OH TODAY: 1:15 - 2:15 pm

Emails

Exam 1

Huffman Coding.

Input: n characters in a text

S: alphabet

\( \text{if } x \in S, f_x: \text{ fraction of times } x \text{ appears in the text} \)

Objective: To obtain a prefix code \( R \) s.t.

\[
\text{ABL} := \sum_{x \in S} f_x |r(x)| \quad \text{is minimized}
\]

Avg. bits per letter

- Binary trees in which characters in \( S \) are associated with leaves give us prefix codes.
- Binary trees corresponding to optimal prefix codes must be full binary trees.

- We need:
  - Shape of the binary tree
  - Assignment of classes in $S$ to leaves in the tree.

- Observation: Two lowest freq. letters are assigned to leaves at the bottommost level. Since the binary tree corresponding to an optimal prefix code is full, there must be two leaves in the bottommost level that are siblings.

Lemma: There exist optimal prefix codes with corresponding binary tree $T^*$ in which the two lowest freq. letters are assigned to sibling leaves in the bottommost level.
Huffman $(S)$

If $|S| \leq 2$ then

Dane (assign 0 to one char & 1 to the other)

else

$y^*, z^*,$ two lowest freq letters in $S$

$y^*, z^* \in S' \cup \{w\}$

$f_w = f_{y^*} + f_{z^*}$

$T' \leftarrow$ Huffman $(S')$

Travers $T'$ and add $y^*$ and $z^*$ as children of $w$ in $T'$. Call this
new tree $T$

return $T$

**Example**

```
  a  b  c  d  e  f  g
0.35  0.22  0.18  0.10  0.06  0.03
```

```
  a  b  c  d  e  f  g
0.35  0.22  0.18  0.1  0.06  0.09
```

```
  a  b  c  d  e  f  g
0.35  0.22  0.18  0.1  0.15
```

```
  a  b  c  defg
0.35  0.22  0.18  0.25
```

```
  a  bc  defg
0.35  0.40  0.25
```

**Running time**: $|S| = n$

$T(n) = T(n-1) + O(n)$

$= O(n^2)$.

$T(n) = T(n-1) + O(ln^2)$

$= O(n \ln n)$. 
Correctness. We will prove the claim using induction on \(|S|\).

**Th.** Let \(k \geq 1\) be an \(n \geq 0\). Assume that when \(|S| = k\), Huffman yields an optimal tree.

**BC:** \(|S| = 1, |S| = 2\) →

**IS:** Want to prove that the tree \(T\) on alphabet \(F\) of size \(k+1\) returned by Huffman corresponds to an optimal prefix code.

**Lemma:** Let \(F\) be a binary tree in which \(y \leq x\).
the two lowest freq letters are assigned to sibling leaves at the bottommost level. Let $F'$ be the tree obtained by removing $y^*$ and $z^*$ from $F$. Then

$$\text{ABL}(F) = \text{ABL}(F') + f_{y^*} + f_{z^*}$$

**Proof Sketch:**

$F:

\begin{align*}
\text{ABL}(F) &= f_{y^*}(\text{depth}_{F}(y^*)) + f_{z^*}(\text{depth}_{F}(z^*)) \\
&= f_{y^*}(\text{depth}_{F'}(u) + 1) + f_{z^*}(\text{depth}_{F'}(u) + 1) \\
&= f_{y^*}(\text{depth}_{F'}(u) + 1) + f_{z^*}(\text{depth}_{F'}(u) + 1)
\end{align*}$
IS (cont’d): Assume for contradiction that $T$ is not an optimal tree, but $Z$ is. Among all opt. solns, we have chosen $Z$ s.t. the clas $y^* \& z^*$ are assigned to sibling leaves at the bottommost level. We know that such a tree exists by an earlier lemma. Let $Z'$ be the tree obtained from $Z$ by removing $y^* \& z^*$. By the prev lemma

\[
\text{ABL}(T) = \underbrace{\text{ABL}(T')} + \underbrace{f_{y^*} + f_{z^*}}
\]
\[ \text{ABL}(z) = \left| \text{ABL}(z') \right| + f_y + f_z \]

\[ +ve = \leq 0 \]

Contradiction.

Divide & Conquer:

\( a \quad b \quad c \quad d \quad e \)

\( 0.32 \quad 0.25 \quad 0.20 \quad 0.18 \quad 0.05 \)

- \( a \quad d \)
- \( b \quad c \quad e \)

- \( e \quad d \quad c \quad a \quad b \)
Graph Algorithms

Q: Given an undirected graph \( G = (V, E) \) and a vertex \( s \in V \) and \( t \in V \), is there a path from \( s \) to \( t \) in \( G \)? Consider the fill alg. that finds all vertices reachable from \( s \) in \( G \).

\[
R \leftarrow \{s\}
\]

\[
\text{while there is an edge } e = (u, v) \text{ that}
\]

\[
\text{return the path } (s, u, v, \ldots, t)
\]
Given the cut \((R, V \setminus R)\) do the partition of vertex.

\[ R \leftarrow R \cup \{v\} \]

\[ \text{return } R \]

Correctness:

**Lemma**: Vertex \(v \in R\) iff \(v\) belongs to the connected component containing \(s\) in \(G\).

\[ v \in R \Rightarrow v\ \text{belongs to the connected component containing } s. \]

**Proof**: let \(v\) be the first vertex brought into \(R\) that does not belong to \(R\).
cc conterary s m G. Who brings v into R? Visitor v. This means that v belongs to the cc conterary s.

Then there is a path in G from s to v given by s ~ u ~ v, contradicting that v \not\in cc conterary s.

(\Rightarrow) v \in cc conterary s \Rightarrow v \in R.

Since v \in cc conterary s, there must be a path P from s to v in G. Let y be the first vertex (going from s to v) in P s.t. y \not\in R. Let x be the vertex preceding v in P. x \in R.
If \( y \in R \) then \( y \in V \setminus R \). Then the edge \((x, y)\) must cross the cut \((R, V \setminus R)\), i.e., the alg. the alg. would not have ended, a contradiction.