Greedy Algorithms 2

Overview for Today

- Quick Recap
- Explore advanced techniques and variations
  - Fraction Knapsack
  - Activity Selection Problem
  - Job Sequencing Problem
- Discuss Problems

Fractional Knapsack Problem

- Given the weights and profits of **N** items, in the form of **{profit, weight}** put these items in a knapsack of capacity **W** to get the maximum total profit in the knapsack. In **Fractional Knapsack**, we can break items for maximizing the total value of the knapsack.
- ***Input:*** *arr[] = {{60, 10}, {100, 20}, {120, 30}}, W = 50*
- ***Output:*** *240*
- ***Explanation:*** *By taking items of weight 10 and 20 kg and 2/3 fraction of 30 kg. Hence total price will be 60+100+(2/3)(120) = 240*

Solution to Fractional Knapsack Problem

- *The basic idea of the greedy approach is to calculate the ratio* ***profit/weight*** *for each item and sort the item on the basis of this ratio. Then take the item with the highest ratio and add them as much as we can (can be the whole element or a fraction of it).*
- *This will always give the maximum profit because, in each step it adds an element such that this is the maximum possible profit for that much weight.*

Activity Selection Problem

- You are given **n** activities with their start and finish times. Select the maximum number of activities that can be performed by a single person, assuming that a person can only work on a single activity at a time.

- ***Input:*** *start[] = {10, 12, 20}, finish[] = {20, 25, 30}*
- ***Output:*** *0 2*
- ***Explanation:*** *A person can perform at most two activities. The maximum set of activities that can be executed is {0, 2} [These are indexes in start[] and finish[]]*

Solution to Activity Selection Problem

- *The greedy choice is to always pick the next activity whose finish time is the least among the remaining activities and the start time is more than or equal to the finish time of the previously selected activity. We can sort the activities according to their finishing time so that we always consider the next activity as the minimum finishing time activity*

- Follow the given steps to solve the problem:
- Sort the activities according to their finishing time
- Select the first activity from the sorted array and print it
- Do the following for the remaining activities in the sorted array
  - If the start time of this activity is greater than or equal to the finish time of the previously selected activity then select this activity and print it

Job Sequencing Problem

- Given an array of jobs where every job has a deadline and associated profit if the job is finished before the deadline. It is also given that every job takes a single unit of time, so the minimum possible deadline for any job is 1. Maximize the total profit if only one job can be scheduled at a time.

Example Job Sequencing Problem

- ***Input:*** *Four Jobs with following deadlines and profits*

JobID	*Deadline*	*Profit*
a	*4*	*20*
b	*1*	*10*
c	*1*	*40*
d	*1*	*30*

- ***Output:*** *Following is maximum profit sequence of jobs: c, a*
- ***Input:*** *Five Jobs with following deadlines and profits*

JobID	*Deadline*	*Profit*
a	*2*	*100*
b	*1*	*19*
c	*2*	*27*
d	*1*	*25*
e	*3*	*15*

- ***Output:*** *Following is maximum profit sequence of jobs: c, a, e*

Solution to Job Sequencing Problem

- *Greedily choose the jobs with maximum profit first, by sorting the jobs in decreasing order of their profit. This would help to maximize the total profit as choosing the job with maximum profit for every time slot will eventually maximize the total profit.*
- Follow the given steps to solve the problem:
- Sort all jobs in decreasing order of profit.
- Iterate on jobs in decreasing order of profit. For each job , do the following :
  - Find a time slot i, such that slot is empty and i < deadline and i is greatest. Put the job in
    this slot and mark this slot filled.
  - If no such i exists, then ignore the job.

Problems

- https://codeforces.com/problemset/problem/1840/D
- https://codeforces.com/problemset/problem/1761/C

