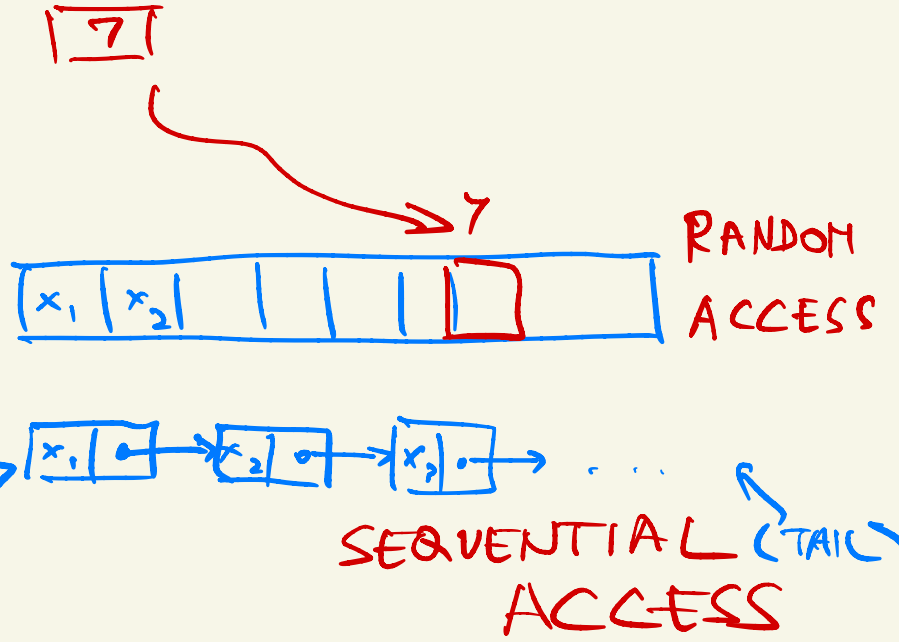
HONORS ALGORITHMS | 2024-01-19 | 1

Data structures

LIST

basic types: ARRAY
LINKED LIST



If SORTED:

	ARRAY	LINKED LIST
FIND	binary search efficient: $O(\log n)$	sequential inefficient: $O(n)$
INSERT/DELETE	requires shifting inefficient: $O(n)$	update links efficient: $O(1)$ } assuming address found
INCREASE/DECREASE KEY	requires re-sorting	

