Dynamic Graph Algorithms

CSL-531

2026

Introduction

Dr. Shahbaz Khan
Department of Computer Science and Engineering,
Indian Institute of Technology Roorkee
shahbaz.khan@cs.iitr.ac.in

Limitations and Bounds

For a given problem P having best static algo time Ts(n) $\Rightarrow O(n^3)$
Dynamic Algorithm Td(n):

$O(mn)$

Upper bound:

max(update,query) = Td(n) $= O(T_s(n))$

$O(m^2)$ X

Lower bound:

max(update,query) = Td(n) $\sqrt{n}$

Incremental, Decremental, Fully Dynamic?

$\Omega(T_s/m)$ $\frac{mn}{m} = \Omega(n)$

WC ✓

Amortized ✓

Dynamic Algorithm for *NP Complete Problem* in **P**?

Travelling Salesman

Complete Weighted graph

Find a cycle of minimum weight that covers all vertices

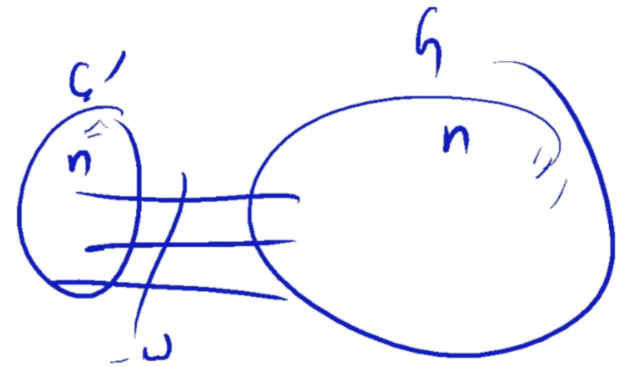
⇒ Simple (No vertex is repeated)

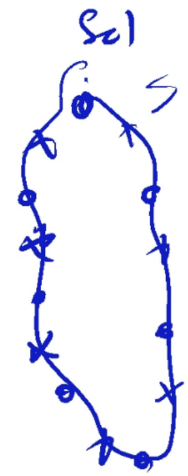
(w/o Simple constraint | Similar to MST)

$w = \frac{MinWT}{10}$

Preprocess $= o(T_s)$

With simple cycle constraint





Vertex deletions

Preprocessing $O(n^2)$ $+ \; n \times T_d \leq T_s$

$T_d = \Omega(T_s/n)$

Edge deletions

(Relax Complete Graph Constraint)

$O(n^2) + n^2 T_d \leq T_s$

$T_d = \Omega(T_s/n^2)$

Preprocessing

Store solution after each update for All possible update sequences

$$\underbrace{2^m}_{\#\text{update sequence}} \times \underbrace{n}_{\#\text{size of soln}} \times \underbrace{m}_{\#\text{updates}}$$

Based on Prob (MST, TSP)

Update in $O(1)$

Limite the preprocessing time = $o(T_s)$

NP hard → Poly

Hardness of Problems

FD vs Inc/Dec

Directed vs Undirected

Algo → Algo

Dir UnDir

Lower Bound ← Lower Bound

Amortized vs WC

$$T_{am} = \frac{Total}{\#upd} \leq \frac{T_{wc} \times \#update}{\#updates} \leq T_{wc}$$

Algo ← Algo

LowerBound → LowerBound

Relation ship btw Reducibility of Algo and Lower Bound

Algo $O(n^2)$ time → $O(n^2)$ time

Is Lower Bound $\Omega(n\sqrt{n})$

Ind

Algo $O(n)$ time

Lower Bound $\Omega(n)$ ← Lower Bound $\Omega(n)$ time

$\nexists$ Algo $o(n)$ time

Assume A_1 takes $O(\sqrt{n})$ time → A_i $O(\sqrt{n})$ time

Classical Problem

Incremental Reachability

Given a graph under edge insertions, maintain all vertices reachable from s

What is optimal update and query time?

Static

$O(m+n)$

Dynamic

$O(1)$ time

$\hookrightarrow \Omega$

Amortized?

WC. time?

$s \rightsquigarrow t$

Reachability (Directed)

$\exists$ path from s to t

Connectivity (Undirected)

Single Source Reachability (S)

↳ Directed Graph $G(V,E)$ having n vertices and m edges with source $s \in V$

→ **Static**: DFS / BFS (Reachability tree)

↳ $O(m+n)$

Incremental:

$$\begin{array}{c} G \\ \downarrow \\ DS \end{array} \xrightarrow{+e_1(u_1,v_1)} DS \xrightarrow{e_2(u_2,v_2)} DS$$

$s \rightsquigarrow v$? $\quad$ $s \rightsquigarrow v$?

