Backtracking

Tuesday, March 3, 2026 8:29 AM

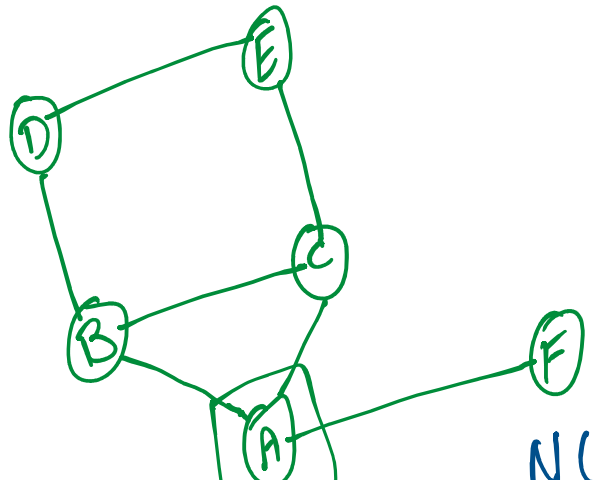
HW5 today
HW 6 Next Tue
GPS 6 " Mon

Recurse + try all possibilities.
(Reuse).

★ Maximum Independent set (MIS)

Optimization Problem:
Given undirect graph $G = (V, E)$ $n = |V|$, $m = |E|$
Find $S \subseteq V$ maximizes $|S|$
Goal: ↓ Sol'n s.t. $\forall u, v \in S$ $(u, v) \notin E$
opt. sol'n

eg.



$S = \{F, C, D\}$
$S = \{F, B, E\}$
} optimum

$S = \{\}$

$S = \{A, E\}, \{A, D\}$

$N(A) = \{B, C, F\}$

$N(A) = \{B, C, F\}$

Alg 0: Bruteforce. Try every possible subset $S \subseteq V$ & check if I.S. Pick the largest sized that is I.S.

Time: $O(2^n \cdot m)$

Alg 1: Backtracking:

Idea: Pick a vertex $v \in V$

case I: v is not in opt sol'n ($\equiv$ MIS)

Then recurse $(G - \{v\})$

↳ Graph constructed by removing v & it's incident edges.

edges.

