Huffman Coding

Variable length prefix free code.

\[ \Sigma \subseteq \text{alphabet} \]

\[ |\Sigma| = n \]

n characters

Probability

for each character \( i \in \Sigma \)

\[ f_i \Rightarrow \text{Count} \]

\[ f_i \Rightarrow \text{Probability} \]

\[ f_i \times n = \text{Count} \]

String length

Alternatively merge the two smallest "frequencies".

Mississippi

Given: \( \Sigma, S \) (alphabet \& string to encode)

1) Compute \( f_i \) for all \( i \in \Sigma \)

2) Create a PQ \((i, f_i)\) \( \forall i \in \Sigma \)

3) While \( |\text{PQ}| \neq 1 \)

\[ O(n) \text{ steps} \]

\[ O(\log n) \] steps

\[ X = \text{deleteMin} (\text{PQ}) \]

\[ Y = \text{deleteMin} (\text{PQ}) \]

Create a new node \( Z = X \cup Y \)

\[ f_z = f_x + f_y \]

\[ Z \in \text{PQ} \]

\[ O(\log n) \]

\[ m = 0 \quad v = 0 \]

\[ m = 0 \quad v = 0 \]
Huffman coding: $O(n \log n) \leq$ If frequency are already given

$O(1)$ + $O(n \log n)$

Proof of correctness: There may be several min-cost encoding tree.

Show that the cost of the Huffman encoding tree is also minimum.

$$\text{Cost}(T) = \sum_{i=1}^{n} f_i \cdot l_i$$

When $f_i$ is probability, then $\text{Cost}(T) = \text{Expected encoding length per character}$

Assumption: some min encoding tree $T$

Our tree

$X \leftarrow \text{deterministic}$

$y \leftarrow \text{deterministic}$

$X$ is the least freq node

$y$ is the lowest key

$f_x \leq f_y$

Create a new tree $T'$, switch $x$ & $y$

$\text{Cost}(T') \leq \text{Cost}(T)$
\[ \text{Lecture 17 Page 3} \]

2) Create \( T'' \) from \( T' \), where we switch \( y \) and \( z \).

\[ C_{\mathcal{S}_X}(T'') \leq C_{\mathcal{S}_X}(T') = C_{\mathcal{S}_X}(T) \]

\[ \Rightarrow C_{\mathcal{S}_X}(T'') = C_{\mathcal{S}_X}(T) \]

Some optimal tree \( T \):

- \( a \) and \( b \) are at the larger depth.
- \( d_T(y) \) : depth of \( y \) in tree \( T \)
- \( d_T(x), d_T(a), d_T(b) \)

\[ C_{\mathcal{S}_X}(T) - C_{\mathcal{S}_X}(T') \]

\[ = \left( d_T(x) \cdot f_X + d_T(a) \cdot f_a \right) - \left( d_T(x) \cdot f_X + d_T(a) \cdot f_x \right) \]

\[ = \left( d_T(x) - d_T(x) \right) \cdot f_X + \left( d_T(a) - d_T(a) \right) \cdot f_a \]

\[ = (d_T(x) - d_T(x)) \cdot f_X + (d_T(a) - d_T(a)) \cdot f_a \]

\[ = (f_a - f_X) \cdot (d_T(a) - d_T(x)) \]

\[ \geq 0 \quad \text{because} \quad f_X \leq f_a \quad \text{and} \quad d_T(a) - d_T(x) \geq 0 \]

\[ \Rightarrow C_{\mathcal{S}_X}(T) - C_{\mathcal{S}_X}(T') \geq 0 \]

\[ \Rightarrow C_{\mathcal{S}_X}(T) \geq C_{\mathcal{S}_X}(T') \]
Set Cover Problem

- NP Complete -> hard
  - no polynomial time solution

Greedy algo -> give us an approximate solution
  - close to the optimal value

Assume there is a graph
Optimization:
Choose the least # of
nodes/cities
to build a
hospital

Sort the vertices by decreasing degree, select one or
a line until all nodes are "covered," making sure
adjust the degrees by removing already covered vertex!
K < optimal value
Is K = 3 optimal?

Input: \( G = (V, E) \)
- \( N_x \) = neighborhood of \( x \) (all neighbors of \( x \))

Problem statement:
Find the minimal number \( k \) of nodes
\( \{1, 2, \ldots, k\} \)
Such that \( \bigcup_{i=1}^{k} N_i = V \)

Greedy is not optimal
Our the greedy solution is \( O(k \log n) \) factor
away from the optimal.

\( \Rightarrow \) if the optimal solution is \( k \)
then the greedy solution value is \( O(k \log n) \)

If \( k \) is optimal value
\[ O = \{1, 2, \ldots, k\} \]

Total vertex in \( G \) is \( |V| = n \).

**Observation:** There is a vertex \( v \) in \( O \) that covers at least \( \frac{n}{k} \) other vertices.

**Proof:** Assume that for all \( x \in O \)
\[
|N_x| < \frac{n}{k}
\]

\[
\sum_{x \in O} |N_x| < \frac{n}{k} + \frac{n}{k} + \ldots + \frac{n}{k} = \frac{n}{k} \cdot k = \frac{n}{k} \cdot \frac{n}{k} < \frac{n}{k}
\]

That is a contradiction.

\[ n_t : \# of uncovered vertices after \( t \) steps. \]

\[ n_0 = n \leftarrow \text{Initially, every node is uncovered} \]

\[ n_1 \leq n - \left( \frac{n}{k} \right) \]

\[ n_1 \leq n \left( 1 - \frac{1}{k} \right) \]

\( n_1 \) nodes remaining, \( k - 1 \) Choice remaining or least higher degree node will cover at least \( \frac{n}{k-1} \) other vertices.

\[ \geq \frac{n_1}{k-1} \geq \frac{n_1}{k} \]

\[ n_2 \leq n_1 - \frac{n_1}{k} = n \left( 1 - \frac{1}{k} \right) \leq n \left( 1 - \frac{1}{k} \right) \left( 1 - \frac{1}{k} \right) \]

\[ n_2 \leq n \left( 1 - \frac{1}{k} \right)^2 \]
\[
\begin{align*}
\eta_t & \leq \eta \left( \frac{1 - \frac{1}{k}}{k} \right)^t \leq \eta \left( e^{-\frac{1}{k}} \right)^t \\
\eta_t & \leq \eta \left( e^{-\frac{1}{k}} \right)^t \leq e^{\frac{1}{k}} \\
\text{the solution is} \quad t &= k \log n \\
\eta \left( e^{-\frac{1}{k}} \right)^t & \leq \eta \log n \\
n \cdot e^{-\frac{1}{k}} &= \frac{n}{e^{\log n}} = \frac{n}{n} = 1
\end{align*}
\]

Greedy solution takes at most
\[
t = k \log n \quad \text{steps} = \# \text{ greedy steps}
\]