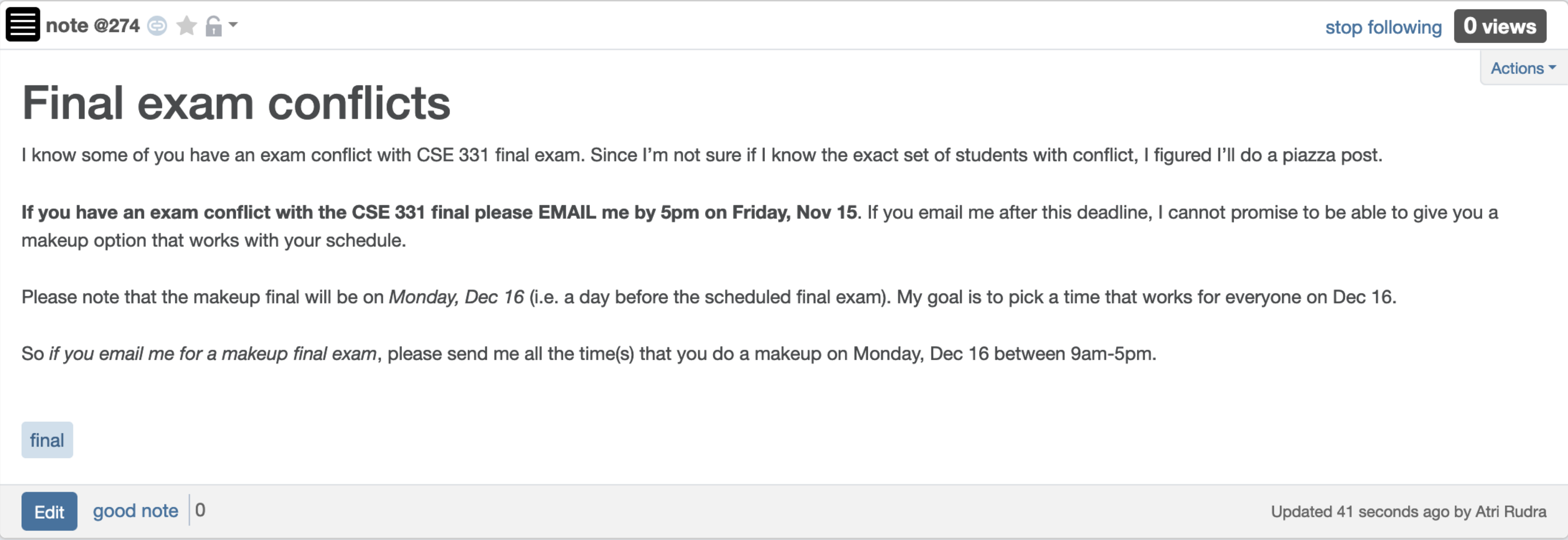
Lecture 29

CSE 331

Nov 8, 2024

Final exam conflict



Final exam conflicts

I know some of you have an exam conflict with CSE 331 final exam. Since I'm not sure if I know the exact set of students with conflict, I figured I'll do a piazza post.

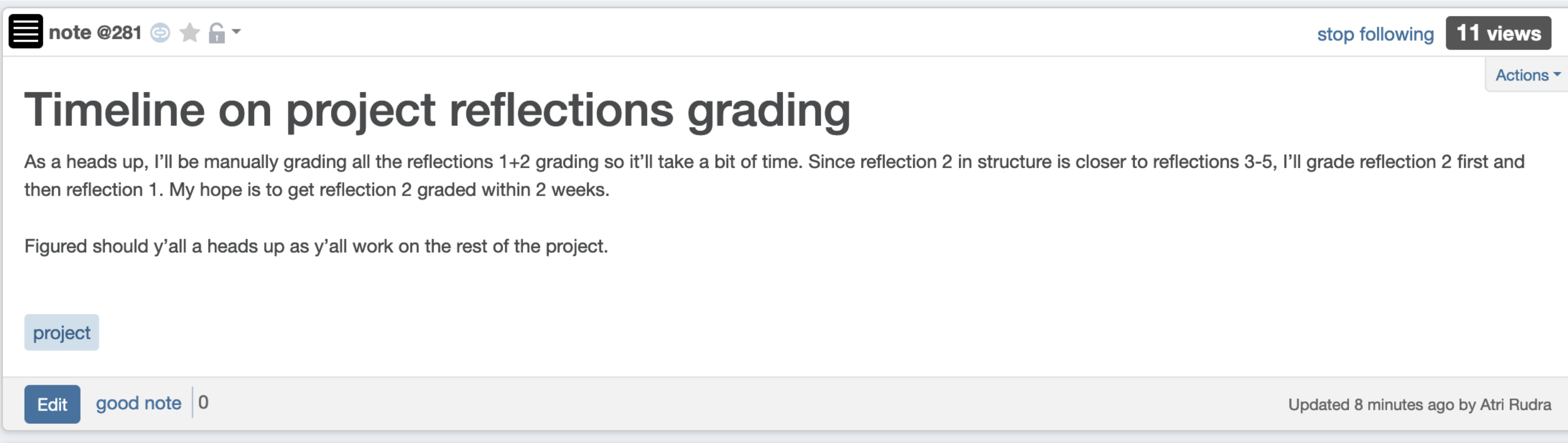
If you have an exam conflict with the CSE 331 final please EMAIL me by 5pm on Friday, Nov 15. If you email me after this deadline, I cannot promise to be able to give you a makeup option that works with your schedule.

Please note that the makeup final will be on *Monday, Dec 16* (i.e. a day before the scheduled final exam). My goal is to pick a time that works for everyone on Dec 16.

So *if you email me for a makeup final exam*, please send me all the time(s) that you do a makeup on Monday, Dec 16 between 9am-5pm.

Updated 41 seconds ago by Atri Rudra

Reflections 2+1 grading timeline



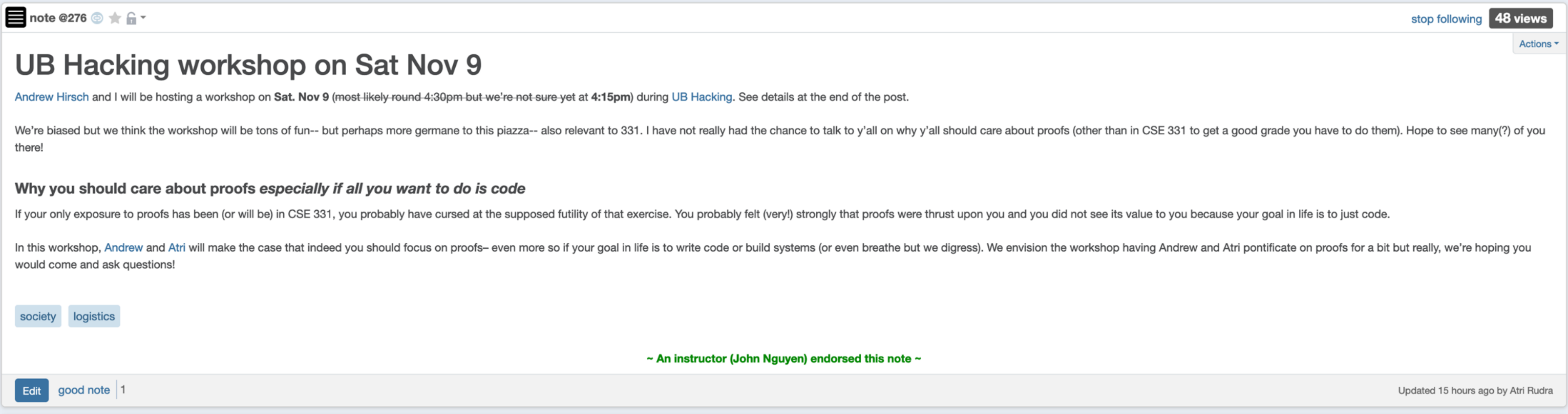
Timeline on project reflections grading

As a heads up, I'll be manually grading all the reflections 1+2 grading so it'll take a bit of time. Since reflection 2 in structure is closer to reflections 3-5, I'll grade reflection 2 first and then reflection 1. My hope is to get reflection 2 graded within 2 weeks.

Figured should y'all a heads up as y'all work on the rest of the project.

Updated 8 minutes ago by Atri Rudra

UB Hacking Workshop on Sat



UB Hacking workshop on Sat Nov 9

Andrew Hirsch and I will be hosting a workshop on **Sat. Nov 9** (~~most likely round 4:30pm but we're not sure yet~~ at **4:15pm**) during UB Hacking. See details at the end of the post.

We're biased but we think the workshop will be tons of fun-- but perhaps more germane to this piazza-- also relevant to 331. I have not really had the chance to talk to y'all on why y'all should care about proofs (other than in CSE 331 to get a good grade you have to do them). Hope to see many(?) of you there!

