Lecture 34

CSE 331

Nov 17, 2014

HW grading schedule

Sorry for the delays!

Weighted Interval Scheduling

Input: n jobs (s_i, f_i, v_i)

Output: A schedule S s.t. no two jobs in S have a conflict

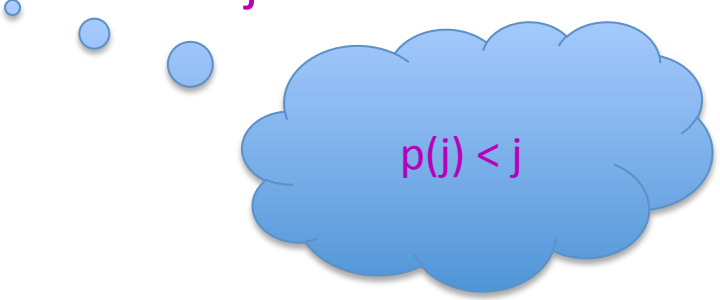
Goal: max $\Sigma_{i \text{ in } S} v_j$

Assume: jobs are sorted by their finish time

Couple more definitions

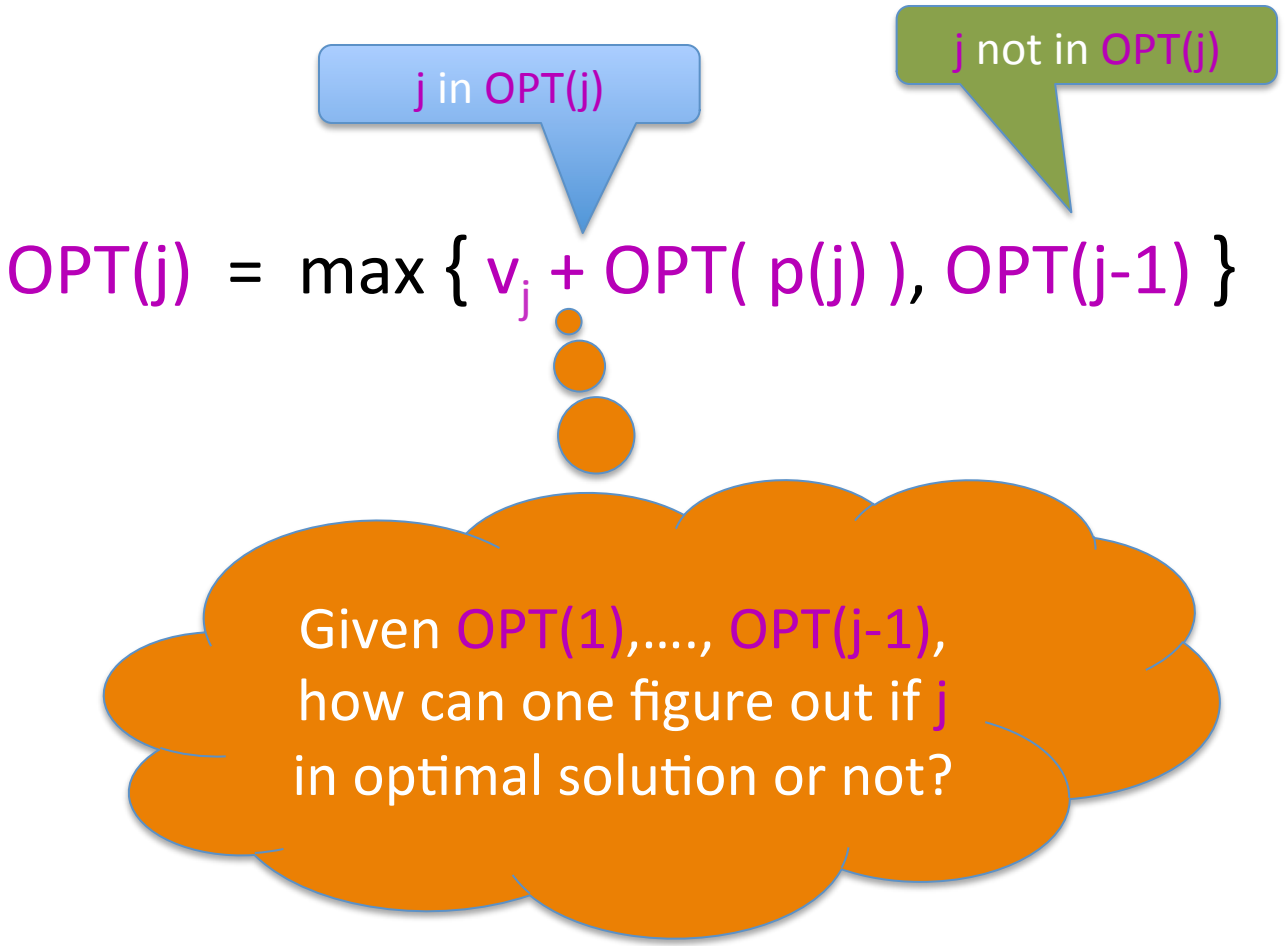
p(j) = largest i<j s.t. i does not conflict with j

