Entropy and Shannon’s Theorem

Part I

Entropy

Part II

Extracting randomness

Storing all strings of length $n$ and $j$ bits on

1. $S_{n,j}$: set of all strings of length $n$ with $j$ ones in them.
2. $T_{n,j}$: prefix tree storing all $S_{n,j}$.

$T_{0,0}$

$T_{1,1}$  $T_{1,0}$
Binary strings of length 4

1. \( S_{4,0} = \{0000\} \implies \#(0000) = 0 \).
2. \( S_{4,1} = \{0001, 0010, 0100, 1000\} \implies \#(0001) = 0 \).
   \#(0100) = 2.
   \#(1000) = 3.
3. \( S_{4,2} = \{0011, 0101, 0110, 1001, 1010, 1100\} \implies \#(0011) = 0 \).
   \#(0101) = 1.
   \#(0110) = 2.
   \#(1001) = 3.
   \#(1010) = 4.
   \#(1100) = 5.
4. \( S_{4,3} = \{0111, 1011, 1101, 1110\} \implies \#(0111) = 0 \).
   \#(1011) = 1.
   \#(1101) = 2.
   \#(1110) = 3.
5. \( S_{4,4} = \{1111\} \implies \#(1111) = 0 \).

Prefix tree \( \forall \) binary strings of length \( n \) with \( j \) ones

\[
T_{n,j} \quad \# \text{ of leafs:} \quad |T_{n,j}| = |T_{n-1,j}| + |T_{n-1,j-1}|
\]
\[
\binom{n}{j} = \binom{n-1}{j} + \binom{n-1}{j-1}
\]
\[
|T_{n,j}| = \binom{n}{j}.
\]

Encoding a string in \( S_{n,j} \)

1. \( T_{n,j} \) leafs corresponds to strings of \( S_{n,j} \).
2. Order all strings of \( S_{n,j} \) order in lexicographical ordering.
3. \( \equiv \) ordering leafs of \( T_{n,j} \) from left to right.

Decoding a string in \( S_{n,j} \)

1. \( T_{n,j} \) leafs corresponds to strings of \( S_{n,j} \).
2. Order all strings of \( S_{n,j} \) order in lexicographical ordering.
3. \( \equiv \) ordering leafs of \( T_{n,j} \) from left to right.

4. Input: \( s \in S_{n,j} \): compute index of \( s \) in sorted set \( S_{n,j} \).
5. \text{EncodeBinomCoeff}(s) denote this polytime procedure.

4. \( x \in \{1, \ldots, \binom{n}{j}\} \): compute \( x \)th string in \( S_{n,j} \) in polytime.
5. \text{DecodeBinomCoeff}(x) denote this procedure.
Encoding/decoding strings of $S_{n,j}$

**Lemma**

$S_{n,j}$: Set of binary strings of length $n$ with $j$ ones, sorted lexicographically.

1. **EncodeBinomCoeff**($\alpha$): Input is string $\alpha \in S_{n,j}$, compute index $x$ of $\alpha$ in $S_{n,j}$ in polynomial time in $n$.
2. **DecodeBinomCoeff**($x$): Input index $x \in \{1, \ldots, \binom{n}{j}\}$. Output $x$th string $\alpha$ in $S_{n,j}$, in time $O(\text{polylog } n + n)$.

Proof...

1. There are $\binom{n}{j}$ input strings with exactly $j$ heads.
2. Each has probability $p^j(1 - p)^{n-j}$.
3. Map string $s$ like that to index number in the set $S_j = \{1, \ldots, \binom{n}{j}\}$.
4. Given that input string $s$ has $j$ ones (out of $n$ bits) defines a uniform distribution on $S_{n,j}$.
5. $x \leftarrow \text{EncodeBinomCoeff}(s)$
6. $x$ uniform distributed in $\{1, \ldots, N\}$, $N = \binom{n}{j}$.
7. Seen in previous lecture...
8. ... extract in expectation, $\lfloor \log N \rfloor - 1$ bits from uniform random variable in the range $1, \ldots, N$.

Extracting randomness

**Theorem**

Consider a coin that comes up heads with probability $p > 1/2$. For any constant $\delta > 0$ and for $n$ sufficiently large:

(A) One can extract, from an input of a sequence of $n$ flips, an output sequence of $(1 - \delta)n\log_2(p)$ (unbiased) independent random bits.

(B) One can not extract more than $n\log_2(p)$ bits from such a sequence.

Exciting proof continued...

1. $Z$: random variable: number of heads in input string $s$.
2. $B$: number of random bits extracted.
   
   $$E[B] = \sum_{k=0}^{n} \Pr[Z = k]E[B \mid Z = k],$$

3. Know: $E[B \mid Z = k] \geq \left\lfloor \log \binom{n}{k} \right\rfloor - 1$.
4. $\varepsilon < p - 1/2$: sufficiently small constant.
5. $n(p - \varepsilon) \leq k \leq n(p + \varepsilon)$:
   
   $$\binom{n}{k} \geq \binom{n}{\lfloor n(p + \varepsilon) \rfloor} \geq \frac{2^{n\log_2(p + \varepsilon)}}{n + 1},$$

6. ... since $2^{n\log_2(p)}$ is a good approximation to $\binom{n}{np}$ as proved in previous lecture.
Super exciting proof continued...