Why you should care about proofs *especially if all you want to do is code*

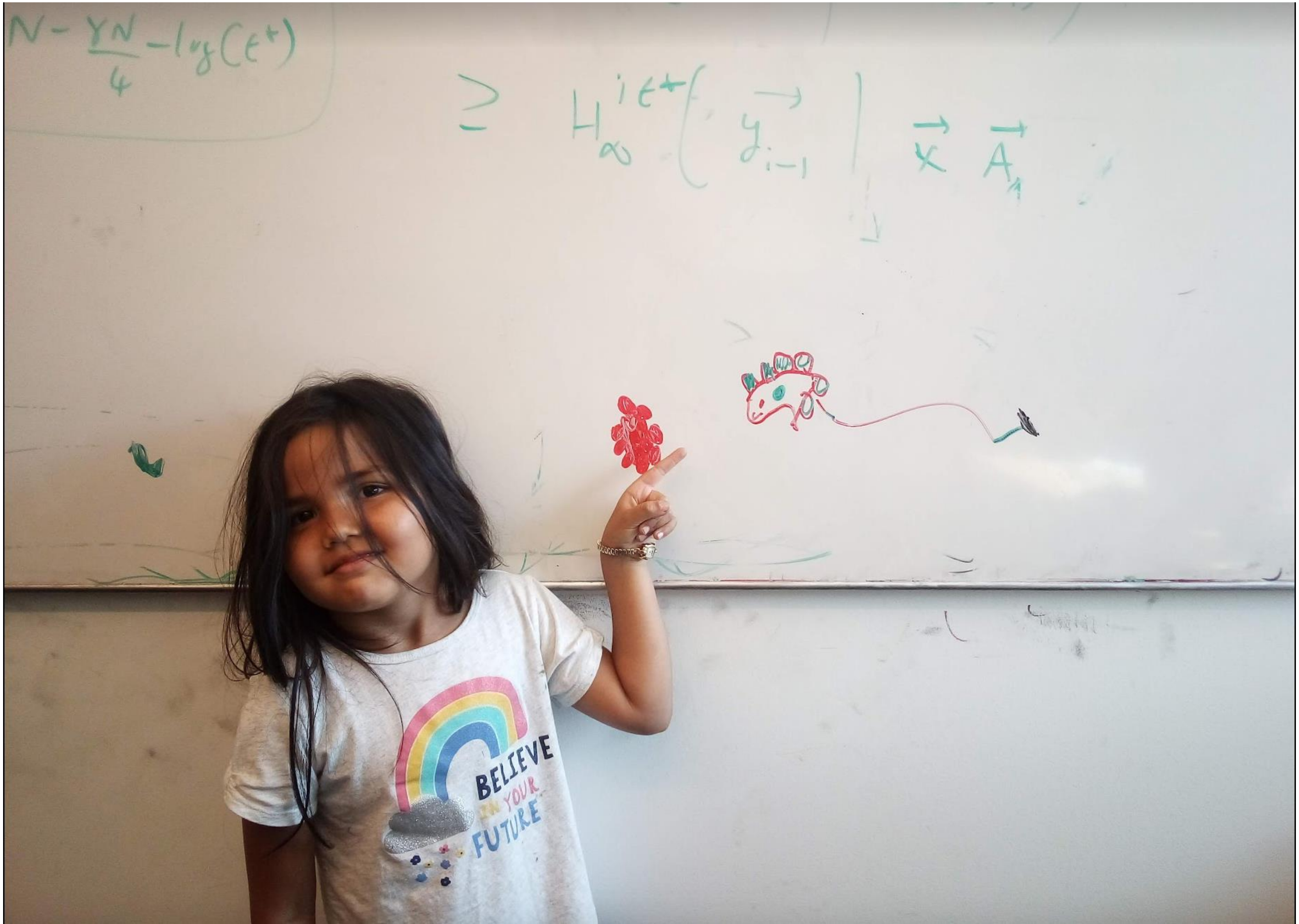
If your only exposure to proofs has been (or will be) in CSE 331, you probably have cursed at the supposed futility of that exercise. You probably felt (very!) strongly that proofs were thrust upon you and you did not see its value to you because your goal in life is to just code.

In this workshop, Andrew and Atri will make the case that indeed you should focus on proofs– even more so if your goal in life is to write code or build systems (or even breathe but we digress). We envision the workshop having Andrew and Atri pontificate on proofs for a bit but really, we're hoping you would come and ask questions!

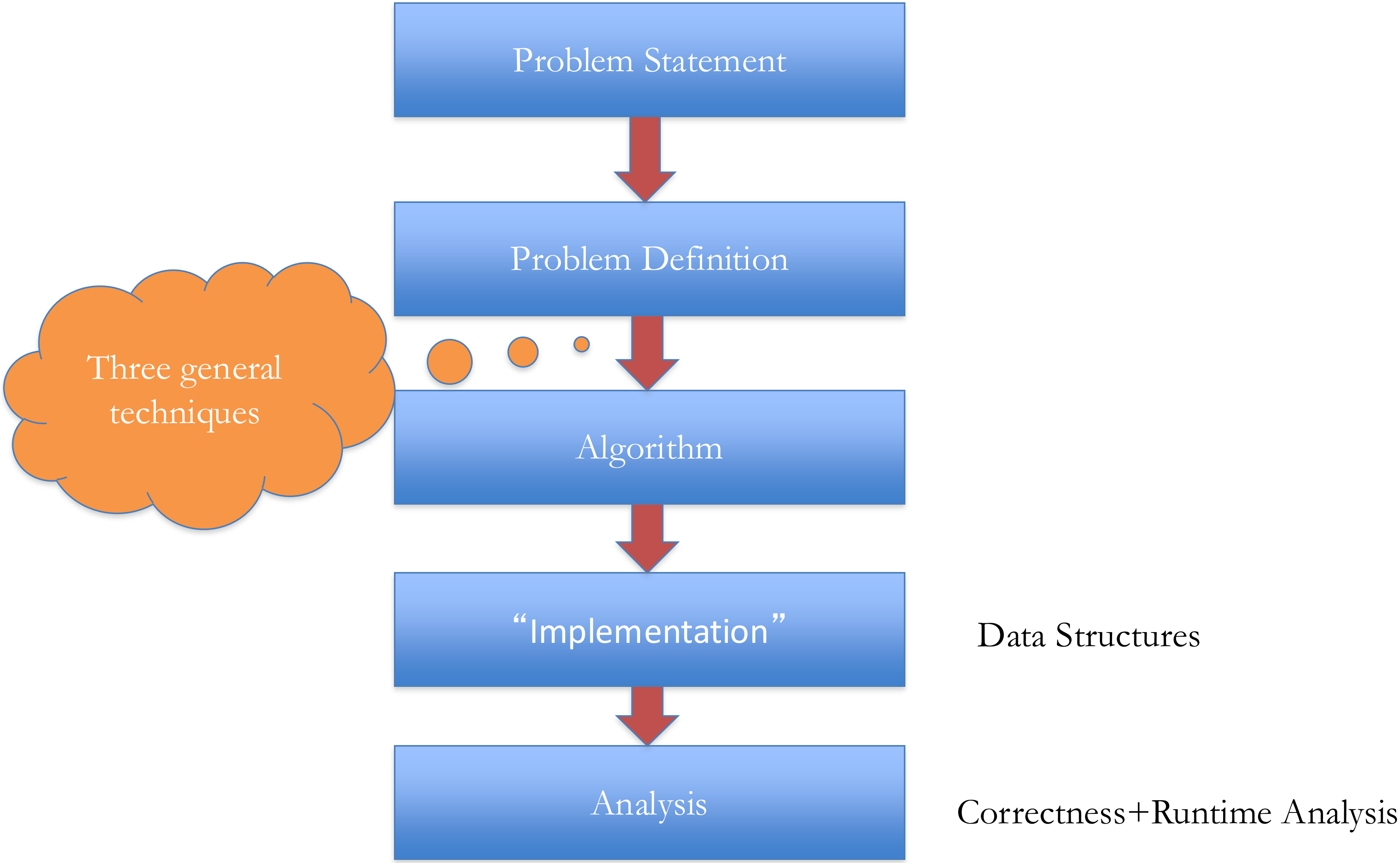
~ An instructor (John Nguyen) endorsed this note ~

Updated 15 hours ago by Atri Rudra

Questions/Comments?