= 0 if no such i exists

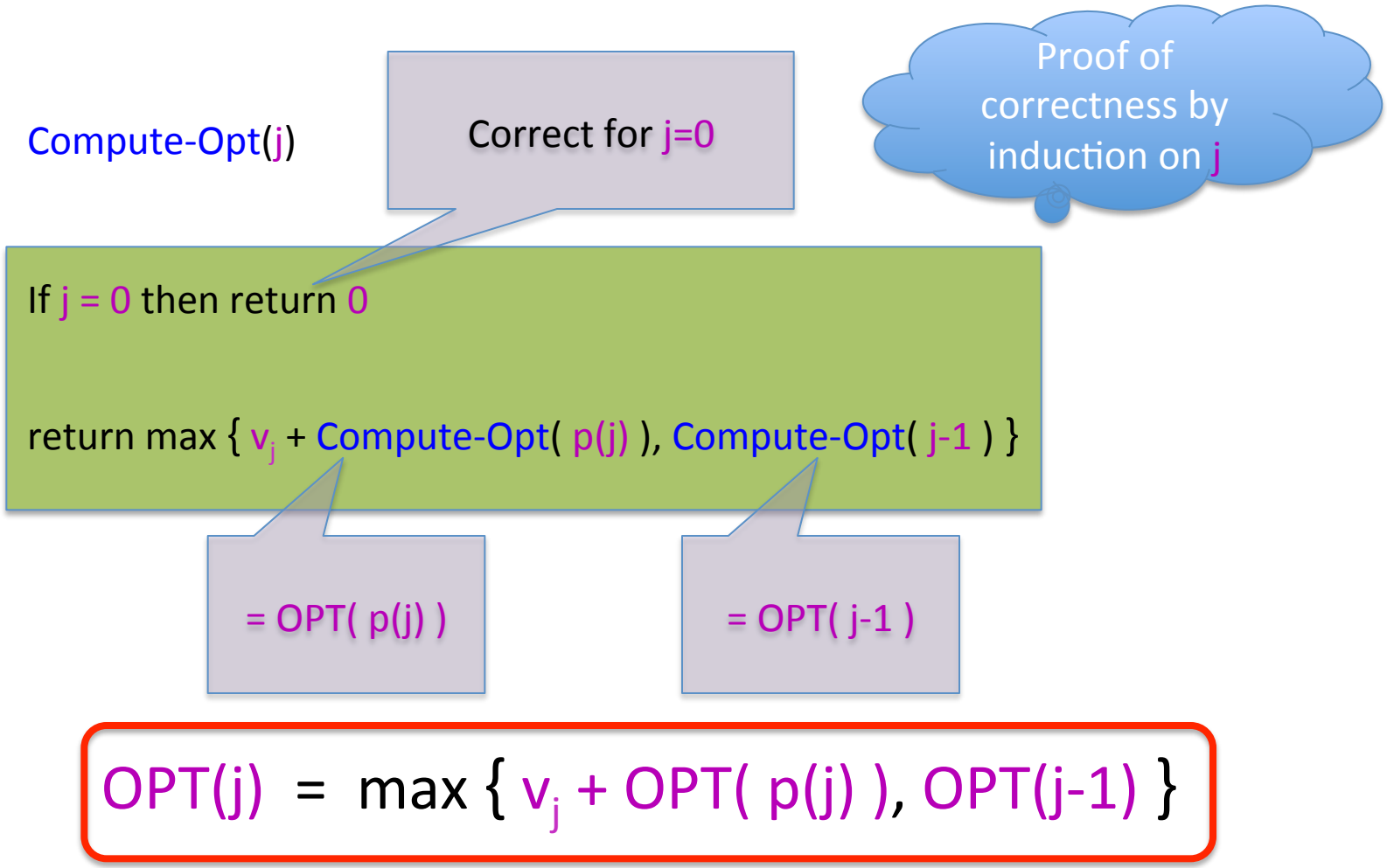


OPT(j) = optimal value on instance 1,..,j

Property of OPT

