7s: Huffman

Thursday, March 25, 2021
3:58 PM

Midterm Revision Guidance:

- I give hints on feedback - try to work on own first
- Come to me or tutors
- Can work with a peer on a problem **if** you are both similarly stuck
- Include mini-reflection with revision: https://www.cs.middlebury.edu/~skimmel/Courses/302S22/misc.html#QuizRev

Candidate Lecture

- Tuesday 4:30 (Intersection of Computing and the Law)

Binary Codes

Binary code: $f: \overset{\text{set of symbols}}{\Sigma} \rightarrow \overset{\text{all binary strings}}{\{0,1\}^*}$

e.g. Morse code, ASCII, Braille, $\underset{\text{unicode}}{\text{UTF-16}}$

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

ok 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) $a \rightarrow 00$
$b \rightarrow 01$
$c \rightarrow 10$

B) $a \rightarrow 0$
$b \rightarrow 1$
$c \rightarrow 01$

$$\boxed{\begin{array}{l} 011 \rightarrow cb \\ 011 \rightarrow abb \end{array}}$$

C) $a \rightarrow 0$
$b \rightarrow 10$
$c \rightarrow 11$

$\ell(f(a)) = 1$
$\ell(f(b)) = 2$
$\ell(f(c)) = 2$

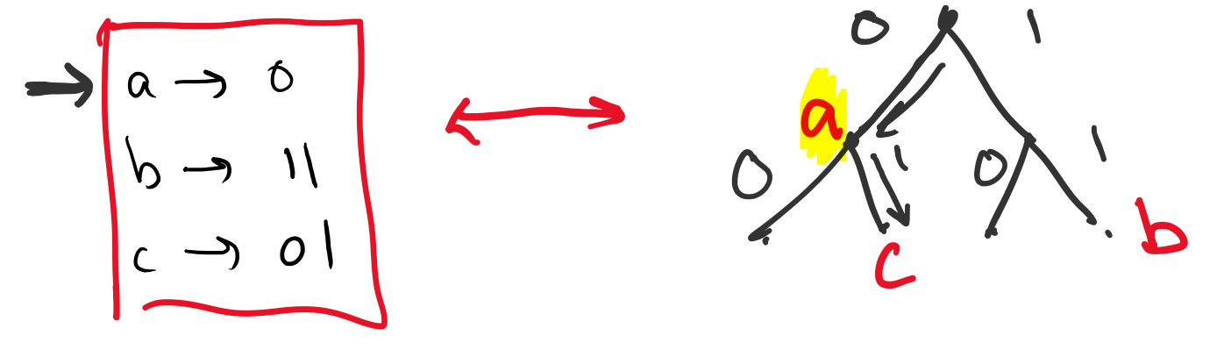
No confusion

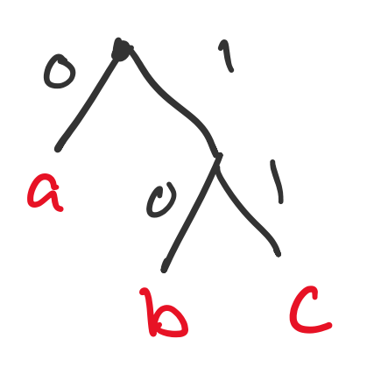
$\underbrace{10}_{b}\underbrace{11}_{c}\underbrace{0}_{a}$

$$\underset{\uparrow\, a}{.5 \cdot 1} + \underset{\uparrow\, b}{.3 \cdot 2} + \underset{\uparrow\, c}{.2 \cdot 2} = 1.5$$

Average Length: $$L(f) = \sum_{i \in \Sigma} \overset{\text{length}}{\ell}(\overset{\text{binary encoding}}{f(i)}) \cdot pr(i)$$

Binary Codes ↔ Binary Trees





def: A code is prefix free if all letters are at leaves of the tree.

Good! No ambiguity when decoding.