Thank You

20 June 2023 18:00

$\underline{W}$

50

①

	Profit	Weight
①	60	10
2	100	20
3	120	30

$\underline{240}$ $\quad 60 + 100 + \frac{2}{3} \times 120 = 240$

$\underline{W} \rightarrow 10 + 20 + \frac{2}{3} \times 30 = \underline{50}$

Item → Profit / Weight

$\underline{3}$

P: vector<pair<double, int>>

{Profit / weight, index} index

index

set<pair< ,

{ ②, ③

1.5 2

4

set

1.5	2
1.2	4
1	1

p/w

n^2

max

nlogn

p/w , index

O

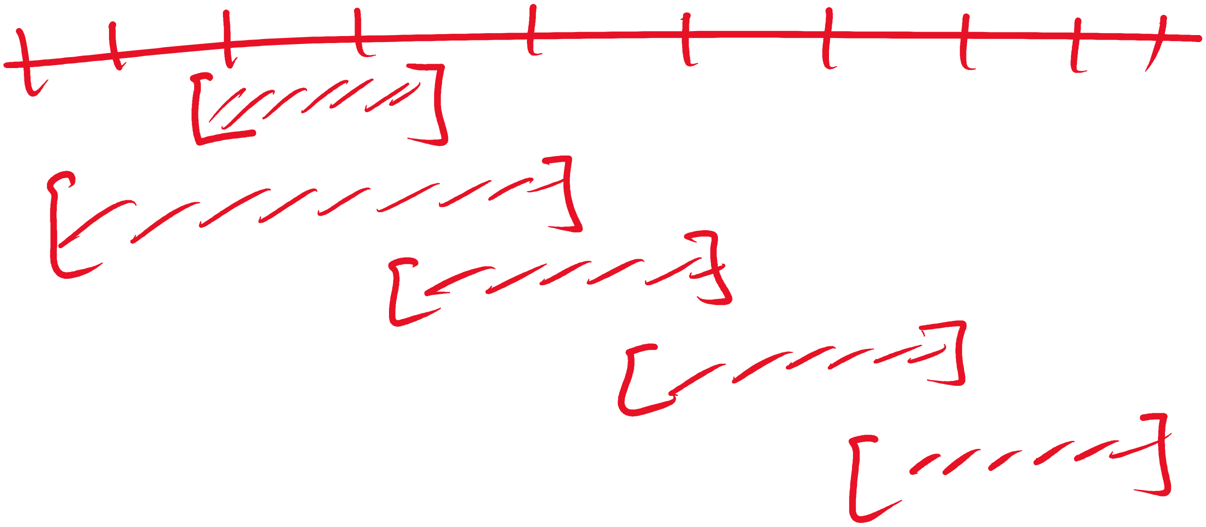
	1st	2nd	3rd
input :-	10	12	20
	20	25	30

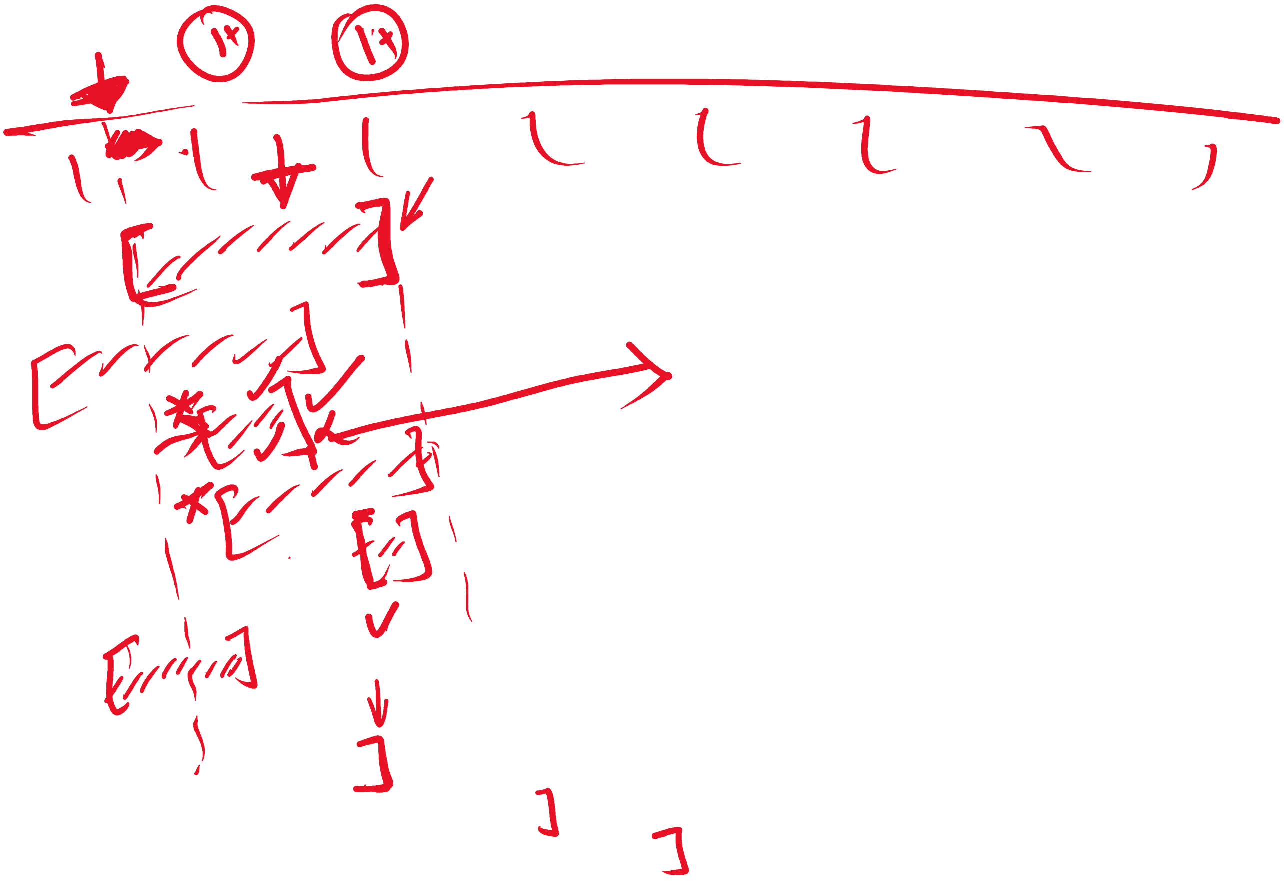
[ ]