Data structure

$$R[v] = \begin{cases} 1 & \text{if } s \rightsquigarrow v \\ 0 & \text{if } s \not\rightsquigarrow v \end{cases}$$

Preprocessing

Initialize $R[v] = 0$ for all

Start DFS/BFS from s $\Rightarrow$ Tree T

$\forall v \in T \Rightarrow R[v] = 1$

Update:

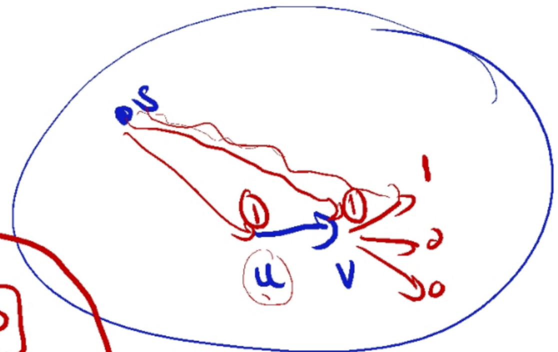
(u, v)

If $(R[u] = 1$ && $R[v] = 0)$

$\rightarrow R[v] = 1;$

$\rightarrow$ for all $(v, w) \in E$

Update(v, w);



R[u]	R[v]	
0	0	X
0	1	X
1	0	
1	1	X

Space $G + R \rightarrow O(n)$

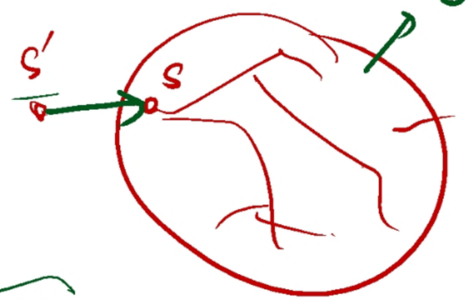
Time: $W.C \Rightarrow O(m+n)$

$Am = O(1)$

Query(v)

return $R[v]$ $\leftarrow O(1)$

WC example



If a vertex u

$R[u]=0 \rightarrow R[u]=1$

$O(\deg(u))$

Total $\sum_{\substack{u \in R[\cdot]=0 \\ \downarrow \\ R[\cdot]=1}} 1+\deg(u)$

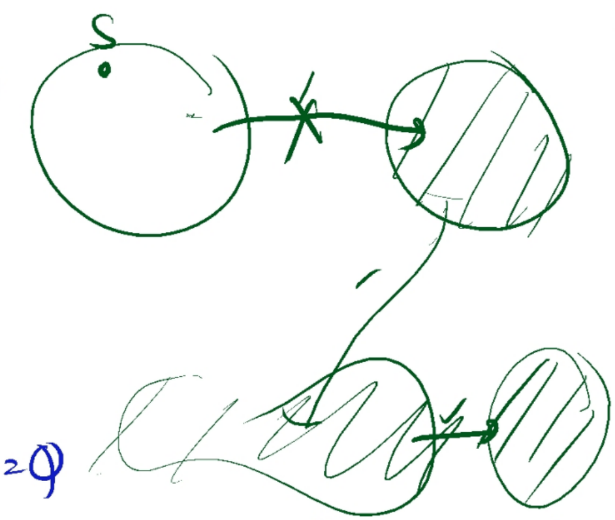
$WC = \sum_{u \in V} 1+\deg(u) = O(m+n)$

Total time

e_1 (u,v) — V_1 — $\sum_{v \in V_1} \deg()$

e_2 — V_2 — " V_2

e_3 — V_3 — " V_3



$V_1 \cap V_2 = \phi$, $V_2 \cap V_3 = \phi$

Total $\sum_{\substack{v \in V \\ \text{if } R[v]=1}} 1+\deg(v) = O(m+n)$

Amortized time $= \frac{O(m+n)}{m} = O(1)$

Amortized Analysis

Methods:

1- **Aggregate**

Upper bounds total cost of n operations T(n),

so average cost <= T(n)/n

Coremen

Textbook Reference

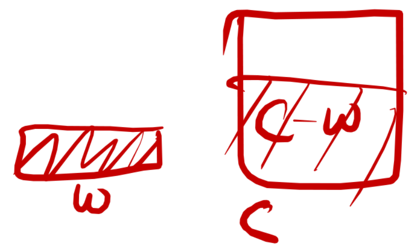
CLRS Chapter 17

2- **Accounting**

Each operation is assigned an amortized cost to store extra credit to pay for the expensive operation later on.

Amortized Cost = Actual Cost + Extra Credit

Unit Operation Perspective



3- **Potential**

Credit is computed in form of a potential which is used in heavy operations

Amortized Cost = Actual Cost + Change in Potential

Overall Perspective

$W + \Delta\Phi \leq 2$

Aggregate Method

BIT COUNTER
Consider the binary representation of the numbers. Incrementing by one unit, several bits may change in the binary representation.

Compute the
AVERAGE NUMBER OF BITS FLIPPED per increment.

Upper bounds total cost of n operations T(n), so

Am. cost <= T(n)/n

T(n) = Number of bit flips after n increments

we # flip = $O(\log n)$ Start from 0 to n

$1 \leq$ Avg. flips $\leq O(\log n)$

≤ 2

log n

0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0