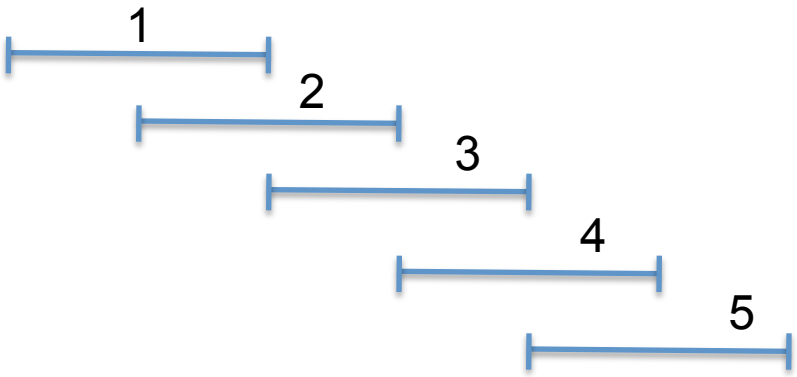


A recursive algorithm



$$OPT(j) = \max \{ v_j + OPT(p(j)), OPT(j-1) \}$$

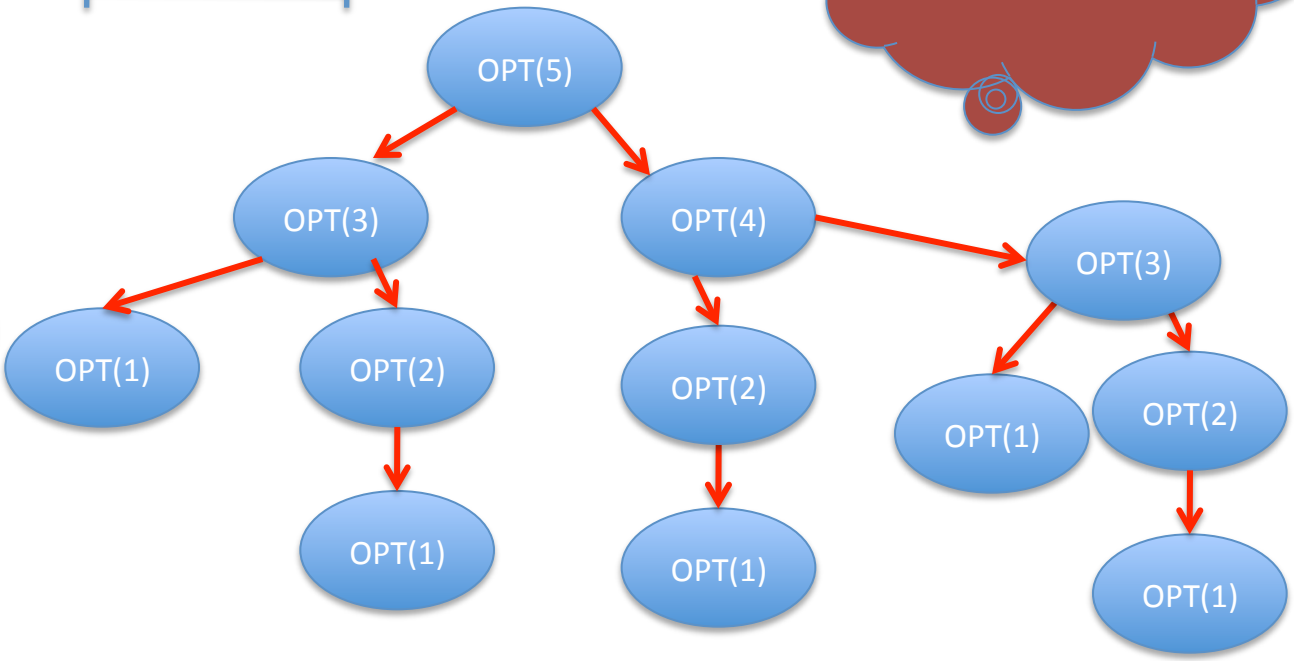
