CS 13

Mathematical Foundations of Computing

CS 13: Mathematical Foundations of Computing

Lecture 12: Huffman Compression

Broken Code

```
dictionary = {0: “A”, 01: “B”, 10: “C”}

0010010
0 0 10 01 0
0 01 0 01 0
0 01 0 0 10
```

Prefix-Free Code

```
dictionary = {0: “A”, 10: “B”, 110: “C”}

001011011000

0 0 10 110 110 0 0

A A B  C   C   A A
```

Compressing Text

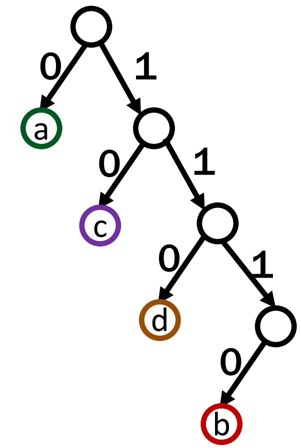
aabaacdacdacdac → {a:7, b:1, c:4, d: 3}

→ {a→0,b→1110,c→10,d→110}

→ 001110001011001011001011001 0

Decompressing Text

001110001011001011001011001010

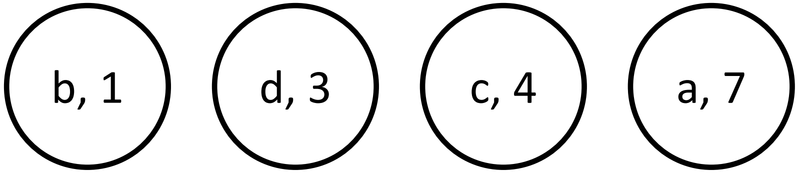


0 0 1110 0 0 10 110 0 10 110 0 10 110 0 10

a a b a a c d a c d a c d a c

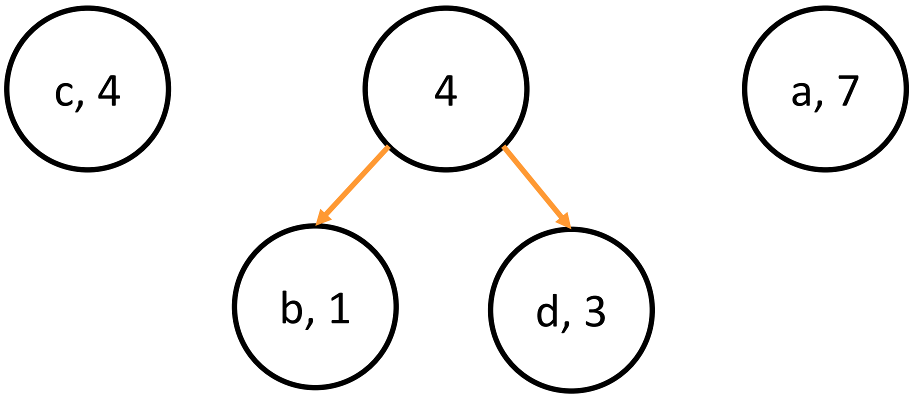
Huffman Coding

aabaacdacdacdac → {a:7, b:1, c:4, d: 3}



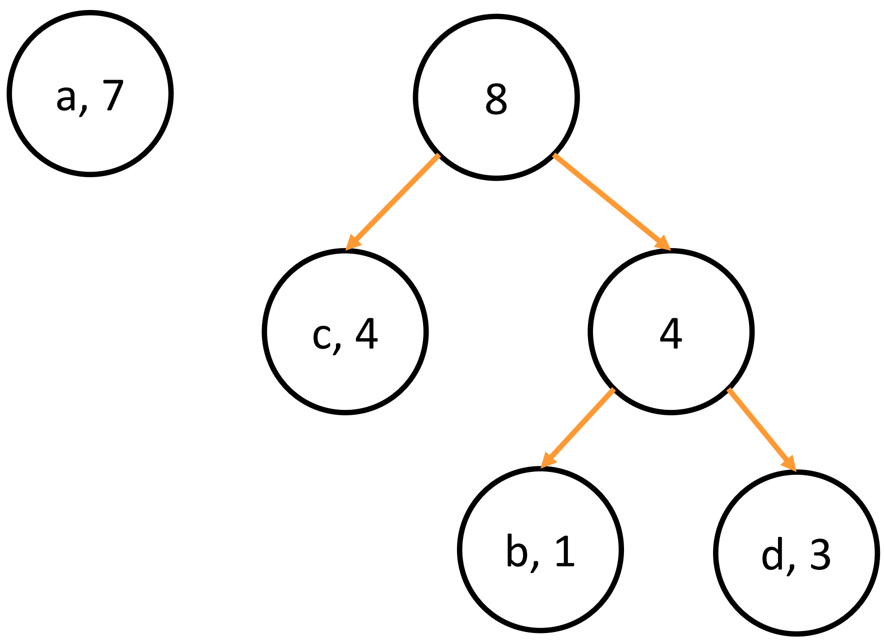
Huffman Coding

aabaacdacdacdac → {a:7, b:1, c:4, d: 3}



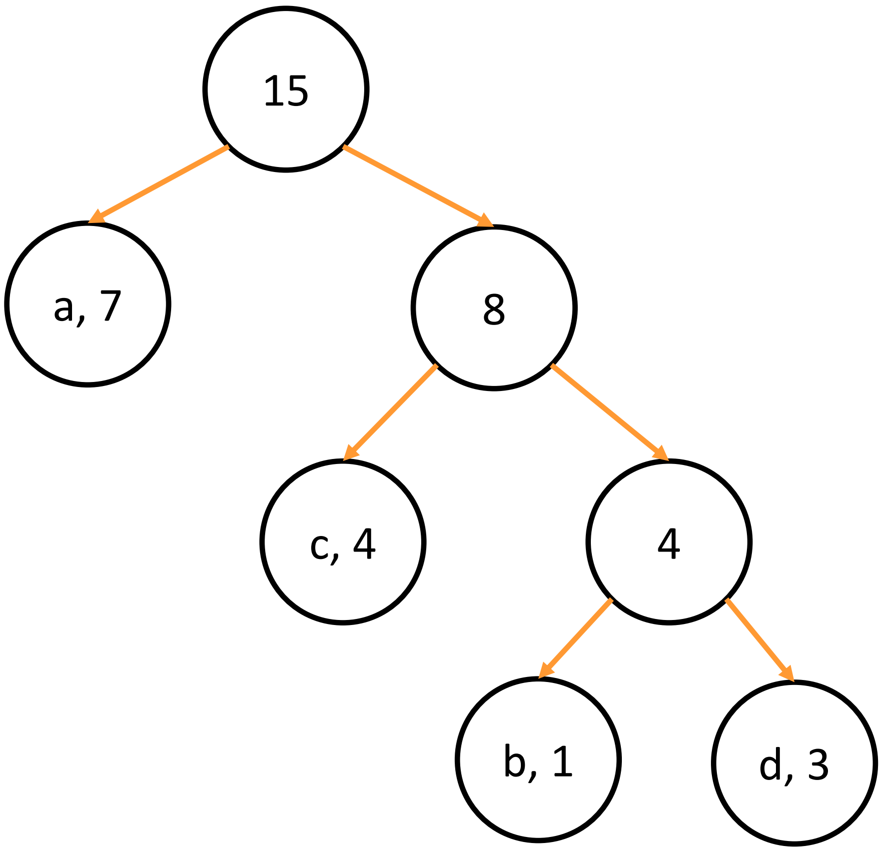
Huffman Coding

