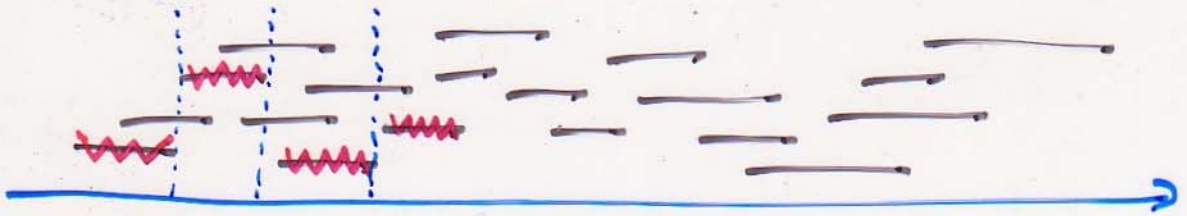
CS 223 Lec 15

- greedy algorithm for activity selection:

1. Let $S = \{$ all activities $\}$
2. while $S \neq \emptyset$ {
   - find $a_m \in S$ with earliest finishing time
   - select a_m
   - $S = S_{m,n+1}$

}



Knapsack Problems

- knapsack holds $\overset{\max}{\wedge}$ weight W

- various objects: $\{o_1, \ldots, o_n\}$

$w(o_i) = w_i$ weight

$v(o_i) = v_i$ value.

0-1 Knapsack Problem

Find a subset $S \subseteq \{o_1, \ldots o_n\}$ s.t.

$$\sum_{i \in S} w_i \leq W$$

and $\sum_{i \in S} v_i$ = value of S

is maximized.

Fractional Knapsack

- not constrained to take unit quantities
- can take fractional amounts of objects.
- greedy strategy does work:

1. Calculate $r_i = \frac{v_i}{w_i}$
2. Sort objects so that $r_1 \geq r_2 \geq r_3 \ldots \geq r_n$
3. $i = 1$
   while the total weight $\leq W$
   {
   - select as much of object i as possible
   - i++
   }

Dynamic Programming Soln for 0-1 Knapsack

$O = \{objects\}$

w_i = weight of o_i

V_i = value of o_i

$k[n]$ = total value of the optimal choice of objects whose total weight is n.

$$k[n] = \max_i \left[\left(k[n - w_i] \right) + V_i \right]$$

Initial Conditions

$k[0] = 0$

$k[\text{lightest weight} - 1] = 0$

$k[\text{lightest weight}] = V_{\text{lightest weight}}$

$\vdots$

- to find a solu for knapsack with capacity W:

  scan $k[0, \ldots W]$

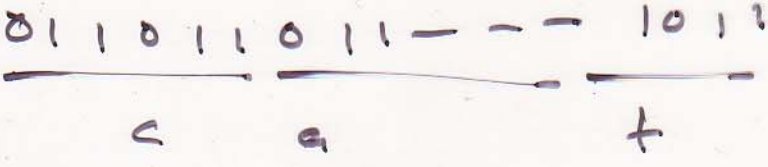
  → find greatest value.

Huffman Codes

- compress text files

	8-bit ASCII code
a	01101111
b	01110000
c	⋮
d	
⋮	

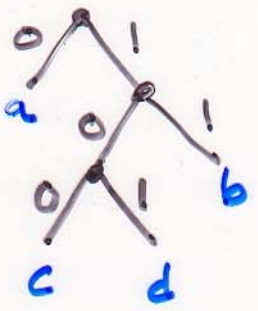
file : "cat"



- setup: each character has a native frequency
- idea: use a variable # of bits to encode a character.

Prefix Codes

{a, b, c, d}



bit stream:

0 1 1 0 1 1 1 0 1 1 1 1 0 1 1 0 0 . . .

a b a b d b d c

a	0
b	1 1
c	1 0 0
d	1 0 1