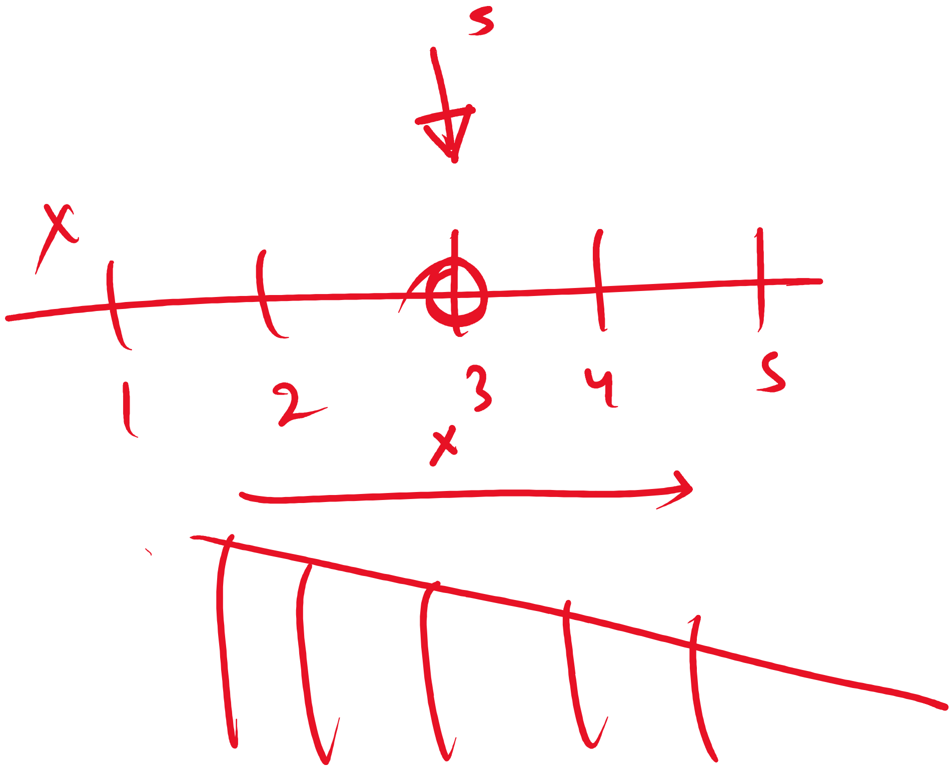
mai

ranges

maximize the no. of ranges

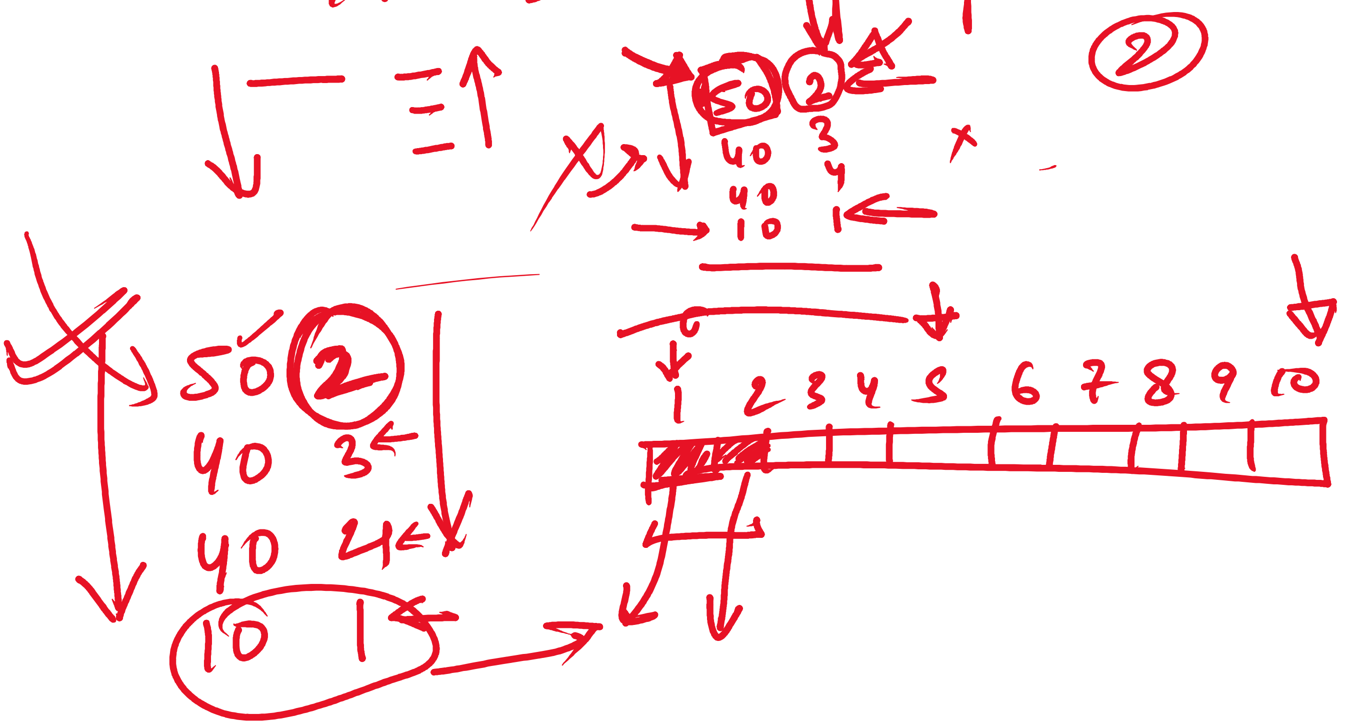




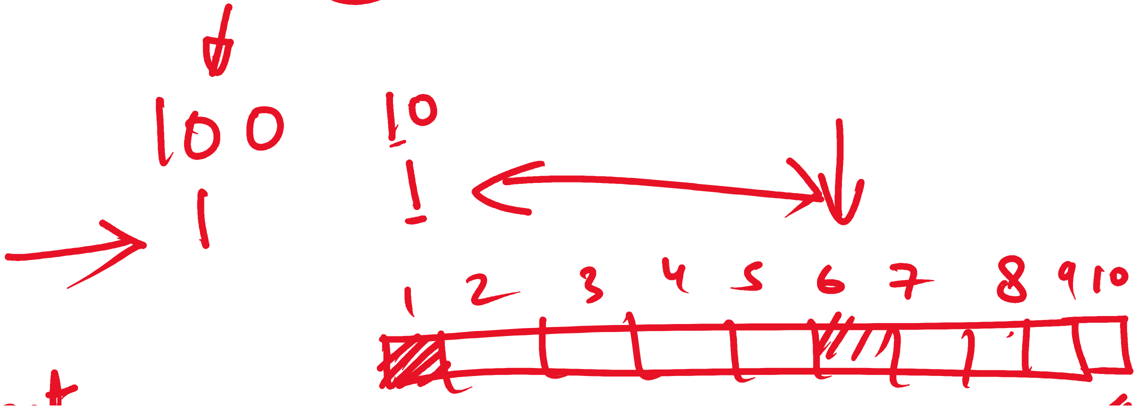

s
x
1
