S. KIMMEL

Goals

- Describe pre-fix free codes & why they are desirable
- Describe Huffman's algorithm
- Prove correctness of Huffman's algorithm

Reminder: Spring Symposium Bonus

QuickSort Clarification

I wrote:

conditional prob.

$$\Pr(z_i, z_j \text{ compared}) = \Pr\left(z_i, z_j \text{ chosen as pivot} \;\middle|\; \text{an element of } \{z_i, z_{i+1}, \cdots z_j\} \text{ chosen as pivot}\right)$$

but $P(A) = P(A|B)P(B)$ (Bayes Rule)

I should have written:

$$\Pr(z_i, z_j \text{ compared}) = \Pr\left(z_i, z_j \text{ chosen as pivot} \;\middle|\; \text{an element of } \{z_i, z_{i+1}, \cdots z_j\} \text{ chosen as pivot}\right) \times \Pr\left(\text{an element of } \{z_i, z_{i+1}, \cdots z_j\} \text{ chosen as pivot}\right)$$

Equals 1 !

Because we get to subarrays of size 1, one is eventually always chosen as a pivot.

S. KIMMEL

Huffman Codes

Binary Code: $f : \Sigma \to \{0,1\}^*$ (f maps characters to bit strings)

e.g. $\Sigma = \{a, b, \ldots z, \_, ', !, ? \ldots\}$ → each letter gets unique string from $\{0,1\}^5$

32 letters

→ ASCII

e.g. $\Sigma = \{a, b, c\}$

Suppose you have a message where a occurs 50%, b occurs 30%, c occurs 20%.

Q. What is the best and most efficient binary encoding of a, b, c to send message?

A) a → 00
b → 01
c → 10

Average length: 2

0110
b c

B) a → 0
b → 1
c → 01

0110 = abba
or
cbba

C) a = 0
b = 11
c = 01

0110

- a or beginning of c
- start of b, end of c
- start of b, end of b
- Need to go to 4th to figure out

D) a = 0
b = 10
c = 11

0110
a c a

average length

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

Length of encoding of i

Prob. of i

Average length $= L(f) = \sum_{i \in \Sigma} \ell(f(i)) p_i$

Prefix-free code: $\forall\ i, j \in \Sigma$, $f(i)$ is not prefix of $f(j)$

e.g.

f
a → 0
b → 11
c → 01

Not prefix-free

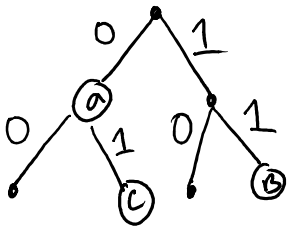
$f(a)$ is prefix for $f(c)$

Don't know if 0 is a or start of c

Problem: Given probability p_i for each $i \in \Sigma$
what is pre-fix free code w/ smallest average length?

Subproblem: how to ensure prefix-free?

Fact: Binary codes ⟷ binary trees



a → 0
b → 11
c → 01

- $\forall b \in \Sigma$, there is a vertex with label b
- Encoding of b is path from root to vertex b

S. KIMMEL

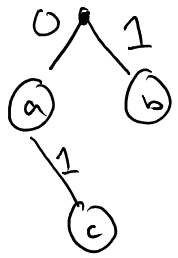
Q: Create trees for each of the following codes. What property must tree have to correspond to prefix-free code?

B) a → 0
b → 1
c → 01

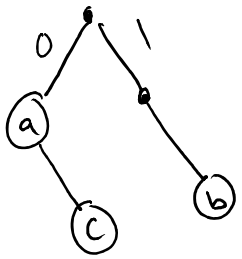
C) a = 0
b = 11
c = 01

D) a = 0
b = 10
c = 11

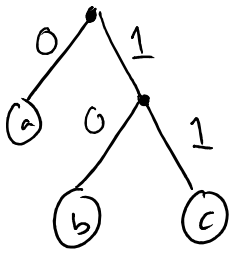
B)