Optimal Binary Code Problem

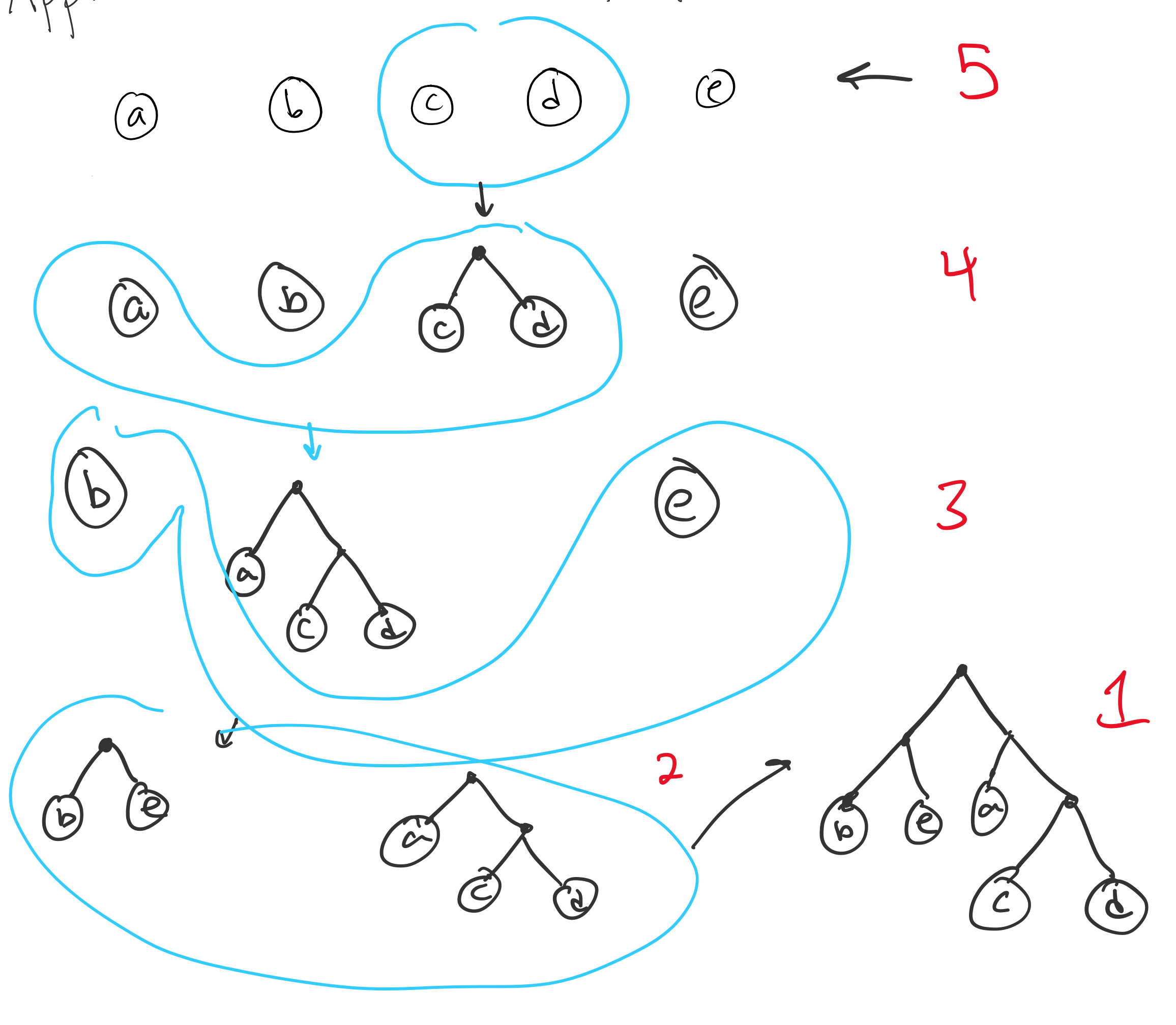
Input: Σ (alphabet of symbols)

$p: \Sigma \rightarrow \mathbb{R}^+$ (probability for each symbol)

Output: $f: \Sigma \rightarrow \{0,1\}^*$

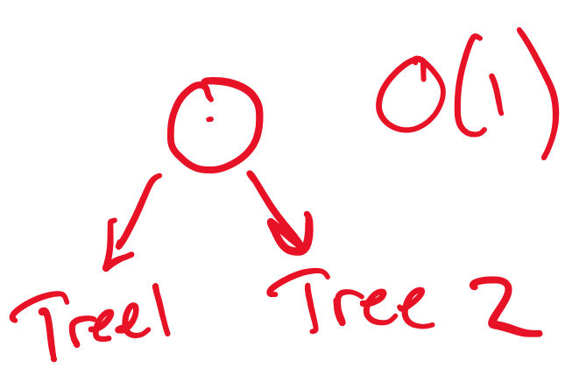
- prefix free
- minimize average length

Approach to Create Prefix-free Trees: Merge Trees



Huffman's Algorithm

```
For each i ∈ Σ                                              } O(1) } O(n)
    • Create a tree with 1 node, label "i"
    • Assign tree weight p_i  ← f-values
n-1× While (more than 1 tree)                              (n²)
    • Merge 2 trees with smallest weight ⇐ O(n)
    • Set weight of merged tree to be sum   ← f-values change
      of weights of the merged trees        ← O(1)
```



Letter	Probability
a	0.3
b	0.25
c	0.2
d	0.15
e	0.1

- Create optimal code/tree using Huffman
- What is av. length of your code? 2.25
- If $|\Sigma| = n$, what is the runtime of Huffman?

How was your break?

$$.3 \cdot 2 + .25 \cdot 2 + .2 \cdot 2 + .15 \cdot 3 + .1 \cdot 3 = 2.25$$