2
3
4
5
x

Job → deadline, profit

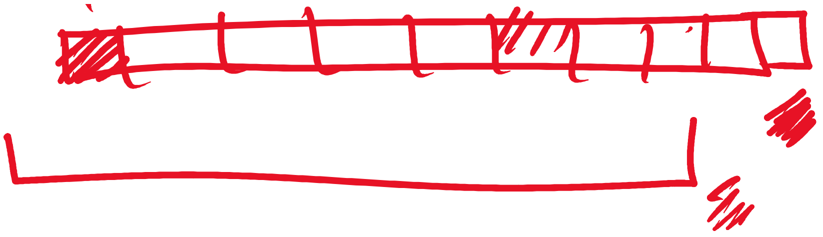
maximize → profit



n n 10^{18}



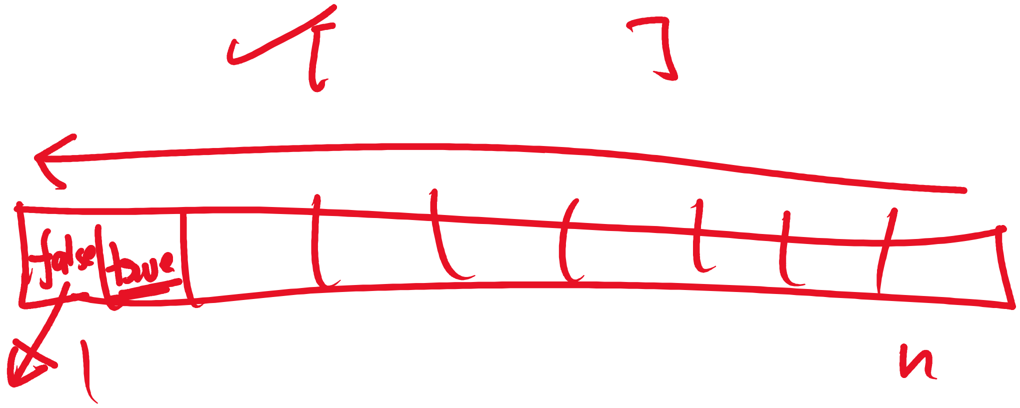
set



n ←

