Even More DP

Tuesday, March 10, 2026 1:50 PM

Recall: "Try all possibilities"

Recursion + Memoization + Reuse → Iterative Alg.

3 steps:

① Define subproblems.

② Recursive Formula to solve subproblems } → Iterative Alg

③ (Memoize) Evaluation Order.

★ Max. v Independent set in a Tree. (MWIT)
Weight

(directed acyclic graph $T = (V, E)$)

Given an undirected acyclic graph $T = (V, E)$

$\Leftrightarrow$ tree

& weights $w(v) \geq 0 \quad \forall v \in V$

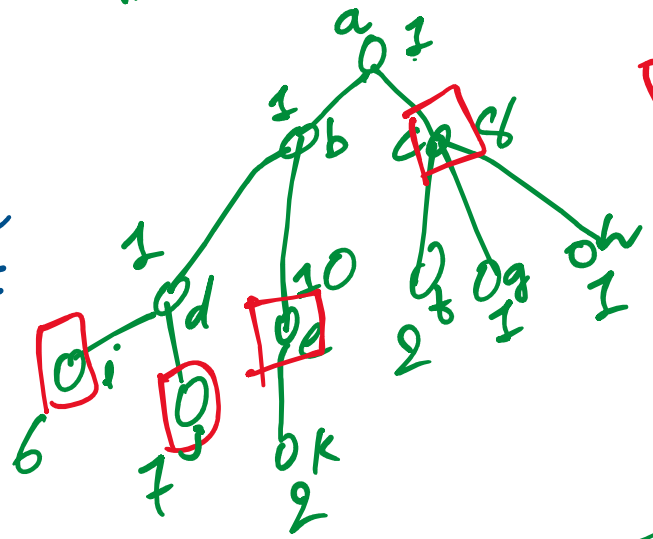
Find maximum $\sum_{v \in S} w(v)$

$S \subseteq V$

s.t. $\forall u, v \in S \quad (u, v) \notin E$.

Ans:

Example:



subtree rooted at b

Terminology: (in example)

Root: a

children: $C(a) = \{b, c\}$

Grand children: $GC(a) = \{d, e, f, g, h\}$

subtree:

Leaf: $\{i, j, k, f, g, h\}$

skip

Root

Intuition: Root → SKIP
↘

Case I: skip → Recurse on the subtrees rooted at the children of v

Case II: take → Recurse on the subtrees rooted at the grandchildren

(1) Define Subproblems:

$\forall v \in V$

$MWIST(v)$ = weight of max weight independent set in the subtree rooted at v.

② Recursive Formula.

Basecase: $v \in V$, v is a leaf

$$MWIT(v) = w(v)$$

Recursive Formula:

$$MWIT(v) = \max \begin{cases} \text{skip} & \sum_{u \in C(v)} \frac{MWIT(u)}{M[u]} \\ \text{take} & w(v) + \sum_{u \in GC(v)} \frac{MWIT(u)}{M[u]} \end{cases}$$

$O(n)$ ← $O(n)$

Ans: MWIT (Root)

③ Evaluation Order.

$M[u] = MWIT(u)$

Which order traversal?

Post order traversal

(before evaluating for u evaluated for children ⇓ evaluated for grandchildren.)

Time: $n = |V|$

subprob = n

Work / subprob = $O(n)$

$\Big\}$ $O(n^2)$

Extra

* Iterative Alg:

obvs ① $PV = [v_1, v_2 \ldots v_i \; v_n]$ post-order traversal of V

② for $i = 1 \ldots n$ {

③ if v_i is a leaf

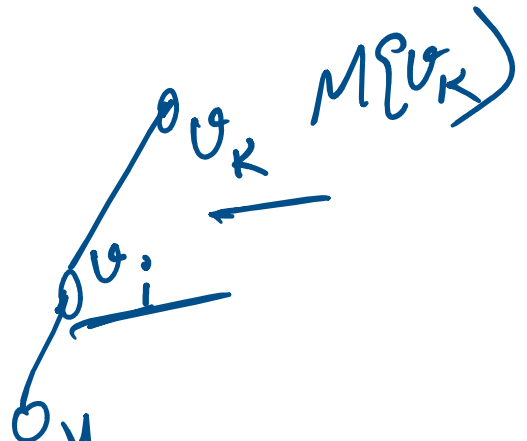
$M[v_i] = w(v_i)$

④ else

$M[v_i] = \max \Big\{ \sum_{u \in C(v_i)} M[u]$

④ else $M[v_i] = \max \begin{cases} u \in C(v_i) \\ w(v_i) + \sum_{u \in GC(v_i)} M[u] \end{cases}$

⑤ } Return: $M[v_n]$:



Amortized time Analysis:

n values computed in M

each $M[u]$ is accesed twice on RHS.

$\Downarrow$

$O(n)$.

$O(n)$.

Parsing

Thursday, March 12, 2026 9:09 AM

Given a context-free-gramer (CFG)

$$G = (T, V, P, S)$$

start Variable

terminals e.g. {0,1}

Variables

Production rules

& $x \in T^*$, $x = a_1 a_2 a_2 \ldots a_n$.

