Recall: Huffman Coding

\[ C = \{ c_1, \ldots, c_n \} \]

0 < \( f(c) \) = frequency of \( c \)
for all \( c \in C \).

Encoding tree: binary tree where:

1. leaves are labeled with all \( c_i \) (n leaves) (1-1 correspondence)
2. Each internal node has 2 children

Optimal tree for \( C \): ("Huffman tree")
is least cost tree
where the cost of tree \( T \) is given by

\[ B(T) = \sum_{i=1}^{n} f(c_i) d_T(c_i) \]

where \( d_T(c) \) is depth of leaf \( c \) in \( T \).

\( B(T) \) = bit length of coded binary file.

To construct a Huffman tree for \( C \):
1. Let Q be a min priority queue (empty)

2. Insert letters c₁, ..., cₙ into Q, keyed by frequency f[cᵢ].

3. For i := 1 to n-1 do:
   \[
   \begin{cases} 
   x := \text{ExtractMin}(Q) \\
   y := \text{ExtractMin}(Q) 
   \end{cases}
   \]
   Form new node z:
   \[
   f[z] := f[x] + f[y] \\
   \text{left}[z] := x \\
   \text{right}[z] := y
   \]
   Insert z into Q.

4. r := \text{ExtractMin}(Q)
   // Q is empty
   Return r as root of the Huffman tree.

Each iteration:

```
x               2z
   \
x               y
```

Recursive version:
if |C| = 1 then
return sole element of C as the root
else
let \( x, y \) be the two els of \( C \) with lowest frequency.

let \( z \in C \) be a new letter.

\[
f[z] := f[x] + f[y]
\]

\[
C' := (C - \{x, y\}) \cup \{z\}
\]

\[
r := \text{Huffman Tree}(C')
\]

// recursive cell

\( r \) is root of a tree \( T' \).

find leaf \( z \) in \( T' \) and replace it with

\[
\begin{array}{c}
T' \\
\Rightarrow \\
T
\end{array}
\]

return root of \( T \).

---

Proof of correctness:

**Lemma:** Let \( T \) be any encoding tree for \( C \).

let \( x, y \) be two nodes in \( C \) of least frequency.

Then there is an tree \( T' \) for \( C \) where \( x, y \) are siblings (on deepest level) in \( T' \) and \( B(T') \leq BG \).
Proof:

Let \(a\) and \(b\) be the two leftmost nodes on the deepest level of \(T\).

(easy: \(a, b\) are siblings).

If \(\exists a, b \in \mathcal{S} = \{x, y, z\}\), then let \(T' = T\). Done.

Otherwise one of \(x\) and \(y\) is not in \([a, b]\). Assume that \(x \notin [a, b]\) and \(a, y\).

\[
\begin{align*}
    d_T(x) & \leq d_T(a) \\
    f[x] & \leq f[a]
\end{align*}
\]

Form \(T'\) to be \(T\) but with \(x\) and \(a\) swapped.

\[
\begin{align*}
    B(T) - B(T') &= f[x]d_T(x) + f[a]d_T(a) \\
                 &\quad - f[x]d_T'(x) - f[a]d_T'(a) \\
                 &\quad + \underbrace{\text{(cancelling terms)}}_{0}
\end{align*}
\]

\[
\begin{align*}
    &\begin{cases}
        d_T'(x) = d_T(a) \\
        d_T'(a) = d_T(x)
    \end{cases}
\end{align*}
\]
\[ = f(x) d_T(x) + f(a) d_T(a) - f(x) d_T(a) - f(a) d_T(x) \]
\[ = (f[a] - f[x]) (d_T(a) - d_T(x)) \]
\[ \geq 0 \quad \geq 0 \]
\[ \geq 0 \]
\[ \therefore B(T) - B(T') \geq 0 \]
\[ \therefore B(T') \leq B(T). \]

If \( y \neq b \) then swap \( y \) and \( b \) in \( T' \) to get some \( T'' \) such that \( B(T'') \leq B(T') \).

Then \( B(T'') \leq B(T) \) and \( x, y \) are silos in \( T'' \) (deepest level) //Lemma

**Theorem:** Let \( x \) and \( y \) be two min freq letters in \( C \). Let

\[ C' = (C - \{x, y\}) \cup \{z\} \]

where \( z \in C \) and freqs of letters in \( C' \) are the same as in \( C \), but \( f[z] = f[x] + f[y] \).
Suppose that $T'$ is any optimal (least cost) encoding tree for $C$.
Let $T$ be same as $T'$ but with leaf $z$ replaced with $xy$.
Then $T$ is an optimal encoding tree for $C$.

"If recursive call returns an optimal $T'$ for $C'$ then original call returns an optimal tree $T$ for $C."$

Correctness of algo follows by induction on $|C|$ (size of the alphabet).

Proof: Assume not true, i.e., the tree $T$ described is not optimal for $C$.
Prove that $T'$ was not optimal for $C'$ (contradiction). Since $T$ is not optimal, there is some encoding tree $T''$ for $C$ such that $B(T'') < B(T)$. 

Nov 13 1:58 PM
Without loss of generality, x and y are siblings in $T''$ (by the lemma).

Remove $x, y$ from $T''$ and label their parent (now a leaf) $z$ with $f[z] = f[x] + f[y]$ to get new tree $T'''$ for $C'$.

Claim: $B(T''') < B(T')$

But $T'$ was optimal for $C'$ by assumption, so this is our contradiction.

Proof of Claim:

Compare $B(T)$ with $B(T')$

$B(T)$ with $B(T''')$
\[ B(T) = B(T') \]

\[
= -f[z]d_{T'}(z) + f[x]d_T(x) + f[y]d_T(y)
\]

\[
\left[d_T(x) = d_T(y) = d_{T'}(z) + 1\right]
\]

\[
= B(T') - f[z]d_{T'}(z) + \left( f[x] + f[y] \right) (d_{T'}(z) + 1) \frac{f[z]}{f[z]}
\]

\[
= B(T') - f[z]d_{T'}(z) + f[z](d_{T'}(z) + 1)
\]

\[
= B(T') + f[z]
\]

\[
= B(T') + f[x] + f[y]
\]

Again:

\[
B(T) = B(T') + f[x] + f[y]
\]

By same analysis:

\[
B(T'') = B(T''') + f[x] + f[y]
\]

So,
\[ S_0, \]
\[ B(T) - B(T'') \]
\[ = (B(T') + f[x] + f[y]) \]
\[ - (B(T'') + f[x] + f[y]) \]
\[ = B(T') - B(T'') \]

Assumed that \( B(T'') < B(T) \)
so \( B(T) - B(T'') > 0 \)

But then
\[ B(T') - B(T'') > 0, \]
so \( B(T''') < B(T') \)
so \( T' \) not optimal for \( C! \) \quad // \text{Claim.} \]

Ends proof of correctness

Time to construct Huffman tree on \( E, c_1, \ldots, c_m \)
\[
= \Theta(n \log n)
\]

\# of initial insertions and loop iterations
\text{time per insertion/extract}

Next time: Amortized Analysis.