Lecture #10

CS 170

Spring 2021

2 More Greedy Algorithms

- Huffman Encoding
  - used to compress data
  - gzip, jpeg, mp3, ...
  - Intro to Entropy

- Horn Clauses
  - special case of satisfiability of a Boolean expression
    $(x \text{ or } \bar{y}) \text{ and } (\bar{z} \text{ or } w) \text{ and } \ldots$
  - used in logic programming (Prolog, ...)

Data Compression Problem -1

Given string of chars: ABACCDBB...
from alphabet $\{A,B,C,D\}$ how many bits do we need to encode it?

Obvious way

4 chars $\Rightarrow$ 2 bits/char $\Rightarrow$ $2n$ bits
for n chars

Can we do better?

Need more information

Data Compression Problem - 2

- Suppose we also know the frequency with which each character appears:

char:	A	B	C	D
freq:	.4	.3	.2	.1

- Can we use shorter codes (fewer bits) for more common characters?

Try $A=0$, $B=1$, $C=00$, $D=01$

what is 000? AAA? AC? CA?

Goal: avoid common "prefix", 0 for A and C

Prefix-free Codes

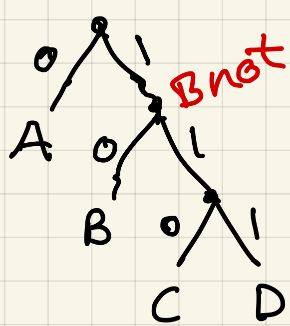
char	A	B	C	D
freq	.4	.3	.2	.1
code	0	10	110	111

total # bits for n chars

$= n\cdot.4\cdot 1 + n\cdot.3\cdot 2 + n\cdot.2\cdot 3 + n\cdot.1\cdot 3$

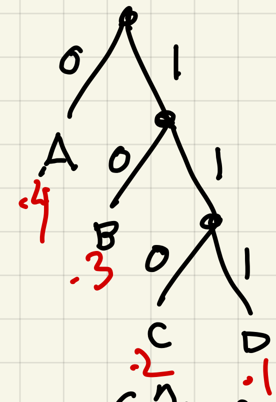
$= 1.9n$ versus $2n$ for 2bit encodings

Def: Code is prefix free if no char is prefix of another



Fact: 1-1 correspondence between prefix free codes and full binary trees (i.e. nodes have 0 or 2 children) with characters at leaves

Prefix free Codes - 2



Goal: Find full binary tree with minimal cost, given

$n = \#$chars, f_i = frequency of char i $\left(\sum_{i=1}^{n} f_i = 1\right)$

$$\text{cost} = \sum_{i=1}^{n} f_i \cdot (\text{length of encoding of char } i)$$

$$= \sum_{i=1}^{n} f_i \, (\text{depth of leaf } i \text{ in tree})$$

sending m chars requires $m \cdot \text{cost}$ bits

Greedy Intuition: Who goes at bottom of tree? i.e. longest encodings? least frequent chars.

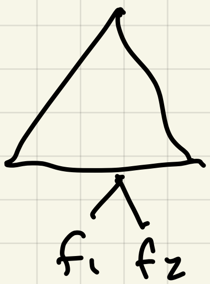
smallest f_i at bottom

Cost of an Optimal Tree

Cost $= \sum_{i=1}^{n} f_i \cdot$ (depth of leaf i in tree)

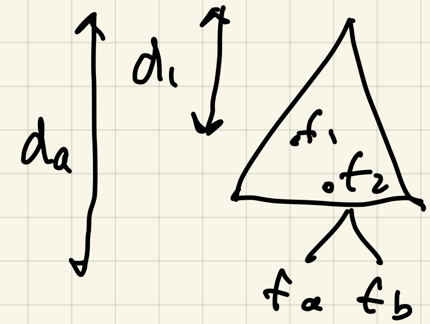
Claim: Suppose $f_1 \le f_2 \le \cdots \le f_n$

f_1 and f_2 are siblings at bottom of tree.

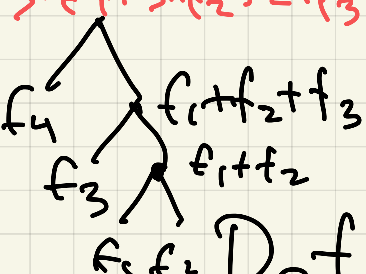


proof: suppose not

then swap f_1 & f_a
f_2 & f_b



cost $= 3\cdot f_1 + 3\cdot f_2 + 2\cdot f_3 + f_4$



Claim lowers cost: swap f_1 & f_a changes cost by

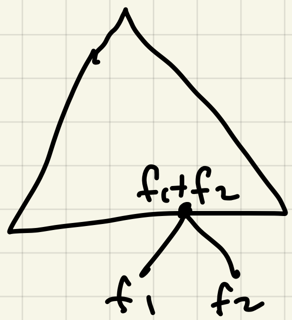
$(d_1 \cdot f_1 + d_a \cdot f_a) - (d_1 \cdot f_a + d_a f_1)$ apply repeatedly?

