ell(u) \triangleq \lceil \log_2 \frac{1}{p(u)} \rceil$. Due to the ceil function, it is clear that $\sum_u 2^{-\ell(u)} \leq 1$ (it was an equality without the ceil). We have

$$\begin{aligned}
\ell(u) &\leq \log_2 \frac{1}{p(u)} + 1 \\
\Rightarrow \mathbb{E}[\ell(u)] &\leq \sum_u p(u) \log \frac{1}{p(u)} + 1
\end{aligned}$$

□

Chapter 3

Lecture 3

3.1 Entropy

It is quite possible that the information source produces more than 1 letter at a time, i.e the source output is a sequence $u_1, u_2, \ldots$ of letters. We can also consider the code $c_n : \mathcal{U}^n \to \{0,1\}^*$.

Example. We can take $\mathcal{U} = \{a, b, c\}$ and have c_2 as $c(aa) = 0$, $c(ab) = 10$, $c(ac) = 110$ etc.

We can extend the notion from previous lecture for code c_n as follows – if $c_n : \mathcal{U}^n \to \{0,1\}^*$ is uniquely decodable, then

$$\mathbb{E}[\ell(u_1 \ldots u_n)] \geq \sum_{u_1,\ldots,u_n} p(u_1, \ldots, u_n) \log_2 \frac{1}{p(u_1, \ldots, u_n)}$$

Moreover, there exists a prefix free code $\widetilde{c}_n$ also s.t.

$$\mathbb{E}[\ell_{\widetilde{c}}(u_1 \ldots u_n)] \leq \sum_{u_1,\ldots,u_n} p(u_1, \ldots, u_n) \log_2 \frac{1}{p(u_1, \ldots, u_n)} + 1$$

A more interesting quantity to look at is the number of bits per letter. Hence, we have

$$\frac{1}{n}\mathbb{E}[\ell(u_1 \ldots u_n)] \geq \frac{1}{n} \sum_{u_1,\ldots,u_n} p(u_1, \ldots, u_n) \log_2 \frac{1}{p(u_1, \ldots, u_n)}$$

$$\frac{1}{n}\mathbb{E}[\ell_{\widetilde{c}}(u_1 \ldots u_n)] \leq \frac{1}{n} \sum_{u_1,\ldots,u_n} p(u_1, \ldots, u_n) \log_2 \frac{1}{p(u_1, \ldots, u_n)} + \frac{1}{n}$$

Notice that as n gets large, the two bounds are extremely close.

Definition 9. When $U \in \mathcal{U}$ is a random variable, we define its entropy as

$$H(U) = \sum_u p(u) \log \frac{1}{p(u)}$$

where $p(u) = \Pr[U = u]$.

Thus, in terms of entropy, we can state that for any uniquely decodable code $c : \mathcal{U} \to \{0,1\}^*$ and for any random variable $U \in \mathcal{U}$,

$$\mathbb{E}[\ell(U)] \geq H(U) \text{ and } \exists \text{ prefix free } \widetilde{c} \text{ s.t. } \mathbb{E}[\ell_{\widetilde{c}}(U)] \leq H(U) + 1$$

Again, replacing U $U_1, \ldots U_n$, we get for any uniquely decodable $c_n : \mathcal{U}^n \to \{0,1\}^*$ such that

$$\frac{1}{n}\mathbb{E}[\ell(u_1 \ldots u_n)] \geq \frac{1}{n} H(u_1, \ldots, u_n)$$

Moreover there exists a prefix free code $\widetilde{c}_n$ s.t.

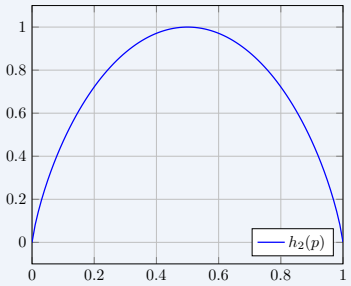
$$\frac{1}{n}\mathbb{E}[\ell_{\widetilde{c}}(u_1 \ldots u_n)] \leq \frac{1}{n} H(u_1, \ldots, u_n) + \frac{1}{n}$$

Example. Consider $\mathcal{U} = \{a, b\}$ with $p(a) = p$ and $p(b) = 1 - p$.

The entropy for the above is given as

$$H(U) = p \log \frac{1}{p} + (1-p) \log \frac{1}{p}$$

The above quantity is defined as the binary entropy function $h_2(p)$. It has a maxima at $p = \frac{1}{2}$ and $h_2(0.5) = 1$.



3.2 The Optimization Problem and Optimal Codes

Consider the following - say we have an alphabet $\mathcal{U}$ and a probability distribution p defined on $\mathcal{U}$ as input. At the output, we want the expected length of the codes to be minimum. Thus the output is $c : \mathcal{U} \to \{0,1\}^*$ s.t. c is prefix free and s.t. $\sum_u p(u)\ell(u)$ is as small as possible. We can frame this problem as

$$\min \left\{ \sum_{u \in \mathcal{U}} p(u)\ell(u) \; ; \; \sum_u 2^{-\ell(u)} \leq 1 \right\} \text{ and } \ell(u) \in \mathbb{N}$$

The integer constraint makes the problem much tougher to solve. If that constraint is removed, the trivial solution is $\ell(u) = -\log p(u)$.

Example. We can see another problem where integer constraint increases complexity. Consider the problem - given $w_1, \ldots, w_k \geq 0$, and let $w = \sum w_i$. Find $S \subset \{1, \ldots, k\}$ s.t.

$$\min_S \left| \sum_{i \in S} w_i - \sum_{i \notin S} w_i \right| \equiv \min_S \left| \sum_{i \in S} w_i - \frac{w}{2} \right| \equiv \min_{x_1, \ldots, x_k \in \{0,1\}} \left| \sum_{w_i x_i} - \frac{w}{2} \right|$$

If we remove the integer constraint, we can easily see that $x_i = \frac{1}{2}$ solves the problem.

We have the following properties of optimal codes:

1. If $p(u) < p(v)$ then $\ell(u) \geq \ell(v)$. The proof is trivial, because if that's not the case, we can swap $\ell(v)$ and $\ell(u)$ and improve the expected length of the code by $[p(v) - p(u)][\ell(v) - \ell(u)]$.

   Corollary. One of the longest codewords must belong to one of the least probable letters.

2. There have to be at least two longest codewords (assuming $|\mathcal{U}| > 1$). Further, the two longest codewords must be siblings. This is also immediate - if there is a single longest codeword, it can be moved down the binary tree, and the codeword will get shorter.