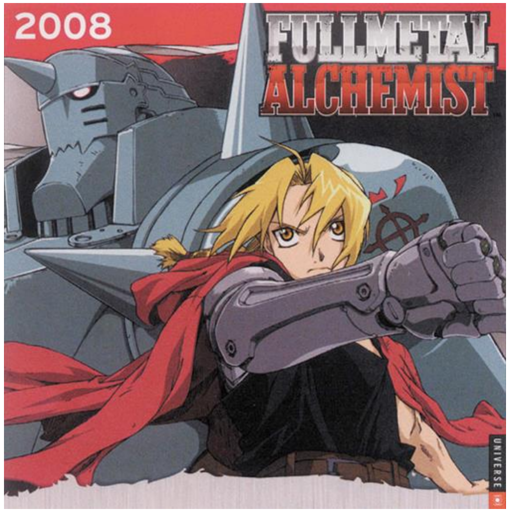


High level view of CSE 331



Greedy Algorithms

Natural algorithms



Reduced exponential running time to polynomial

Divide and Conquer

Recursive algorithmic paradigm

Reduced large polynomial time to smaller polynomial time

A new algorithmic technique

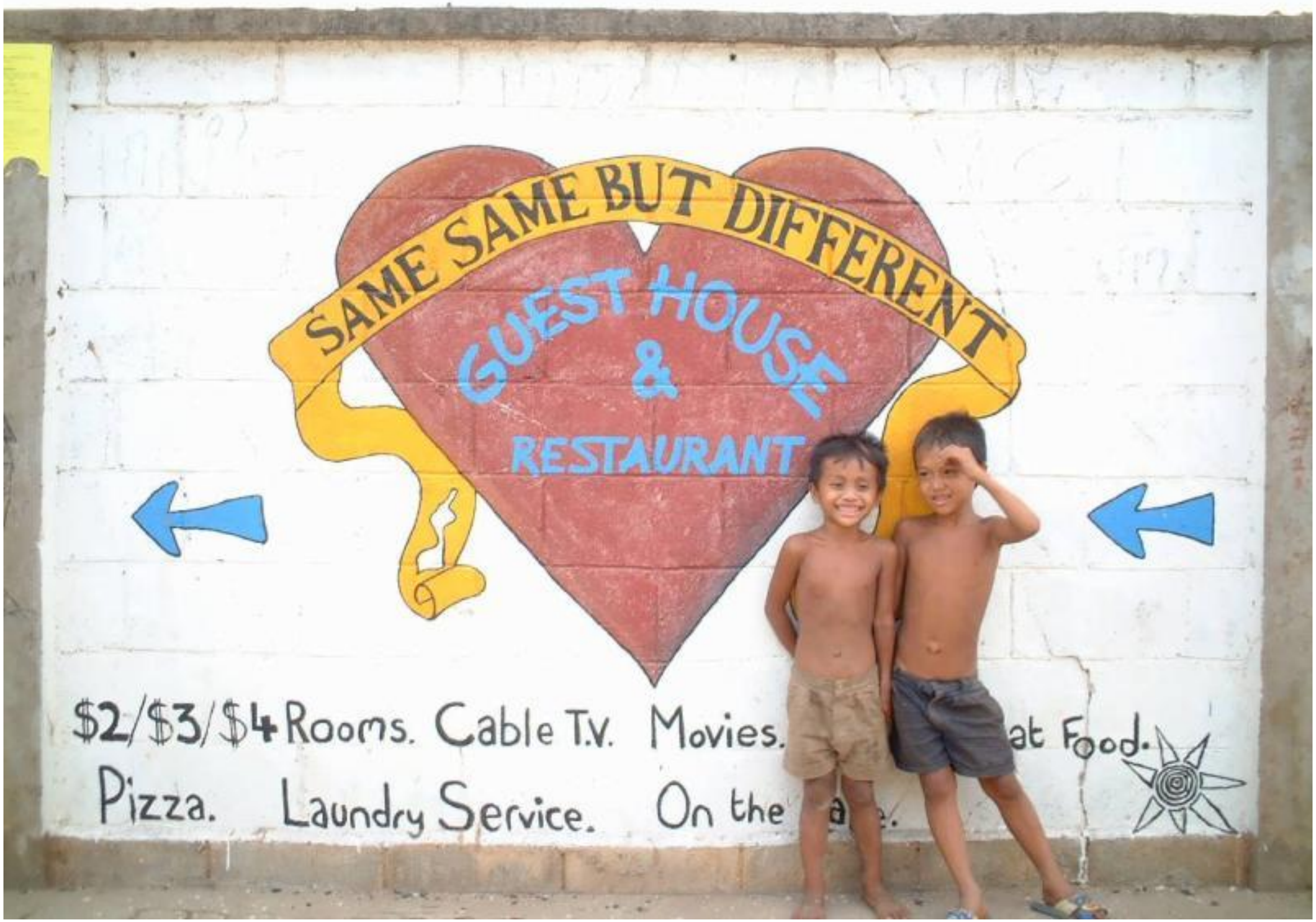
Dynamic Programming

Dynamic programming vs. Divide & Conquer

SAME SAME BUT DIFFERENT
GUEST HOUSE
&
RESTAURANT
$2/$3/$4 Rooms. Cable T.V. Movies. at Food.
Pizza. Laundry Service. On the

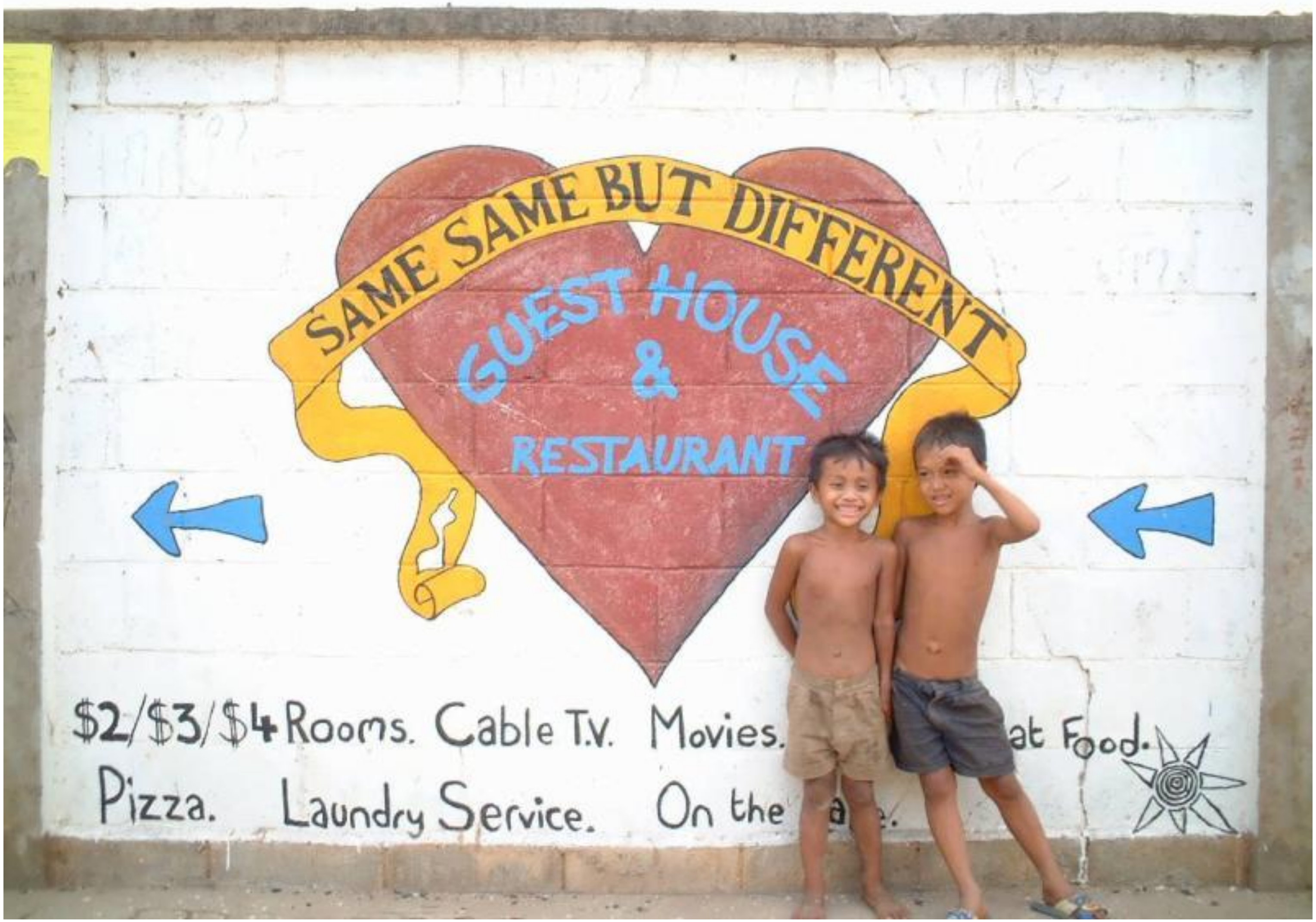
Same same because

Both design recursive algorithms


SAME SAME BUT DIFFERENT
GUEST HOUSE
&
RESTAURANT
$2/$3/$4 Rooms. Cable T.V. Movies. at Food.
Pizza. Laundry Service. On the

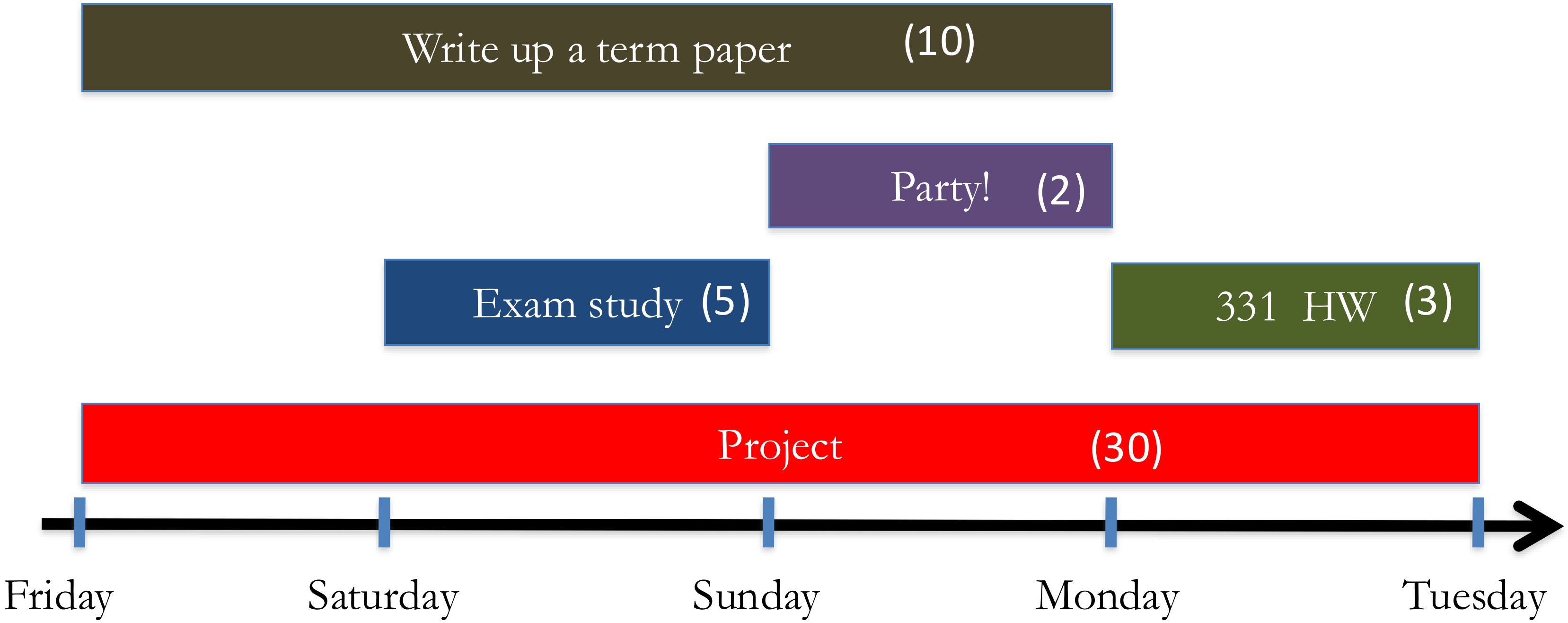
Different because

Dynamic programming is smarter about solving recursive sub-problems


SAME SAME BUT DIFFERENT
GUEST HOUSE
&
RESTAURANT
$2/$3/$4 Rooms. Cable T.V. Movies.
at Food.
Pizza. Laundry Service. On the