Exponential Running Time



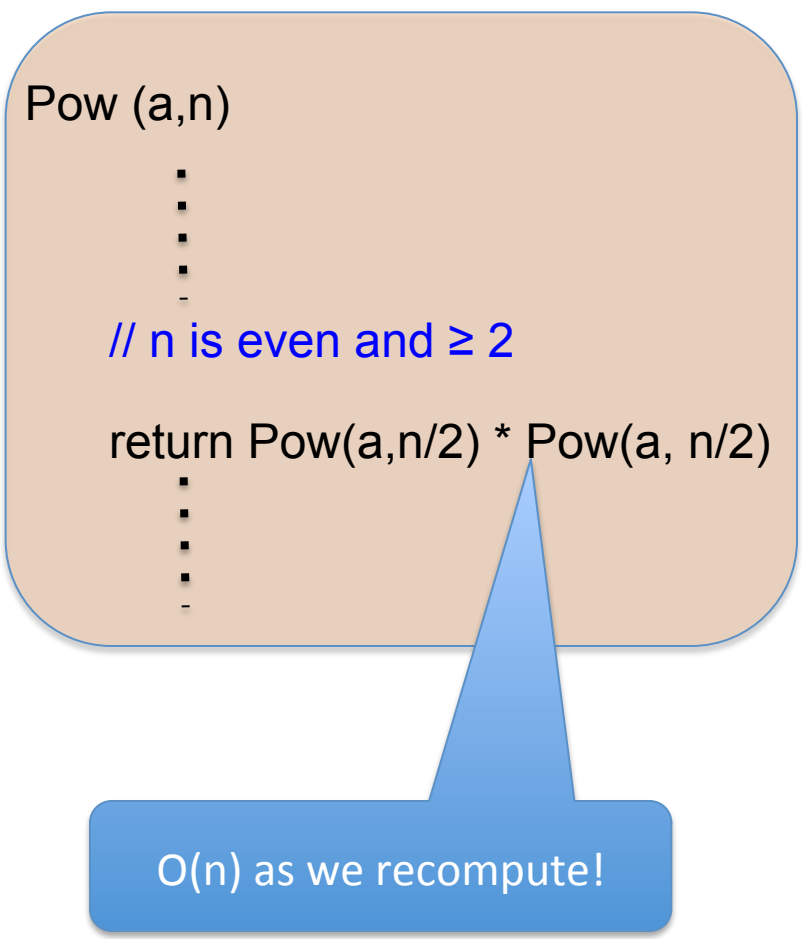
p(j) = j-2

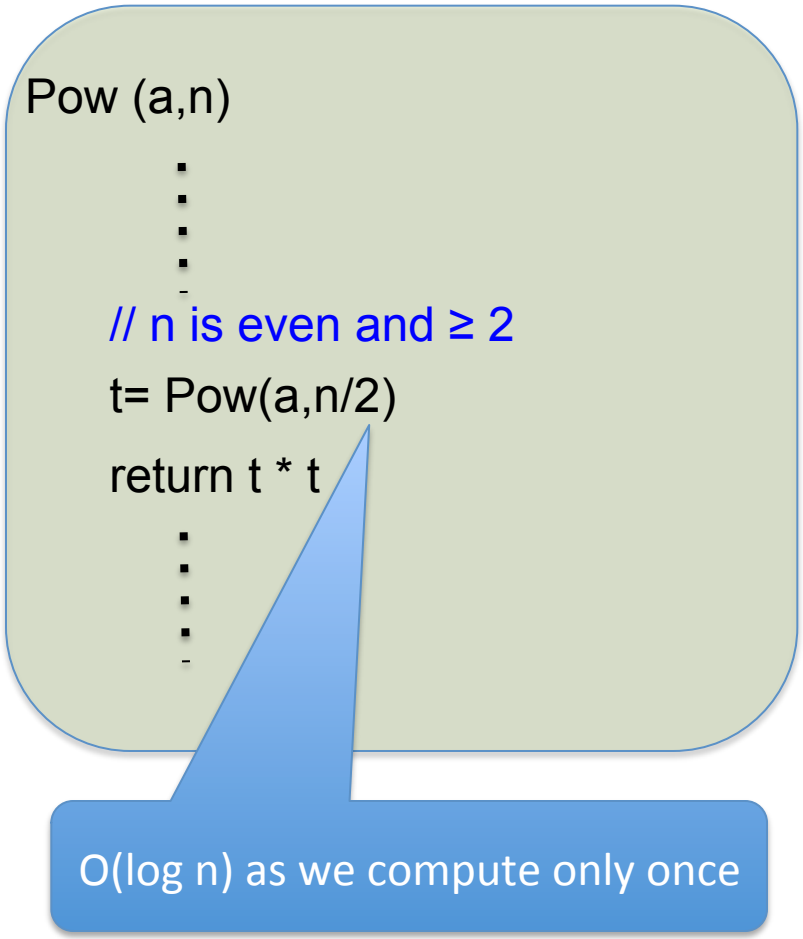
Only 5 OPT values!