   Corollary. There is an optimal code in which the two least probable letters are assigned sibling codewords

   Proof. Consider a code c in which the two least probable letters are not siblings. Just swap one of the codewords to make them siblings. □

In an optimal code for $(\mathcal{U}, p)$, we know that the two least likely letters say k and $k-1$ must have codewords $[\ldots]0$ and $[\ldots]1$. Let the common prefix $[\ldots]$ have length $\widetilde{\ell}$. Then we can write

$$\begin{aligned} \mathbb{E}[\ell(U)] &= p(1)\ell(1) + p(2)\ell(2) + \cdots + p(k-2)\ell(k-2) + [p(k) + p(k-1)](1 + \widetilde{\ell}) \\ &= p(1)\ell(1) + \cdots + [p(k) + p(k-1)]\widetilde{\ell} + [p(k) + p(k-1)] \end{aligned}$$

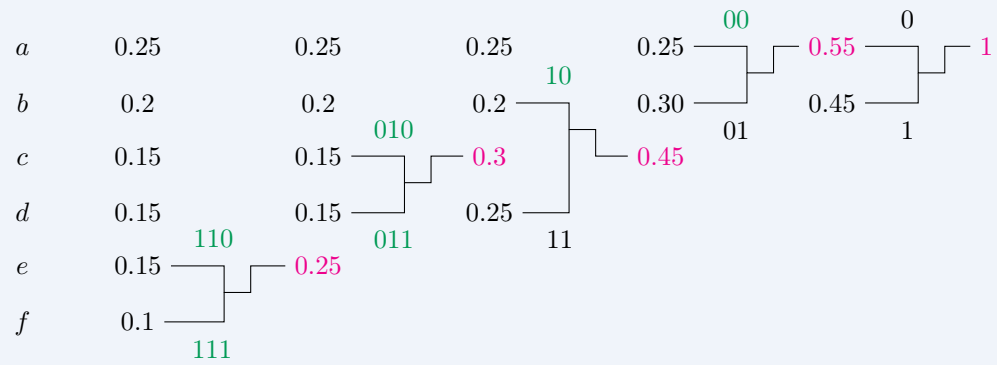
Thus our optimization problem is as follows -

$$\min_{\ell(1),\ldots,\ell(k)} \sum_{i=1}^{k} p(i)\ell(i) \text{ s.t. } \sum_i 2^{-\ell(i)} \leq 1$$

$$\equiv \min_{\ell(1),\cdots,\ell(k-2),\widetilde{\ell}} \sum_{i=1}^{k-2} p(i)\ell(i) + [p(k-1)+p(k)]\widetilde{\ell} + [p(k)+p(k-1)] \text{ s.t. } \left(\sum_{i=1}^{k-2} 2^{-\ell(i)}\right) + +2^{-\widetilde{\ell}-1} + 2^{-\widetilde{\ell}-1} \leq 1$$

The constraint is same as $2^{-\ell(1)}+2^{-\ell(2)}+\cdots+2^{-\ell(k-2)}+2^{-\widetilde{\ell}} \leq 1$. Thus, we reduced the optimization problem to $k-1$ variables, i.e. $(\mathcal{U}=\{1,\ldots,k\}, p=(p(1),\ldots,p(k)) \to (\mathcal{V}=\{1,\ldots,k-1\}, q=(p(1),\cdots,p(k-2),p(k-1)+p(k))$. We can keep on repeating this procedure and bring the problem down even from the $(\mathcal{V},q)$ code.

Example. We show the Huffman procedure on the following code: $\mathcal{U}=\{a,b,\ldots,f\}$ and $p=(0.25,0.2,0.15,0.15,0.15,0.1)$.



Following the procedure, we get the codes (marked in green for the corresponding letter in the alphabet). The nodes marked in magenta are the fake nodes, and can be used to calculate the expected length of the code. The expected length of the code is given by

$$\mathbb{E}[\ell(u)] = \sum (\text{fake node values}) = 0.25 + 0.3 + 0.45 + 0.55 + 1 = 2.55$$

Chapter 4

Lecture 4

4.1 Entropy, Conditional Entropy and Mutual Information

Recall that for a random variable $U \in \mathcal{U}$, we define

$$H(U) \triangleq \sum_{u \in \mathcal{U}} p(u) \log_2 \frac{1}{p(u)} = \mathbb{E}\left[\log_2 \frac{1}{p(u)}\right] \text{ where } p(u) = \Pr[U = u] = p_U(u)$$

If U and V are random variables with $p(u, v) = \Pr[U = u, V = v] = p_{UV}(uv)$ then

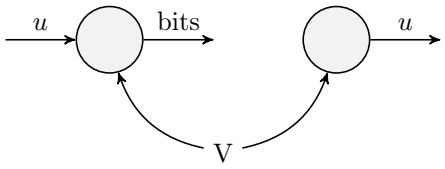
$$H(UV) \triangleq \sum_{u,v} p(uv) \log_2 \frac{1}{p(uv)} = \mathbb{E}\left[\log_2 \frac{1}{p(uv)}\right]$$

and in general for $U_1, \ldots, U_n$, we have

$$H(U_1, \ldots, U_n) = \sum_{u_1, \ldots, u_n} p(u_1, \ldots, u_n) \log_2 \frac{1}{p(u_1, \ldots, u_n)}$$

Remark. From now, we denote $u^n \triangleq (u_1 \ldots u_n)$ as a sequence of letters and $U^n = (U_1 \ldots U_n)$ as a random vector.

Now consider the case where we have some information between the encoder and decoder, denoted by the random variable V. For each possible $v \in V$, we can design a source code for $\mathcal{U}$. Recall that



$p(u|v) = \Pr[U = u|V = v] = \frac{p_{UV}(uv)}{p_V(v)}$. For a given $v \in V$, we see that

$$H(U|V = v) = \sum_{u \in \mathcal{U}} p(u|v) \log \frac{1}{p(u|v)}$$

and this denotes the minimum number of bits per letter to describe $\mathcal{U}$. The overall expected bits per letter is then

$$H(U|V) = \sum_{v \in V} p_V(v) H(U|V = v)$$

Remark. Possible cognitive dissonance: Consider a function $f : U \to V$. We have that $f(U)$ is a random variable, but $H(U)$ is a number. Similar to $\mathbb{E}[U]$, $H(U)$ is a number. Now, consider $\mathbb{E}[U|V]$ - this is a random variable. But still $H(U|V)$ is a number.

Now, the conditional entropy of U given V is given as

$$H(U|V) = \sum_{u,v} p(uv) \log_2 \frac{1}{p(u|v)} = \mathbb{E}\left[\log_2 \frac{1}{p(u|v)}\right]$$

Remark.

$$\mathbb{E}\left[\log_2 \frac{p_U(u)}{p_V(u)}\right] = \sum_{u \in \mathcal{U}} p_U(u) \frac{p_U(u)}{p_V(u)} = D(p_U \parallel p_V)$$

$$EE\left[\frac{p_U(u)}{p_V(v)}\right] = H(V) - H(U)$$

Theorem 6 (Chain Rule).

$$H(UV) = H(V) + H(U|V) = H(U) + H(V|U)$$

$$H(U^n) = H(U_1) + H(U_2|U_1) + \cdots + H(U_n|U^{n-1})$$

Proof.

$$\begin{aligned} p_{UV}(uv) &= p_V(v) p_{U|V}(u|v) \\ \log_2 p_{UV}(uv) &= \log_2 p_V(v) + \log_2 p_{U|V}(u|v) \\ \Rightarrow H(UV) &= H(V) + H(U|V) \end{aligned}$$

For the general case,

$$\begin{aligned} p_{(U_1,\ldots,U_n)}(u_1,\ldots,u_n) &= p_{U_1}(u_1) p_{U_2|U_1}(u_2|u_1) \ldots p_{U_n|U_1\ldots U_{n-1}}(u_n|u_1\ldots u_{n-1}) \\ &= \prod_{i=1}^{n} p_{U_i|U^{-1}}(u_i|u^{i-1}) \\ \Rightarrow \log_2 p(u^n) &= \sum_{o} \log_2 p(u_i|u^{i-1}) \\ \Rightarrow H(U^n) &= H(U_1) + H(U_2|U_1) + \cdots + H(U_n|U^{n-1}) \end{aligned}$$

□

Based on some source coding representation, we denote the value of V in representing U as $H(U) - H(U|V)$. We notice that

$$H(U) - H(U|V) = H(U) + H(V) - H(UV) = H(V) - H(V|U)$$

is symmetric and U and V.