aabaacdacdacdac → {a:7, b:1, c:4, d: 3}



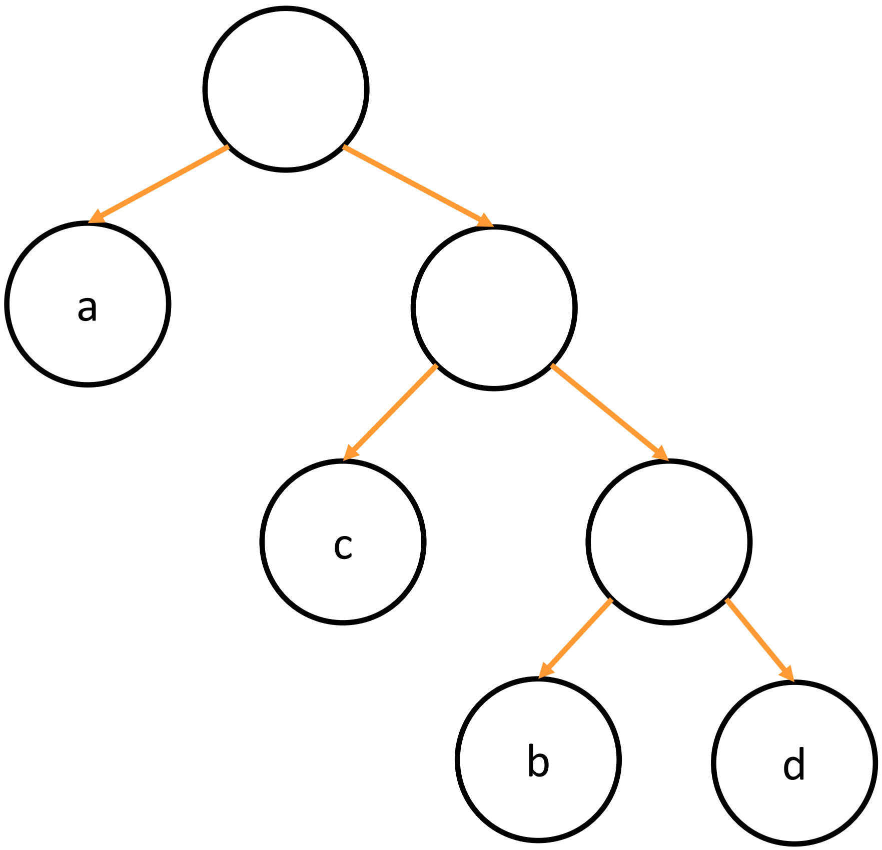
Huffman Coding

aabaacdacdacdac → {a:7, b:1, c:4, d: 3}



Huffman Coding

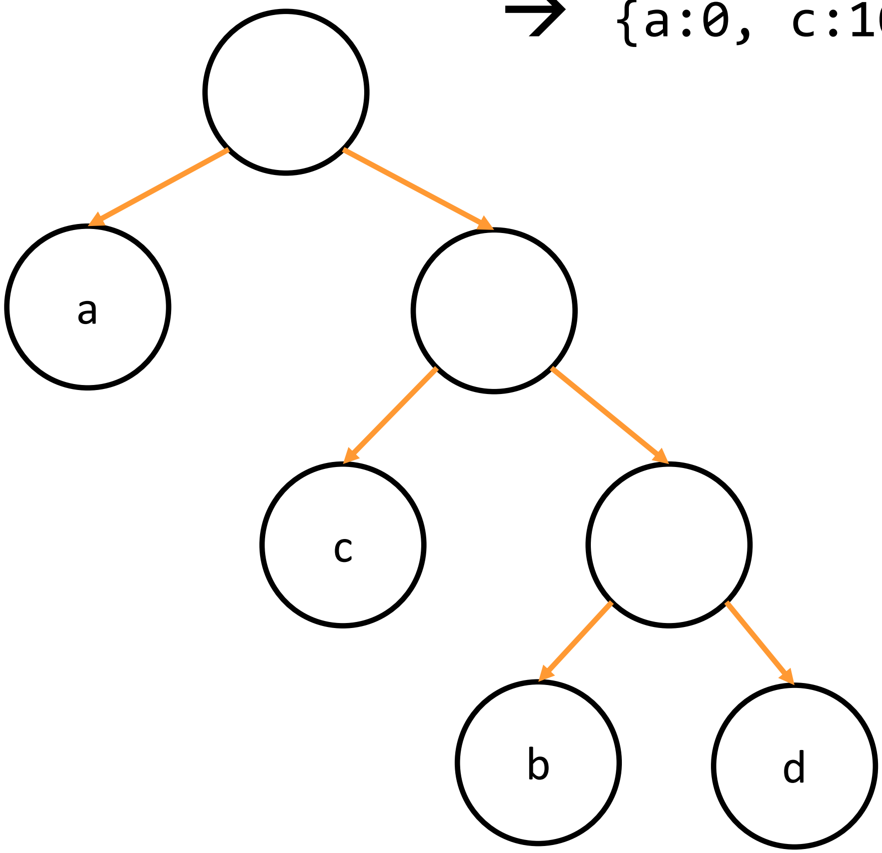
aabaacdacdacdac → {a:7, b:1, c:4, d: 3}

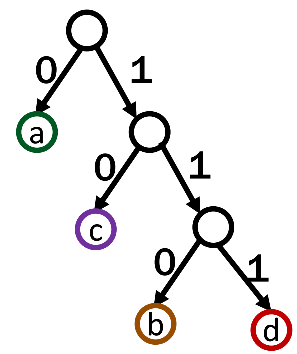


Huffman Coding

aabaacdacdacdac → {a:7, b:1, c:4, d: 3}

→ {a:0, c:10, b:110, d: 111}





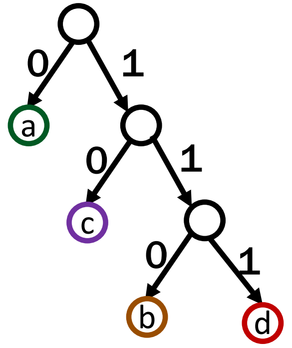
Prefix-Free Codes are Full Binary Trees

Definition: "full binary tree"