End of Semester blues

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



Write up a term paper (10)

Party! (2)

Exam study (5)

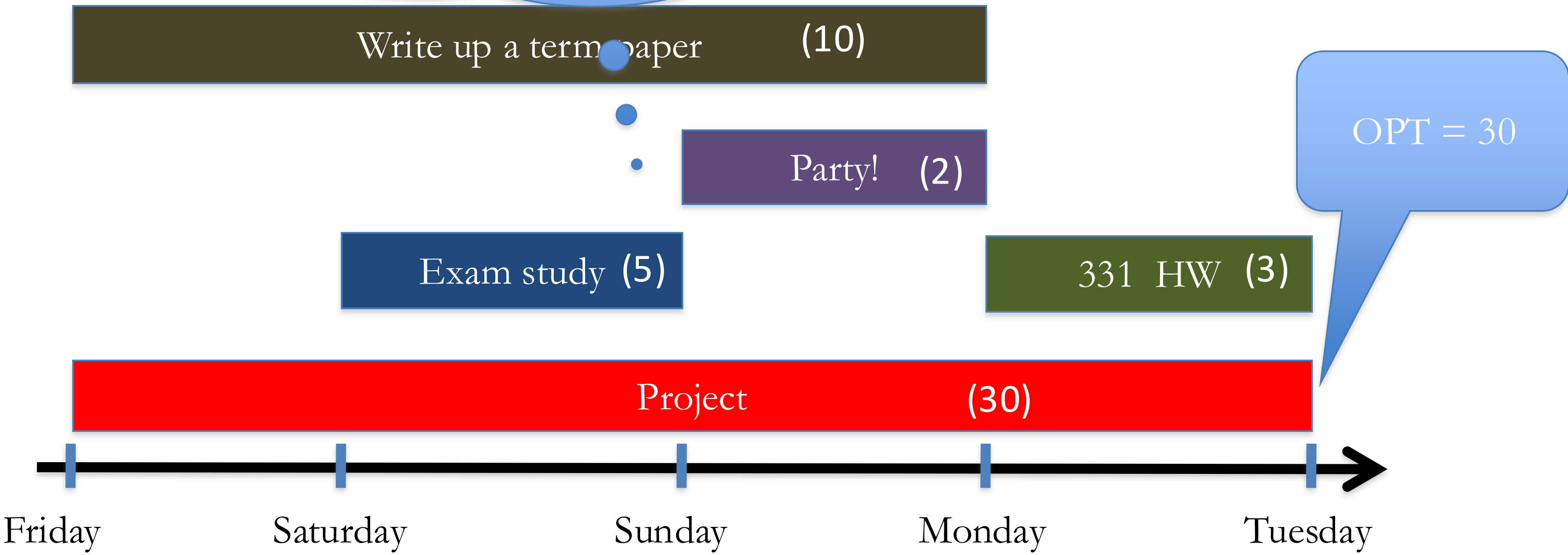
331 HW (3)

Project (30)

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

Friday

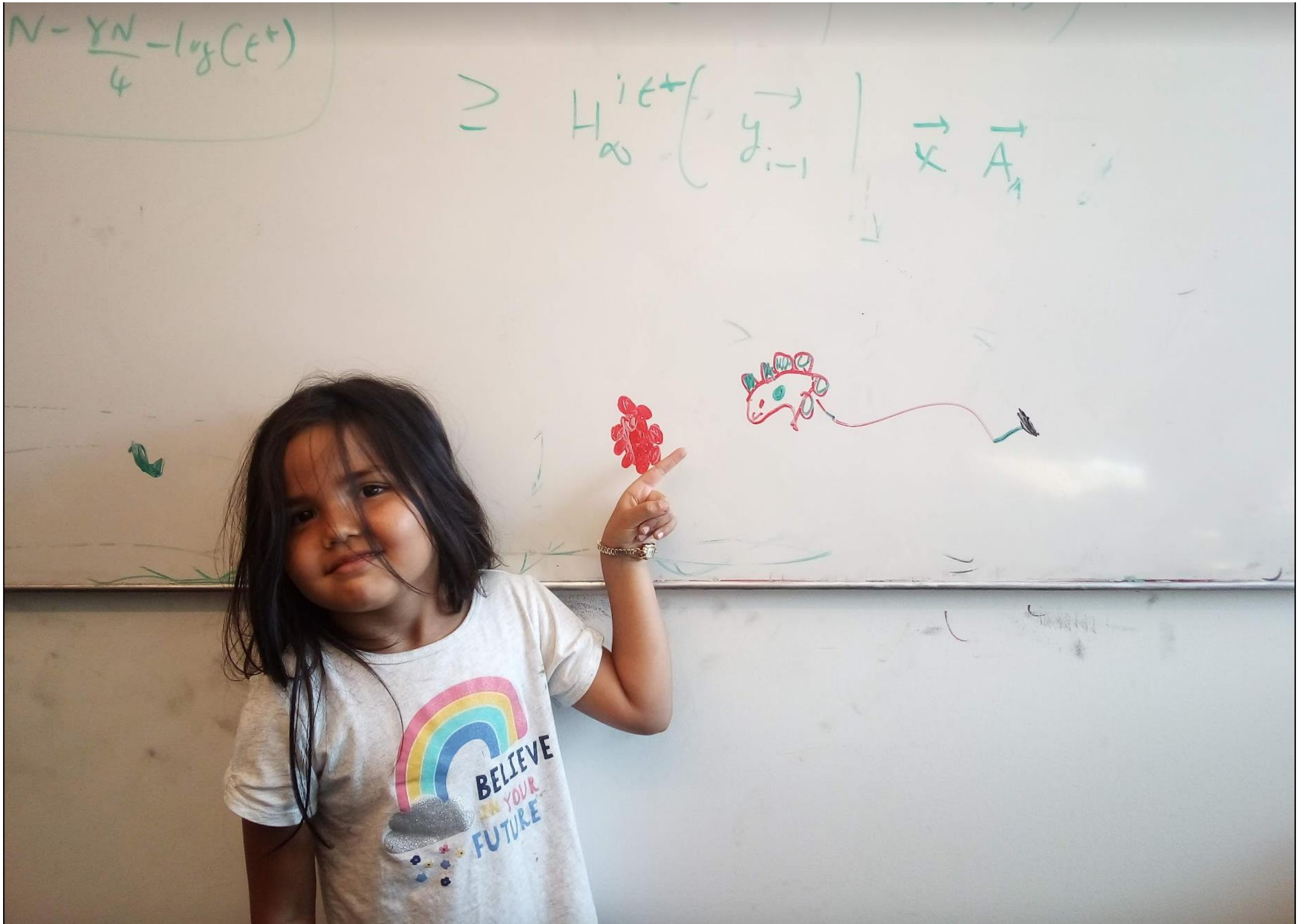
Saturday

Sunday

Monday

Tuesday

Questions/Comments?



Today’s agenda

Formal definition of the problem

Start designing a recursive algorithm for the problem

SAME SAME BUT DIFFERENT
GUEST HOUSE
&
RESTAURANT
$2/$3/$4 Rooms. Cable T.V. Movies. at Food.
Pizza. Laundry Service. On the

Weighted Interval Scheduling

Input: n jobs/intervals. Interval i is triple (s_i, f_i, v_i)