Definition 10. We define the Mutual Information between two random variables U and V as

$$I(U;V) = H(U) + H(V) - H(UV) = H(U) - H(U|V) = H(V) - H(V|U) = I(V;U)$$

Similarly, we define the Conditional Mutual Information of the two given W as

$$\begin{aligned} I(U;V|W) &= H(U|W) + H(V|W) - H(UV|W) \\ &= H(U|W) - H(U|VW) = H(V|W) - H(V|UW) \\ &= H(UW) + H(VW) - H(UVW) - H(W) \end{aligned}$$

Theorem 7. $I(U;V|W) \geq 0$ and equality holds if and only if $\forall\, u, v, w : p(uv|w) = p(u|w)p(v|w)$ whenever $p(w) > 0$.

Proof.

$$\begin{aligned}
I(U;V|W) &= H(U|W) + H(V|W) - H(UV|W) \\
&= \mathbb{E}\left[\log_2 \frac{p(uv|w)}{p(u|w)p(v|w)}\right] \\
&= \sum_w \sum_u \sum_v p(uvw) \log_2 \frac{p(uv|w)}{p(u|w)p(v|w)} \\
&= \sum_w p(w) \sum_{u,v} p(uv|w) \log_2 \frac{p(uv|w)}{p(u|w)p(v|w)} \\
&= \sum_w p(w) D(p(uv|w) \parallel p(u|w)p(v|w)) \geq 0
\end{aligned}$$

For equality, $p(uv|w) = p(u|w)p(v|w)$, which means that U and V are conditional independent given W or $U - W - V$ form a markov chain. □

Corollary. $I(U;V) \geq 0$ and equlaity holds if and only if U and V are independent.

Corollary. $H(U|V) \leq H(U)$ and equality holds if and only if U and V are independent. This means that conditioning will never increase entropy. Similarly, $H(U|VW) \leq H(U|W)$ and equality holds if and only if $U - W - V$.

Theorem 8 (Chain rule for Mutual Information)**.**

$$I(U^n;V) = I(U_1;V) + I(U_2;V|U_1) + \cdots + I(U_n;V|U^{n-1})$$

$$I(U^n;V|W) = \sum_{i=1}^{n} I(U_i;V|U^{i-1}|W)]$$

Proof.

$$\begin{aligned}
I(U^n,V) &= H(U^n) - H(U^n|V) \\
&= H(U_1) + H(U_2|U_1) + \cdots + H(U_n|U^{n-1}) - H(U_1|V) - H(U_2|U,V) - \dots \\
&= I(U_1;V) + I(U_2;V|U_1) + \cdots + I(U_n;V|U^{n-1})
\end{aligned}$$

On the similar lines, the second statement can be proven. □

Theorem 9 (Data Processing Inequality)**.** Suppose $U - W - V$, then $I(U;W) \geq I(U;V)$ and equality holds if and only if $U - V - W$, and $I(V;W) \geq I(U;V)$ and equality holds if and only if $V - U - W$.

Proof.

$$\begin{aligned}
I(U;VW) = I(U;V) + I(U;W|V) &= I(U;W) + I(U;V|W) \\
\Rightarrow I(U;V) + I(U;W|V) &= I(U;W) \text{ since } I(U;V|W) = 0 \\
\Rightarrow I(U;V) &\leq I(U;W)
\end{aligned}$$

□

Chapter 5

Lecture 5

5.1 Entropy Inequalities

Theorem 10. $H(U) \geq 0$ with equality if and only if $\exists\ u_0 \in \mathcal{U}$ s.t. $\Pr[U = u_0] = 1$, i.e U isn't random. Also, $H(U) \leq \log_2 |\mathcal{U}|$ with equality if and only if U follows a uniform distribution, i.e for any $u_0 \in \mathcal{U}$, $\Pr[U = u_0] = \frac{1}{|\mathcal{U}|}$

Proof. The first part is easy to prove, since $p(u) \geq 0$ and $\frac{1}{p(u)} \geq 1 \Rightarrow \log_2 \frac{1}{p(u)} \geq 0$. Thus, $H(U) \geq 0$.
Now, we claim that $H(U)$ is maximal if U is a uniform distribution on $\mathcal{U}$.
We prove this by contradiction - consider that U is not uniform, then $\exists\ (u_0, u_1) \in \mathcal{U}$ s.t. $p(u_0) < p(u_1)$. Now, consider a random variable $\widetilde{U}$ s.t.

$$p_{\widetilde{U}}(\widetilde{u}) = \begin{cases} \frac{1}{2}[p(u_0) + p(u_1)] & u = u_0 \\ \frac{1}{2}[p(u_0) + p(u_1)] & u = u_1 \\ p(u) & \text{else} \end{cases}$$

Now, note that since $p(u_0) < p(u_1)$, we have that $2p(u_0) < p(u_0) + p(u_1) < 2p(u_1)$. Thus,

$$\begin{aligned} H(\widetilde{U}) - H(U) &= (p(u_0) + p(u_1)) \log_2 \left(\frac{2}{p(u_0) + p(u_1)} \right) - p(u_0) \log \frac{1}{p(u_0)} - p(u_1) \log \frac{1}{p(u_1)} \\ &= p(u_0) \log_2 \frac{2p(u_0)}{p(u_0) + p(u_1)} + p(u_1) \log_2 \frac{2p(u_1)}{p(u_0) + p(u_1)} \\ &\geq p(u_0) \log_2 \frac{2p(u_0)}{2p(u_0)} + p(u_1) \log_2 \frac{2p(u_1)}{2p(u_0)} \\ &= p(u_1) \log_2 \frac{p(u_1)}{p(u_0)} > 0 \end{aligned}$$

Thus, we found a distribution which has a higher entropy than U, but this is a contradiction. Thus, the only maximizer of $H(U)$ is the uniform distribution. □

Theorem 11. Suppose U and V have the same alphabet $\mathcal{U}$ and let $\delta = \Pr[u \neq v]$. Then,

$$H(U|V) \leq h_2(\delta) + \delta \log_2(|\mathcal{U}| - 1)$$

Proof. Let's define W as the indicator variable $\mathbb{1}[U \neq V]$. Now,

$$\begin{aligned} H(UW|V) &= H(W|V) + H(U|WV) \\ &\leq H(W) + H(U|VWV) = h_2(\delta) + H(U|WV) \end{aligned}$$

Now, to show that $H(U|WV) \leq \delta \log(|\mathcal{U}| - 1)$ - notice that $H(U|WV = wv) = 0$ if $w = 0$ else

$H(U|WV = wv) \leq \log(|\mathcal{U}| - 1)$ if $w = 1$.

$$\begin{aligned} H(U|VW) &= \sum_{wv} p_{WV}(w, v) H(U|VW = vw) \\ &\leq \sum_{v} p(1, v) \log(|\mathcal{U}| - 1) \\ &= \Pr(w = 1) \log(|\mathcal{U}| - 1) = \delta \log(|\mathcal{U}| - 1) \end{aligned}$$

□

Example. Suppose $\mathcal{U} = \mathcal{V} = \mathcal{W}$ and $\Pr(U = 0) = \Pr(U = 1) = 0.5$ and $\Pr(V = 0) = \Pr(V = 1) = 0.5$. Consider

$$W = U \oplus V = \begin{cases} 0 & \text{if } uv = 00 \text{ or } 11 \\ 1 & \text{else} \end{cases}$$

Then,

1. $H(U) = H(V) = H(W) = 1$
2. $H(UV) = H(U) + H(V) - I(U; V) = 1 + 1 - 0 = 2$
3. For $H(UW)$, consider when $U = 0$ then $W = 0 \oplus V = V$ and $\Pr(W) = (0.5, 0.5)$ and if $U = 1$ then $W = 1 \oplus V = \neg V$ and $\Pr(W)$ is the same as before. Thus, the distribution of W doesn't change when conditioned on U and $\Pr(W = w|U = 0) = \Pr(W = w|U = 1) \Rightarrow U$ and W are independent and $H(UW) = H(VW) = 2$
4. $H(U|V) = H(UV) - H(V) = 2 - 1 = 1$
5. $H(U|VW) = 0$ since given W and V, U is deterministic
6. $I(U; W) = I(V; W) = I(U; V) = 0$
7. $I(U; VW) = I(U; W) + I(U; V|W) = H(U|W) + H(V|W) - H(UV|W) = 1 + 1 - 1 = 1$