C)



D)



Prefix-free codes correspond to trees where no letter node is ancestor of any other
⇒ all letters are at leaves

As desired Decoding is simple: follow path until hit a leaf

ex: 010110 → a b c a ← using this tree

S. KIMMEL

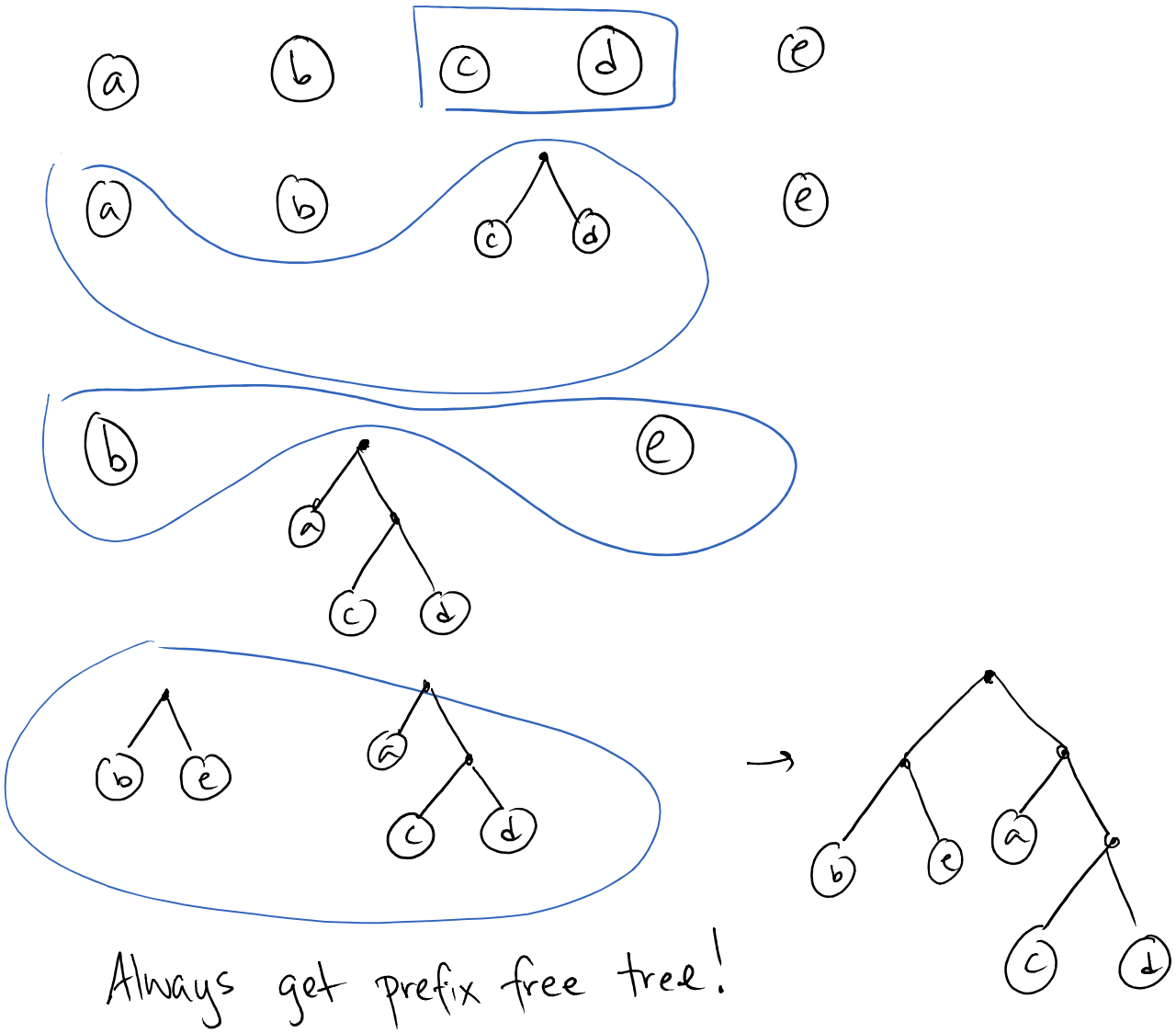
Optimal Encoding Problems

Input : p_i for each $i \in \Sigma$

Output: [Binary tree T with all letters at leaves] & Prefix free code

Smallest average length : $L(T) = \sum_{i \in \Sigma} p_i$ [depth of node i]

Bottom Up Approach: Merge trees



Always get prefix free tree!