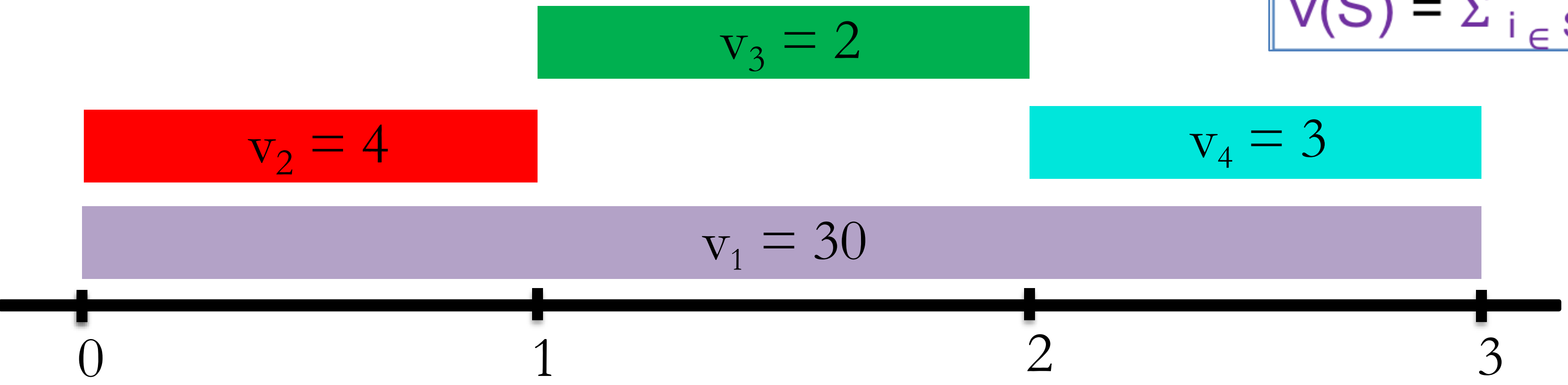
start time

finish time

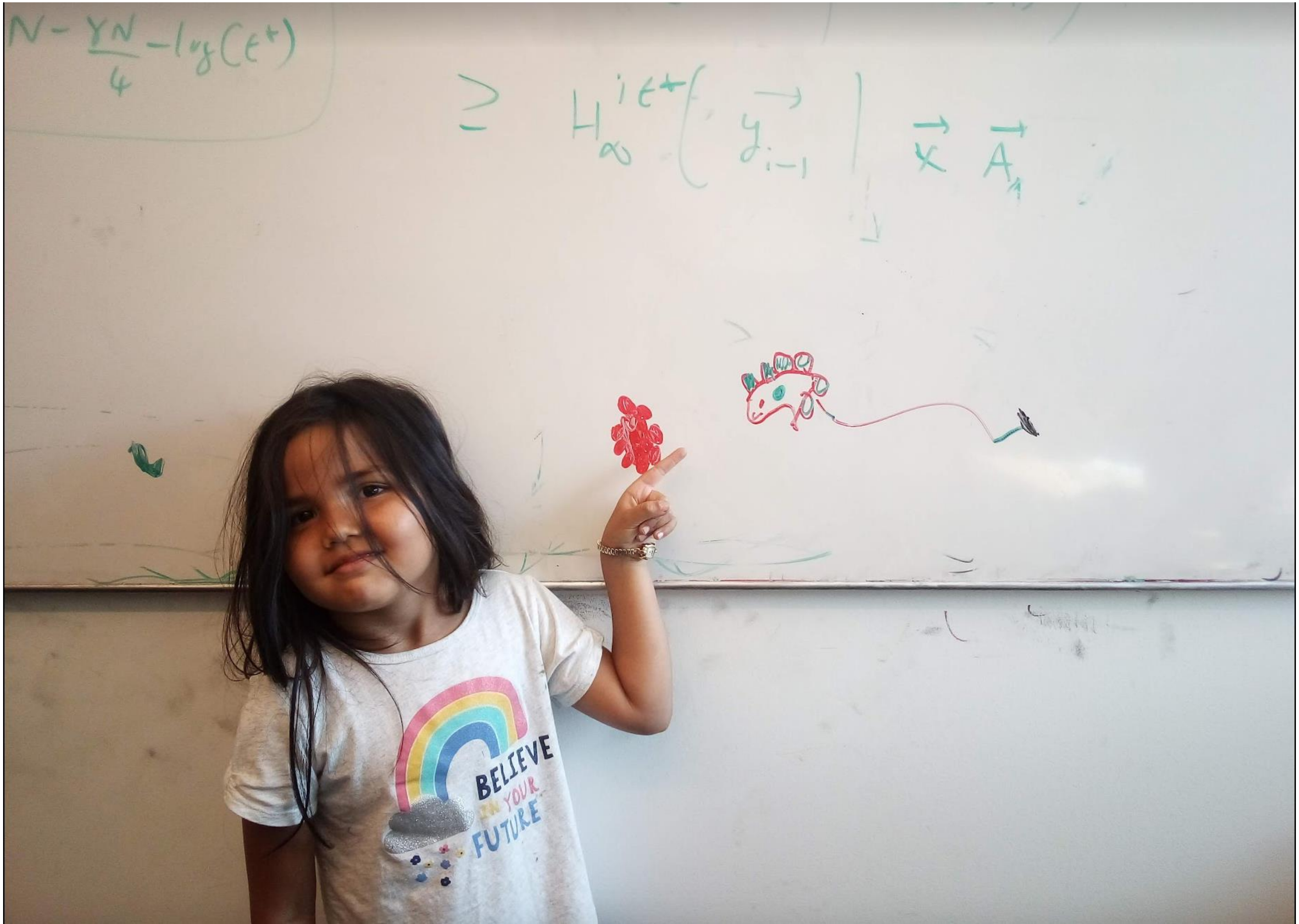
value

Output: A valid schedule $S \subseteq [n]$ that maximizes $v(S)$

$v(S) = \Sigma_{i \in S} v_i$

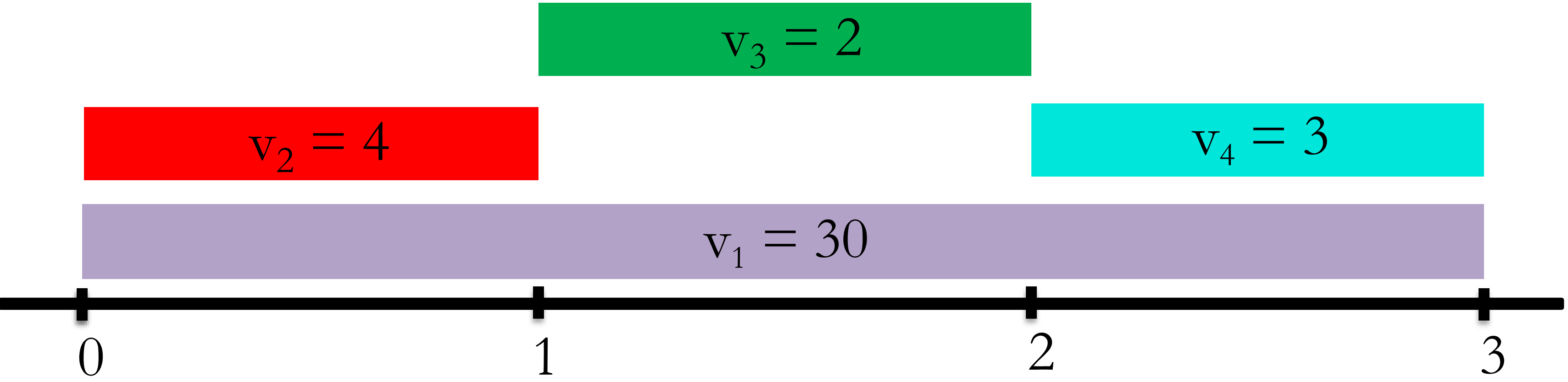


Questions/Comments?



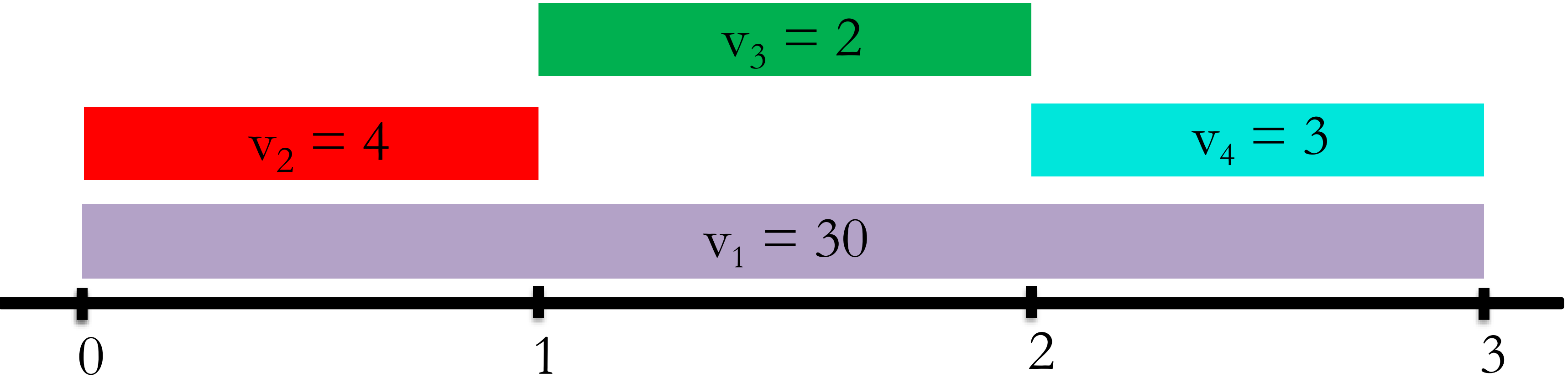
Previous Greedy Algorithm

```
R = original set of jobs
S = $\phi$

While R is not empty
    Choose i in R where $f_i$ is the smallest
    Add i to S
    Remove all requests that conflict with i from R

Return S* = S
```



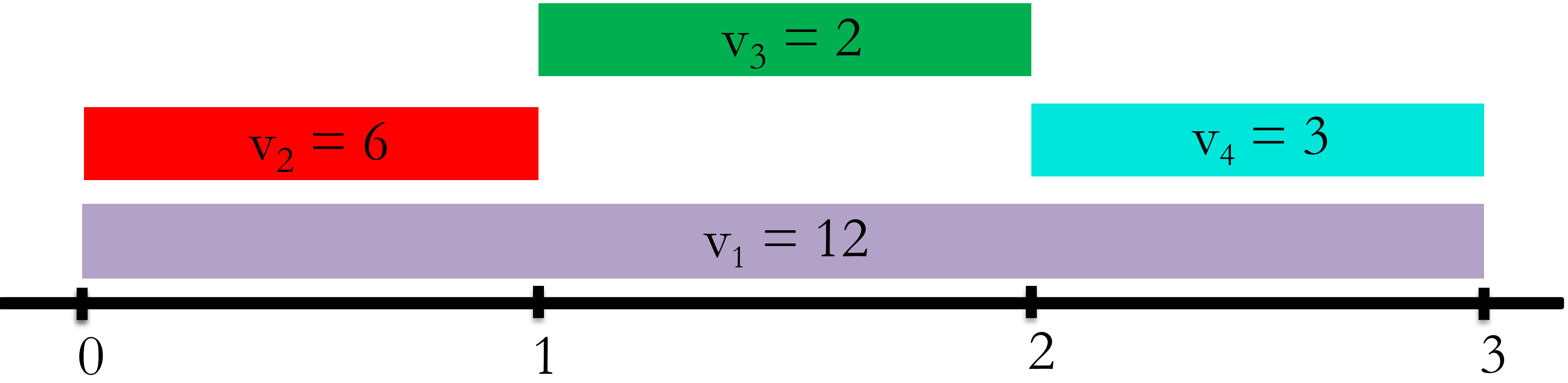
Perhaps be greedy differently?

```
R = original set of jobs
S = $\phi$

While R is not empty
    Choose i in R where $v_i/(f_i - s_i)$ is the largest
    Add i to S
    Remove all requests that conflict with i from R

Return S* = S
```