Theorem 12. $2H(UVW) \leq H(UV) + H(VW) + H(WU)$

Proof. Rewrite the equation as

$$H(UVW) - H(UV) + H(UVW) - H(VW) \overset{?}{\leq} H(WU)$$

The first two terms in the RHS collapse to $H(W|UV)$ and the next two collapse to $H(U|WV)$. We know that conditioning decreases entropy, and thus $H(W|UV) \leq H(W)$ and $H(U|WV) \leq H(U|W)$. Thus, we have

$$LHS \leq H(W) + H(U|W) = H(UW)$$

□

Theorem 13. Consider n points in a 3-dimensional space. Consider shining light from each axis (u, v, w) and let the number of corresponding projections of points on the planes be n_{uw}, n_{vw} and n_{uv}. Then,

$$n_{uv} n_{vw} n_{uw} \geq n^2$$

Proof. Let the random variables (UVW) be uniformly distributed on the n points. Thus, $H(UVW) = \log_2 n$. Now, $H(UV) \leq \log n_{uv}$ since, UV cannot have entropy more than what it would have if UV was uniformly distributed. Thus by the above inequality,

$$2 \log_2 n \leq \log_2 n_{uv} \log_2 n_{vw} \log_2 n_{wu} \Rightarrow n^2 \leq n_{uv} n_{vw} n_{uw}$$

□

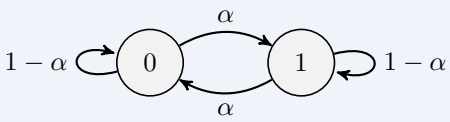
Definition 11. Suppose $U_1, U_2, \dots$ is a stochastic process. We define its *Entropy Rate* as

$$\mathcal{H}(U^\infty) = \lim_{n\to\infty} \frac{1}{n} H(U^n) \text{ if it exists}$$

Example. Suppose $U_1, U_2, \dots$ are i.i.d, then

$$\mathcal{H} = \lim_{n\to\infty} \frac{1}{n} H(U^n) = \lim_{n\to\infty} \frac{1}{n} [H(U_1) + H(U_2|U_1) + \cdots + H(U_n|U^{n-1}) = H(U_1)$$

Example. Consider the markov chain given,



Consider the initial (step 1) probability U_1 of being in state 0 and state 1 to be uniform, i.e $U_1 = 0$ w.p 0.5 and 1 w.p 0.5. Now, U_2 is given by

$$U_2 = \begin{cases} 0 & \frac{1}{2}(1-\alpha) + \frac{\alpha}{2} = \frac{1}{2} \\ 1 & \frac{\alpha}{2} + \frac{1}{2}(1-\alpha) = \frac{1}{2} \end{cases}$$

We see that being in any state is uniform, i.e $U_1 = U_2 = U_3 = U_4 = \dots$.
Now, $H(U_2|U_1 = 0) = h_2(\alpha)$ and $H(U_2|U_1 = 1) = h_2(\alpha) \Rightarrow H(U_2|U_1) = h_2(\alpha)$. Similarly, we can state that $H(U_{n+1}|U_n) = h_2(\alpha)$.

$H(U_n) = H(U_1) + H(U_2|U_1) + \cdots + H(U_n|U^{n-1})$
$H(U_n) = H(U_1) + H(U_2|U_1) + \cdots + H(U_n|U_{n-1})$ (Markov Process: previous state dependency)

$$\Rightarrow \mathcal{H} = \lim_{n\to\infty} \frac{1}{n} H(U_1) + \frac{(n-1)}{n} h_2(\alpha) = h_2(\alpha)$$

Chapter 6

Lecture 6

6.1 Stationary Processes and Entropy

Definition 12. A process $U_1, U_2, \ldots$ is a stationary process if

$$\forall\, k \geq 0, \forall\, n, \forall; (u_1, \ldots, u_n) \;:$$
$$\Pr[((U_{k+1}, \ldots U_{k+n}) = (u_1, \ldots, u_n)] = \Pr[(U_1, \ldots, U_n) = (u_1, \ldots, u_n)]$$

Lemma 4 (Cesaro's Lemma). If $x_n \to x$ (i.e if the sequence x_n converges with the limit x), then the sequence $y_n \triangleq \frac{\sum_{i=1}^{n} x_i}{n} \to x$

Proof. $x_n \to x \equiv \forall\, \varepsilon > 0\ \exists\, n(\varepsilon) \mid \forall\, n \geq n(\varepsilon) : |x_n - x| < \varepsilon$

$$\begin{aligned}
|y_n - x| &= \left| \frac{(x_1 - x) + (x_2 - x) + \cdots + (x_n - x)}{n} \right| \\
&\leq \frac{|x_1 - x| + |x_2 - x| + \cdots + |x_n - x|}{n} \\
&< \frac{\sum_{i=1}^{n(\varepsilon)-1} |x_i - x| + n\varepsilon}{n} \\
&= \varepsilon + \frac{1}{n} \sum_{i=1}^{n(\varepsilon)-1} |x_i - x| \leq 2\varepsilon \text{ if } n \text{ is large enough}
\end{aligned}$$

Thus, $y_n \to x$ □

Theorem 14. Suppose $U_1, \ldots, U_n$ is a stationary process, then

- $\lim_n \frac{1}{n} H(U^n)$ exists and is equal to $\lim_n H(U_n | U^{n-1})$
- Moreover, both $\frac{1}{n} H(U^n)$ and $H(U_n | U^{n-1})$ are monotone non-increasing sequences

Proof. Let $K_n = H(U_n | U^{n-1})$ and $H_n = \frac{1}{n} H(U^n)$. By the chain rule, we can see that

$$H_n = \frac{1}{n}[K_1 + \cdots + K_n]$$

Also, since conditioning decreases entropy, we have that

$$\begin{aligned}
K_{n+1} = H(U_{n+1} | U^n) &= H(U_{n+1} | U_n, \ldots, U_2, U_1) \\
&\leq H(U_{n+1} | U_n, \ldots, U_2)
\end{aligned}$$

Due to stationarity - we have $H(U_{n+1} | U_n, \ldots, U_2) = H(U_n | U_{n-1} \ldots U_1) \Rightarrow K_{n+1} \leq K_n$

Now,

$$\begin{aligned} n(n+1)(H_n - H_{n+1}) &= (n+1)(K_1 + \cdots + K_n) - n(K_1 + \cdots K_{n+1}) \\ &= (K_1 + \cdots + K_n) - nK_{n+1} \\ &= (K_1 - K_{n+1}) + \cdots + (K_n - K_{n+1}) \\ &\geq 0 \end{aligned}$$

Thus, $H_n \geq H_{n+1}$, and since both are monontonically non-increasing, the limit exists. Further, using Cesaro's Lemma, K_n and H_n have the same limit □

Theorem 15. Suppose $U_1, U_2, \ldots, U_n$ is stationary with entropy rate $\mathcal{H}$ and let $\varepsilon > 0$. Then, there exists a data compression method that uses less than $H + \varepsilon$ bits per letter on the average to describe the stationary process.