A full binary tree is a tree where every node has either zero or two children.

Every prefix-free code can be represented by a full binary tree

`{a:0, c:10, b:110, d: 111}` ⟷



The leaves represent symbols and the path represents the code.

Optimal Prefix-Free Codes

Let $\mathrm{len}_{\mathrm{code}}(s)$ to be the number of bits required by code to represent s. Let $\mathrm{depth}_{\mathrm{code}}(s)$ to be number of edges from the root to the leaf representing s in the tree corresponding to code.

Given symbol frequencies, f_i, and symbols s_i, an optimal prefix-free code minimizes:

$$\mathrm{cost}(\mathrm{code}) = \sum_{i}^{n} f_i \cdot \mathrm{len}_{\mathrm{code}}(s_i) = \sum_{i}^{n} f_i \cdot \mathrm{depth}_{\mathrm{code}}(s_i)$$

Huffman Codes = Optimal Prefix-Free Codes

It turns out Huffman's Algo generates optimal prefix-free codes!

Deep Siblings Lemma

In an optimal prefix-free code tree, two of the least frequent symbols are siblings at the greatest depth.

$$\text{cost}(\text{code}) = \sum_{i}^{n} f_i \cdot \text{len}_{\text{code}}(s_i) = \sum_{i}^{n} f_i \cdot \text{depth}_{\text{code}}(s_i)$$

Huffman Codes = Optimal Prefix-Free Codes

It turns out Huffman's Algo generates optimal prefix-free codes!

Deep Siblings Lemma

In an optimal prefix-free code tree, two of the least frequent symbols are siblings at the greatest depth.

$$\text{cost}(\text{code}) = \sum_{i}^{n} f_i \cdot \text{len}_{\text{code}}(s_i) = \sum_{i}^{n} f_i \cdot \text{depth}_{\text{code}}(s_i)$$

Note that the tree is full; so, the deepest leaves must be siblings. Then, we show the least frequent symbols are always the deepest leaves.

Huffman Codes = Optimal Prefix-Free Codes

It turns out Huffman's Algo generates optimal prefix-free codes!

Deep Siblings Lemma

In some optimal prefix-free code tree, two of the least frequent symbols are siblings at the greatest depth.

$$\mathrm{cost}(\mathrm{code}) = \sum_{i}^{n} f_i \cdot \mathrm{len}_{\mathrm{code}}(s_i) = \sum_{i}^{n} f_i \cdot \mathrm{depth}_{\mathrm{code}}(s_i)$$

Note that the tree is full; so, the deepest leaves must be siblings. Then, we show the least frequent symbols are always the deepest leaves.

Suppose for contradiction that they aren't the deepest leaves. Then, there must be some other symbol at a deepest leaf. Swapping that symbol with the least frequent symbol will result in a smaller cost sum. This means the tree wasn't optimal.

Huffman Codes = Optimal Prefix-Free Codes

$$\text{cost}(\text{code}) = \sum_{i}^{n} f_i \cdot \text{len}_{\text{code}}(s_i) = \sum_{i}^{n} f_i \cdot \text{depth}_{\text{code}}(s_i)$$

We go by induction on the number of symbols.

BC (n = 2). There is only one full binary tree with two leaves.
IH. Suppose the claim is true for all codes with n symbols.
IS. We show the claim is true for $n + 1$ symbols.

Huffman Codes = Optimal Prefix-Free Codes

$$\text{cost}(\text{code}) = \sum_{i}^{n} f_i \cdot \text{len}_{\text{code}}(s_i) = \sum_{i}^{n} f_i \cdot \text{depth}_{\text{code}}(s_i)$$

We go by induction on the number of symbols.

