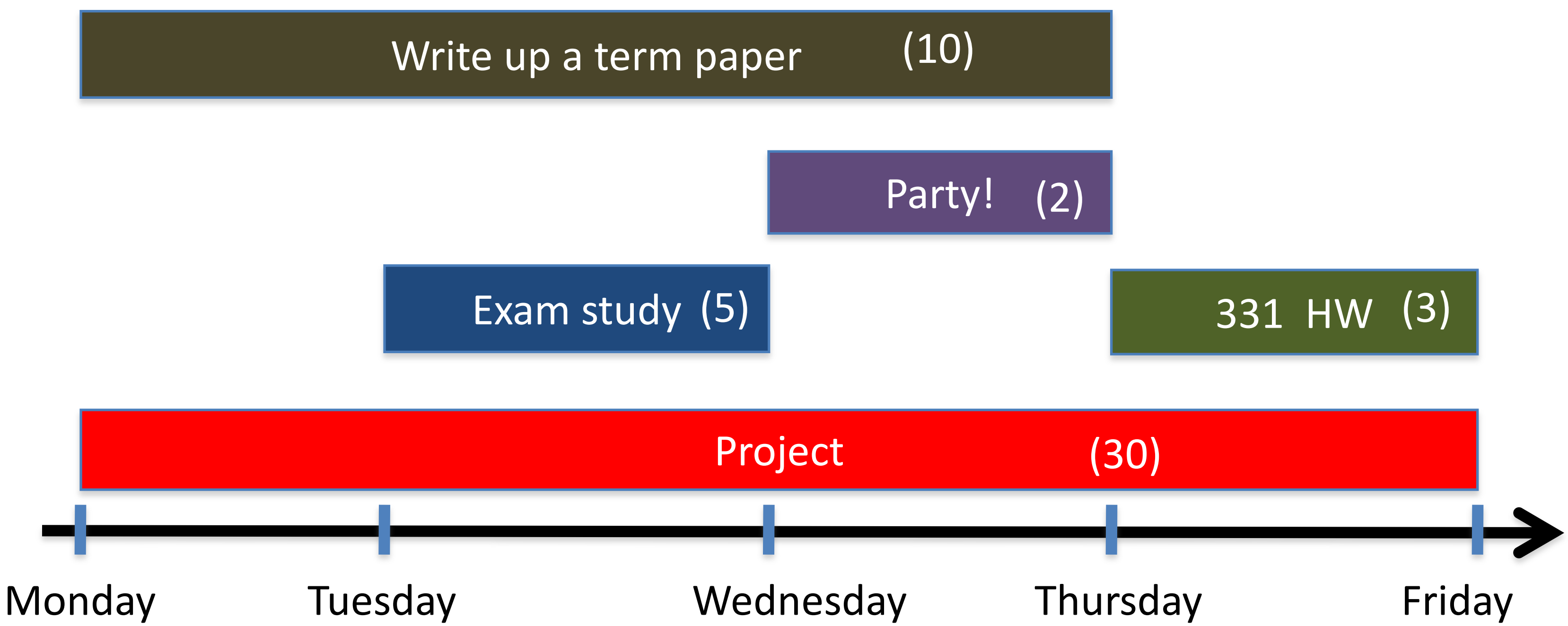
Lecture 29

CSE 331

Apr 13, 2020

End of Semester blues

Can only do one thing at any day: what is the optimal schedule to obtain maximum value?



Previous Greedy algorithm

Order by end time and pick jobs greedily

Greedy value = 5+2+3= 10

Write up a term paper (10)

Party! (2)

Exam study (5)

331 HW (3)

Project (30)

OPT = 30

Monday Tuesday Wednesday Thursday Friday

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

Today’s agenda

Finish designing a recursive algorithm for the problem

Couple more definitions

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

= 0 if no such i exists

p(j) < j

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

Property of OPT

j in OPT(j)

j not in OPT(j)

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

Given OPT(1),…., OPT(j-1),
how can one figure out if j
in optimal solution or not?