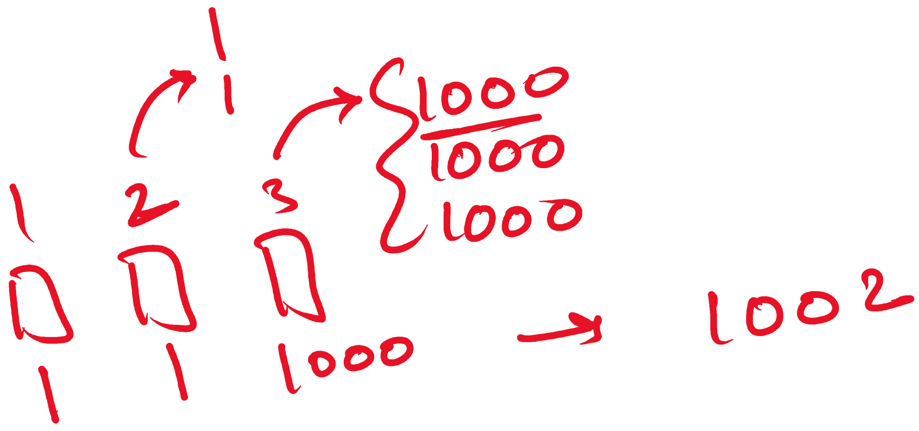
$n+1$

n 10^6



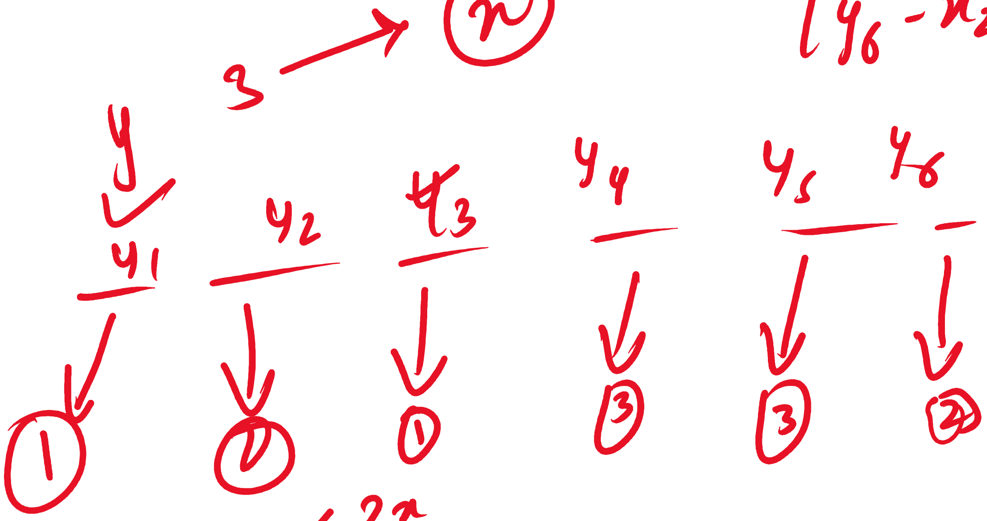
10^{18}

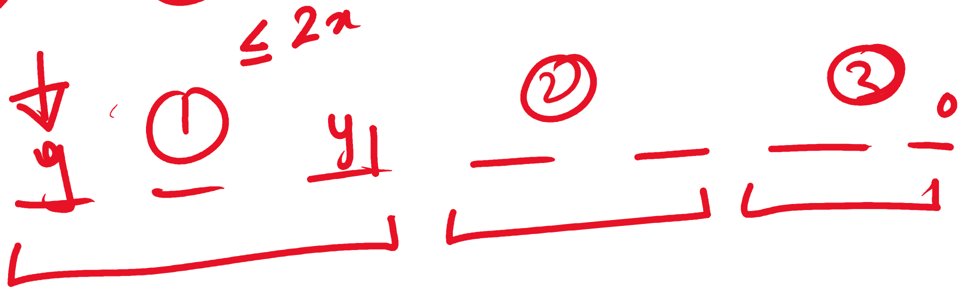
$2n$

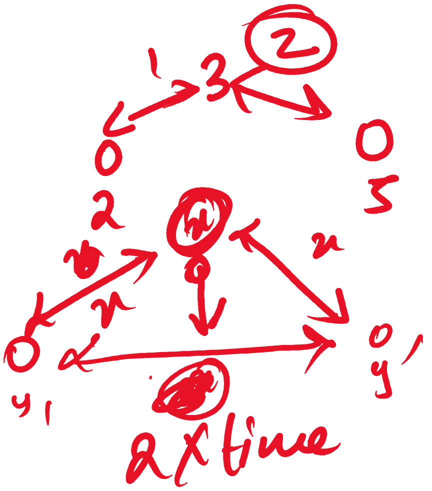


