Consider data source modeled by RV $X$. Assume we know distribution $P_X$. 
E.g. $X$ could be a letter and we assume $P(X) = P_{English}(x)$.

How well can we compress? Today we consider symbol codes, which compress one symbol (letter, source message,...) at a time.

\[ X \rightarrow C \rightarrow \{0,1\}^* \rightarrow D \rightarrow \hat{X} = X \]

Compressor, encoder
\[ \text{de compressor}, \ decoder \]

\textbf{WANT:}

\[ \text{at least one more bit than entropy} \]

\textbf{Goal:}
Show that lossless compression one symbol at a time can achieve
\[ H(X) \leq L < H(X) + 1, \text{ where } L = \text{average length of codeword}. \]

\textbf{At least one more bit than entropy}

\textbf{Notation:}
\[ S^+ = \bigcup_{n \geq 1} S^n = \text{non empty strings over } S \]
\[ l(w) = \text{length of string } w \in S^+ \]

\textbf{Symbol code:}
\[ C : A \rightarrow \{0,1\}^+ \text{ for alphabet } A \]
\[ C_i(x) = \text{how we compress } x \]

\textbf{Average length:}
\[ L(C_i(P)) = L(C_i(X)) = \sum_{x \in A} P(x) l(C_i(x)) = E[ l(C_i(X))] \]

\textbf{Extended code:}
\[ C_i^+ : A^+ \rightarrow \{0,1\}^+, \quad C_i^+(x_1...x_n) := C_i(x_1)...C_i(x_n) \]

\textbf{Two important classes of codes:}
\[ C_i \text{ is called:} \]

\textbf{Uniquely Decodable (UD) if:}
\[ w \neq w' \Rightarrow C_i^+(w) \neq C_i^+(w') \quad \forall w, w' \in A^+ \]

\textbf{Prefix Free Code if no codeword }$C_i(x)$ is prefix of any other
FACT: Any prefix code is UD!

Kraft-McMillan Inequality: If $G$ is UD then
\[
\sum_{x \in \mathcal{A}} 2^{-\ell(G(x))} \leq 1
\]
Optimal codes should saturate this ("complete" code)

**Proof:** Let $S := \sum_{x} 2^{-\ell(G(x))}$ and $\ell_{\text{max}} := \max_{x} \ell(G(x))$. Then:

\[
S^N = \sum_{x_1 \cdots x_N} 2^{-\ell(G^+(x_1 \cdots x_N))} \leq N \cdot \ell_{\text{max}} \quad \text{for linear growth}
\]

\[
\sum_{x_1 \cdots x_N} 2^{-\ell(G^+(x_1 \cdots x_N))} \leq \sum_{l=1}^{N \cdot \ell_{\text{max}}} \binom{N}{l} \leq 2^N \leq 2^N \cdot 2^{-\ell_l} \leq 2^\ell
\]

Kraft's converse: Let $\ell_x \geq 1$ for $x \in \mathcal{A}$ be integers s.t. $\sum_{x} 2^{-\ell_x} = 1$. Then there is a prefix code $G$ with $\ell(G(x)) = \ell_x$ for all $x \in \mathcal{A}$

**Proof:** Construct as follows: algorithm, but not very efficient

1. Order the numbers:
   \[
   \ell_{x_1} \leq \ell_{x_2} \leq \ldots
   \]
   \[
   \text{ Constr. } \ell_{x_1} = \ell_{x_2} = \ldots
   \]

Thus, prefix codes are as good as any UD code!!!
For $k=1,2,\ldots$ choose $C_i(x_k) \in \{0,1\}^k$, s.t. none of the $C_i(x_1),\ldots,C_i(x_k)$ is prefix. This is possible, since

$$\sum_{i=1}^{k-1} 2^{x_k} - x_i = 2^{x_k} \sum_{i=1}^{k-1} 2^{-x_i} < 2^{x_k} \sum_{x} 2^{-x} \leq 2^{x_k}$$

But what does this mean for the average length? Need one more tool...