COUNTING SORT

Suppose all keys $key(i) \in [m]$

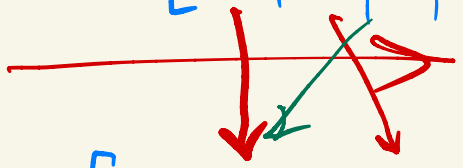
n items

SORT in $O(n)$ operations

$A = [key(1), \ldots, key(n)]$

$B = [1 \ldots m]$

$A = [3, 5, 3, 2]$



$B = [1\ 2\ 3\ 4\ 5\ 6]$

DO: pseudocode

single pass through A n

" " B m

total $O(n+m)$

B	1	2	3	4	5	6
B*	0	1	2	0	1	0

$C = [2, 3, 3, 5]$

linear time if $m = O(n)$

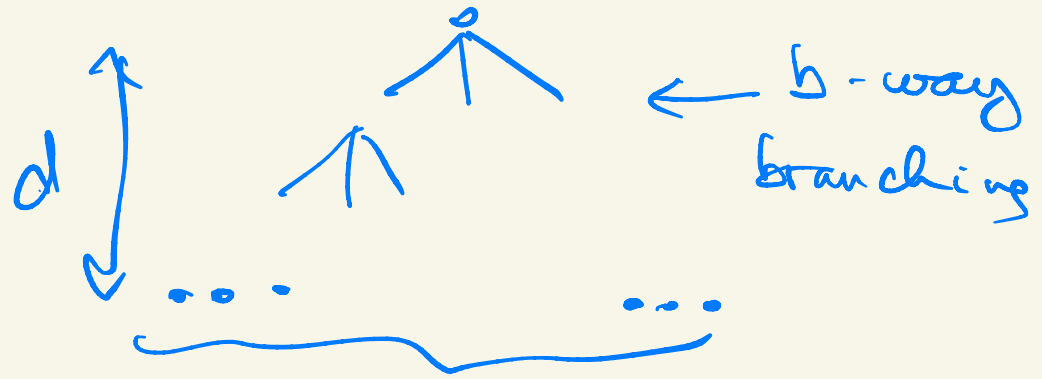
m-way branching

$$d \geq \log_m (n!) = \frac{\log (n!)}{\log m}$$

Case $m = n$

$$d \geq \log_n (n!) = \frac{\log (n!)}{\log n} \sim \frac{n \ln n}{\ln n} = n$$

info thy lower bound

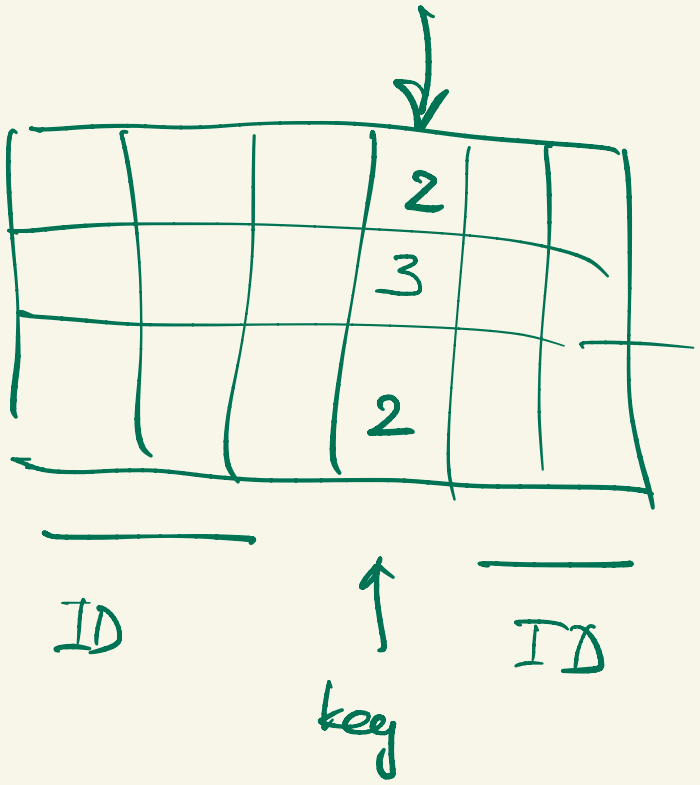


leaves b^d

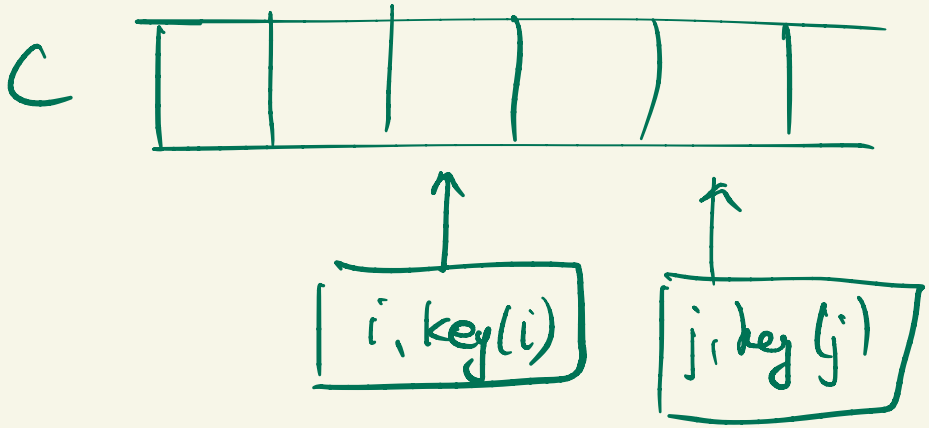
n outcomes all distinguished

$$\Rightarrow \quad n \leq b^d$$

$$\boxed{d \geq \log_b n}$$



Sort array by column k

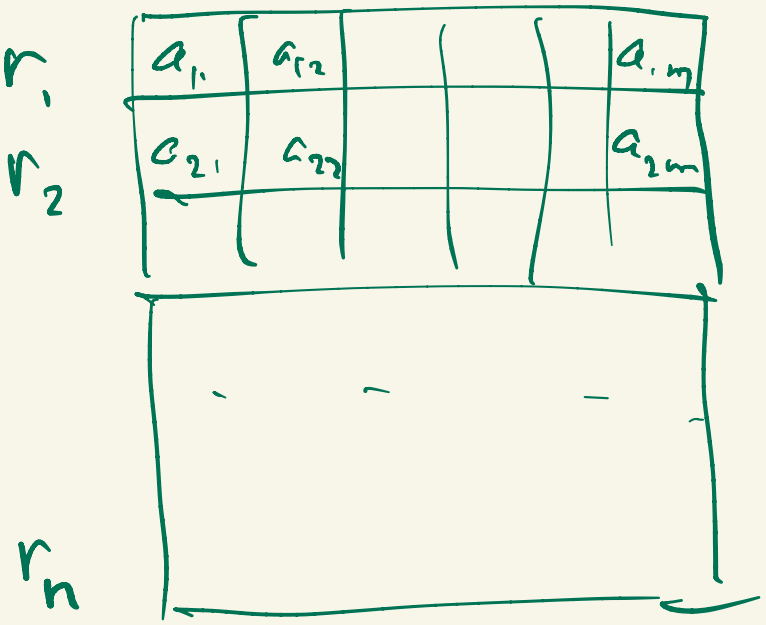


then $key(i) \leq key(j)$

and if $key(i) = key(j)$

then $i \leq j$

(a_{ij})