Proof. First, we pick n large enough so that $\frac{H(U_1,\ldots,U_n)+1}{n} < \mathcal{H} + \varepsilon$. This is possible since $\frac{1}{n}[H(U_1, \ldots, U_n) + 1] \to \mathcal{H}$. Now, we design a Huffmann code C_n for $(U_1, \ldots, U_n)$ - $C_n : \mathcal{U}^n \to \{0,1\}^*$ and use it as follows -

$$\underbrace{U_1, \ldots, U_n}_{\substack{\downarrow C_n \\ \text{bits}}}, \underbrace{U_{n+1}, \ldots, U_{2n}}_{\substack{\downarrow C_n \\ \text{bits}}}, \underbrace{U_{2n+1}, \ldots, U_{3n}}_{\substack{\downarrow C_n \\ \text{bits}}} \cdots$$

Since C_n is Huffman, we have $\mathbb{E}[\#bits] \leq H(U_1, \ldots, U_n) + 1$

$$\frac{\mathbb{E}[\#bits]}{n} \leq \frac{H(U_1, \ldots, U_n) + 1}{n} < \mathcal{H} + \varepsilon$$

Since each of the blocks uses less than $\mathcal{H} + \varepsilon$ bits, the entire scheme uses less than $\mathcal{H} + \varepsilon$ bits. □

6.2 Compression of IID Sources and Typicality Inequalities

Definition 13. Suppose $U_1, U_2, \ldots$ are IID random variables. Given a distribution p on $\mathcal{U}$, n and $\varepsilon > 0$ we define the set of *typical* sequences as

$$T(n, p, \varepsilon) = \left\{ u^n \in \mathcal{U}^n, \forall_{u \in U}\ p(u)(1-\varepsilon) \leq \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{u_i = u\} \leq p(u)(1+\varepsilon) \right\}$$

Suppose $U_1 \sim q$ and $u^n \in T(n, p, \varepsilon)$. Let us examine $\Pr[U^n = u^n]$

$$\begin{aligned} \Pr[U^n = u^n] &= \prod_{i=1}^{n} \Pr[U_i = u_i] \quad (\because \text{ independence}) \\ &= \prod_{i=1}^{n} q(u_i) = \prod_{u \in \mathcal{U}} q(u)^{\sum_{i=1}^{n} \mathbb{1}(u_i = u)} \end{aligned}$$

Note that $\sum_{i=1}^{n} \mathbb{1}(U_i = u) \approx np(u)(1 \pm \varepsilon)$.

$$\Rightarrow \prod_{u \in \mathcal{U}} q(u)^{np(u)(1+\varepsilon)} \leq \Pr[U^n = u^n] \leq \prod_{u \in \mathcal{U}} q(u)^{np(u)(1-\varepsilon)}$$

Remark.

$$\prod_{u \in \mathcal{U}} q(u)^{np(u)(1-\varepsilon)} = 2^{-(1-\varepsilon)n \sum_u p(u) \log_2 \frac{1}{q(u)}}$$

$$\sum_{u \in Ucal} p(u) \log_2 \frac{1}{q(u)} = H(p) + D(p \parallel q)$$

Theorem 16. Suppose $U_1, U_2, \ldots$ are i.i.d and $U_i \sim q$, and $u^n \in T(n,p,\varepsilon)$, then

$$2^{n(1+\varepsilon)[H(p)+D(p \parallel q)]} \leq \Pr[U^n = u^n] \leq 2^{-n(1-\varepsilon)[H(p)+D(p \parallel q)]}$$

We can also write this as $\Pr[U^n = u^n] \dot{\sim} 2^{-n[H(p)+D(p \parallel q)]}$

Proof. Follows from the explanation above along with the remark. □

Corollary. Taking $p = q$ in the above theorem, we can write

$$\Pr[U^n = u^n] \dot{\sim} 2^{-nH(U)}$$

Theorem 17.

$$|T(n,p,\varepsilon)| \leq 2^{n(1+\varepsilon)H(p)}$$

Proof. Let $U_1, U_2, \ldots$ be i.i.d and $\sim p$. Then

$$\begin{aligned}
\Pr[U^n \in T(n,p,\varepsilon)] = \sum_{u^n \in T(n,p,\varepsilon)} \Pr[U^n = u^n] &\leq 1 \\
\Rightarrow \sum_{u^n \in T(n,p,\varepsilon)} 2^{-(1+\varepsilon)nH(p)} &\leq 1 \\
\Rightarrow |T| 2^{-n(1+\varepsilon)H(p)} &\leq 1 \\
\Rightarrow |T| &\leq 2^{n(1+\varepsilon)H(p)}
\end{aligned}$$

□

Corollary. Suppose $U_1, U_2, \ldots, U_n$ are i.i.d and $\sim q$, then

$$\Pr[U^n \in T(n,p,\varepsilon) \leq 2^{-n[D(p \parallel q)-\varepsilon[H(p)+D(p \parallel q)]}$$

Proof.

$$\begin{aligned}
\Pr[U^n \in T] = \sum_{u^n \in T} \Pr[U^n = u^n] &\leq \sum_{u^n \in T} 2^{-n(1-\varepsilon)[H(p)+D(p \parallel q)]} \\
&= |T| 2^{-n(1-\varepsilon)}[H(p) + D(p \parallel q)] \\
&\leq 2^{n(1+\varepsilon)H(p) - n(1-\varepsilon)H(p) - n(1-\varepsilon)D(p \parallel q)} \\
&= 2^{-n[D(p \parallel q) - \varepsilon[2H(p)+D(p \parallel q)]} \\
&\approx 2^{-nD(p \parallel q)}
\end{aligned}$$

□

Theorem 18. Suppose $U_1, \ldots$ are i.i.d and $\sim p$, $\varepsilon > 0$ then $\Pr[U^n \in T(n,p,\varepsilon)] \to 1$ as n increases, and in particular for n large enough we have $\Pr[U^n \in T(n,p,\varepsilon)] \geq 1 - \varepsilon$

Proof. We consider the probability $\Pr[U^n \notin T(n,p,\varepsilon)]$

$$\begin{aligned}
\Pr[U^n \notin T(n,p,\varepsilon)] &= \Pr\left[\exists\, u \in \mathcal{U} : \left|\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{U_i = u\} - p(u)\right| > \varepsilon p(u)\right] \\
&\leq \sum_{u \in \mathcal{U}} \Pr\left[\left|\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{U_i = u\} - p(u)\right| > \varepsilon p(u)\right]
\end{aligned}$$

We examine the event $\left|\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{U_i = u\} - p(u)\right| > \varepsilon p(u)$ and denote $X_i = \mathbb{1}\{U_i = u\}$, and we see that X_i are independent since U_i are independent, and $\mathbb{E}[X_i] = p(u)$. Thus, by the law

of large numbers,

$$\Pr\left[\left|\frac{1}{n}\sum_{i=1}^{n}\mathbb{1}\{U_i = u\} - p(u)\right| > \varepsilon p(u)\right] \to 0 \text{ as } n \text{ gets large}$$

Thus, for $n > n_0(u, \varepsilon)$

$$\Pr\left[\left|\frac{1}{n}\sum_{i=1}^{n}\mathbb{1}\{U_i = u\} - p(u)\right| > \varepsilon p(u)\right] \leq \frac{\varepsilon}{|\mathcal{U}|}$$

So taking $n > \max\{n_0(u, \varepsilon), u \in \mathcal{U}\}$, we have that

$$\sum_{u\in\mathcal{U}} \Pr\left(\left|\frac{1}{n}\sum_{i=1}^{n}\mathbb{1}\{U_i = u\} - p(u)\right| > \varepsilon p(u)\right) < \varepsilon \Rightarrow \Pr[U^n \notin T(n, p, \varepsilon)] < \varepsilon$$