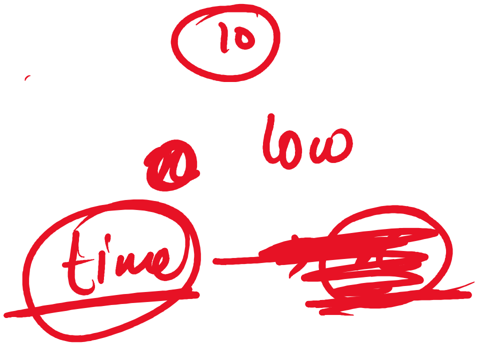
$$\max(|y_1 - x_1|, |y_3 - x_1|)$$

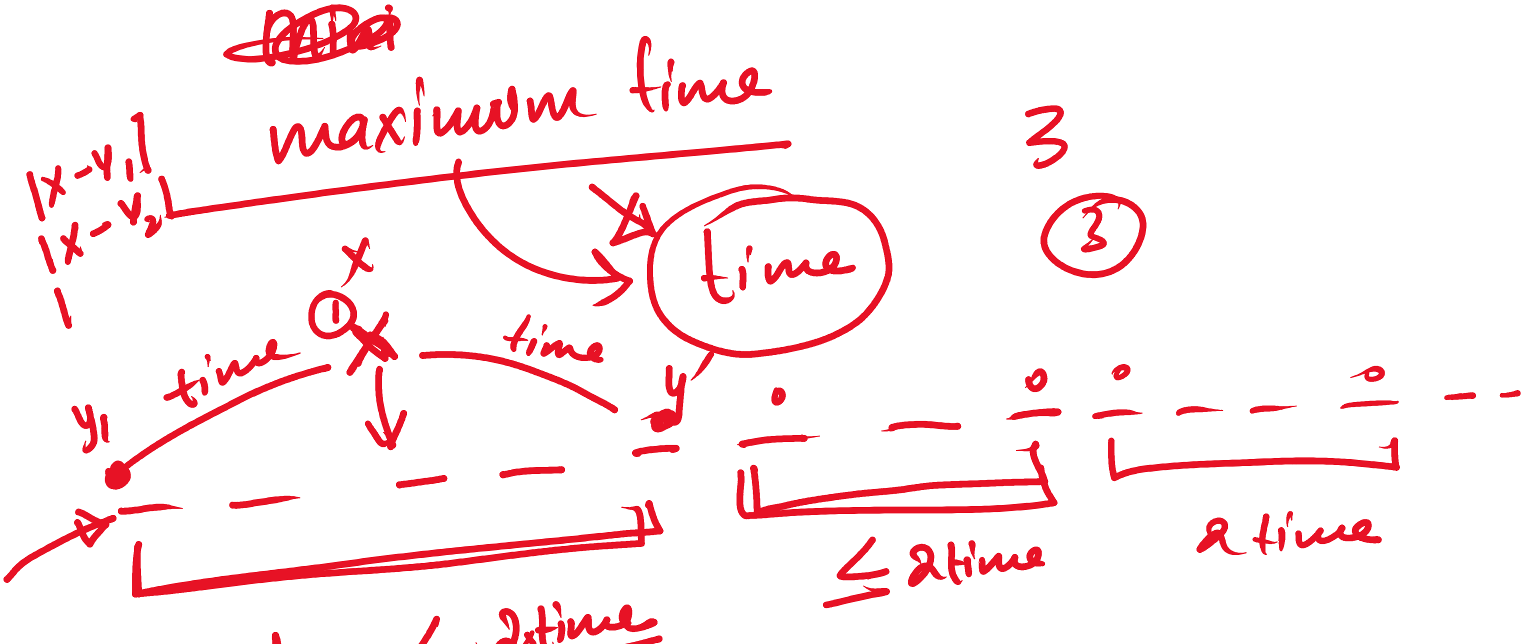
$$\max(|y_2 - x_2|, |y_6 - x_2|)$$











