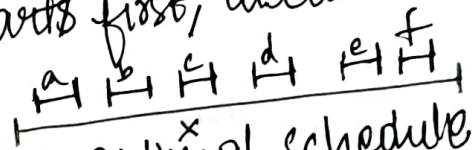
ASSIGNMENT - 4

NAME:- NIHAR MUNIRAJU
ID:- 2072857
email:- nmuniraj@depaul.edu.

1 a) Choose the Course x that ends last, discard classes that conflict with x, and recurse.

Ans The Optimal Solution having the Optimal schedule might also have ends which are shorter than always being longer. And also considering the greedy algorithm it could only choose a course with a long ends. So Henceforth discarding the classes that conflict and recurse might won't work for all greedy algorithm as an Optimum Solution.

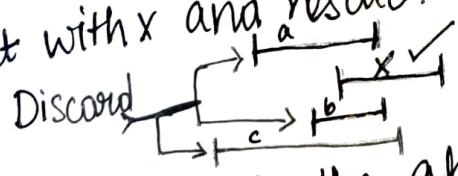
b) Choose the Course x that starts first, discard all classes that conflict with x and recurse:



Ans The Optimal Solution having the optimal schedule might also starts late but ends fasts compared to starting first. The Greedy Algorithm choosing a course where starting first and removing all the conflicts with it recurse might not be work.

c) Choose the course x that starts last, discard all classes that conflict with x and rescue:

Sine[x]<[a, b, c]

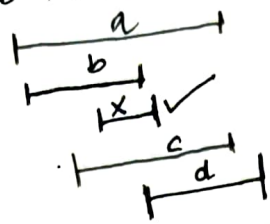


Ans

For every case in the above example, the greedy algorithm ends fasters with even being starting last. The Optimal schedule that includes the course of work to start in the last and also end as first which proves the Correctness of the greedy Algorithm.

d) Choose the course x with shortest duration, discard all classes that conflicts with x and recurse:

Ans



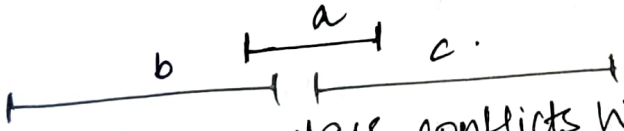
⇒ By this input the greedy Algorithm might only think having the shortest duration might help to finish the Course x. And having an optimal schedule which can contain other longer ends also or shorter ends to make sure it works. But this way it Doesnt Work.

e) If no classes conflict, choose them all. Otherwise, discard the course with longest duration and recurse.

Ans This way doesnt Work, the greedy Algorithm always thinks of choosing each different single duration of interval in the middle, and also the optimal schedule might contain the other different single duration of intervals in the longest duration.

f) If no classes conflict, choose them all. Otherwise, discard the course that conflicts with the most other courses and recurse.

Ans



This way having no class conflicts wont Work, the optimal schedule might contain a classes with different duration of intervals but whereas the greedy Algorithm might not contain many intervals but will have a interval to choose.

1) If any course x completely contains another course discard x and recurse. Otherwise the coursey that ends last, discard all classes that conflict with y, and recurse.

Ans If there is an optimal schedule which has the course that starts and ends last then the greedy algorithm might not contain any course x that ends last is also that starts last. But if an optimal schedule does not includes and discard x which is from the conflict of the class y then it contains a valid schedule of the same size.

2)

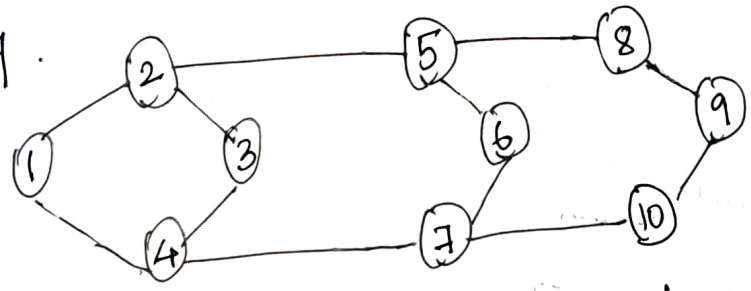
⇒ The Algorithm initializes the set I to the empty set.

