COL 702 Lecture 19 Oct 8

Dynamic Programming characterization

1. Recurrence for the solution which unfolds into many common sub-problems

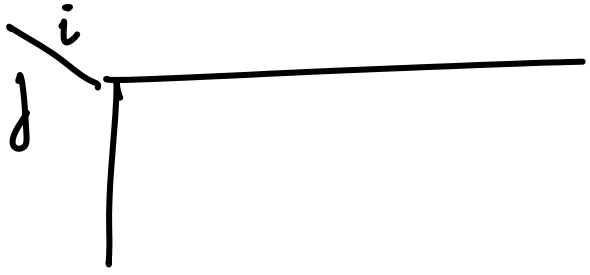
for optimization problems, the optimal soln itself is expressed as a recurrence : principle of optimality

- Proof of correctness is implicit in the recurrence

2. Define a table to store previously computed solns

Table is typically indexed by the parameters of the recurrence

$S(i, j, \ldots) = S(i+1, j-1, \ldots) + \ldots$



3. Analysis. We compute the time required to fill up the table
- although the final solution that we are interested in is "one entry"

$\leq$ . . Time to . compute an table entry
all entries
. that depends on some other entries

4. Space-time tradeoff
to economize the size of the table

Knapsack problem

w_i : weights
p_i : profits
B: capacity

Max $\sum x_i p_i$

s.t $\sum x_i w_i \leq B$

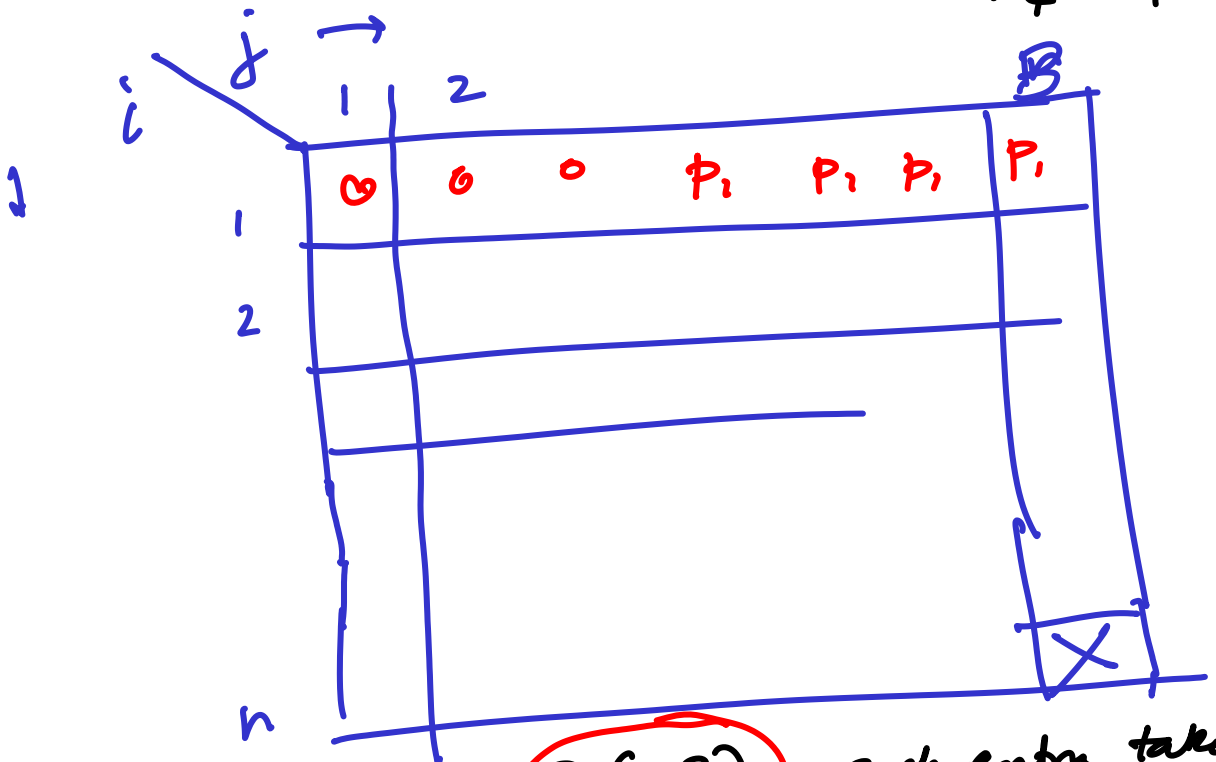
$x_i \in \{0,1\}$

$K(i,j)$: optimal soln for knapsack with capacity j and objects $\{1 \dots i\}$

generalised soln $1 \le i \le n$ $1 \le j \le B$

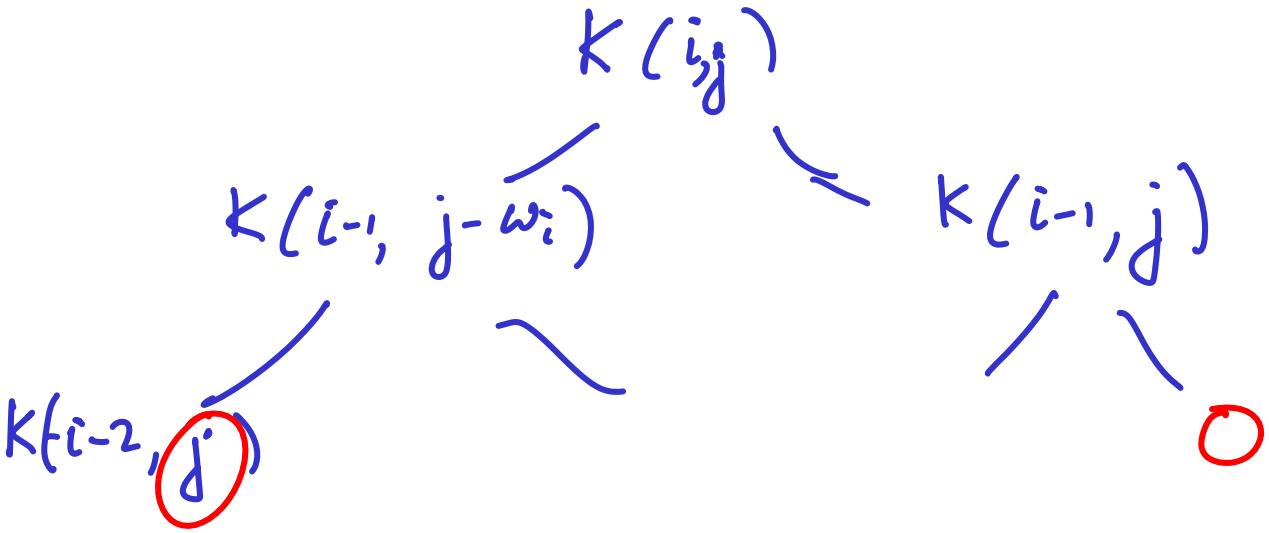
$K(n, B)$: the optimal soln of the original problem

$$K(i,j) = \max \begin{cases} p_i + K(i-1, j-w_i) & \text{if } x_i \in \text{optimal soln} \\ K(i-1, j) & \text{otherwise } x_i \notin \text{optimal soln} \end{cases}$$

