Algorithm Analysis & Design

Big 'O' notation & Worst case - Diff.

→ Worst case behavior of an algo in all possible instances.

	I_1	. -	-	$I_{m'}$
A_1	t_{11}	t_{12}	. - -	$t_{1m'}$
⋮				
A_i	. - -	- -	t_{ij}	
$A_{i'}$				
A_m				

Max of these would be the worst case.

$\max\{t_{i'j}\} \leq c' \cdot n \log n$; $\max\{t_{ij}\} = c \cdot n^2$

Graphs on n vertices $= 2^{\binom{n}{2}}$

Eg: If A_i : Bubble sort, $A_{i'}$: Merge sort.

→ To choose best algorithm, we pick the minimum among the max. of each row.

Book: Tim Ruff Gardn

Beyond worst case analysis

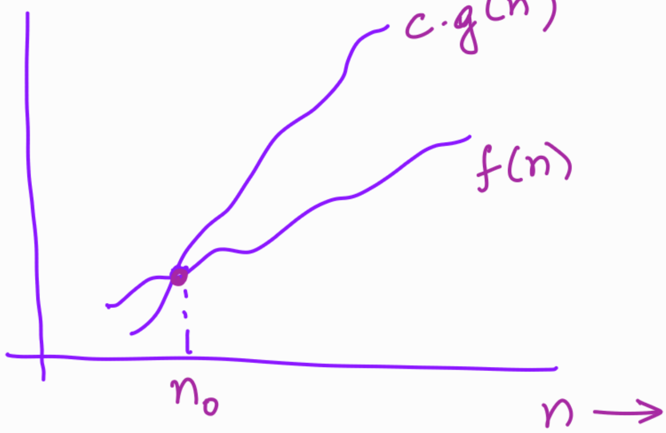
ASYMPTOTICS

Big 'O' notation → Analytical notation

→ Let f and g be func from $\mathbb{R} \to \mathbb{R}$. We say that
$(>0)\ (>0)$

$f(x) = O(g(x))$ if $\exists$ a const: c & n_0 large enough s.t $\forall$

$n \geq n_0, \quad f(n) \leq c \cdot g(n)$



Book: Art of comp. program

→ We can also say that $5n^2$ is $O(n^3)$ but that wouldn't make much sense because the projection is way lot worse than it actually is.

Open Q: Is worst case analysis actually enough to analyse algo?

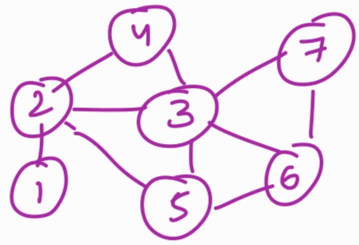
4-5 problem sets.
(Not expected any code)
(Short & concise answers)
→ Weightage will be decided halfway
→ Anonymous portal.

Chap exercises.

Book: 1) Klienberg & Tardos
2) Erikson (Web version) (UIUC) ↳ Has some typos
3) DasGupta, Papadimitrou Vazirani
4) Goodrich Tammassie.

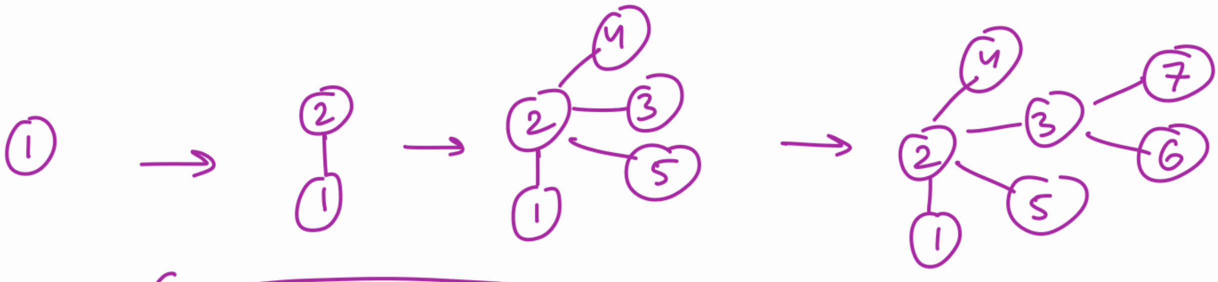
Basic graph algo.

1) BFS



{ Visited
Neighbours

BFS(1) :



Discovery tree

↳ Root of discovering a vertexe.

Layered tree.

→ Layer 0 : Start node

→ Layer 1 : Neighbours of start node.

→ $\forall_j \geq 2$, layer L_j contains all vertices that are not already in $L_1, \dots, L_{j-1}$ & those that have an edge to a vertexe in L_{j-1}

→ There is some sort of ordering that kicks in. If there is no order, then each time diff... search tree.

Book keeping involved : 1) Explored / Visited

- For every vertexe visited, check its neighbo

2) Layer address → Depth / Shortest dist. from root.

- Can combine with visited array.

- If null ⇒ not visited,
  else if some no. ⇒ Visited.

n: no. of vertices
m: no. of edges.

No. of queries $= \sum_{v} d_v$ → Degree of vertices.

$= 2|E| = 2m$

Running time : $\underline{\underline{O(m+n)}}$.

→ Layer no. implicitly holds the info abt. "shortest dist" of a vertex from root..

(or)

All ele in layer L_j have a shortest dist. of j from root

Proof: (Induction)

Base case : Layer $j = 1$
(Contents of layer 1 are neighbours of root node)

Inductive hypo. : For L_{j-1}, holds true

Inductive step. : $v \in L_j$, u is a predecessor of v s.t
$u \in L_{j-1}$ & $(u, v) \in E$ and $v \notin L_1 \cup \dots \cup L_{j-1}$