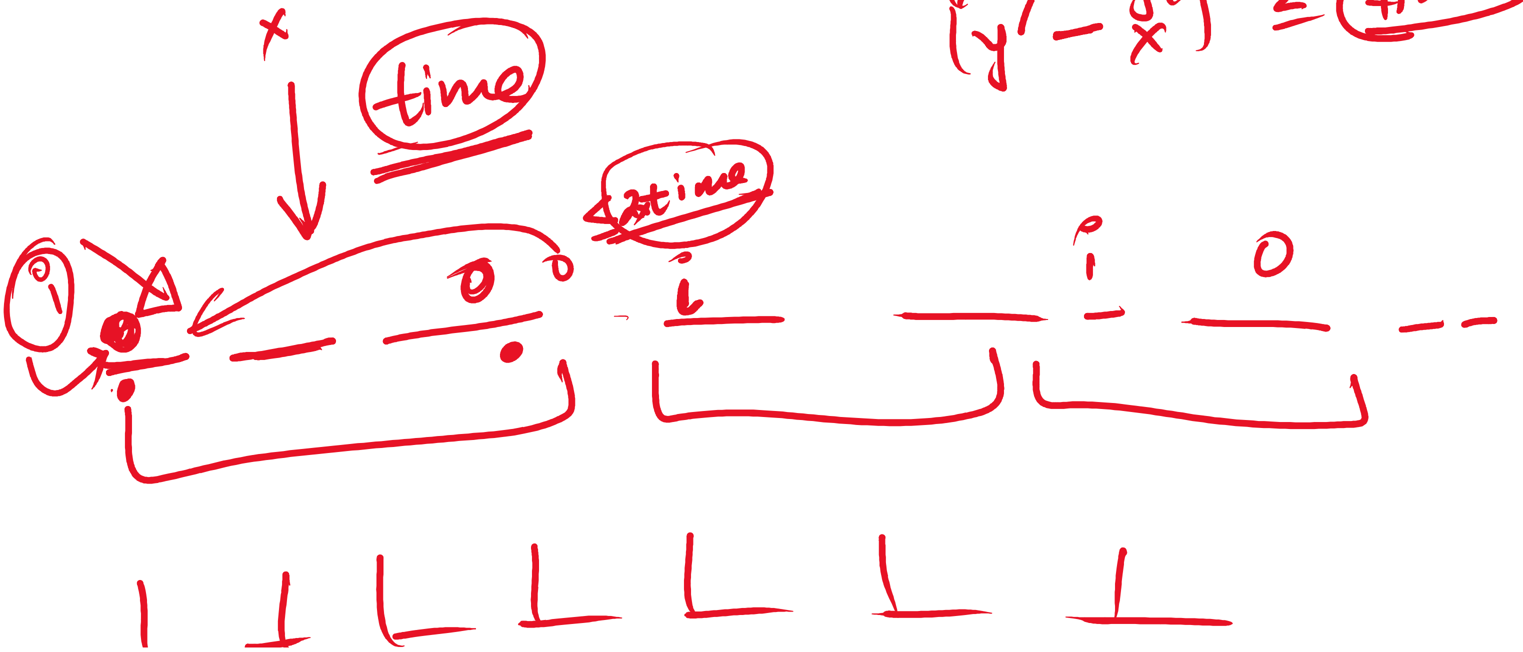
$\boxed{X}$ first ≤ 2*time

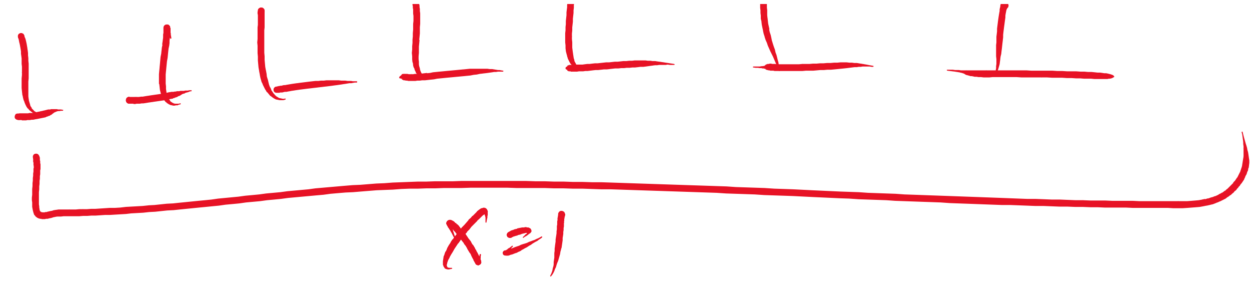
time

$$|y' \underset{\uparrow x}{-} y_1| \le 2time$$

