Chapter 8

In this chapter we will be introduced to new set of algorithms that help you with a wide set of problems and help you reach an approximate solution to some problems that are impossible to solve like the traveling salesman problem.

The classroom scheduling problem

Let's say this is the schedule of your classroom

Art	9 AM	10 AM
Eng	9:30 AM	10:30 AM
Math	10 AM	11 AM
CS	10:30 AM	11:30 AM
music	11 AM	12 AM

you of course have noticed that that these classes overlap so our aim is to hold as many classes as possible sounds hard at first but the algorithm for solving it is pretty easy here is how it works
1-pick the class that ends the soonest this is the first class to hold
2-now you have to pick the class that starts after it and so on
you keep doing this and eventually you'll end up with the most number of classes possible the classes are Art, Math, Music
this kind of algorithms is called greedy algorithms you are working towards the optimal solution to your local case and eventually you end up with the global optimal solution obviously greedy algorithms don't always work but they are easy to write.

The knapsack problem

You are a thief in a store with a knapsack your knapsack can only 35 kg you try to maximize the value of the objects you steal let's try the greedy strategy
1-pick the most valuable item that fits into the knapsack.
2-pick the next most valuable item that fits into the knapsack.
sounds easy right? but this time it won't work let's try an example let's say these are the items
TV: 3000 $ 30 kg laptop: 2000$ 20 kg guitar: 1500$ 15 kg
if you follow the greedy algorithm you will take the TV but that will leave you with no space left and only 3000$ while if you picked the laptop and the guitar you'd have ended up with 3500$ the greedy algorithm here didn't give the optimal solution but we got really close we will know how to solve this problem in the next chapter

The set-covering problem

You are starting a radio show you want it to broadcast over 50 states you have to decide which stations to play on to reach all the 50 states it costs money to play on each station so you have to minimize number of stations each station covers a region and these regions overlap clearly

the greedy algorithm won't work here so to git the best solution you will try all the possible subsets there 2^n possible subset and then you pick the set with the smallest number of stations and here you are the problem with this solution is that it takes too long it's O(2^n) it's possible to do if the set is small but with each small increment to the number of stations the number of subsets gets drastically higher there's no way to solve it fast enough

Approximation algorithms

Here is where greedy algorithms are helpful while it might not give you the correct answer it gits really close so this is a small sacrifice to make when the algorithms to get the correct answer isn't efficient.

Let's try again with the set-covering problem
1-pick the station that covers the most states that haven't been covered yet it's ok if some of this states have been covered but we want the most number of new states
2- repeat until all the states are covered
this is called an approximation algorithm while the exact solution takes too long to be calculated approximation will work you can judge an approximation algorithm by
1-how fast they work
2-how close they are to the optimal solution

Greedy algorithms were good option here because not only they were easy to write but being simple means, they are usually fast in this case greedy algorithm takes O(n^2)

NP-complete problems

To solve the set-covering problem you had to cover all the possible sets doesn't that ring a bell? this was the same thing we had to do back in chapter 1 with the travelling salesman problem to solve the travelling salesman problem you had to cover all the possible combinations of routes to solve the problem which would take O(n!) and like the set-covering problems the number of routes becomes bigger drastically as the number of cities increase and in both problems you had to calculate all the possible solutions then chose the best one both of these problems are called NP-complete this problems are considered very hard to solve and it's widely thought that there is no possible way to write an algorithm that solve it quickly.

How do you tell if a problem is NP-complete

The difference between Np-complete and an easy problem is very small and NP-complete problems show up everywhere and there is no straight way to know but here are some tips that might be helpful
1-Your algorithm runs quickly with a handful of items but really slows down with more items.
2- "All combinations of X" usually point to an NP-complete problem.
3-Do you have to calculate "every possible version" of X because you can't break it down into smaller sub-problems? Might be NP-complete.

4-If your problem involves a sequence (such as a sequence of cities, like traveling salesperson), and it's hard to solve, it might be NP-complete
5-If your problem involves a set (like a set of radio stations) and it's hard to solve, it might be NP-complete
6-Can you restate your problem as the set-covering problem or the traveling-salesperson problem? Then your problem is definitely NP-complete.

And that's all for this chapter