S.KIMMEL

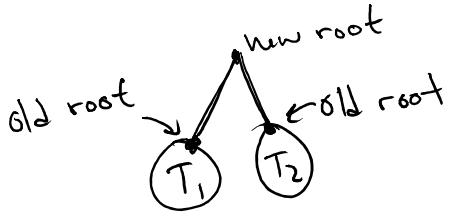
Q: What is the length of $f(i)$ for $i \in \Sigma$ if use merging strategy to create tree

A) # of mergers involving i

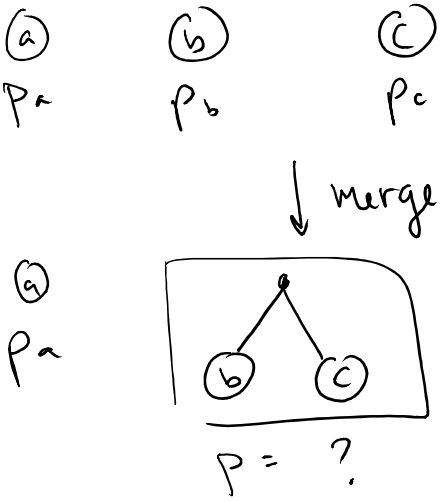
B) $\log_2 i$

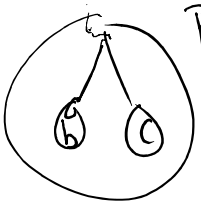
C) $2^{[\#\text{ of mergers involving } i]}$

D) # of siblings of node i



Q:





Prob of entering subtree is prob of b or c occuring

$p(b \vee c) = p(b) + p(c)$

(Prob of Union of disjoint events)

S. KIMMEL

Huffman's Strategy

While (> 1 tree to be merged)

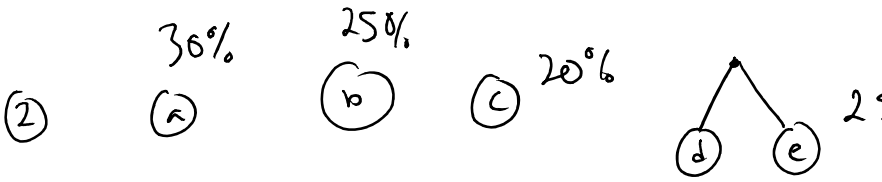
- Find 2 trees with smallest probability
- Merge into new tree with new probability = sum of old probabilities

Q: Create tree using this algorithm for

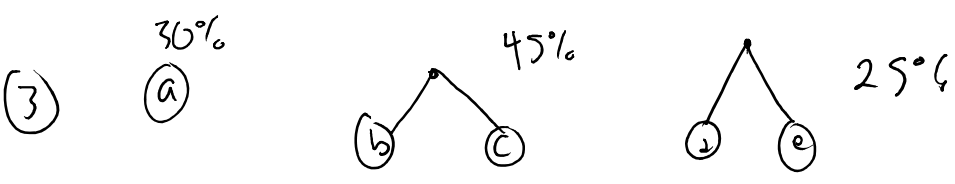
	a	b	c	d	e
	30%	25%	20%	15%	10%

- What is average length?
- Write + encode a message

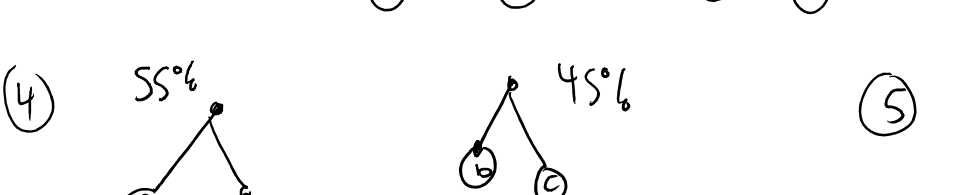
(1)



