Binary Codes

$\Sigma = \{a, b, c, d\}$

$\downarrow$

def: Given an alphabet Σ, a binary code is a function $f: \Sigma \to \{0,1\}^*$

ex: Morse code, ASCII, Braille

Suppose you have a message where the letter "a" occurs 50% of the time, "b" 30%, and "c" 20%. Which is the best binary encoding of $\Sigma = \{a, b, c\}$?

A): $f(a) = 00$, $f(b) = 01$, $f(c) = 10$

B) $f(a) = 0$, $f(b) = 1$, $f(c) = 01$

C) $f(a) = 0$, $f(b) = 10$, $f(c) = 11$

A) $f(a)=00$
$f(b)=01$
$f(c)=10$

Clear for decoding, but doesn't take advantage of different prob.

B) $f(a)=0$
$f(b)=1$
$f(c)=01$

Ambiguous for decoding

01 → c
→ a b

C) $f(a)=0$
$f(b)=10$
$f(c)=11$

No ambiguity for decoding, average bits per is small

\# of bits in $f(i)$

Average letter length: $L(f) = \sum_{i \in \Sigma} |f(i)| \cdot p(i)$

$2 \cdot .5 + 2 \cdot .3 + 2 \cdot .2 = 2$

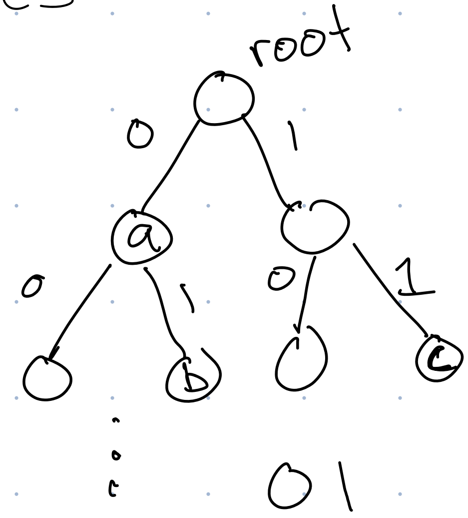
ex: $|f(a)| \cdot p(a) + |f(b)| \cdot p(b) + |f(c)| \cdot p(c) =$