?
ICANHASCHEEZBURGER.COM

A recursive algorithm

Correct for $j=0$

Proof of correctness by induction on j

Compute-Opt(j)

If $j = 0$ then return 0

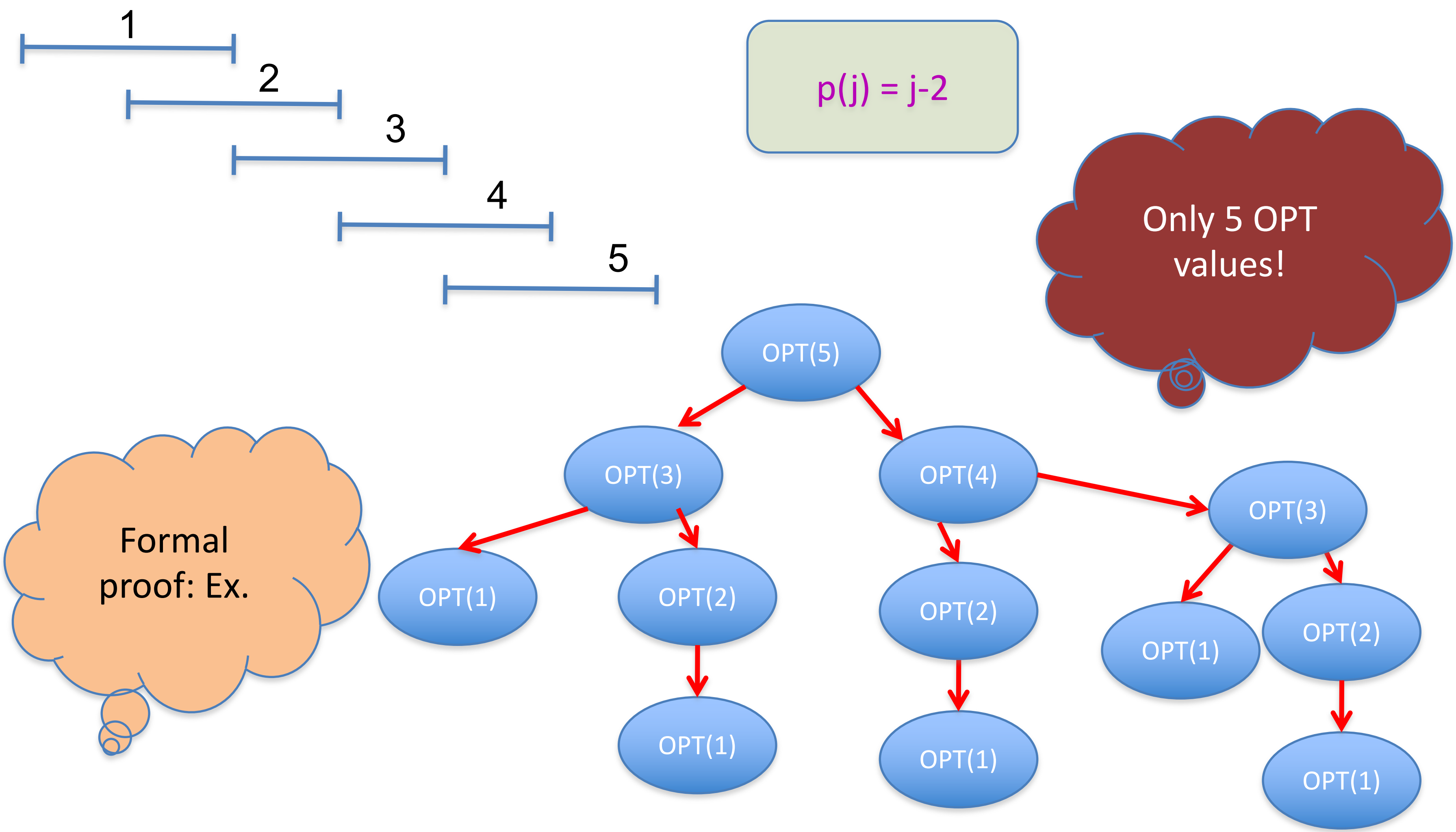
return max { v_j + Compute-Opt($p(j)$), Compute-Opt($j-1$) }

$= OPT(p(j))$

$= OPT(j-1)$

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

Exponential Running Time




?
ICANHASCHEEZBURGER.COM

Using Memory to be smarter

Using more space can reduce runtime!

How many distinct OPT values?

A recursive algorithm

M-Compute-Opt(j)

M-Compute-Opt(j) = 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]

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


?
ICANHASCHEEZBURGER.COM

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)