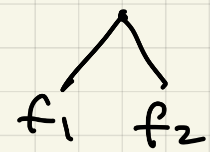
$= \underbrace{(d_a - d_1)}_{\ge 0} \cdot \underbrace{(f_a - f_1)}_{\ge 0} \ge 0$

Def for any node except root, $cost(v) = \sum f_i$ of all leaves in subtree rooted at v

Claim Cost of tree $= \sum_{v \text{ except root}} cost(v)$

Greedy Algorithm for an Optimal Tree - 1



1. pick 2 least frequent chars $f_1 \le f_2 \le \cdots$
2. replace them by 1 char with frequency f_1+f_2
3. find optimal tree with $n-1$ chars
4. replace leaf for f_1+f_2 by 

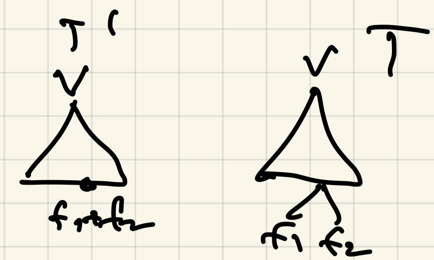
proof: Induction on $n = \#$chars

Base: $n = 1$ or 2 chars, use 1 bit

T' = tree found on $f_1+f_2, f_3, f_4 \cdots f_n$ (optimal by induction)

$T = T'$ with 

added to f_1+f_2



if $v \in T$ and T' then cost(v) same

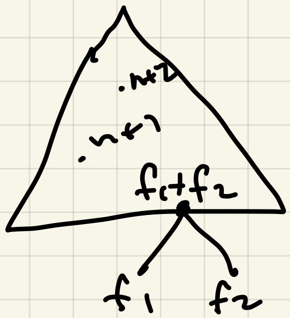
$\Rightarrow$ cost(T) = cost$(T') + f_1 + f_2$

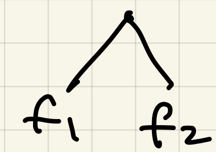
if T not optimal $\Rightarrow \exists$ better tree for $\underbrace{f_1, f_2}_{\text{siblings at bottom}}, f_3 \ldots$

$\Rightarrow T'$ not optimal (replace f_1, f_2 by f_1+f_2

contradiction

Greedy Algorithm for an Optimal Tree -2



1. pick 2 least frequent chars $f_1 \le f_2 \le \cdots$
2. replace them by 1 char with frequency f_1+f_2
3. find optimal tree with n-1 chars
4. replace leaf for f_1+f_2 by 