□

Chapter 7

Lecture 7

7.1 Typicality Continued

Example. Consider $\mathcal{U} = \{a, , b, c\}$ with probabilities $(0.5, 0.25, 0.25)$. We can show that $H(p) = 1.5$. Then, $\mathcal{U}^n$ has cardinality 3^n and $T(n, p, \varepsilon)$ has cardinality of approximately $2^{nH(p)} = 2^{1.5n} \approx 2.8^n$. As n increases, the ratio of the cardinalities decreases, and if U^n is i.i.d and $\sim p$, then it falls in the decreasing region w.p 1

Corollary. $|T(n, p, \varepsilon)| > (1 - \varepsilon)2^{nH(p)(1-\varepsilon)}$ for n large enough

Proof.

$$\begin{aligned}(1-\varepsilon) < \Pr[U^n \in T] &= \sum_{u^n \in T} \Pr[U^n = u^n] \\ &\leq \sum_{u^n \in T} 2^{-nH(p)(1-\varepsilon)} = |T| 2^{-nH(p)(1-\varepsilon)} \\ \Rightarrow |T| &> (1-\varepsilon)2^{nH(p)(1-\varepsilon)}\end{aligned}$$

□

Corollary. If $U_1, U_2, \ldots$ are i.i.d and $\sim q$, for n large enough we have

$$\Pr[U^n \in T(n, p, \varepsilon)] \geq (1-\varepsilon)2^{-n[D(p \parallel q) + \varepsilon(2H(p) + D(D(p \parallel q))]}$$

Proof.

$$\begin{aligned}\Pr[U^n \in T(n, p, \varepsilon)] = \sum_{u^n \in T} \Pr[U^n = u^n] &\geq \sum_{U^n \in T} 2^{-n(1+\varepsilon)[D(p \parallel q) + H(p)]} \\ &= |T| 2^{-n(1+\varepsilon)[D(p \parallel q) + H(p)]} \\ &\geq (1-\varepsilon)2^{nH(p)(1-\varepsilon)} 2^{-n(1+\varepsilon)[D(p \parallel q) + H(p)]} \\ &= 2^{-n[D(p \parallel q) + \varepsilon(2H(p) + D(D(p \parallel q))]}\end{aligned}$$

□

Suppose we replace $\mathcal{U}$ by $\mathcal{U} \times \mathcal{V}$, $Uby(U, V)$ and $p_U by p_{UV}$, we see that

$$T(n, p_{UV}, \varepsilon) = \{(u_1, v_1), \ldots, (u_n, v_n)\} \equiv (u^n, v^n) \; \forall \; u \in \mathcal{U}, v \in \mathcal{V}$$

Also, the bounds on the indicator extends as

$$n(1-\varepsilon)p_{UV}(u, v) \leq \sum \mathbb{1}\{U_i = u, V_i = v\} \leq n(1+\varepsilon)p_{UV}(u, v)$$

Lemma 5.

$$(u^n, v^n) \in T(n, p_{UV}, \varepsilon) \Rightarrow u^n \in T(n, p_U, \varepsilon)$$

Proof. Observe that $\mathbb{1}\{u_i = u\} = \sum_{v \in \mathcal{V}} \mathbb{1}\{u_i = u, v_i = v\}$. Consequently,

$$\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{u_i = u\} = \sum_{v \in \mathcal{V}} \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{u_i = u, v_i = v\}$$

$$\sum_{v \in \mathcal{V}}\sum_{v \in \mathcal{V}} (1-\varepsilon) p_{UV}(u, v) = (1-\varepsilon) p_U(u)$$

□

Remark. Note that the reverse doesn't hold true.

As a special case, suppose we are given p_{UV} and we consider a pair $(\widetilde{U}, \widetilde{V})$ of random variables such that $\widetilde{U}$ and $\widetilde{V}$ are independent and $\widetilde{U} \sim p_U$ and $\widetilde{V} \sim p_V$. Thus, $(\widetilde{U}, \widetilde{V}) \sim p_U p_V$. When we ask -

$$\Pr(\widetilde{U}, \widetilde{V}) \in T(n, p_{UV}, \varepsilon) \approx 2^{-n(D(p_{UV} \parallel p_U p_V) \pm \varepsilon[\dots]}$$

But now, notice that $D(p_{UV} \parallel p_U p_V) = I(U; V)$, thus it is related to how difficult it is for the pair to pretend they came from a joint distribution when they are independent.

7.2 Application: Lossy Coding

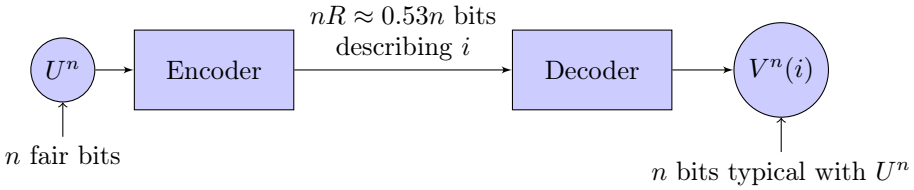
Consider p_{UV} on binary sequences $\{0,1\}^* \times \{0,1\}^*$ with

$$p(00) = 0.45, \quad p(01) = 0.05, \quad p(10) = 0.05, \quad p(11) = 0.45$$

Say we generate M binary sequences of length n ahead of time from p_V, thus

$$M \underbrace{\begin{bmatrix} 0 & 1 & 0 & \dots \\ \dots & \dots & \dots & \dots \\ \vdots & \ddots & \ddots & \vdots \end{bmatrix}}_{n} \triangleq \begin{matrix} V^n(1) \\ \vdots \\ V^n(M) \end{matrix}$$

Now, we observe a $(U_1, \dots, U_n)$ data i.i.d $\sim p_U$. Let's search the table of there exists as $V^n(i)$ for which $(U^n, V^n(i)) \in T(n, p_{UV}, \varepsilon)$. Note that for the above probability distribution, $I(U; V) \approx 0.52$ and we take $M = 2^{nR}$ where $R > I(U; V)$. We can describe i with $\log_2 M = nR$ bits.



We see that with only around 50% of the bits, we can reach 90% accuracy (since 0.45 + 0.45 = 0.90). In a really naive scheme, we send 90% of the bits, and in a lesser naive scheme, we send 80% of the bits and guess randomly to get 90% accuracy, but here we only need to send 50% of the information. The only overhead is the one-time transfer of the probability table.

7.3 Application: Almost Lossless Data Compression with Fixed Length Binary Representations

Given p_U, fix $\varepsilon > 0$ and pick n large enough such that

$$\Pr(U_1 \dots U_n \in T(n, p, \varepsilon)) > 1 - \varepsilon$$

where $U_1 \dots U_n$ is i.i.d $\sim p_U$. Since $|T(n, p_U, \varepsilon)| \leq 2^{nH(p)(1+\varepsilon)}$, we can assign binary sequences of length $k = \lceil nH(p)(1+\varepsilon) \rceil$ to represent uniquely each of these sequences (elements of $T(n, p_U, \varepsilon)$). The data compression system outputs U^n if $U^n \in T(n, p, \varepsilon)$ else it outputs a sequence of k 0s.

