Lecture 31

CSE 331

A recursive algorithm

Compute-Opt(j)

Correct for j=0

Proof of correctness by induction on j

If j = 0 then return 0

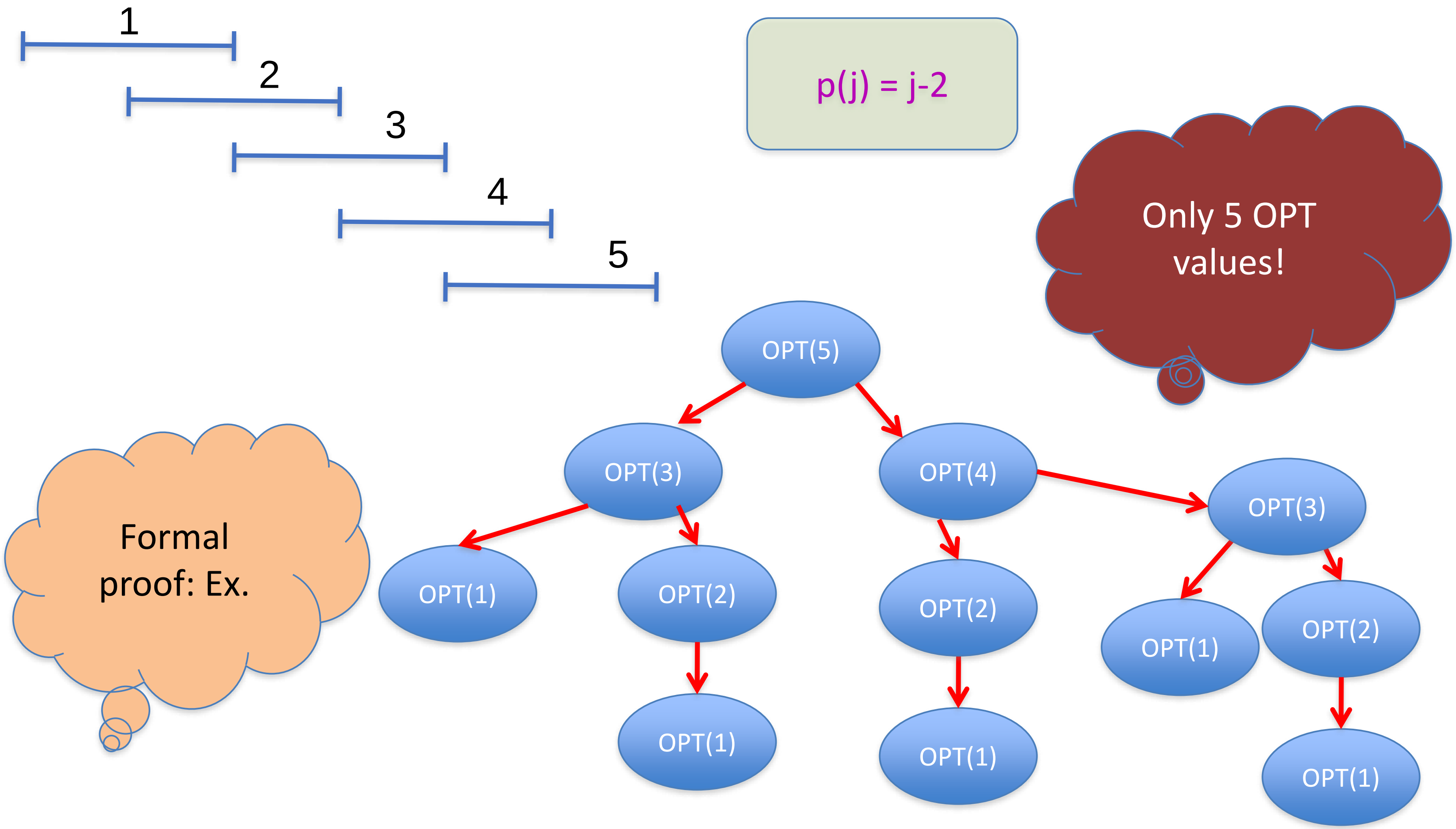
return max { v_j + Compute-Opt(p(j)), Compute-Opt(j-1) }

= OPT(p(j))

= OPT(j-1)

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

Exponential Running Time



Using Memory to be smarter

Using more space can reduce runtime!

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 + M-Compute-Opt(p(j)), 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

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

Given OPT(1), …, OPT(j-1), one can compute OPT(j)

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

Algo run on the board…

Reading Assignment

Sec 6.1, 6.2 of [KT]

When to use Dynamic Programming

Richard Bellman

There are polynomially many sub-problems

OPT(1), …, OPT(n)

Optimal solution can be computed from solutions to sub-problems

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

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

OPT (j) only depends on OPT(j-1), …, OPT(1)

Scheduling to min idle cycles

n jobs, i^{th} job takes w_i cycles

You have W cycles on the cloud

What is the maximum number of cycles you can schedule?

Subset sum problem

Input: n integers $w_1, w_2, \ldots, w_n$

bound W

Output: subset S of $[n]$ such that

(1) sum of w_i for all i in S is at most W

(2) $w(S)$ is maximized

Rest of today’s agenda

Dynamic Program for Subset Sum problem

Algo on the board…