$I = \{\ \}$.

where G is a set of vertices I that no two vertices in I are adjacent. and N be a node with minimum in G where.

$I = \cup(I, \{N\})$

⇒ For the given above Graph G the Pinocchio's Greedy Algorithm might not be correct to find the exact maximum independent sets of G.



The maximum Independent set that can be formed is {1,3,5,7,8,10} so it has a size of 6.

Now, Lets use the Pinnochio's Greedy Algorithm approach. we have the following nodes 1,3,5,7,8,10 with degree 2 and nodes

nodes 2 and 4 with degree 3.

⇒ So, we start with removing node 6 and node 9.

Now there is a 4 cycle left for S = {6} which are two ways.

⇒ From here there are two possibilities

⇒ If we select node 1, we will left 3.

Hence, S = {1, 3, 6}.

⇒ If we select node 2, we be left with node 4.

Hence, S = {2, 4, 6}.

⇒ So, Henceforth the way of size of the set will be 3 (which cannot be the maximum).

Hence, Greedy Algorithm does not always give the maximum Independent set.

3

⇒ Proffessor Gekko has always dreamed of inline skating across North Dakota. And he carries of almost 2 liters of water, and he can also skate 'm' miles before running out of water.

⇒ Starting with the bottles probably then go to the nester 'm' most refelling location he can get to within m millies of where he filled up.

⇒ Looked at another way, at each refilling location, Proffessor Ghekko check whether he can make it to the next refilling location without stopping anywhere.

⇒ If he skip this one then he doesn't need to know how much water he has or how far the next refilling location to implement this way since each fill up can determine which location he can stop to fill it up.

⇒ Suppose n possible reffilling locations the Optimal Solution of the first stop is at the kth location. Then the rest of the Subproblem must be a remaining n-k stations. Otherwise the best Solution is to the subproblem one of the fewer s-1 stops, we can use to come up with fewer than s stops for our contradicting optimality.

⇒ The Greedy choice is k refilling locations beyond the start before m miles. The Greedy Soln chosses kth location as first stop.

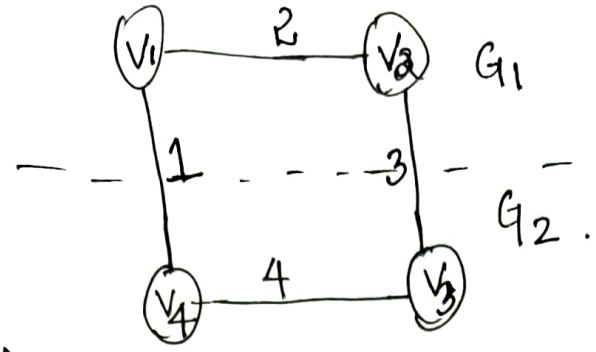
⇒ Proffessor, would run out of water if j<k he could choose kth location instead, having least amount of water when he leaves to the kth location as if he will come to the jth location. And then he would refill the water location choosen in the kth location.

---

4) ⇒ For Computing divide and conqueror in minimum spanning trees where the Graph G=(V,E) as a position of various set of two vertices V1 & V2 such that |V1| & |V2| may differ almost by 1.

⇒ Let E1 be the set of edges that are incident only on vertices in V1, and let E2 be the edges that are incident only on vertices in V2.

⇒ Two Subgraphs G1 = (V1, E1) and G2 (V2, E2). Selecting the minimum weight edge in E that crosses the cut (V1, V2) to the edge to unite the resulting two minimum spanning trees into a single spanning tree.



⇒ where $G_1 = \{V_1, V_2\}$ with V_1

and $G_2 = \{V_3, V_4\}$ with V_2.

⇒ It has a height of the minimum spanning tree of G1. has 2 and minimum spanning tree of G2 is 4 and the minimum weight edge crossing the cut is 1.

∴ V2-V1-V4-V3 which has a weight much greater than. V4-V1-V2-V3 with weight.

⇒ Therefore we can see that the minimum spanning tree of G, algorithm fails to obtain it.