$$|x - y_1| \le \boxed{time}$$

$$|y' - x| \le \boxed{time}$$

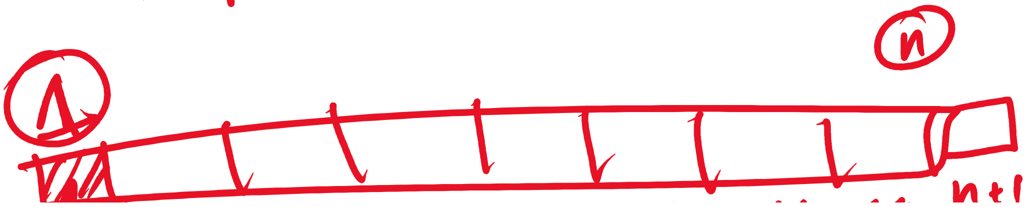


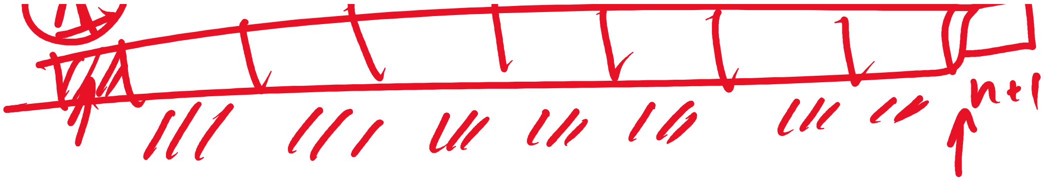


6

time = 2

n → 1 unit




n+1