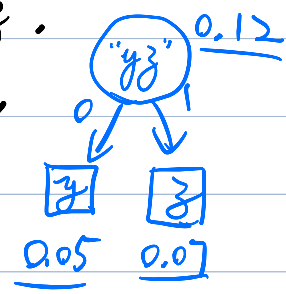
Recap:

Huffman's Greedy Approach [1952] (build the tree bottom up)

1. Make a node with 2 children leaf nodes for the two lowest freq. symbols y & z.
2. Replace y & z with the "metasymbol" "yz". its freq. is the sum of freq. of y and z.
3. Rinse & Repeat until we have a single metasymbol, it'll be the root of the huffman tree.



Pseudocode:

```
    Huffman(S):
1.      If |S|=2:
2.          Return tree with root and two leaves
3.      Let y and z be lowest freq. symbols in S
4.      S'=S
5.      Remove y and z from S'
6.      Insert new symbol w in S' w/ f_w = f_y + f_z
7.      T' = Huffman(S')
8.      T = add children y and z to leaf w in T'
9.      Return T
```

Runtime: $T(n) = T(n-1) + \underbrace{O(n)}_{\text{Step 3}} \Rightarrow T(n) = O(n^2)$

Optimization: use a Priority Queue with

Insert / ExtractMin $O(\log n)$ time

*Covered in CSCI-UA, 102 : Data Structures

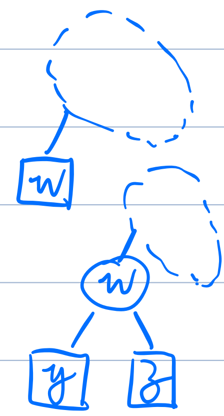
Implemented w/ Heaps (Binary, Binomial, Fibonacci....)

$\Rightarrow$ Step 3, 5, 6 takes $O(\log n)$ time

$\Rightarrow T(n) = T(n-1) + O(\log n) \Rightarrow T(n) = O(n \log n)$

1. Correctness (Optimality of Huffman)

Claim 1: $ABL(T') = ABL(T) - f_w$



Proof: T' vs T: in T', w is leaf
in T, w has two children, y & z.

$f_w = f_y + f_z$

$depth_T(w) = depth_T(y) - 1 = depth_T(z) - 1$

$$ABL(T) = \sum_{x \in S} f_x \, depth_T(x)$$

$$= f_y \, depth_T(y) + f_z \, depth_T(z) + \sum_{x \neq y,z} f_x \, depth_T(x)$$

$$= f_y(depth_T(w)+1) + f_z(depth_T(w)+1) +$$

$$= (f_y + f_z)(depth_T(w)+1) +$$

$$= f_w \, depth_T(w) + f_w +$$

$$= ABL(T') + f_w$$

\#

Claim 2: There is always an optimal prefix code that have the two lowest frequency symbols as siblings.

Proof Sketch: Must both be on the lowest level
$\Rightarrow$ reordering in the same level does not affect optimality

Thm: Huffman gives a prefix code with optimal ABL.

Proof: Induction on $n = |S|$

- Base Case: $n=2$, ABL is 1, trivially optimal.
- Ind. Hyp.: True for $n-1$ Symbols
- Inductive step:

By Ind. Hyp., T' for $S' = (S \setminus \{y,z\}) \cup \{w\}$ has optimal ABL since $|S'| = n-1$.

Assume towards contradiction T is <u>not</u> optimal.

$\Rightarrow \exists T^*$ s.t. $ABL(T^*) < ABL(T)$

By claim 2, $\exists T^*$ where y and z are siblings.

Let $T^{\perp}$ be T^* with y, z removed and parent labeled w.

By Claim 1: $ABL(T^{\perp}) = ABL(T^*) - f_w$

$ABL(T') = ABL(T) - f_w$

$\Rightarrow ABL(T^{\perp}) < ABL(T')$

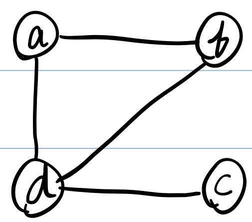
Contradicts with T' optimal.

$\#$

2. Graphs

$$G = (V, E)$$

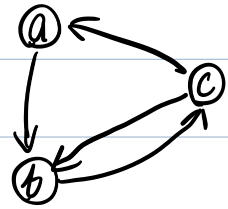
V: vertices, E: edges



undirected

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

$E = \{\underbrace{\{a, b\}}_{\text{unordered pairs}}, \{a, d\}, \{b, d\}, \{c, d\}\}$



directed

$V = \{a, b, c\}$

$E = \{\underbrace{(a, b)}_{\text{ordered pairs}}, (b, c), (c, b), (c, a)\}$

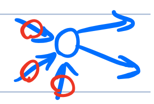
Terminology/Notations:

$n = |V|$, $m = |E|$

Degree of a vertex u: # of edges connected to u $(\deg(u))$

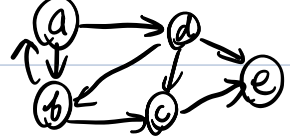
In-degree of a vertex u: # edges into u

$|\{(v, u) : (v, u) \in E\}|$



Out-degree of a vertex u: # edges out of u

Path: sequence of nodes/edges from one node to another

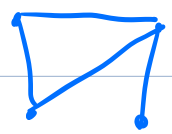


e.g. (a, d, b, c, e) $((a,d), (d,b), (b,c), (c,e))$

(Simple path: a path w/o repeating vertices
e.g. (a,d,b,c,e) is simple path, (a,b,a,d) is not)

Cycle: A simple path that starts and ends at same vertex
e.g. (a,d,b,a)

G is connected if there is a path b/w every pair of vertices



Connected

Connected components

not connected

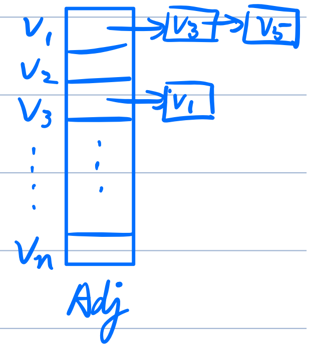
Applications of Graphs
- Google Page Rank
- Facebook social graph
- Maps
- Internet / traffic routing
- Many More ...

Graph Representations:

① Adjacency List

Array Adj of $|V|$ linked lists,
one linked list for each vertex
In each linked list, Adj [u] stores
u's neighbors.

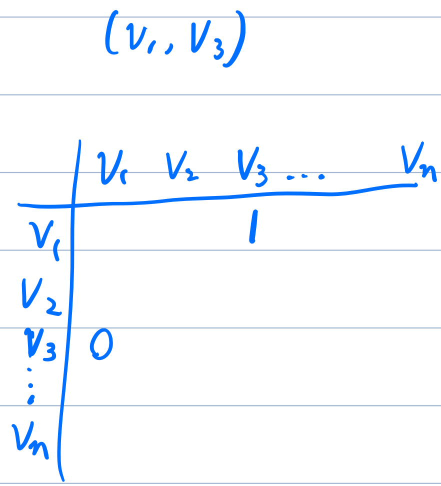
$$Adj[u] = \{v \in V \mid (u,v) \in E\}$$



Default Representation

Space : $\Theta(|V|+|E|)$

② Adjacency Matrix



$|V| \times |V|$ bit-matrix

$A[u,v]=1 \Leftrightarrow (u,v) \in E$

Space: $\Theta(|V|^2)$

Good if $|E| \approx |V|^2$, very fast for look up
if (u,v) exists!