Recovery of U^n from the k bits is exact as long as $U^n \in T$, bit since $U^n \in T$ w.p more than $1 - \varepsilon$, we are fine.
The efficiency of the system (bits/letter) is given as

$$\frac{\text{bits}}{\text{letter}} = \frac{k}{n} < \frac{nH(1+\varepsilon)H(p) + 1}{n} = (1+\varepsilon)H(U) + \frac{1}{n} \approx H(U)$$

Chapter 8

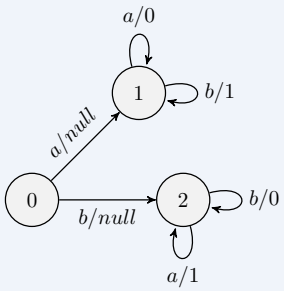
Lecture 8

8.1 Universal Data Compression

Consider an alphabet $\mathcal{U}$, with $|\mathcal{U}| < \infty$ and an infinite sequence $u_1 u_2, \ldots, u_i \in \mathcal{U}$. We consider the adversaries to our (yet to be described) universal method as a Finite State Machine which is invertible. In essence, a finite state machine can be described by

- A finite state set of states $\mathcal{S}$, with $|\mathcal{S}| = s$, and a starting point $s_0 \in \mathcal{S}$
- A function `next-state`: $\mathcal{S} \times \mathcal{U} \to \mathcal{S}$ such that $s_{i+1} = \texttt{next-state}(s_i, u_{i+1})$ for $i = 0, 1, \ldots$
- A function `output`: $\mathcal{S} \times \mathcal{U} \to \{0,1\}^*$ scuh that $y_{i+1} = \texttt{output}(s_i, u_{i+1})$

Example. Consider $\mathcal{U} = \{a, b\}$ and $\mathcal{S} = \{0, 1, 2\}$ with $s_0 = 0$



Here, we notice that the output doesn't have any information of the first letter, and it is not invertible, i.e given the output $000\ldots$, the input can be $aaa\ldots$ or $bbb\ldots$. But if we can look athe final state of the machine, we can identify the input. Thus the internal memory of the machine keeps the information, and the machine is called *information losselss*. Thus together with the final state, we can determine the input given the output.

Remark. *Notation:* If $z \in \mathcal{S}$, $w \in \mathcal{U}^*$, $w = v_1 v_2 \ldots v_m$, then $\texttt{next-state}(z, w)$ is the final state of the machine after reading the entire w. Say $w = v_1 v_2$, then

$$\texttt{next-state}(z, w) = \texttt{next-state}(\texttt{next-state}(z, v_1), v_2)$$

and, $\texttt{output}(z, w)$ is the concatenation of the outputs while reading w, thus

$$\texttt{output}(z, v_1 v_2) = \texttt{output}(z, v_1)\texttt{output}(\texttt{next-state}(z, v_1), v_2)$$

For an information lossless machine - for any state $z \in \mathcal{S}$ and any $w, w' \in \mathcal{U}^*$, with $w \neq w'$ then either

1. $\texttt{output}(z, w) \neq \texttt{output}(z, w')$ [output isn't same] **OR**
2. $\texttt{next-state}(z, w) \neq \texttt{next-state}(z, w')$ [different state]

Lemma 6. Suppose $u_1 \dots u_n = w_1 \dots w_m$, $u_i \in \mathcal{U}, w_j \in \mathcal{U}^*$. Suppose $w_i \neq w_j$ when $i \neq j$ (i.e it is a distinct parsing of $u_1 \dots u_n$ into words $w_1 \dots w_m$), then for any non-negative integer k -

$$n \geq k\left[m - |\mathcal{U}|^k\right]$$

Proof. Consider $\mathcal{U}^*$ contains 1 word of length 0, $|\mathcal{U}|$ words of length 1, ... and $|\mathcal{U}|^{k-1}$ words of length $k-1$. From the sum of geometric progressions, we know that $\sum_{i=0}^{k-1} |\mathcal{U}|^i < |\mathcal{U}|^k$. Among the m w's, at least $m - |\mathcal{U}|^l$ must be longer than k or more and thus total length of these must be greater than $k\left[m - |\mathcal{U}|^k\right]$ □

Corollary. Suppose $m^*(u^n)$ is the largest m such that $u^n = w_1 \dots w_m$ with distinct w's. Then,

$$\lim_{n\to\infty} \frac{m^*(u^n)}{n} = 0 \text{ i.e } m^* \text{ grows sublinearly}$$

Proof. We have $m^*(u^n) < \frac{n}{k} + |\mathcal{U}|^k$ for any non-negative k. Divide by n and take the limit to get $0 \leq \lim_{n\to\infty} \frac{1m^*(u^n)}{n} \leq \frac{1}{k}$. But since this is true for all k, we can take k to be very large. □

Example. For $\mathcal{U} = \{a, b\}$, we have $m^*(a) = 2$, $m^*(ab) = 3$, $m^*(aa) = 2$, $m^*(aaa) = 3$.

Remark. Say we bound k such that $|\mathcal{U}|^k = \frac{m}{2}$. This means that $n \geq \frac{m}{2} \log_{|\mathcal{U}|} \frac{m}{2}$. Moreover, if we take $|\mathcal{U}|^k = 0.001m$, then $n \approx m \log_{|\mathcal{U}|} m$ and thus n grows superlinearly in m.

Definition 14. A collection $y_1 \dots y_m$ of binary words $y_i \in \{0,1\}^*$ is said to be k-distinct if $\forall\, t \in \{0,1\}^*$, $|\{i : y_i = t\}| \leq k$.

Example. $(null, 0, 1, 00, 01, 10, 11)$ is 1-distinct and $(null, null, 0, 00, 11, 000)$ is 2-distinct.

Lemma 7. Suppose $(y_1, \dots, y_m)$ is k-distinct. If we write $m = k + k \cdot 2 + k \cdot 2^2 + \dots + k \cdot 2^{j-1} + r$ for $0 \leq r < k \cdot 2^j$, then

$$\sum_{i=1}^{m} \text{length}(y_i) \geq \sum_{i=0}^{j-1} k \cdot i \cdot 2^i + j \cdot r$$

Proof. We have $\{0,1\}^* = \{null, 0, 1, 00, 01, 10, 11, \dots\}$ and thus there are 2^i binary words of length i. Rewrite m as $k(2^j - 1) + r$ and using $\sum_{i=0}^{j-1} i2^i = (j-2)2^j + 2$ to get the required inequality (since this is the best case possible for k-distinct words). □

Corollary. If $(y_1, \dots, y_m)$ are k-distinct words, then

$$\sum_{i=1}^{m} \text{length}(y_i) \geq m \log \frac{m}{8k}$$

Proof. From the previous lemma, we have

$$\sum_{i=1}^{m} \text{length}(y_i) \geq k(j-2)2^j + 2k + rj = (j-2)m + kj + 2r \geq (j-2)m$$

Now, since $0 \leq r < k \cdot 2^j$, we have $m < k(2^j - 1) + k \cdot 2^j < k \cdot 2^{j+1}$. This means that $\frac{m}{8k} < 2^{j-2}$ and thus $(j-2) > \log_2 \frac{m}{8k}$ and thus $m(j-2) > m \log_2 \frac{m}{8k}$ □

Theorem 19. Suppose we have a s-state information lossless finite state machine an we feed it $u_1 \dots u_n = w_1 \dots w_m$ where they are a distinct parsing, then

$$\text{length}(\texttt{output}(s_0, u^n)) \geq m \log_2 \frac{m}{8s^2}$$

Proof. Consider the states (with $z_0 = s_0$)

	z_0		z_1		z_2		$\dots$	$z\_\{m-1\}$		z_m
Input $u =$		w_1		w_2		w_3	$\dots$		w_m	
Output $=$		y_1		y_2		y_3	$\dots$		y_m	

We claim that $(y_1, \dots, y_m)$ is s^2-distinct. We can see that this is true since if it isn't s^2-distinct, then we contradict the fact that the machine is information lossless. Consider $\{y_\alpha = t : \alpha \in A\}$, with $|A| > s^2$ (since $z_i z_j$ pairs are s^2 in number). Thus, we have $\alpha, \beta \in A$ such that $(z_{\alpha-1}, z_\alpha) = (z_{\beta-1}, z_\beta)$, and $(y_\alpha = y_\beta)$ but this contradicts since $w_\alpha \neq w_\beta$ (due to distinct-parsing). □