$1 \cdot .5 + 2 \cdot .3 + 2 \cdot .2 = 1.5$

Binary Trees & Binary Codes

$f(a) = 0$
$f(b) = 01$
$f(c) = 11$

$\longleftrightarrow$



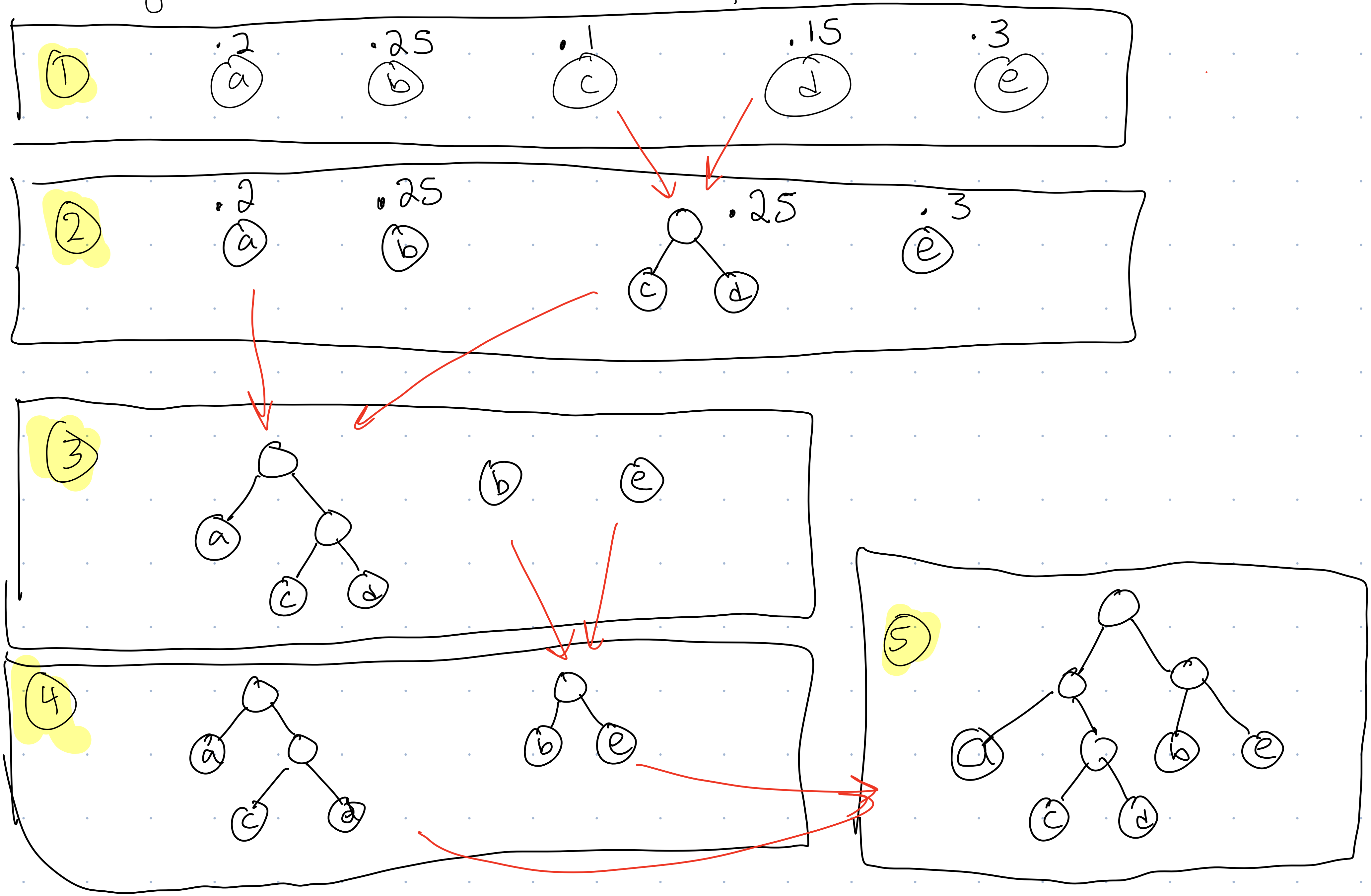
$|f(i)| = d(i) =$ depth of node i

$a \rightarrow \boxed{0}$
$b \rightarrow \boxed{0}\,1$

Ambiguity arises if multiple letters lie on same path.

<u>def</u>: A code is "prefix free" if all letters are at leaves in corresponding binary tree

Merge Trees to Create Prefix Free Codes



Optimal Binary Encoding Problem

Input:
- Σ (alphabet of symbols)
- $p : \Sigma \to \mathbb{R}$ (probabilities / frequencies for each symbol)

Output:

$f : \Sigma \to \{0,1\}^*$ such that

- f is prefix free ← constraint
- minimize average letter length $L(f)$

Objective function

Huffman's Algorithm $O(n^2)$

For each $i \in \Sigma$: $O(n)$

- Create a tree with node labelled i $O(1)$
- Give tree weight $p(i)$

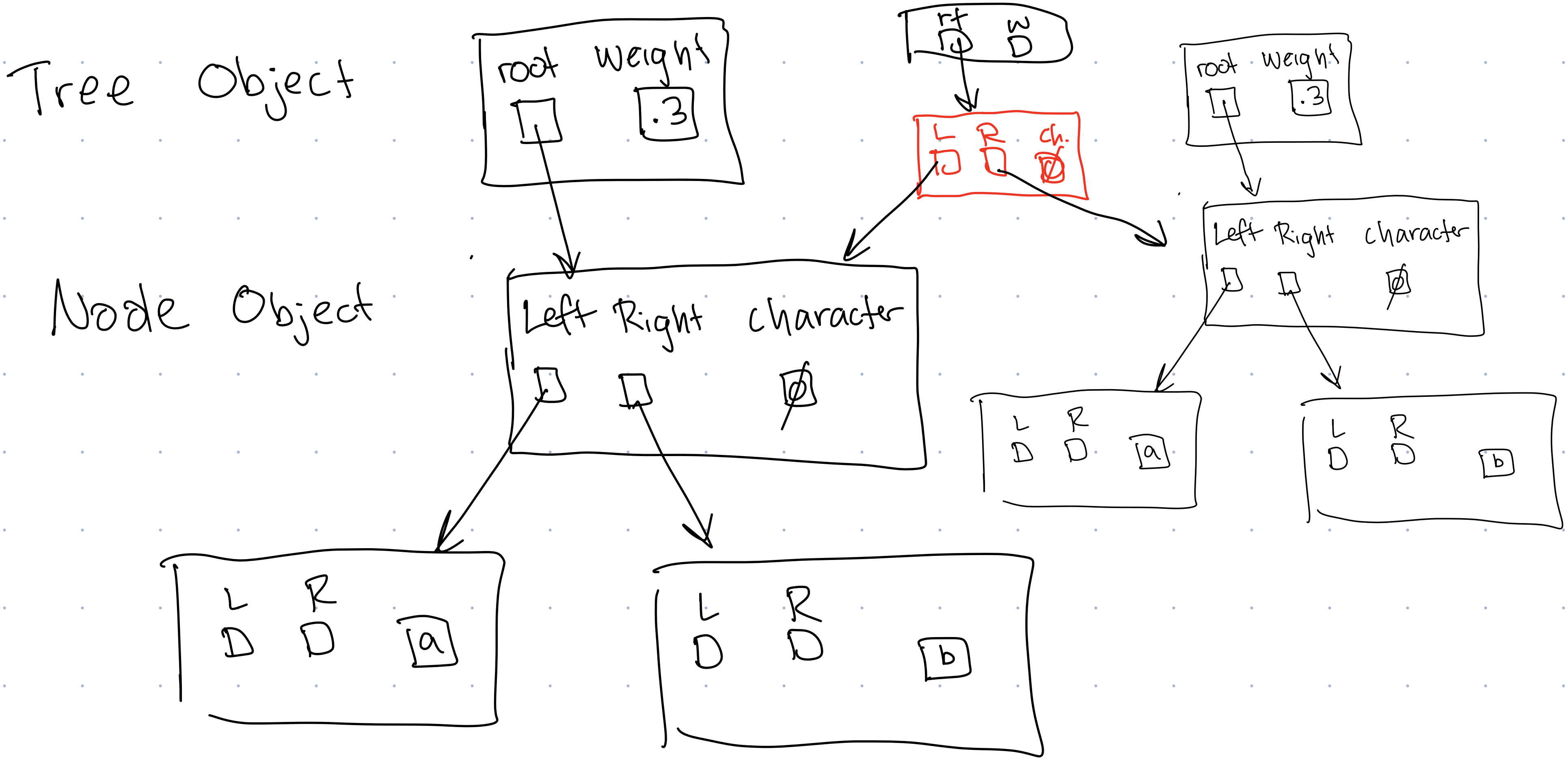
While there is more than one tree in our forest: $O(n)$

- $\boxed{\text{Merge 2 trees}}$ $O(1)$ $\boxed{\text{with smallest weight}}$ $O(n)$
- Set weight of merged tree to be sum of 2 weights $O(1)$

---

i	$p(i)$
a	.3
b	.25
c	.2
d	.15
e	.1

- Use Huffman's algorithm to create a binary code
- What is the average letter length of your code 2.25
- What is the runtime of Huffman's in terms of $|\Sigma| = n$? Ideas to improve?
- Why greedy?
- Midd Bucket list


Tree Object
Node Object
root
weight
.3
Left
Right
character
∅
L
R
a
b
rt
w
ch.

Huffman's Algorithm

For each $i \in \Sigma$: $\left.\begin{array}{l}\text{• Create a tree with one node, label } i\\ \text{• Give tree weight } p(i)\end{array}\right\} O(n\log n)$

While there is more than one tree: $O(n)$

- $\boxed{\text{Merge}}$ $O(1)$ $\boxed{\text{two trees with smallest weights}}$ → 2 pops $O(\log n)$
- Set weight of merged tree to be sum of weights
- Reinsert into heap ← $O(\log n)$

$O(n\log n)$

Min Heap to store trees

- Initialize n items in heap → $O(n\log n)$
- Pop min-value off heap → $O(\log n)$
- Push new item into heap → $O(\log n)$

Why greedy: Order using a simple score function and take the object(s) with the best score(s)

Huffman's Algorithm

For each $i \in \Sigma$:

- Create a tree with one node, label i
- Give tree weight $p(i)$

While there is more than one tree: n

- Merge two trees with smallest weights
- Set weight of merged tree to be sum of weights

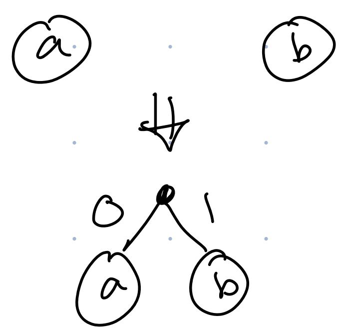
Improve runtime?