(2)

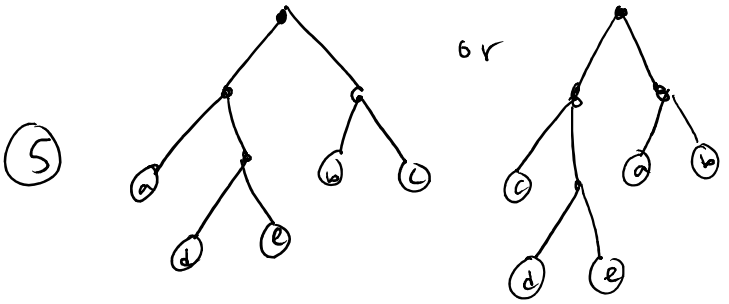


(3)



(4)

(5)



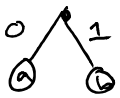
$$L = .1 \cdot 3 + .15 \cdot 3 + .75 \cdot 2 = 2.25$$

S. KIMMEL

Thm: Huffman's Algorithm Produces a Tree T with minimum average length

$$L(T) = \sum_{i \in \Sigma} p_i [\text{depth node } i]$$

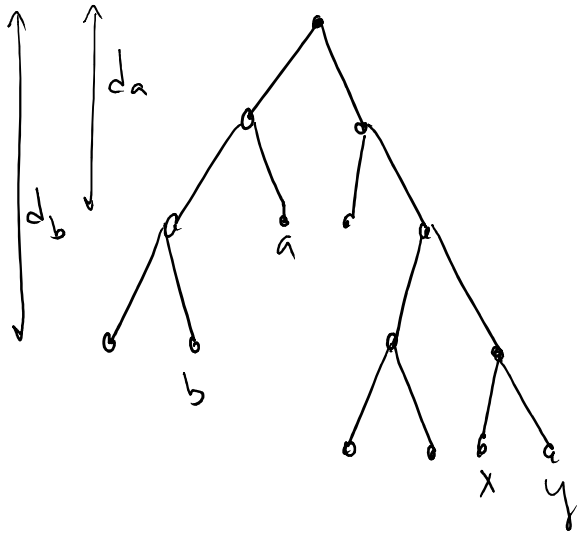
Pf: Induction on $n = |\Sigma|$. Assume $n \geq 2$

Base Case: If $n = 2$, 

is optimal

Inductive Step: Let a, b be the letters with the lowest frequency.

1. There is a tree with optimal L s.t. a, b are siblings. [Use exchange!] For contradiction, suppose T^* is optimal tree.

T^*: 

Let x, y be siblings at deepest level of tree

Let T be tree where $x \leftrightarrow a$
$y \leftrightarrow b$

S.KIMMEL

Q What is $L(T^*) - L(T)$?

$$L(T^*) = \sum_{i \in \Sigma - \{a,b,x,y\}} p_i d_i + p_a d_a + p_b d_b + (p_x + p_y) d_{x/y}$$

$$L(T) = \sum_{i \in \Sigma - \{a,b,x,y\}} p_i d_i + p_a d_{x/y} + p_b d_{x/y} + p_x d_a + p_y d_b$$