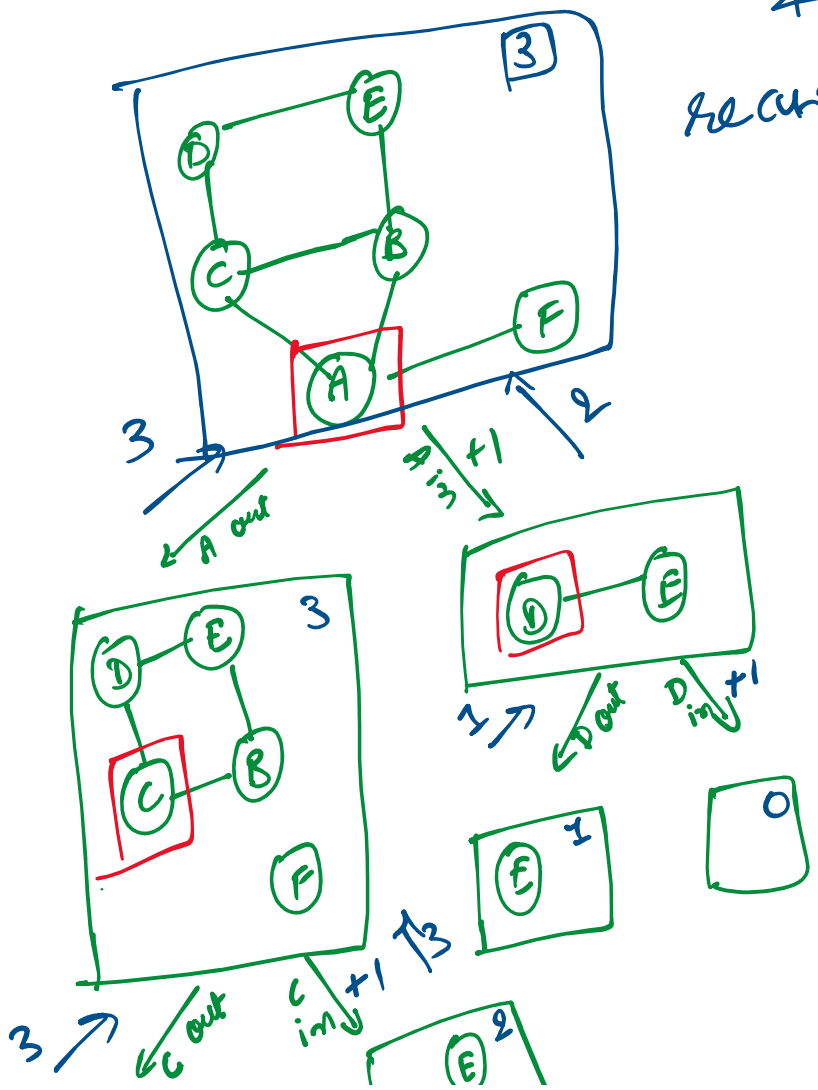
Case II: v is in opt sol'n

then $S = S \cup \{v\}$

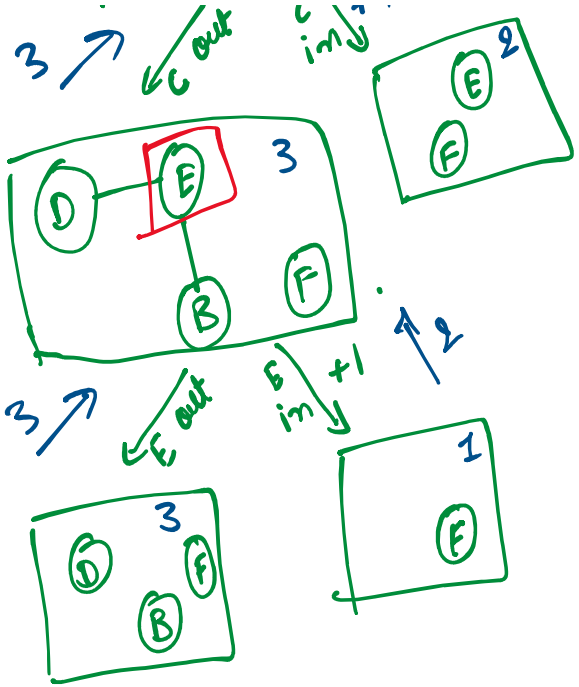
&

Recurse $(G - \{v\} - N(v))$

$N(v) = \{u \mid (u,v) \in E\}$




```
MIS(G) {                                   → T(n)
  // Returns size of MIS in G.
  // G = (V,E), n = |V|
  (1) if n = 0 return 0.
  (2) pick any vertex v ∈ V
  (3)
      Return max { MIS(G - v),                  ← T(n-1)
                   1 + MIS(G - v - N(v)) }      → T(n-1-deg(v))   (|N(v)|)
}
```



↑

↳ $T(n-1-\deg(v))$

$T(n) = T(n-1) + T(n-1-\deg(v)) + O(m)$

Analysis 0: $\deg(v) \geq 0$

$T(n) = 2\,T(n-1) + O(n)$

$= O(2^n \cdot n)$

Analysis 1:

All deg = 0 vertices are always in opt. sol'n

Observation:

First put all 0 nbrs in S'

Remove them from G & nbrs start.

$\deg(v) \geq 1.$

$T(n-1) + T(n-2) + O(n)$

$\deg(v) = 1$.

$$T(n) = T(n-1) + T(n-2) + O(n)$$

$$= O(1.618^n \cdot n)$$

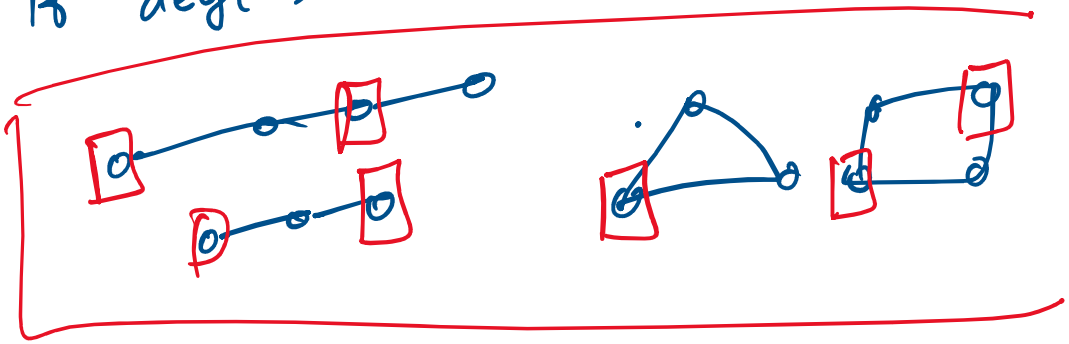
$$x^n = x^{n-1} + x^{n-2}$$

$$\Rightarrow x^2 = x + 1 \Rightarrow x = \frac{1+\sqrt{5}}{2}$$

= Golden ratio

Analysis 2: Pick v with largest $\deg(v)$

if $\deg(v) \leq 2$ Hen easy.



$\deg(v) \geq 3$

$$T(n) = T(n-1) + T(n-4) + O(n)$$

$$= (1.381^n \cdot n)$$

$$= O(1.991^n)$$

Best known: $O(1.1991^n)$

[Xiao & Nagamochi '17]

Longest Increasing Subsequence (LIS)

Tuesday, March 3, 2026 8:29 AM

Given (LIS) a sequence of #s

$a_1\ a_2\ a_3\ \ldots\ a_n$

Find longest subsequence $i_1, i_2, \ldots, i_\ell$

s.t. $i_1 < i_2 < \ldots < i_\ell$

$a_{i_1} < a_{i_2} < \ldots < a_{i_\ell}$

eg. 10 2 3 1 5 10 7 3 4 11 ← 5

1 5 7 11 ← 4

Alg 0: Bruteforce $O(2^n, n)$.

i/p : $\langle a_1, \ldots, a_n \rangle$

Alg 1: Case I: a_n is not in opt sol'n (LIS)

recurse $\langle a_1, \ldots, a_{n-1} \rangle$

Case II: a_n is in opt sol'n.

recurse $\langle a_1, \ldots, a_{n-1} \rangle$

but now we can only pick elements smaller than a_n.

