Lecture Notes #9

Proof of the Optimality of Huffman Code

Lemma 1 *Let C be an alphabet in which each character $c \in C$ has frequency $f[c]$. Let x and y be two characters in C having the lowest frequencies. Then, there exists optimal prefix code for C in which the codewords for x and y have the same length and differ only in the last bit.*

Proof. (The basic idea : Let T be the tree representing any optimal prefix code of C. Modify it to obtain a tree T'' representing another optimal prefix code such that x and y appear as sibling leaves of maximum depth.)

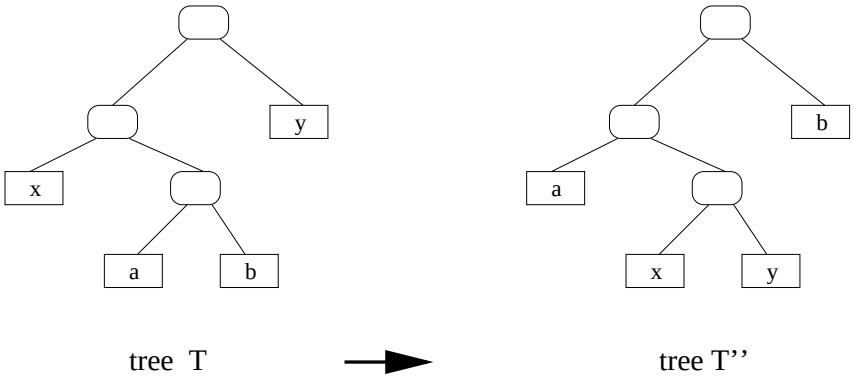

Figure 1: Transformation used in the proof of Lemma 1.

Let a and b be two characters that are sibling leaves of maximum depth in T. Without loss of generality, assume that $f[a] \leq f[b]$ and $f[x] \leq f[y]$. Since $f[x] \leq f[a]$ and $f[y] \leq f[b]$, we obtain a tree T' by exchanging a and x, and the obtain a tree T'' from T' by exchanging b and y (see Figure 2). Then,

$$\begin{aligned} B(T) - B(T') &= \sum_{c \in C} f(c) d_T(c) - \sum_{c \in C} f(c) d_{T'}(c) \\ &= f[x] d_T(x) + f[a] d_T(a) - f[x] d_{T'}(x) - f[a] d_{T'}(a) \\ &= f[x] d_T(x) + f[a] d_T(a) - f[x] d_T(a) - f[a] d_T(x) \\ &= (f[a] d_T(a) - f[x] d_T(a)) - (f[a] d_T(x) - f[x] d_T(x)) \\ &= (f[a] - f[x])(d_T(a) - d_T(x)) \\ &\geq 0. \end{aligned}$$

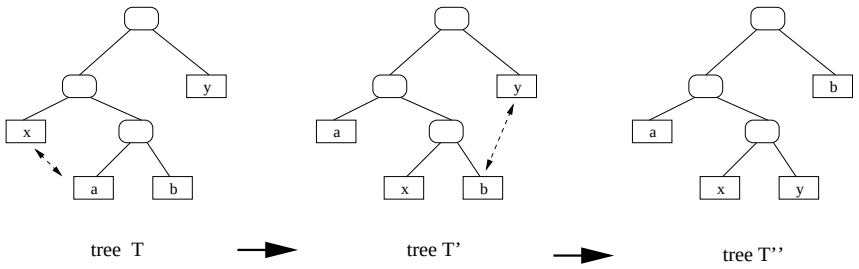

Figure 2: Transformation used in the proof of Lemma 1.

This is because $f[x]$ is the lowest frequency, and a is a leaf of maximum depth in T (i.e. $f[a] - f[x] \geq 0$ and $d_T(a) - d_T(x) \geq 0$).

Similarly, exchanging y and b does not increase the cost, and $B(T') - B(T'') \geq 0$.