Can this work?

```
$R$ = original set of jobs
$S = \phi$

While $R$ is not empty
    Choose $i$ in $R$ where $v_i/(f_i - s_i)$ is the largest
    Add $i$ to $S$
    Remove all requests that conflict with $i$ from $R$

Return $S^* = S$
```



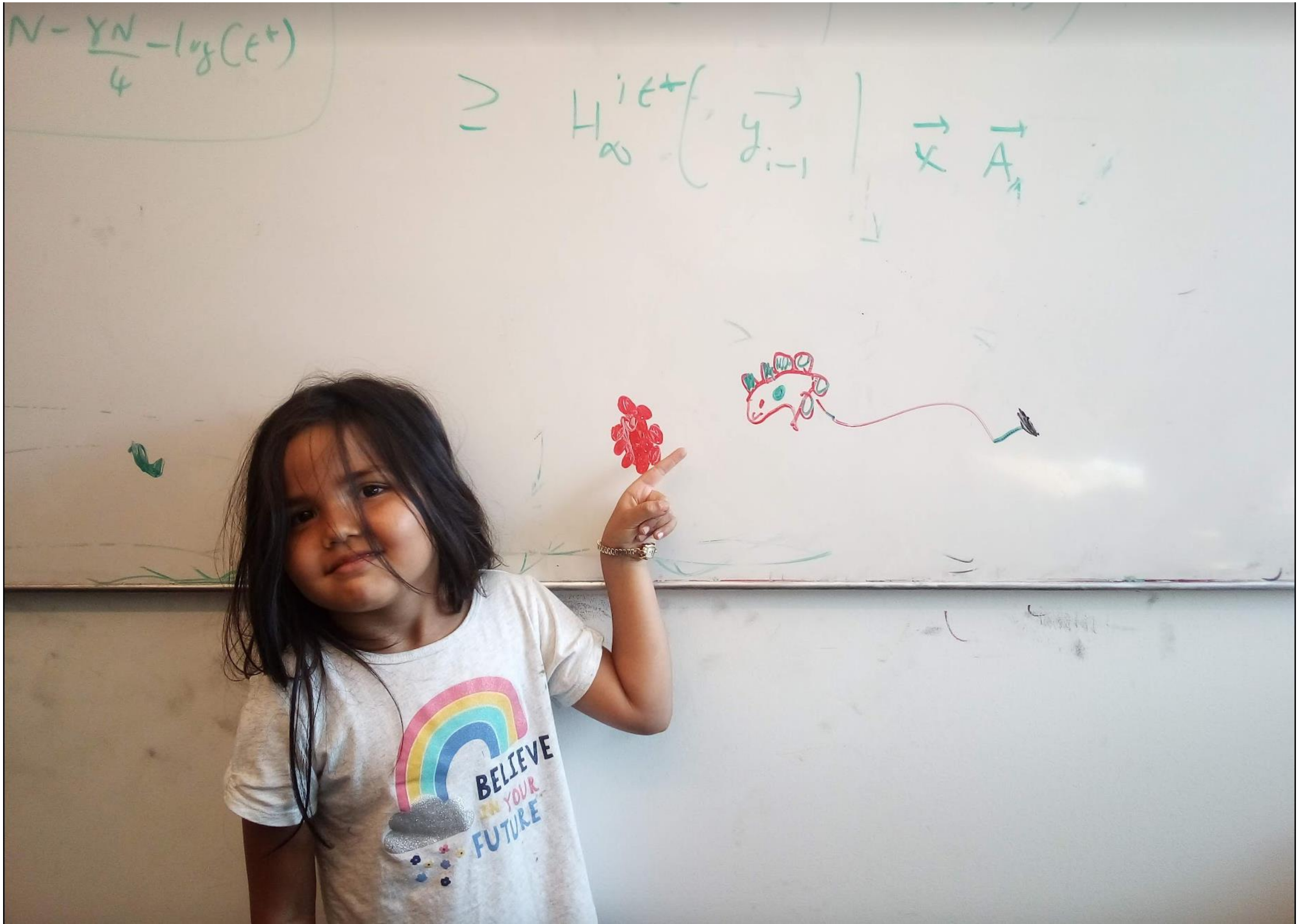
Avoiding the greedy rabbit hole

https://www.writerightwords.com/down-the-rabbit-hole/

Provably IMPOSSIBLE for a large class of greedy algos

There are no known greedy algorithm to solve this problem

Questions/Comments?



Perhaps a divide & conquer algo?

Divide the problem in 2 or more many EQUAL SIZED INDEPENDENT problems

Recursively solve the sub-problems

Patchup the SOLUTIONS to the sub-problems

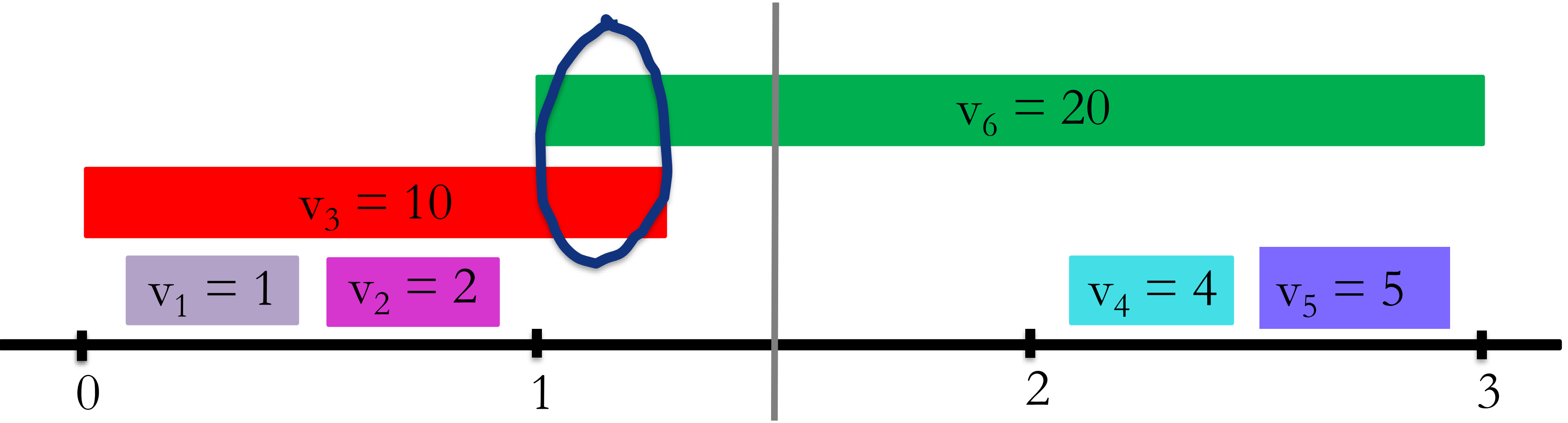
Perhaps a divide & conquer algo?

```
RecurWeightedInt([n])

    if $n = 1$ return the only interval

    $L$ = first $n/2$ intervals
    $R$ = last $n/2$ intervals

    $S_L$ = RecurWeightedInt($L$)

    $S_R$ = RecurWeightedInt($R$)

    PatchUp($S_L$, $S_R$)
```

Would this general scheme work?

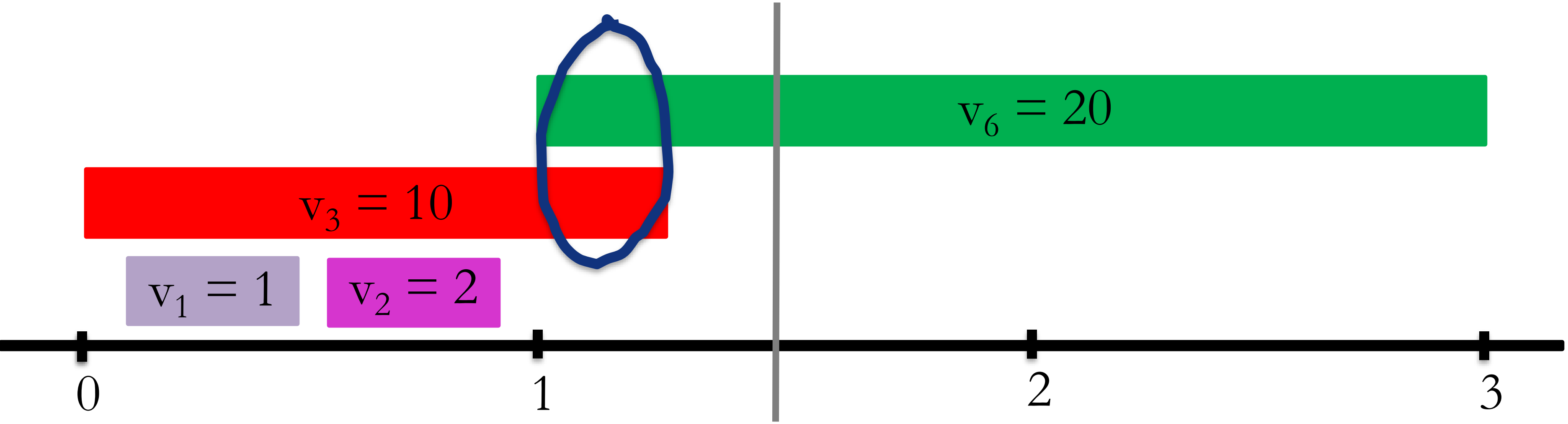
Divide the problem in 2 or more many EQUAL SIZED INDEPENDENT problems

Sub-problems NOT independent!