Chapter 9

Lecture 9

9.1 Information Losslessness

Theorem 20. For any finite state information lossless machine,

$$\lim_{n\to\infty}\frac{1}{n}\text{length}(\texttt{output}(u^n)) \geq \lim_{n\to\infty}\frac{1}{n}m^*(u_n)\log m^*(u^n)$$

Proof. Divide both sides by n and take limit in the previous theorem (note that since the theorem is true for all m, it is also true for $m^*(u^n)$). □

9.2 Lempel-Ziv Universal Data Compression Method (1978)

(0) Let $\mathcal{D} = \mathcal{U}$ be the initial dictionary and let $i = 0$ (initial letters read so far)

(1) Represent a word in $\mathcal{D}$ using $\lceil \log_2 |\mathcal{D}| \rceil$ bits. Read $u_{i+1} \dots u_{i+\ell}$ so that $w = (u_{i+1} \dots u_{i+\ell}) \in \mathcal{D}$. Output the representation of w

(2) Remove w from $\mathcal{D}$ and insert $(wu), u \in \mathcal{U}$ into $\mathcal{D}$ and repeat (1)

Example. Take $\mathcal{D} = \{a, b, c\}$ with representations $(00, 01, 10)$. Consider the input sequence as $abaabcca \dots$.

1. Word read: a, $\mathcal{D} = \{aa, ab, ac, b, c\}$, Output: (00)
2. Word read: b, $\mathcal{D} = \{aa, ab, ac, ba, bb, bc, c\}$, Output: 00(011)
3. Word read: aa, $\mathcal{D} = \{aaa, aab, aac, ab, ac, ba, bb, bc, c\}$, Output: 00011(000)

The decoder works in the same way - It first reads 000 and decides that we replaced a, and expands dictionary in the same way.

If we analyze LZW, then consider $u_1 \dots u_n = w_1 \dots w_m$ which is the LZ-parsing. Observe that it is a distinct-parsing since we never add a word back to the dictionary. We can calculate the number of produced bits as

$$n_{\text{bits}}^{LZ} = \lceil \log_2(|\mathcal{U}|) \rceil + \lceil \log_2(|\mathcal{U}| + |\mathcal{U}| - 1) \rceil + \dots + \lceil \log_2(|\mathcal{U} + (m-1)(|\mathcal{U}| - 1)) \rceil$$

We can bound this as

$$\begin{aligned} n_{\text{bits}}^{LZ} &\leq m\lceil \log_2(|\mathcal{U} + (m-1)(|\mathcal{U}| - 1)) \rceil \leq m\lceil \log_2(m|\mathcal{U}|) \rceil \\ &\leq m[\log_2(m|\mathcal{U}|) + 1] = m\log_2(2m|\mathcal{U}|) \end{aligned}$$

Theorem 21.

$$\lim_{n\to\infty}\frac{1}{n}\text{length}(\texttt{output}_{LZ}(u^n)) \le \lim_{n\to\infty}\frac{1}{n}m^*(u^n)\log m^*(u_n)$$

Proof. Divide both sides of the above inequality by n and take limit. □

From this, we conclude that LZ beats any information lossless machine.

Remark. Note that in FSM we ignored $-\frac{m^*(u^n)}{n}\log 8s^2$ and in LZW we ignored $\frac{m^*(u^n)}{n}\log 2|\mathcal{U}|$. The difference is $\frac{m^*(u^n)}{n}\log 16s^2|\mathcal{U}|$ and this favors FSM. In general, we should have n large enough so that we overcome this advantage. In practice, LZ works much better than the pessimistic bound we put on it but in case we know about the data in prior with what we're dealing, LZ might be worse.

Remark. Suppose $U_1U_2\dots$ is a stationary process, then

$$\lim_{n\to\infty}\mathbb{E}\left[\frac{1}{n}\text{length}(\texttt{output}_{LZ}(U^n))\right] \le \frac{1}{k}[H(U^k)+1]$$

We can achieve the right bound by a FSM where K is the number of blocks of sequence (we can implement the FSM for doing Huffman coding for U^k). Thus the limit is $\le \mathcal{H}$. Now, since it is impossible to beat the entropy rate in lossless compression, it must be an equality and thus LZ universally compresses stationary sources with no penalty.

9.3 Universality

Suppose we have an alphabet $\mathcal{U}$ and $\mathcal{P}$ is a class/set of probability distributions on $\mathcal{U}$. Now, say we design a source code $c:\mathcal{U}\to\{0,1\}^*$. Let $q(u)=2^{-\ell_c(u)}$ (note that whenever we design a code, we inherently define a probability distribution on it).
Suppose U is a random variable with distribution $p\in\mathcal{P}$. Then, we can write $\mathbb{E}[\ell_c(u)]=H(U)$ as the penalty of c w.r.t any adversary which knows p. Since this quantity is equal to $D(p\parallel q)$, we write this as the regret. We can associate to any code design the maximal regret given as $\max_{p\in\mathcal{P}} D(p\parallel q)$. A sensible way to design the code is to minimize this, and thus $\min_q \max_{p\in\mathcal{P}} D(p\parallel q)$.
Now, consider an alphabet $\mathcal{U}^n$ and $c_{\widetilde{LZ}}(u^n)\rightsquigarrow q^n_{LZ}$ on $\mathcal{U}^n$ (where $\widetilde{LZ}$ is a modified version to include termination. We have seen that

$$\lim_{n\to\infty}\max p\in\mathcal{P}_n\frac{1}{n}D(p^n\parallel q^n_{LZ})=0$$

where p^n is the restriction of p to first n letters and $\mathcal{P}_n$ is a stationary process. Moreover, we can interpret this as

$$\begin{aligned}
\frac{1}{n}D(p^n\parallel q^n_{LZ}) &= \frac{1}{n}\sum_{u^n}p^n(u^n)\log\left[\frac{p(u_1)p(u_2|u_1)\dots p(u_n|u^{n-1})}{q(u_1)q(u_2|u_1)\dots q(u_n|u^{n-1})}\right]\\
&= \frac{1}{n}\sum_{i=1}^{n}\sum_{u^n}p^n(u^n)\log\left[\frac{p(u_i|u^{-1})}{q(u_i|u^{i-1})}\right] = \frac{1}{n}\sum_{i=1}^{n}\sum_{u^i}p(u^i)\log\left[\frac{p(u_i|u^{-1})}{q(u_i|u^{i-1})}\right]\\
&= \frac{1}{n}\sum_{i=1}^{n}\sum_{u^{i-1}}p(u^{i-1})\sum_{u^i}p(u_i|u^{i-1})\log\left[\frac{p(u_i|u^{-1})}{q(u_i|u^{i-1})}\right]\\
&= \frac{1}{n}\underbrace{\sum_{i=1}^{n}\sum_{u^{i-1}}}p(u^{i-1})\,D(p(u_i|u^{i-1}\parallel q(u_i|u^{i-1}))
\end{aligned}$$

The first term represents the average over time, and the second one represents the average over the past letters. Now, since the limit goes to 0 as $n\uparrow$, for most i's and for a high $p^i(\cdot)$ probability set of u^{i-1}'s, $D(p(u_i|u^{i-1}\parallel q(u_i|u^{i-1}))$ is small and thus we conclude that LZ is a ***learning algorithm***, and for stationary sources, it was learning the values (since $D(\cdot\parallel\cdot)\downarrow$).