Thm: Huffman's Algorithm produces a prefix free code that minimizes average letter length.

Pf: We will prove correctness by induction on $n = |\Sigma|$.

Base case: If $n = 2$, there are two letters, a, b.

Huffman



This is optimal because there is no code with less than 1 bit per letter.

Inductive step: Assume for induction that Huffman's alg is optimal for any alphabet with k characters. Consider an alphabet Σ s.t. $|\Sigma| = k+1$.

Let a, b be letters with smallest weight in Σ.

Define $\Sigma^- = \Sigma - \{a, b\} \cup \{a/b\}$. Note $|\Sigma^-| = k$

Set $p(a/b) = p(a) + p(b)$ $\quad$ a/b is a single letter

ex:

$\Sigma = \{e, f, g, h\}$ $\quad p(e) = .1 \quad p(f) = .7 \quad p(g) = .15 \quad p(h) = .05$

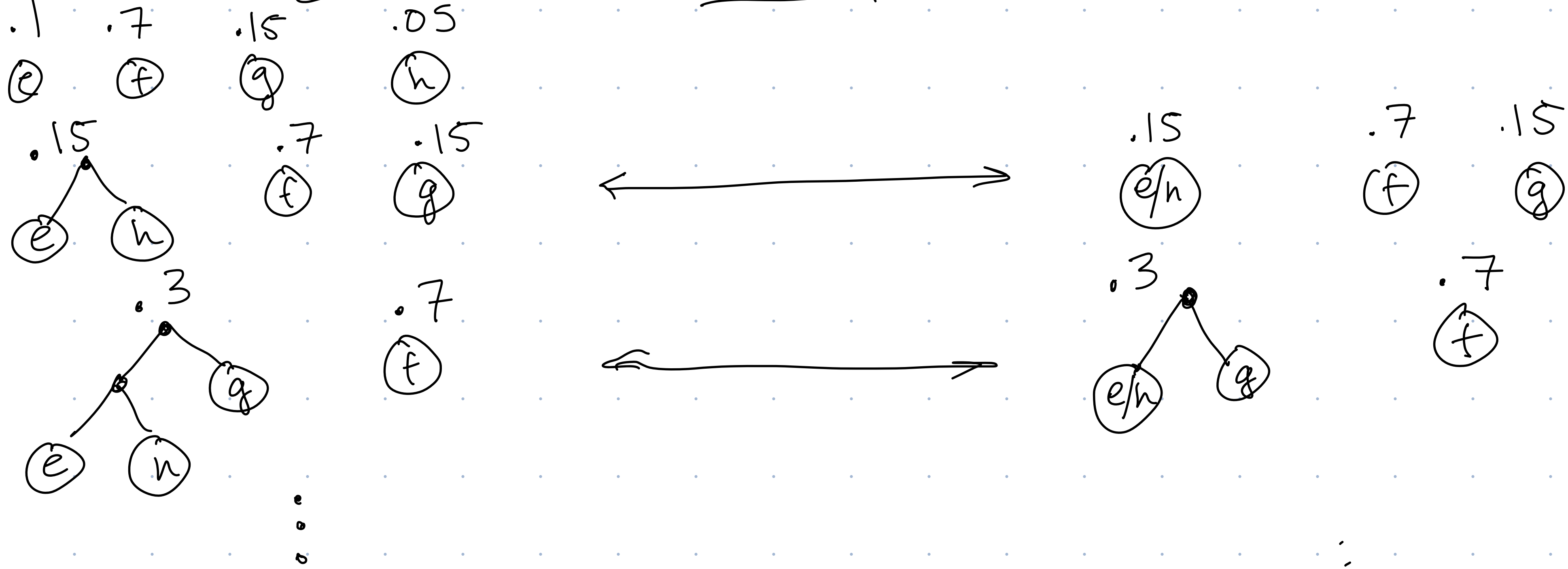
then

ex:

$\Sigma = \{e, f, g, h\}$ $\quad p(e) = .1 \quad p(f) = .7 \quad p(g) = .15 \quad p(h) = .05$

$\bar{\Sigma} = \{e/h, g, f\}$ $\quad p(e/h) = .15 \quad p(f) = .7 \quad p(g) = .15$