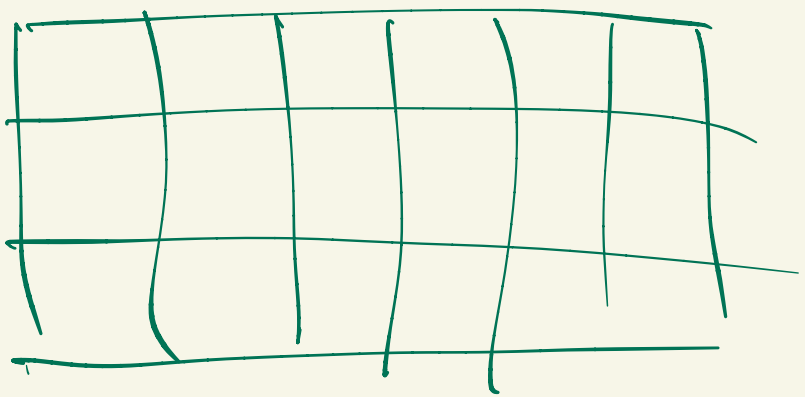


Sort rows lexicographically (dictionary order)

$r_i \prec r_j$ if $(\exists k)(a_{i,1} = a_{j,1} \cdots a_{i,k-1} = a_{j,k-1}$ and $a_{ik} < a_{jk})$

RADIX SORT

(a_{ij}) $i = 1 \dots n$, $j = 1 \dots q$



$a_{ij} \in [m]$

$n \times q$ array

for $j = q$ downto 1

counting sort by column j

EX verify correctness

COST $O((n+m)q)$

for $m = O(n)$ we get $O(nq)$ ← linear

output: permutation $f : [n] \to [n]$ i.e. bijection of $[n]$ to itself

over $[m]$

Ex. Sort strings of variable lengths in linear time, assuming $m = const.$

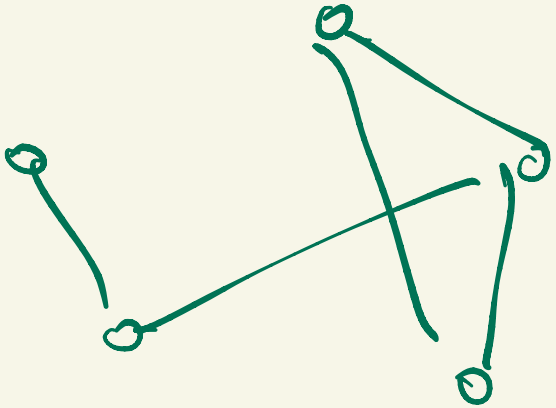
GRAPHS

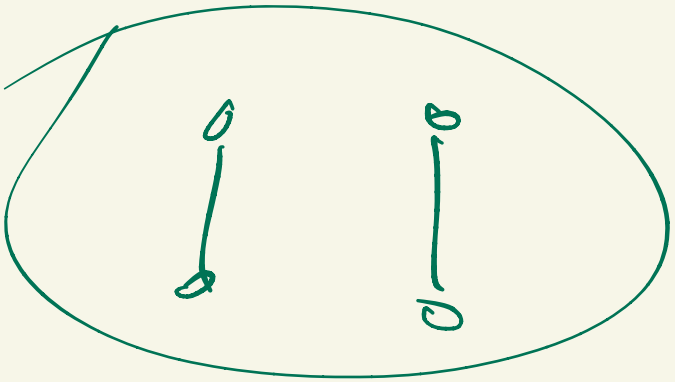
(nodes, links)

(V, E)

vertices / vertex

edges





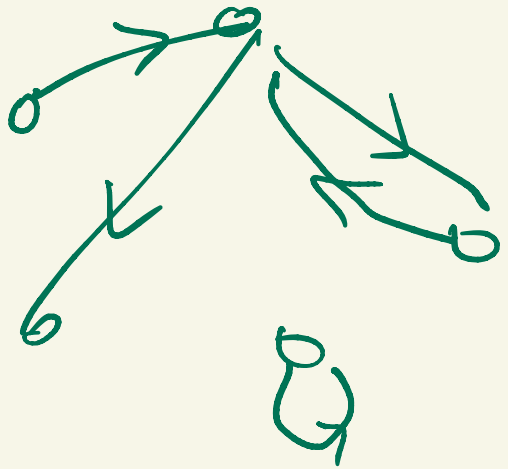
Given n vertices, # graphs on this set of vertices?