Formal proof: Ex.



Using Memory to be smarter





How many distinct OPT values?

A recursive algorithm

M-Compute-Opt(j)

If $j = 0$ then return 0

If $M[j]$ is not null then return $M[j]$

$M[j] = \max \{ v_j + \text{M-Compute-Opt}(p(j)), \text{M-Compute-Opt}(j-1) \}$

return $M[j]$

M-Compute-Opt(j)
= OPT(j)

Run time = O(# recursive calls)

Bounding # recursions

M-Compute-Opt(j)

If j = 0 then return 0

If M[j] is not null then return M[j]

M[j] = max { v_j + M-Compute-Opt(p(j)), M-Compute-Opt(j-1) }

return M[j]

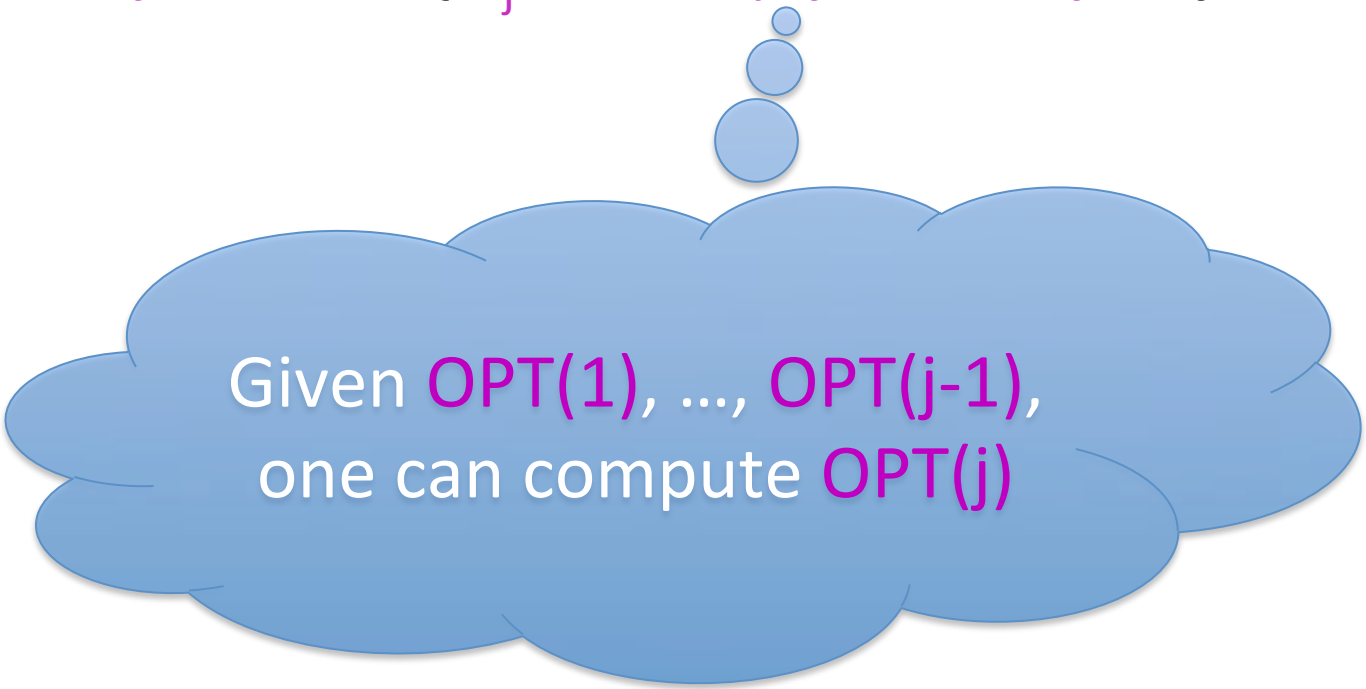
O(n) overall

Whenever a recursive call is made an M value is assigned

At most n values of M can be assigned

Property of OPT

$$OPT(j) = \max \{ v_j + OPT(p(j)), OPT(j-1) \}$$



Recursion+ memory = Iteration

Iteratively compute the OPT(j) values

Iterative-Compute-Opt

M[0] = 0

For j=1,…,n

M[j] = max { v_j + M[p(j)], M[j-1] }

M[j] = OPT(j)

O(n) run time

Reading Assignment

Sec 6.1, 6.2 of [KT]

When to use Dynamic Programming

Richard Bellman

There are polynomially many sub-problems

Optimal solution can be computed from solutions to sub-problems

There is an ordering among sub-problem that allows for iterative solution