Σ Huffman $\bar{\Sigma}$

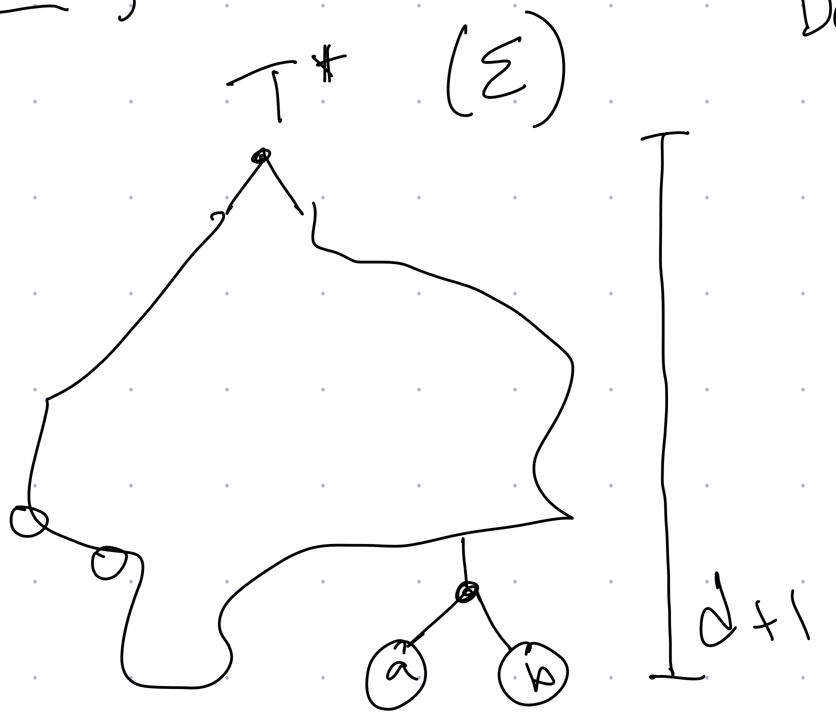


In general:

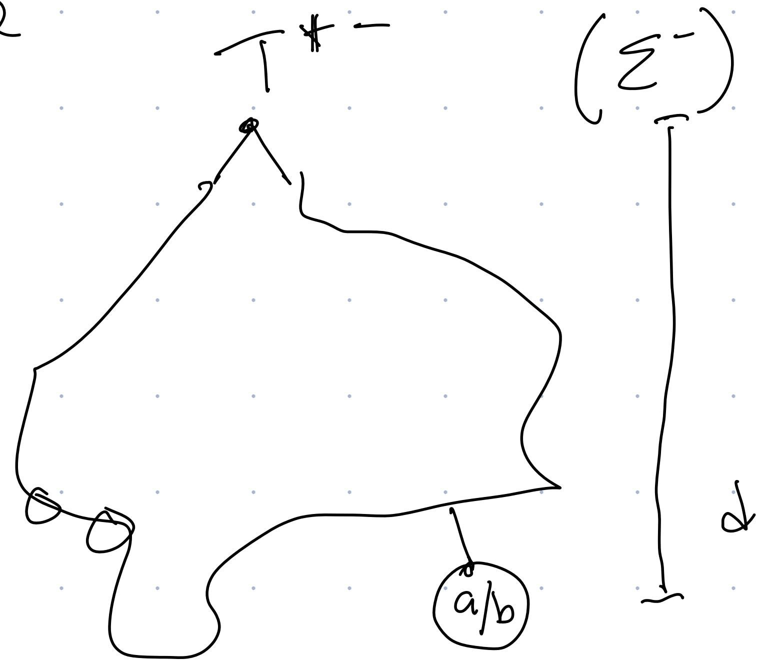
By inductive assumption T^- is optimal because it has k characters and was produced by Huffman's.

Lemma: There is an optimal tree for Σ where a, b are siblings.

Suppose for contradiction that T is not optimal. Let $T^* \neq T$ be optimal with a, b siblings (can do this by <u>Lemma</u>).



Define



Then

$p(a), p(b), d$

$$L(T^*) = \sum_{i \neq a,b \in \Sigma} p(i)\,d(i) + \boxed{}$$

$$L(T^{*-}) = \sum_{i \neq a/b} p(i)\,d(i) + \boxed{}$$

So

$$L(T^*) - L(T^{*-}) = \boxed{}$$

Similarly

$$L(T) - L(T^-) = \boxed{}$$

Thus

$$L(T^*) - L(T^{*-}) = L(T) - L(T^-)$$

Rearranging: $L(T) - L(T^*) = \boxed{}$

This is a contradiction because

$\boxed{}$

Lemma: There is an optimal tree for Σ with a, b (characters with smallest p-values) siblings.