Check if $x \in L(G)$. $\rightarrow$ O/P = TRUE or FALSE

example:

in CNF

$$\begin{cases} S \rightarrow AS \mid BC \\ A \rightarrow BC \\ B \rightarrow BB \mid 1 \\ C \rightarrow CC \mid 0 \end{cases}$$

$x = 010111$ O/P FALSE

$= 101010$ TRUE

LC → CNF

Assumption: G is in CNF

Each production rule is of the form

either $A \to BC$

or $A \to t$, for a $t \in T$.

★ CKY (Cocke - Younger - Kasami '70).

① subproblems:

$\forall\ 1 \le i \le j \le n$, $\forall A \in V$

$Par(i, j, A)$ = TRUE if $a_i a_{i+1} \cdots a_j$ is generated by G starting at A.

= FALSE o.w.

$= \text{FALSE}$ o.w.

② Recursive Formula

Base case: $\forall i = 1 \ldots n, \quad \forall A \in V$

$P_{dn}(i, i, A) = \text{TRUE}$ if $(A \to a_i) \in P$.

$= \text{FALSE}$ o.w.

★ $i < j \equiv \underbrace{a_i a_{i+1} \ldots}_{B} \underbrace{\ldots a_j}_{C}$ has atleast two characters.

$A \to BC$

$$P_{dn}(i, j, A) = \bigvee_{(A \to V_1 V_2) \in P} \bigvee_{k = i \ldots (j-1)} \left(P_{dn}(i, k, V_1) \wedge P_{dn}(k+1, j, V_2) \right)$$

for some $V_1, V_2 \in V$

③ Evaluation Order.

$P[i, j, A] = Par(i, j, A)$

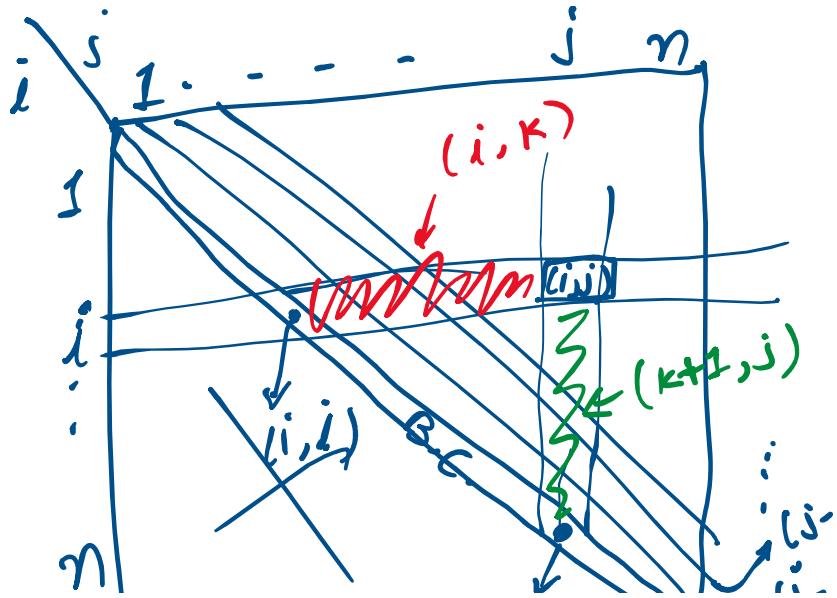
$i = 1 \ldots n$

$j = 1 \ldots n$

$A \in V$

$\rightarrow$ $n \times n \times |V|$ dimensional array.

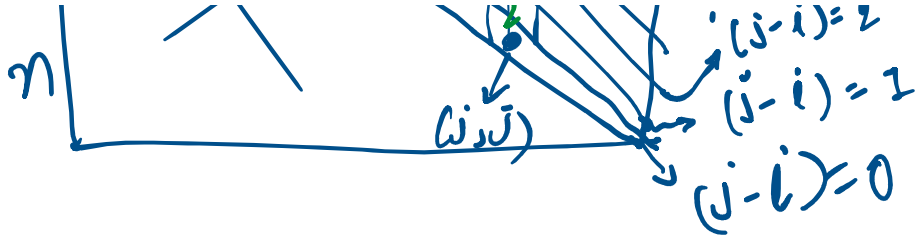
Slice $n \times n$ for $A \in V$



B.C.: $Par(i, i, A)$ $\forall i = 1 \ldots n$ $\forall A \in V$

R.F. $i < j$

$\bigvee_{k=i..(j-1)} (Par(i, k, V_1) \wedge Par((k+1), j, V_2))$



☆ In increasing order of (j-i) {
In any order of variables V ∈ .

☆ Run Time :

subproblems = $n^2 * |V|$

time / subprob. = $O(n * |P|)$

} → $O(n^3 * |V| * |P|)$

Valiant '75 : $O(n^{2.373} * |P|)$

Now, mostly linear-time parsers for most programming languages.