Gives an UB.

Init. UB = ∞

$T(n) \leftarrow$ LIS $(\langle a_1, \ldots, a_n \rangle, UB)$

length of LIS in $\langle a_1, \ldots, a_n \rangle$ LIS $< UB$

$T(n) \leftarrow LIS(\ldots$

// Return length of LIS in $\langle a_1, \ldots, \ldots$ s.t. all ele in LIS $< UB$

① If $n=0$ then return 0. $\leftarrow O(1)$

② If $a_n < UB$

$$\text{Return } \max \begin{cases} LIS(\langle a_1, \ldots, a_{n-1} \rangle, UB) & \leftarrow T(n-1) \\ 1 + LIS(\langle a_1, \ldots, a_{n-1} \rangle, a_n) & \rightarrow T(n-1) \end{cases}$$

③ Else $(a_n \geq UB)$

Return $LIS(\langle a_1, \ldots, a_{n-1} \rangle, UB)$

}

Time: $T(n) = T(n-1) + T(n-1) + O(1)$

Time : $T(n) = T(n-1) + \ldots$

$= O(2^n)$

Analysis : Q: How many different subproblems in my recursion tree?

$(\langle a_1, \ldots a_{n-1}, a_n \rangle, \; UB)$

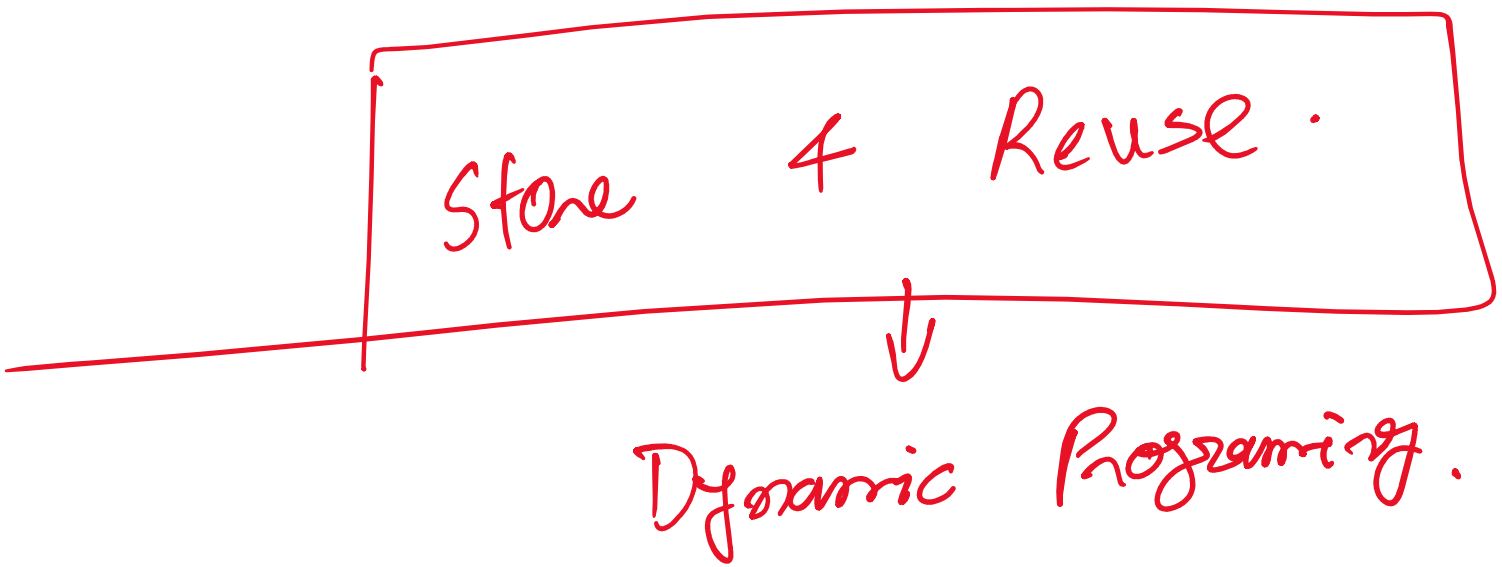
$\hookrightarrow \in \{\infty, a_1, \ldots, a_n\}$

All prefixes n of them

$(n+1)$ of them

$\Downarrow$

$O(n^2) = n \cdot (n+1)$ many subproblems!

$O(n^2) = n \cdot (n+1)$

Solving same subprob. over & over & over again!