**Gibbs inequality**: Let $P, Q$ prob. distributions. Then:

$$\sum_x P(x) \log \frac{1}{Q(x)} = H(C) - I \iff P = Q$$

**Proof**: LHS-RHS $= \sum_x P(x) \log \frac{P(x)}{Q(x)} = -\sum_x P(x) \log Q(x)$ & use Jensen.

**Lower bound**: $LCC(C(P)) \geq H(C(P))$ for every UD code. Information content!

Equality holds iff $l(C(\mathcal{X})) = \log \frac{1}{P(\mathcal{X})} (k_x)$.

**Proof**: Define

$$\mathcal{Q}(x) = \frac{2^{-l(\mathcal{G}(x))}}{S}$$

where $S = \sum_x 2^{-l(\mathcal{G}(x))}$ Kraft-McMillan

$$H(C(P)) \leq \sum_x P(x) \log \frac{1}{Q(x)} = LCC(C(P)) + \log S \leq LCC(C(P))$$

**Existence of good codes**: Prefix codes with $LCC(C(P)) < H(X) + 1$

**Proof**: Define $L_x = \lceil \log \frac{1}{P(x)} \rceil > 1$ round up equalizing condition from above

$$\sum_x 2^{-L_x} \leq \sum_x P(x) = 1 \rightarrow \text{by Kraft’s converse, there exists a prefix code } C \text{ with } l(C(x)) = L_x$$

$$L(X,C) = \sum_x P(x) L_x < \sum_x P(x) \left( \log \frac{1}{P(x)} + 1 \right) = H(X) + 1$$
NB: The code constructed in the proof is in general not optimal. E.g.: 

$$H(X) = \log_2(3) = 1.585...$$

<table>
<thead>
<tr>
<th>x</th>
<th>P(x)</th>
<th>l(x)</th>
<th>C(x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1/3</td>
<td>2</td>
<td>01</td>
</tr>
<tr>
<td>B</td>
<td>1/3</td>
<td>2</td>
<td>101</td>
</tr>
<tr>
<td>C</td>
<td>1/3</td>
<td>2</td>
<td>00</td>
</tr>
</tbody>
</table>

$$L(C, X) = 2$$

But we can clearly do better: 

$$L = 1.666...$$

To find an optimal prefix (and therefore UD) code, we can use the following algorithm:

**Huffman's Coding Algorithm:**

- **Input:** probability dist. P on X
- **Output:** binary tree corresponding to prefix code C with minimal L(C, P)

  **algo:**
  1. Start with a "forest" of n isolated leaves
  2. While more than one tree: merge two trees with smallest probabilities

**Example:**

<table>
<thead>
<tr>
<th>X</th>
<th>P(x)</th>
<th>H(P) = 2.28...</th>
<th>C(x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.25</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>0.25</td>
<td>0.45</td>
<td>010</td>
</tr>
<tr>
<td>C</td>
<td>0.2</td>
<td></td>
<td>0.55</td>
</tr>
<tr>
<td>D</td>
<td>0.15</td>
<td>0.3</td>
<td>0.3</td>
</tr>
<tr>
<td>E</td>
<td>0.15</td>
<td></td>
<td>011</td>
</tr>
</tbody>
</table>

**Summary:**

**Source Coding Theorem for Prefix Codes:** Let C be the optimal UD/prefix code for X i.i.d. (e.g., Huffman's). Then: 

$$H(X) \leq L(C, X) < H(X) + 1$$

**Problem:** Completely useless when X is e.g. a bitstream.

**Solution:** Compress blocks of N symbols at a time:

[Diagram showing compression and decompression processes]

I.e., build code on C^N for joint distribution of X_{11..N}

$$X^N = (X_{11..N})$$
Result: If $X_1, \ldots, X_N \overset{iid}{\sim} P$ then the optimal prefix code satisfies

$$H(C) \leq \frac{1}{N} \sum_{i=1}^{N} H(X_i) \leq H(C) + \frac{1}{2} \rightarrow 0 \text{ as } N \rightarrow \infty$$

$\Rightarrow H(C)$ is optimal asymptotic average rate of compression of iid source.

**Pf:** $H(X_1, \ldots, X_N) = N \cdot H(C)$ because iid.

Remark: iid assumption is not realistic, but a good starting point.

Two bits of TERMINOLOGY to remember:

* "Compression" = "source coding"
* Average rate of compression = $\frac{(\text{average}) \# \text{bits used to compress message of length } N}{N}$

**NOTATION:** $R$ for rate, $R$ for average rate