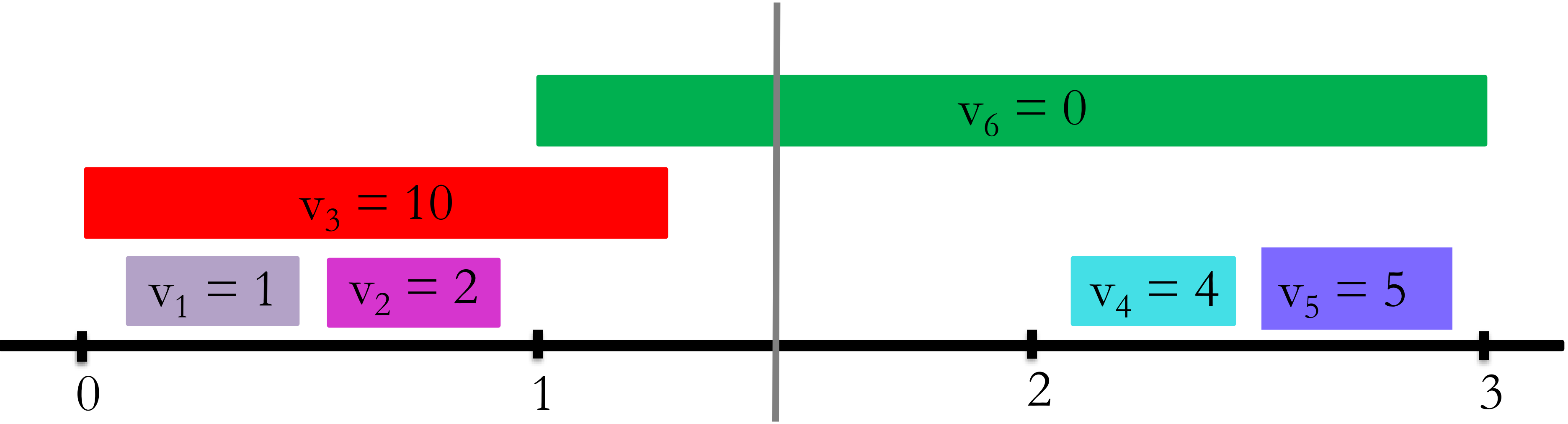


Perhaps patchup can help?

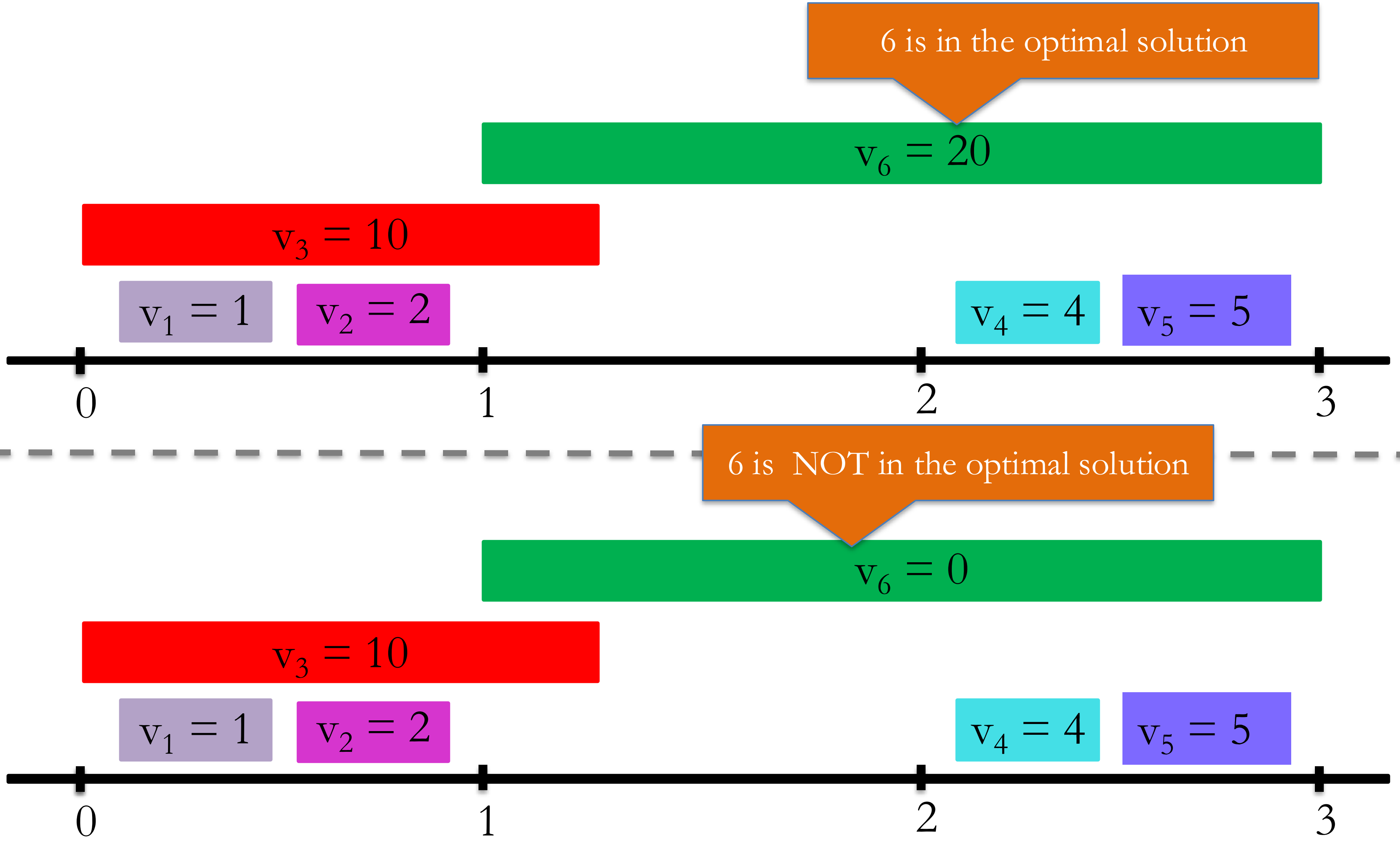
Patchup the SOLUTIONS to the sub-problems



Sometimes patchup NOT needed!



Check for two cases?