$2^{\binom{n}{2}}$

edge $\{i,j\}$ $= \{j,i\}$

Digraphs (directed graphs)

edge (i,j) $i \to j$

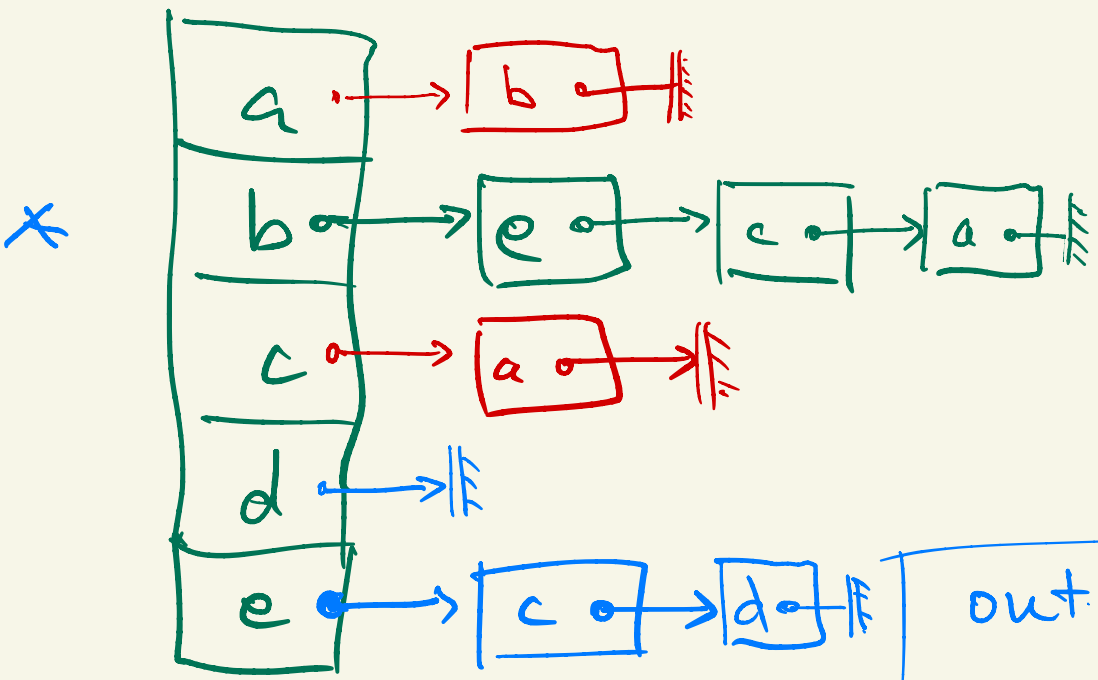


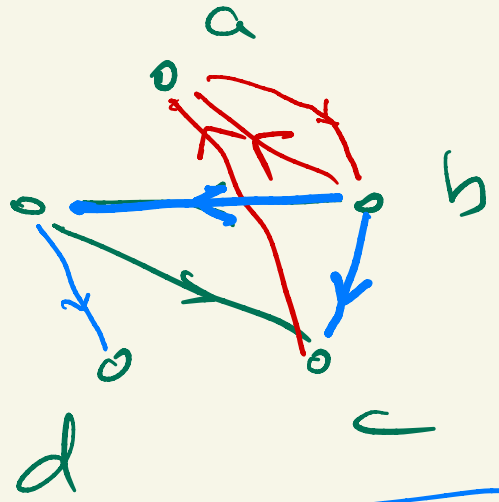
n^2 possible edges

$\# \; 2^{n^2}$

Standard representation:

array of linked lists

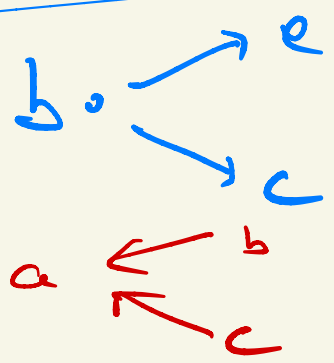




array of vertices

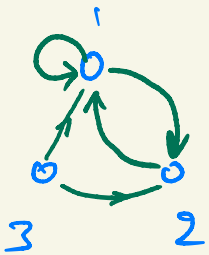
out-neighbors

in-neighbors



$i \to j$ (i,j) are <u>adjacent</u>

this is the <u>adjacency relation</u>



adjacency <u>matrix</u> $a_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{o/w} \end{cases}$

$n \times n$ $(0,1)$-matrix

$$\begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix}$$

<u>Size: n^2</u>

<u>linked list rep</u>: size $O(n+m)$ $m = \#$ <u>edges</u>

<u>EX</u>. in linear time
- sort all adj. lists
- remove repetitions
- reverse every edge