Therefore, $B(T'') \leq B(T') \leq B(T)$. Since T is optimal, $B(T) \leq B(T'')$, which implies $B(T) = B(T'')$. Thus, T'' is an optimal tree for C with the property for x and y claimed in the lemma. □

Lemma 1 implies that building an optimal tree by mergers can begin with greedy choice of merging two characters of lowest frequency.

Lemma 2 *Let*

- *x and y be two characters in C with minimum frequency;*
- *C' be the alphabet C with x and y removed and new character z added (i.e. $C' = (C - \{x, y\}) \cup \{z\}$);*
- *f for C' be defined the same as for C except that $f[z] = f[x] + f[y]$;*
- *T' be any tree representing an optimal prefix code for C'.*

Then, the tree T, obtained from T' by replacing the leaf node z with an internal node with x and y as children (see Figure 3), represents an optimal prefix code for C.

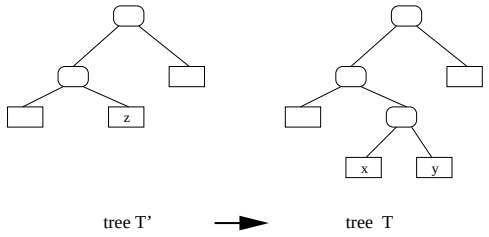

Figure 3: Obtaining T for C from T' for C' in Lemma 2.

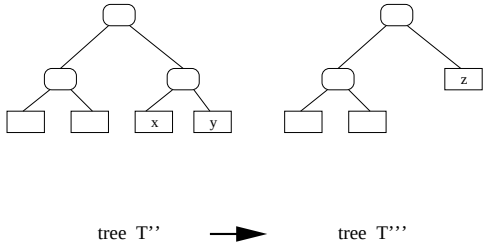

Figure 4: Obtaining T''' for C' from T'' for C in Lemma 2.

The logic of the proof of Lemma 2 outlined below may be helpful for you to understand the proof. There are two components.

Component 1:

Assume any tree T' optimal for C', and obtain T for C from T' in the way stated in the Lemma. Evaluate T to get cost $B(T) = B(T') + f[x] + f[y]$.

Component 2:

Assume that tree T is not optimal for C but tree T'' is

$\Rightarrow$ Obtain T''' for C' from T'' so that
$B(T''') = B(T'') - f[x] - f[y] < B(T) - f[x] - f[y] = B(T')$
(i.e. $B(T''') < B(T')$)

$\Rightarrow$ **Reach the contradiction to the assumption that T' is optimal for C'**

$\Rightarrow$ Conclude that T is optimal

Proof. of Lemma 2

For each $c \in C - \{x, y\}$, $d_T(c) = d_{T'}(c)$, and hence $f[c]d_T(c) = f[c]d_{T'}(c)$.

Since $d_T(x) = d_T(y) = d_{T'}(z) + 1$, we have

$$\begin{aligned} f[x]d_T(x) + f[y]d_T(y) &= (f[x] + f[y])(d_{T'}(z) + 1) \\ &= f[z]d_{T'}(z) + (f[x] + f[y]) \end{aligned}$$

because $f[x] + f[y] = f[z]$. Then,

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

i.e.

$$B(T') = B(T) - f[x] - f[y].$$

Now we prove the lemma by contradiction. Suppose that T does not represent an optimal prefix code for C. Then there exists a tree T'' such that $B(T'') < B(T)$. Without loss of generality assume that T'' has x and y as siblings (by Lemma 1). Obtain T''' from T'' by replacing the parent of x and y with leaf z of $f[z] = f[x] + f[y]$. Then, by Lemma 1, we have

$$\begin{aligned} B(T''') &= B(T'') - f[x] - f[y] \\ &< B(T) - f[x] - f[y] \\ &= B(T') \end{aligned}$$

yielding a contradiction to the assumption that T' represents an optimal prefix code for C'. Thus, T must represent an optimal prefix code for C. □

Lemma 2 shows that the problem of constructing optimal prefix codes has the optimal-substructure property.

Theorem 1 *Huffman produces an optimal prefix code.*

Proof. Immediate from Lemmas 1 and 2. □