BC (n = 2). There is only one full binary tree with two leaves.
IH. Suppose the claim is true for all codes with n symbols.
IS. We show the claim is true for $n + 1$ symbols. Let H_{n+1} be the tree generated by Huffman's Algorithm for the frequencies

$$f_0 < f_1 < \cdots < f_n$$

Huffman Codes = Optimal Prefix-Free Codes

$$\text{cost}(\text{code}) = \sum_{i}^{n} f_i \cdot \text{len}_{\text{code}}(s_i) = \sum_{i}^{n} f_i \cdot \text{depth}_{\text{code}}(s_i)$$

We go by induction on the number of symbols.

BC (n = 2). There is only one full binary tree with two leaves.
IH. Suppose the claim is true for all codes with n symbols.
IS. We show the claim is true for $n+1$ symbols. Let H_{n+1} be the tree generated by Huffman's Algorithm for the frequencies

$$f_0 < f_1 < \cdots < f_n$$

Let T be some optimal tree for this set of frequencies.

Huffman Codes = Optimal Prefix-Free Codes

$$\text{cost}(\text{code}) = \sum_{i}^{n} f_i \cdot \text{len}_{\text{code}}(s_i) = \sum_{i}^{n} f_i \cdot \text{depth}_{\text{code}}(s_i)$$

We go by induction on the number of symbols.

BC (n = 2). There is only one full binary tree with two leaves.
IH. Suppose the claim is true for all codes with n symbols.
IS. We show the claim is true for $n + 1$ symbols. Let H_{n+1}be the tree generated by Huffman's Algorithm for the frequencies

$$f_0 < f_1 < \cdots < f_n$$

Let T be some optimal tree for this set of frequencies. We show

$$\text{cost}(H_{n+1}) \leq \text{cost}(T)$$

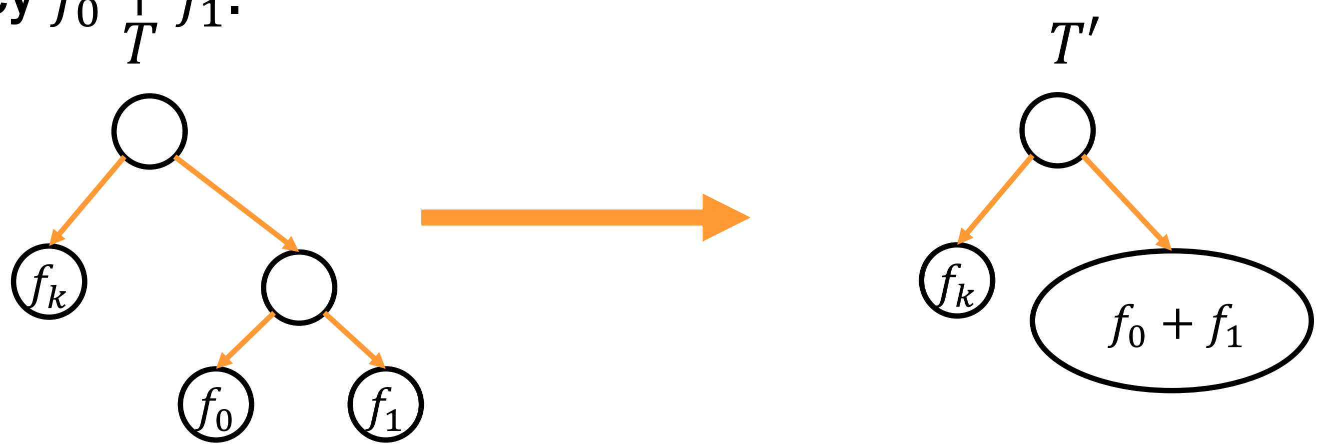
Thus, showing H_{n+1} is also an optimal code.

Huffman Codes = Optimal Prefix-Free Codes

$$\text{cost}(\text{code}) = \sum_i^n f_i \cdot \text{len}_{\text{code}}(s_i) = \sum_i^n f_i \cdot \text{depth}_{\text{code}}(s_i)$$

Let H_{n+1} be the tree generated by Huffman's Algorithm for the frequencies $f_0 < f_1 < \cdots < f_n$. Let T be some optimal tree for this set of frequencies.