6 is in the optimal solution

$v_6 = 20$

$v_3 = 10$

$v_1 = 1$

$v_2 = 2$

$v_4 = 4$

$v_5 = 5$

6 is NOT in the optimal solution

$v_6 = 0$

$v_3 = 10$

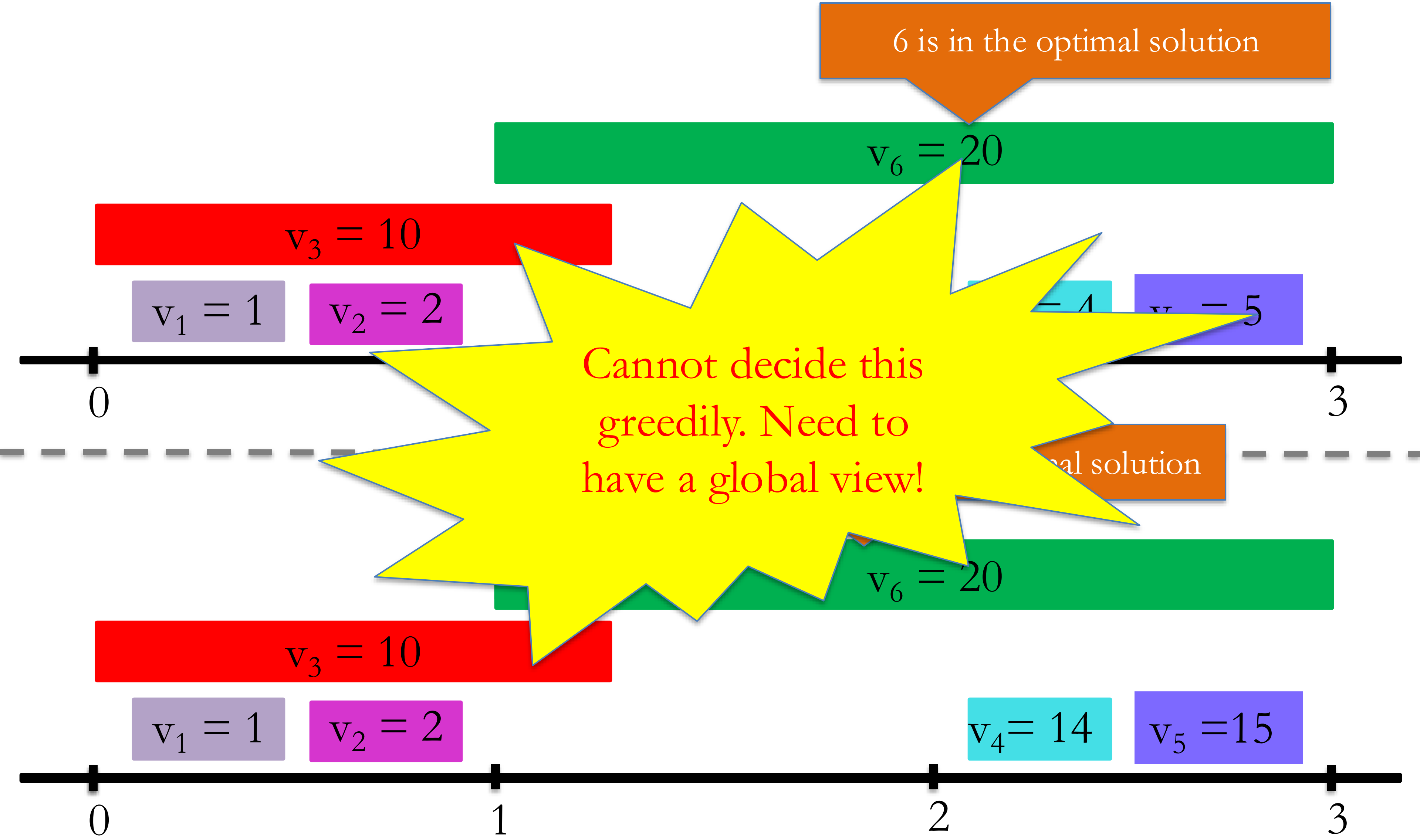
$v_1 = 1$

$v_2 = 2$

$v_4 = 4$

$v_5 = 5$

Check if v_6 is the largest value?

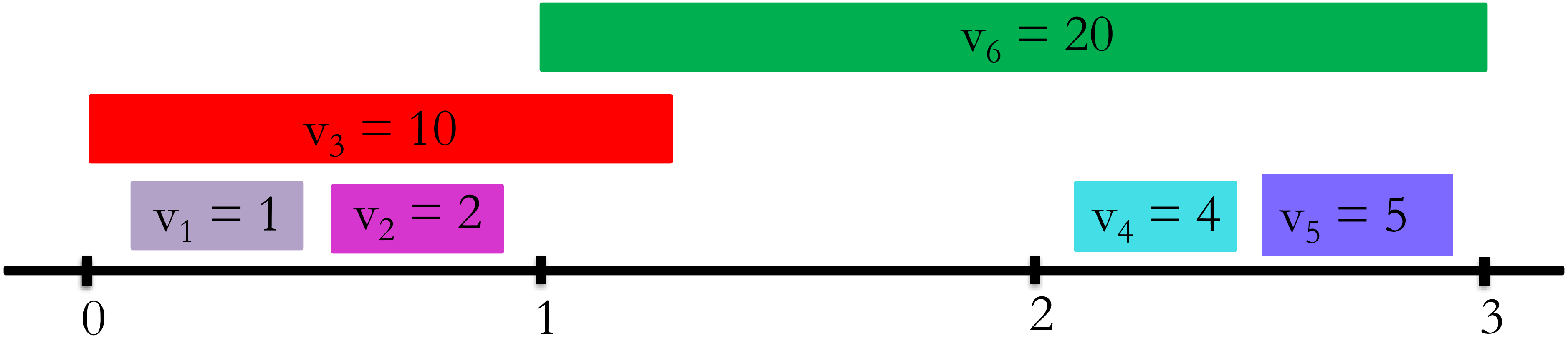


6 is in the optimal solution

Cannot decide this greedily. Need to have a global view!

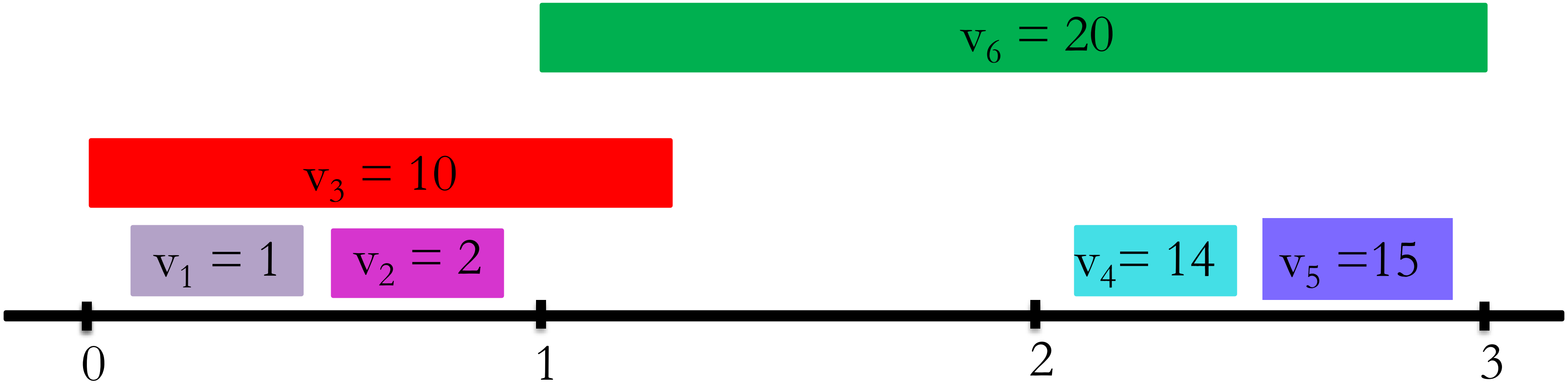
al solution

Check out both options!



Case 1: 6 is in the optimal solution

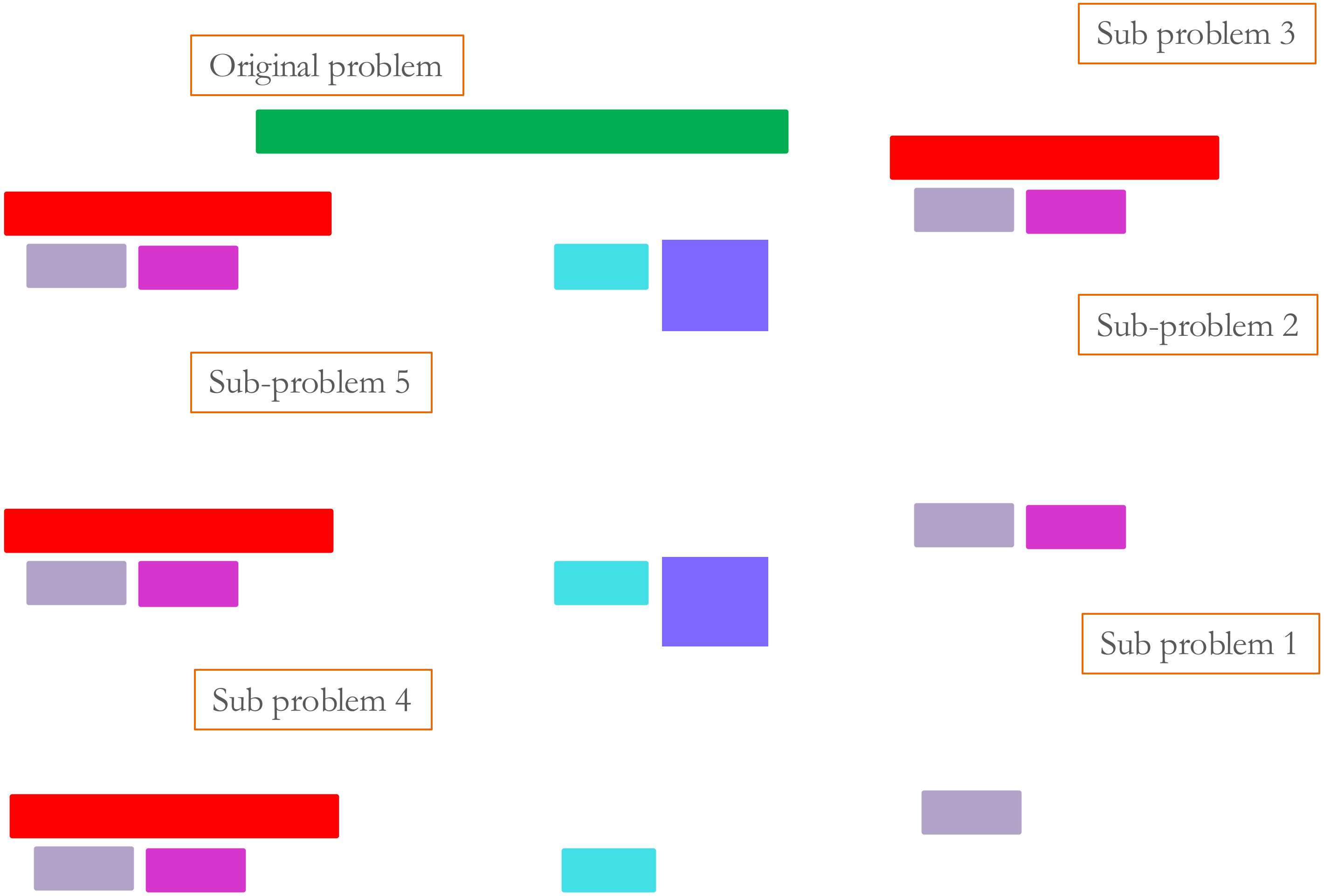
6 is not in optimal solution



?
ICANHASCHEEZBURGER.COM

So what sub-problems?

Divide the problem in 2 or more many ~~EQUAL SIZED~~ ~~INDEPENDEN~~T problems



Sub problem 3

Original problem

Sub-problem 2

Sub-problem 5

Sub problem 1

Sub problem 4

?
ICANHASCHEEZBURGER.COM