Create Priority Queue H of $\{1, 2, \ldots, n\}$

ordered by $f_1 \le f_2 \le \cdots \le f_n$

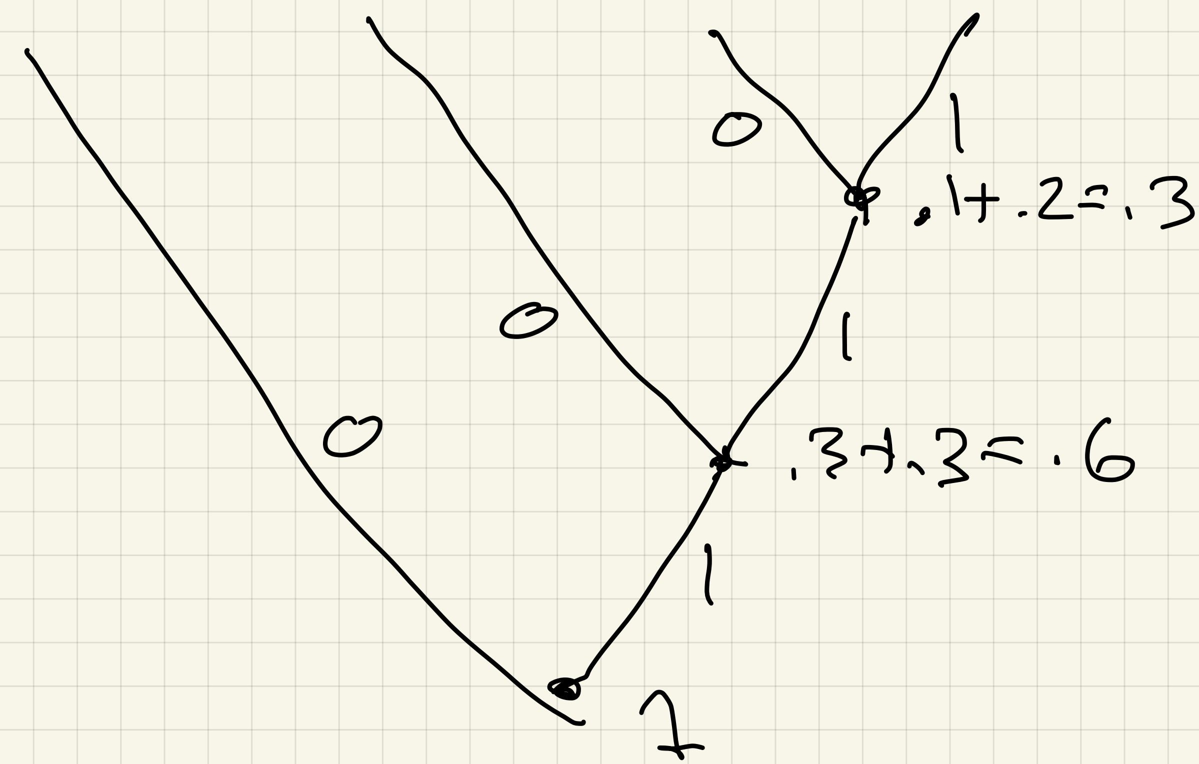
for $k = n+1$ to $2n-1$

i = deletemin(H), j = deletemin(H)
create node k with children i and j
$f_k = f_i + f_j$; insert(k)

cost of Alg $= O(n \cdot \text{cost(deletemin)} + n \cdot \text{cost(insert)}) = O(n \log n)$ binary heap

Huffman Encoding Example

	0	10	110	111
char	A	B	C	D
freq	.4	.3	.2	.1



Horn Formulas

- Can we satisfy a Boolean expression?
  - Notation w = Boolean variable (T or F)
    $\wedge$ = and, $\vee$ = or, $\bar{w}$ = not w
  - Are there values (T or F) of Bool. vars. that make this True?
    $$(w \wedge y \wedge z) \wedge (x \wedge y \Rightarrow w) \wedge (\underbrace{\bar{u} \vee \bar{v} \vee \bar{z}}_{\text{clause}}) \cdots$$
- General case: NP Hard (chap 8)
- Horn Formulas – special case – clauses must b
  1) implication $(u \wedge v \wedge w) \Rightarrow x$
     with all positive vars (no $\bar{u}, \bar{v}$ etc)
     includes $\Rightarrow x$ (x must be T)
     recall $a \Rightarrow b \equiv \bar{a} \vee b$
  2) negative clauses $(\bar{x} \vee \bar{y} \vee \bar{z})$

Horn Formula: Example

$x \equiv$ murder took place in kitchen

$y \equiv$ butler innocent

$z \equiv$ colonel asleep at 8pm

$w \equiv$ murder took place at 8pm

$u \equiv$ colonel innocent

$v \equiv$ professor innocent

implication $(z \wedge w) \Rightarrow u$

negative clause $(\bar{u} \vee \bar{y} \vee \bar{v})$

See Prolog

Greedy Algorithm for Horn Formulas

3 kinds of formulas:

1) $(z \wedge w) \Rightarrow u$

2) $\Rightarrow x \quad (x = T)$

3) $(\bar{u} \vee \bar{v} \vee \bar{y})$

Set all variables to false (3) ok, 1) ok, 2) not ok)

While an implication not satisfied, set rhs = T

If all negative clauses satisfied

Satisfiable: return assignment

else

Not satisfiable

Example:

$\underset{\text{OK}}{(w \wedge y \wedge z) \Rightarrow x}, \underset{\text{OK}}{(x \wedge z) \Rightarrow w}, \underset{T}{x \Rightarrow y}, \underset{T}{\Rightarrow x}, \underset{T}{(x \wedge y) \Rightarrow w}, \underset{\text{oops}}{(\bar{w} \vee \bar{x} \vee \bar{y})}$

Back to Huffman: Intro to Entropy

- What is Entropy?
- What is "Information"?
  - If a random event occurs, and I tell you the outcome, how much Information is that?
  - $I(p)$ = information from being told a random event with probobility p occurred
  - Basecase: flip fair coin, be told H or T : $I(\frac{1}{2}) = 1$ bit
  - If $p > \frac{1}{2}$, $I(p) < 1$, if $p < \frac{1}{2}$, $I(p) > 1$
  - Two indep. events, probs p_1 and $p_2 \Rightarrow I(p_1 \cdot p_2) = I(p_1) + I(p_2)$

  $\Rightarrow I(p) = \log_2\left(\frac{1}{p}\right)$

Intro to Entropy - 2

EE 126, 229A
Stat 212A

- Suppose n possible outcomes, probabilites $p_1, p_2, \dots, p_n$
  Expected information $= E(I) = \sum_{i=1}^{n} p_i I(p_i) = \sum_{i=1}^{n} p_i \log_2\left(\frac{1}{p_i}\right)$
  = Entropy, measures how "random" distribution is
  Ranges from 0 (one $p_i = 1$, rest 0) to $\log_2 n$ (all $p_i = \frac{1}{n}$)
- In Huffman, suppose all $f_i = p_i = \frac{1}{2^{k_i}}$ for some k_i
  - can show depth of f_i in Huffman tree $= k_i$
  - To encode m chars with frequencies p_i, takes
    $m \cdot \sum_{i=1}^{n} p_i \cdot$ depth of f_i in tree $= m \cdot E(I) = m \cdot$ Entropy
- Thm (Shannon) $m \cdot$ Entropy is a lower bound on any encoding scheme