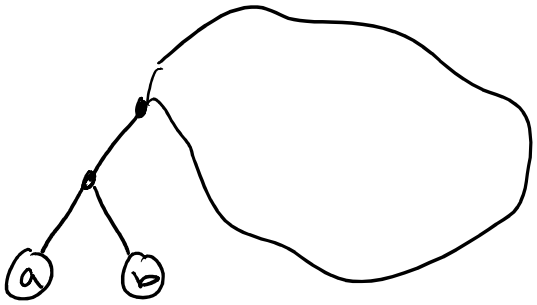
$$L(T^*) - L(T) = p_x(d_{x/y} - d_a) + p_y(d_{x/y} - d_b) + p_a(d_a - d_{x/y}) + p_b(d_b - p_{x/y})$$

$$= (p_x - p_a)(d_{x/y} - d_a) + (p_y - p_b)(d_{x,y} - d_b)$$

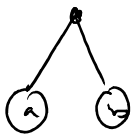
≥ 0 ≥ 0 ≥ 0 ≥ 0

So new tree T must also have optimal average length, but $T \in X_{ab}$. So there is always an optimal tree with a, b siblings.

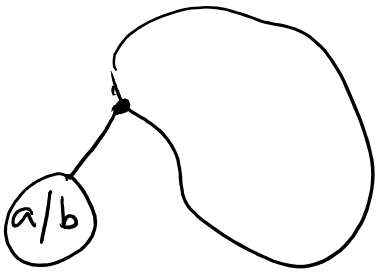
From 1, without loss of generality, there is an optimal tree T where a, b are siblings



Let T' be tree that is same as T, but with



replaced by (a/b)
one letter
(a/b) w/ probability $p_a + p_b$



Q: What is $L(T) - L(T')$

$d_{a/b}$ is depth of node (a/b)

A: $L(T') = \sum_{i \in \Sigma - \{a,b\}} p_i d_i + (p_a + p_b) d_{a/b}$

$$L(T) = \sum_{i \in \Sigma - \{a,b\}} p_i d_i + (p_a + p_b)(d_{a/b} + 1)$$

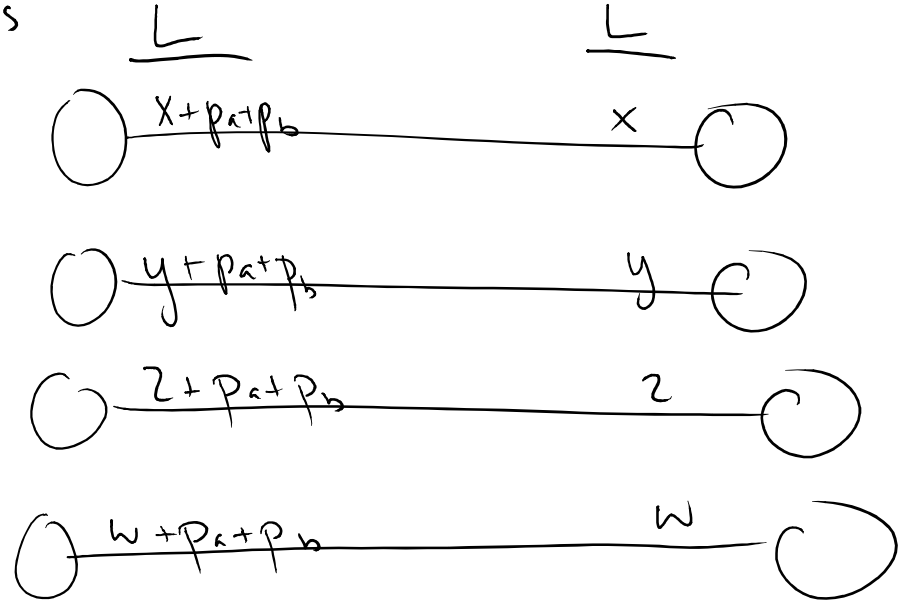
$$L(T) - L(T') = p_a + p_b$$

X_{ab}

Set of all trees where a,b are siblings

X'_{ab}

Set of all trees with symbol (a/b)



Tree with minimum L in X_{ab}, can be found by finding tree with minimum L in X'_{ab}

By inductive assumption, Huffman does this!

and replacing (a/b) with (a) (b)