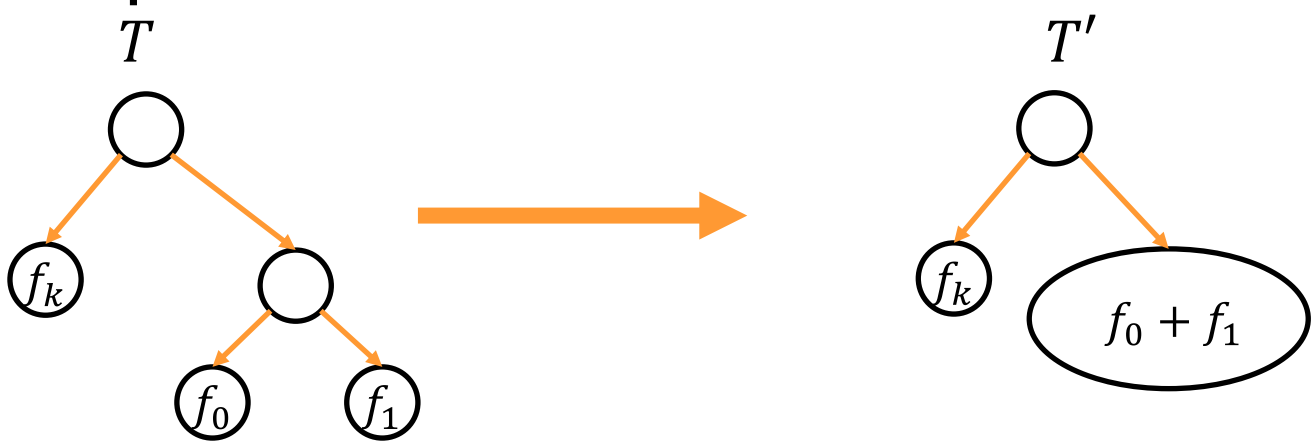
Now, we transform $H_{n+1} \to H'_{n+1}$ and $T \to T'$ by removing their leaves and replacing their parent with a merged symbol with frequency $f_0 + f_1$.



Huffman Codes = Optimal Prefix-Free Codes

$$\text{cost}(\text{code}) = \sum_{i}^{n} f_i \cdot \text{len}_{\text{code}}(s_i) = \sum_{i}^{n} f_i \cdot \text{depth}_{\text{code}}(s_i)$$

Let H_{n+1} be the tree generated by Huffman's Algorithm for the frequencies $f_0 < f_1 < \cdots < f_n$. Let T be some optimal tree for this set of frequencies.



Note that H'_{n+1} is exactly the tree in the previous step of Huffman's algorithm. Then, by our IH, we have

$$\text{cost}(H'_{n+1}) \leq \text{cost}(T')$$

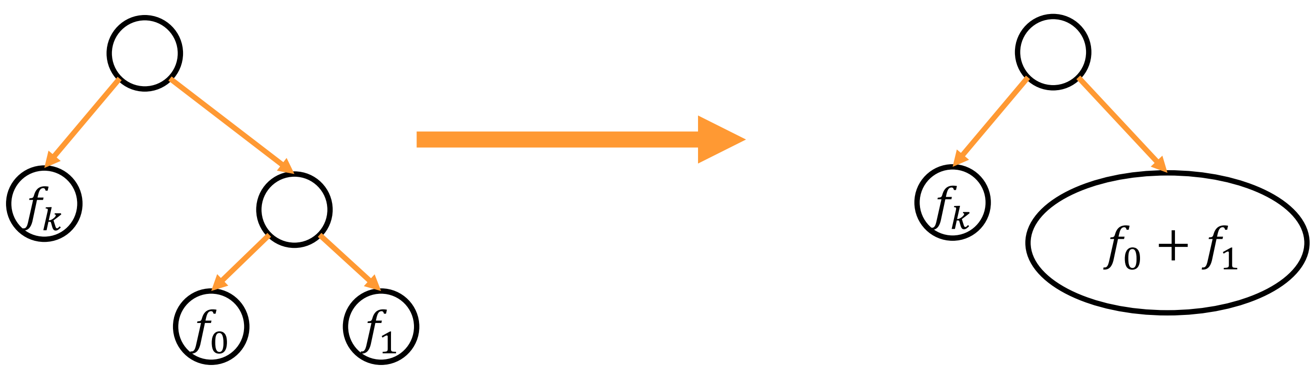
Huffman Codes = Optimal Prefix-Free Codes

$$\text{cost}(\text{code}) = \sum_{i}^{n} f_i \cdot \text{len}_{\text{code}}(s_i) = \sum_{i}^{n} f_i \cdot \text{depth}_{\text{code}}(s_i)$$

