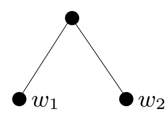
Summary on Lecture 17, February 22, 2017

Weighted Trees and Huffman algorithm

Let $L = (w_1, \ldots, w_t)$ be a list of weights. Recall that we say that a binary weighted tree T is **optimal for the weights** $L = (w_1, \ldots, w_t)$ if $W(T) \leq W(T')$ for any weighted tree T' with the same weights $L = (w_1, \ldots, w_t)$.

Here is the algorithm to find an optimal tree for a given list of weights:

Huffman$(L = \{w_1, w_2, \ldots, w_k\})$:
`{Input: A list of weights:` $L = \{w_1, w_2, \ldots, w_k\}$ `,` $k \geq 2$`}`
`{Output: an optimal tree` $T(L)$`}`
`if` $k = 2$ `then`
   `return the tree`



`else`
   `Choose two smallest weights` u `and` v `of` L`.`
   `Make a list` L' `by removing the elements` u `and` v `and adding the element` $u+v$`.`
   `Let` $T(L') :=$ **Huffman**(L')`.`
   `Form a tree` $T(L)$ `from` $T(L')$ `by replacing a leaf of weight` $u+v$
   `by a subtree with two leaves of weights` u `and` v`.`

   `return` $T(L)$.

Now we returen to the example above to merge the lists L_1, L_2, L_3, L_4, and L_5 with $|L_1| = 15$, $|L_2| = 22$, $|L_3| = 31$, $|L_4| = 34$, and $|L_5| = 42$. We run the algorithm **Huffman**$(L = \{15, 22, 31, 34, 42\})$ and we get the following weighted tree:

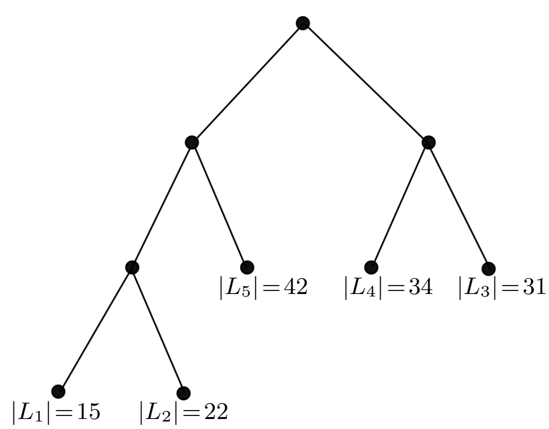


Fig. 3.

We get the the following total number of comparisons:

$$W(T) - 4 = 3 \cdot |L_1| + 3 \cdot |L_2| + 2 \cdot |L_3| + 2 \cdot |L_4| + 2 \cdot |L_1| - 4 = 3 \cdot 15 + 3 \cdot 22 + 2 \cdot 31 + 2 \cdot 34 + 2 \cdot 42 - 4 = 321.$$

Now we will show that the algorithm **Huffman**(L) indeed works. Let $w_1, w_2, \ldots, w_k$ be the weights, and let T be an optimal tree with those weights. We denote by ℓ_j the level of the vertex labeled by w_j.

Lemma 1. *Let T be an optimal tree with the weights $w_1, w_2, \ldots, w_k$. Then if $w_i < w_j$, then $\ell_i \geq \ell_j$.*

Proof. Assume that $w_i < w_j$ and $\ell_i < \ell_j$ for an optimal tree T. We denote by T' the tree which is obtained from T by interchanging the weights w_i and w_j. We obtain:

$$W(T) - W(T') \quad = \quad w_i\ell_i + w_j\ell_j - w_i\ell_j - w_j\ell_i = (w_j - w_i)(\ell_j - \ell_i) > 0$$

Thus $W(T) > W(T')$, i.e. T is not an optimal tree. Contradiction. Hence $w_i < w_j$ implies $\ell_i \geq \ell_j$ for an optimal tree. □

Lemma 2. *Let $w_1 \leq w_2 \leq \cdots \leq w_k$. Then there exists an optimal tree for those weight such that w_1 and w_2 are at the lowest level ℓ.*

Proof. Let T be an optimal tree, and w_i and w_j are at the lowest level ℓ. If $w_1 < w_i$, then $\ell_1 \geq \ell$. This means that $\ell_1 = \ell$ since ℓ is the lowest level. If $w_1 = w_j$, then we can interchange the weights w_1 and w_j without changing the weight of the tree. Similarly, by interchanging w_2 and w_j if necessary, we obtain an optimal tree with w_1 and w_2 at the lowest level. □

Now we are ready to prove that the algorithm **Huffman**(L) indeed works.

Theorem. *Let $w_1 \leq w_2 \leq w_3 \leq \cdots \leq w_k$, and T_0 be an optimal tree for the weights $w_1 + w_2, w_3, \ldots w_k$. Then the tree T, obtained from T_0 by replacing the leaf $w_1 + w_2$ by a subtree with the weights w_1 and w_2, is an optimal tree for the weights $w_1 \leq w_2 \leq w_3 \leq \cdots \leq w_k$.*

Proof. Clearly, there are only finite number of binary trees with k leaves. Then it means that there exists an optimal tree T' with given weights $w_1 \leq w_2 \leq w_3 \leq \cdots \leq w_k$. By Lemma 2, we can assume that the weights w_1 and w_2 have both the lowest weight ℓ. Moreover, since T is a binary tree, we can assume that w_1 and w_2 are children of the same parent. Indeed, if w_1 has a sibling w_i with $i > 2$, we interchange w_2 and w_i. Let p be a common parent of w_1 and w_2.

We denote by T_p the subtree with the root p and two children w_1 and w_2. Then the weight of the tree remains the same. Now we denote by T'_0 the tree obtained from T' by replacing the subtree T_p by a leaf with the weight $w_1 + w_2$. Now we find that

$$W(T') - W(T'_0) = \ell(w_1 + w_2) - (\ell - 1)(w_1 + w_2) = w_1 + w_2$$

Thus $W(T') = W(T'_0) + (w_1 + w_2)$. Similary, we obtain that $W(T) = W(T_0) + w_1 + w_2$. Since T' is an optimal tree for the weights $w_1 \leq w_2 \leq w_3 \leq \cdots \leq w_k$, we obtain that $W(T) \leq W(T')$, or we have that

$$W(T'_0) + (w_1 + w_2) \leq W(T_0) + w_1 + w_2$$

Thus $W(T'_0) \leq W(T_0)$. Since T_0 is an optimal tree, we obtain that $W(T'_0) \geq W(T_0)$, i.e. $W(T_0) = W(T'_0)$, i.e. T'_0 is an optimal tree. This shows that the algorithm **Huffman**(L) delivers an optimal tree. □

Exercise. Show that the complexity of the algorithm **Huffman**(L) is at least $O(k^2)$, where k is the number of weights. Find a way to improve it to $O(k \log_2 k)$.

Exercise. Construct an optimal binary tree for the following sets of weights and compute the weight of the optimal tree.

(a) $L = \{1, 3, 4, 6, 9, 13\}$,

(b) $L = \{1, 3, 5, 6, 10, 13, 16\}$,

(c) $L = \{2, 4, 5, 8, 13, 15, 18, 25\}$,

(d) $L = \{1, 2, 3, 5, 8, 13, 21, 34\}$.