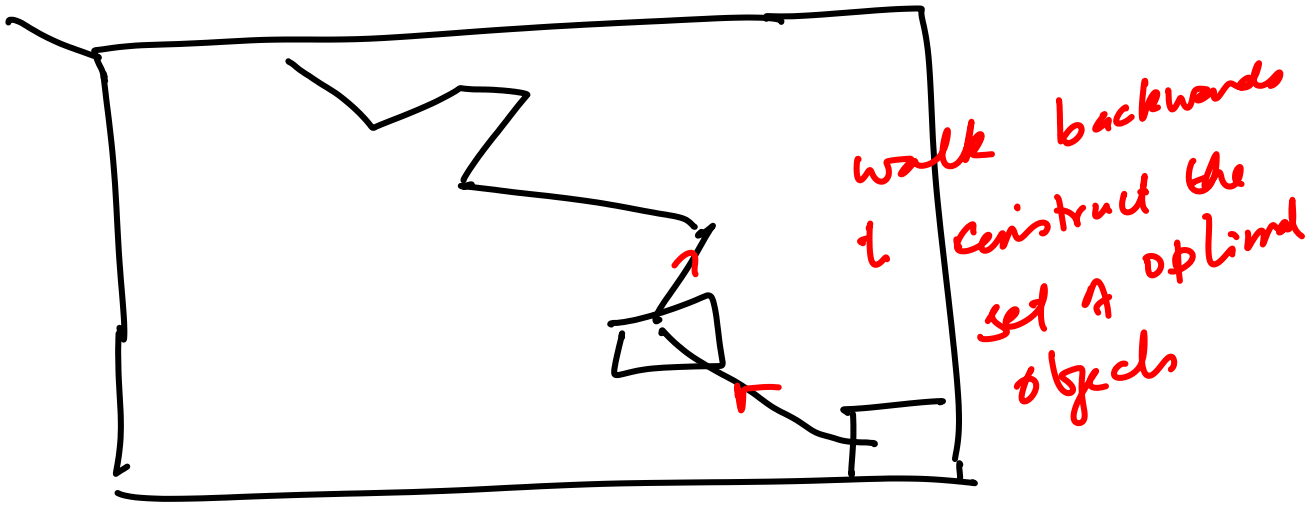


Running Time : $O(nB)$ each entry takes $O(1)$ time

Space : $O(nB)$ $2B$ by maintaining two rows



If we want to know which objects define the optimal soln, we cannot erase the information about the rows



Size of input : $p_1\ p_2 \dots p_n \quad w_1, w_2 \dots w_n,$ B: $\log B$ bits

$2n + \log B$

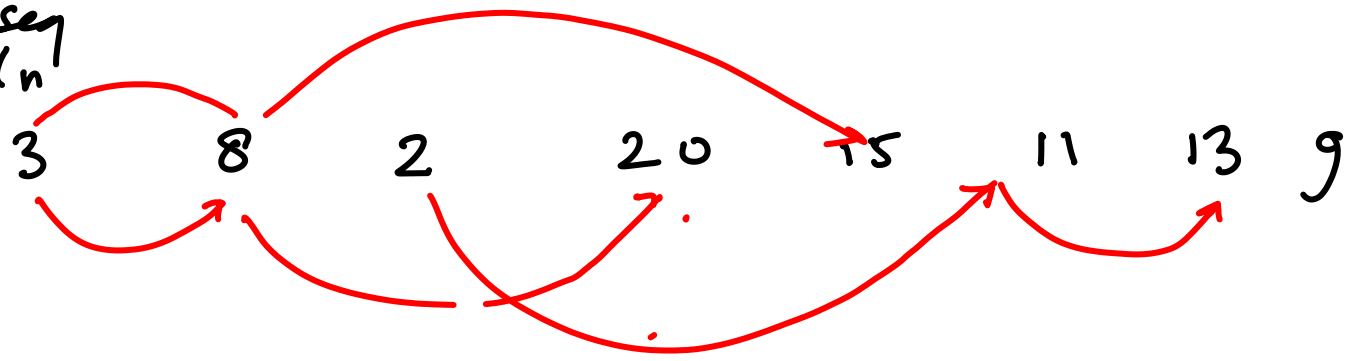
Input size $2n \cdot b + \log B = N$ b: max # bits used for p_i w_i

Run time $O(n \cdot b \cdot B)$

If $\log B = \frac{N}{2}$ $\Rightarrow$ Run time $\frac{N}{2} \, 2^{N/2}$

If $B \leq n^2$

Given seq $x_1, x_2 \ldots x_n$



3, 8, 15

3, 8, 20

2, 11, 13

Monotonically increasing subsequences

$x_{i_1} \; x_{i_2} \; x_{i_3} \ldots x_{i_k}$

$i_1 < i_2 < i_3 \ldots < i_k$ $\quad x_{i_1} \leq x_{i_2} \leq x_{i_3} \ldots$

Find the longest non inc subseq.

Candidate Subproblems

. MIS (i) : Longest non inc sub $x_1, x_{i_2} \ldots x_i$

It is known that any arbitrary sequence of length n has either

(1) a non inc. subseq of length $\sqrt{n}$

or

(2) " non dec subs of length $\sqrt{n}$

[Erdos-Szekeres thm]