\[ E[B] = \sum_{k=0}^{n} \Pr[Z = k] E[B \mid Z = k]. \]

\[ E[B] \geq \sum_{k=\lfloor np - \varepsilon \rfloor}^{\lfloor np + \varepsilon \rfloor} \Pr[Z = k] E[B \mid Z = k] \]
\[ \geq \frac{\beta(n\varepsilon + p - 1)}{\beta(n\varepsilon + p) - 2} \]
\[ \geq \frac{(n\varepsilon + p - 2) - \varepsilon(n + 1)}{\varepsilon} \Pr[Z - np \leq \varepsilon] \]
\[ = (n\varepsilon + p - 2)(1 - 2 \exp(-\frac{\varepsilon^2}{4p})). \]

since \( \mu = E[Z] = np \) and \( \Pr[Z - np \geq \varepsilon pn] \leq \)
\[ 2 \exp\left(-\frac{np}{4} \left(\frac{\varepsilon}{p}\right)^2\right) = 2 \exp\left(-\frac{1}{4p}\right), \] by the Chernoff inequality.

Hyper super duper exciting proof continued...

1. Need to prove upper bound.
2. If input sequence \( x \) has probability \( \Pr[X = x] \), then
   \( y = \text{Ext}(x) \) has probability to be generated
   \[ \geq \Pr[X = x]. \]
3. All sequences of length \( |y| \) have equal probability to be generated (by definition).
4. \[ 2^{|\text{Ext}(x)|} \Pr[X = x] \leq 2^{|\text{Ext}(x)|} \Pr[y = \text{Ext}(x)] \leq 1. \]
5. \( \Rightarrow |\text{Ext}(x)| \leq \lg(1/\Pr[X = x]) \)
6. \( E[B] = \sum_x \Pr[X = x] |\text{Ext}(x)| \]
   \[ \leq \sum_x \Pr[X = x] \frac{1}{\Pr[X = x]} = \mathbb{H}(X). \]

Part III

Coding: Shannon’s Theorem
Claude Elwood Shannon (April 30, 1916 - February 24, 2001), an American electrical engineer and mathematician, has been called “the father of information theory”. His master thesis was how to building boolean circuits for any boolean function.

Claude Elwood Shannon

Claude Elwood Shannon (April 30, 1916 - February 24, 2001), an American electrical engineer and mathematician, has been called “the father of information theory”. His master thesis was how to building boolean circuits for any boolean function.

Claude Elwood Shannon

Claude Elwood Shannon (April 30, 1916 - February 24, 2001), an American electrical engineer and mathematician, has been called “the father of information theory”. His master thesis was how to building boolean circuits for any boolean function.

Claude Elwood Shannon

Claude Elwood Shannon (April 30, 1916 - February 24, 2001), an American electrical engineer and mathematician, has been called “the father of information theory”. His master thesis was how to building boolean circuits for any boolean function.
When the sender sends a string...

\[ S = s_1 s_2 \ldots s_n \]

One ring to rule them all!

Some intuition...

1. senders sent string \( S = s_1 s_2 \ldots s_n \).
2. receiver got string \( T = t_1 t_2 \ldots t_n \).
3. \( p = \Pr[t_i \neq s_i] \), for all \( i \).
4. \( U \): Hamming distance between \( S \) and \( T \):
   \[ U = \sum_i [s_i \neq t_i] \]
   with high probability, where \( \delta \) is tiny constant.
5. By assumption: \( E[U] = pn \), and \( U \) is a binomial variable.
6. By Chernoff inequality: \( U \in [ (1 − \delta)np, (1 + \delta)np ] \)
   with high probability.
7. \( T \) is in a ring \( R \) centered at \( S \), with inner radius \( (1 − \delta)np \) and outer radius \( (1 + \delta)np \).
8. This ring has
   \[ \sum_{i=(1-\delta)np}^{(1+\delta)np} \binom{n}{i} \leq 2 \left( \frac{n}{(1+\delta)np} \right) \leq \alpha = 2 \cdot 2^{nH((1+\delta)p)}. \]
   strings in it.

Many rings for many codewords...

Some more intuition...

1. Pick as many disjoint rings as possible: \( R_1, \ldots, R_\kappa \).
2. If every word in the hypercube would be covered...
3. ... use \( 2^n \) codewords \( \implies \kappa \geq 2^n \geq 2^n \cdot 2^{nH((1+\delta)p)} \approx 2^{n(1-H((1+\delta)p))}. \)
4. Consider all possible strings of length \( k \) such that \( 2^k \leq \kappa \).
5. Map \( i \)th string in \( \{0, 1\}^k \) to the center \( C_i \) of the \( i \)th ring \( R_i \).
6. If send \( C_i \implies \) receiver gets a string in \( R_i \).
7. Decoding is easy - find the ring \( R_i \) containing the received string, take its center string \( C_i \), and output the original string it was mapped to.
8. How many bits?
   \[ k = \lceil \log \kappa \rceil = n \left( 1 - H((1 + \delta)p) \right) \approx n(1-H(p)). \]
What is wrong with the above?

1. Can not find such a large set of disjoint rings.
2. Reason is that when you pack rings (or balls) you are going to have wasted spaces around.
3. Overcome this: allow rings to overlap somewhat.
4. Makes things considerably more involved.
5. Details in class notes.