By our IH, we have $\text{cost}(H'_{n+1}) \leq \text{cost}(T')$**. By construction:**

$$\text{cost}(T') = (f_0 + f_1) \cdot (\text{depth}_T(s_i) - 1) + \sum_{i=2}^{n} f_i \cdot \text{depth}_T(s_i)$$

$$=$$

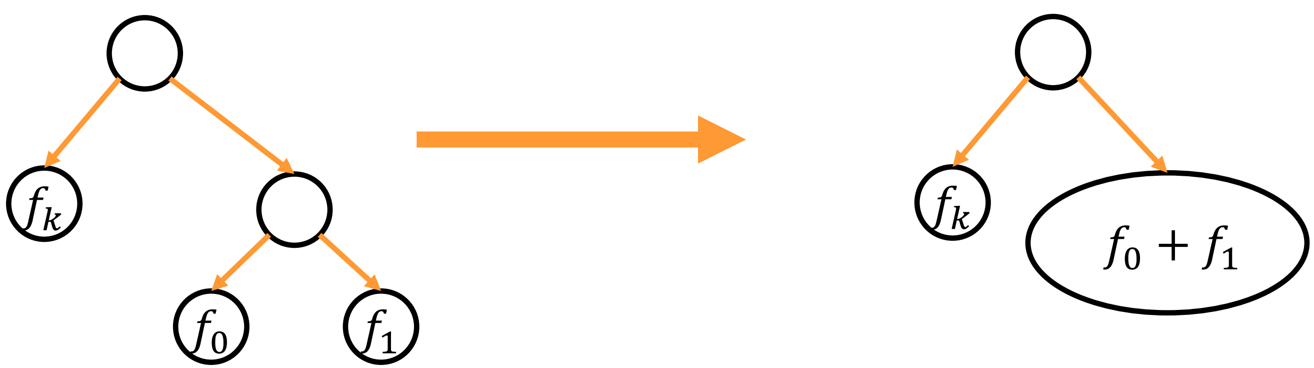


Huffman Codes = Optimal Prefix-Free Codes

$$\text{cost}(\text{code}) = \sum_{i}^{n} f_i \cdot \text{len}_{\text{code}}(s_i) = \sum_{i}^{n} f_i \cdot \text{depth}_{\text{code}}(s_i)$$

By our IH, we have $\text{cost}(H'_{n+1}) \leq \text{cost}(T')$. **By construction:**

$$\begin{aligned}\text{cost}(T') &= (f_0+f_1)\cdot(\text{depth}_T(s_i)-1) + \sum_{i=2}^{n} f_i \cdot \text{depth}_T(s_i) \\ &= \left(\sum_{i=0}^{n} f_i \cdot \text{depth}_T(s_i)\right) - (f_0+f_1) \\ &= \text{cost}(T) - (f_0+f_1)\end{aligned}$$



Huffman Codes = Optimal Prefix-Free Codes

$$\text{cost}(\text{code}) = \sum_{i}^{n} f_i \cdot \text{len}_{\text{code}}(s_i) = \sum_{i}^{n} f_i \cdot \text{depth}_{\text{code}}(s_i)$$

By our IH, we have

(a) $\text{cost}(H'_{n+1}) \leq \text{cost}(T')$.

By construction:

(b) $\text{cost}(T') = \text{cost}(T) - (f_0 + f_1)$

(c) $\text{cost}(H'_{n+1}) = \text{cost}(H_{n+1}) - (f_0 + f_1)$

Thus:

$\text{cost}(H_{n+1}) =$

Huffman Codes = Optimal Prefix-Free Codes

$$\text{cost}(\text{code}) = \sum_{i}^{n} f_i \cdot \text{len}_{\text{code}}(s_i) = \sum_{i}^{n} f_i \cdot \text{depth}_{\text{code}}(s_i)$$

By our IH, we have

(a) $\text{cost}(H'_{n+1}) \leq \text{cost}(T')$.

By construction:

(b) $\text{cost}(T') = \text{cost}(T) - (f_0 + f_1)$

(c) $\text{cost}(H'_{n+1}) = \text{cost}(H_{n+1}) - (f_0 + f_1)$

Thus:

$$\begin{aligned}\text{cost}(H_{n+1}) &= \text{cost}(H'_{n+1}) + (f_0 + f_1) \\ &\leq \text{cost}(T') + (f_0 + f_1) \\ &= \text{cost}(T)\end{aligned}$$

which is what we were trying to prove!