Day 1 Progress

01 October 2020 20:17

As usual our SHERLOCK is ready to face any challenge, but this time MORIARTY made sure that the challenge is not that easy even for Sherlock. Today's time was spent on thinking about the puzzle or a task which our hero might be facing in the comic. This puzzle may be related to most of the puzzles where the hero will have to save people at different places in a very limited time. (Similarities can be found with the case where there will be bombs placed at different parts of the country and our force should be using their manpower and their technology to save the maximum people if not all). After doing some research in some of the algorithms and problems regarding these kind of problems. Even though I was not able to retrieve the complete solution which is correct as well as efficient. But I discovered that there are great similarities between our problem and the generalised case in algorithms.

One of the major similarity is that we can represent people position and bombs position as coordinates in a grid. And we will relate the intensity of the bomb using the vertices in and around the bomb, as an initial idea what I thought was to assume that if the bomb is placed at (c,d) then area formed by x=c and y=d with coordinates will be in great danger. So we need to communicate to the people as to how they should move so that they will get out of this danger zone. Again as a first idea we thought I thought that Sherlock may hack (which itself is a story of its own for some other day) all the telephone services so that he will be using internet to communicate to all the people to move in right or upwards in the grid direction so that they will be in safe zone. As usual our hero will try to save them all using his intellectual power. But let's wait for comic to unfold its story for more details :)

I tried few greedy algorithms which seem to fail, some of them include the nearest edge of each point and maximum among for all such distances, this is very inefficient way of moving the people out and in worst case we may not even be able to save them. We may use these algorithm as an idea which will be suggested by ordinary military men like WATSON but as usual our hero will prove the inefficiency of the algorithm that the side roles suggest him. As of now I was not convinced by the current best algorithm but will explore for some more time on this problem.

Side ideas that are evolved in thinking about this task include the following:

- An episode can be made using the hacking part in the discussion mentioned above which is always nice to have when designing something interesting.

TIME BREAKDOWN:

Around 40 min were spent on getting the basic idea of the issue that our hero might face. And then the part of seeing its resemblance with algorithms was done in 15 min. Around 15 min on thinking how to show the destruction effect of bomb in a grid. Major time was spent on seeing why various algorithms are bad this part I did read few algorithms in internet and some time in thinking myself and failed to find very good solution this is done for 1hr. Inspiration is drawn from a question that my sister asked regarding a mathematics question although the question is not exactly the same it drew enough attention of time that this idea worth the time spent on it. Around 8 min was spent in explaining it to a group member and around 15 min in writing today's report. This report is written in an elaborate way keeping in view of the team members and mentors to criticize the idea so that it can be improved based on the feedback received.

2 Hr 40 min

Past Two Days Report This says the briefing of the Episode-2

28 October 2020 14:15

I have thought about this weeks algorithm proposed by Sri Ram, there are two algorithms proposed by the algorithm team this week.

First algorithm is based on a graph. We need to find a path such that all the nodes are visited and no node is visited more than once. The idea I proposed is that a scenario where our hero will be injected some poison whose medicine is not available in any medical store. The villian himself gives a hint about how he should find the medicine. He continues saying that "Let me guide you", he says that there are n places in which doses of a medicine are available to this poission. Our hero will think why this villian is helping him for no reason and thinks about the evil plot behind this help. Hero will be thinking about how come there is a medicine which he doesn't know. Villian reveals that the medicine is not a normal medicine. He says that the medicine is illegal and costs millions of dollars, there are many criminals who are smuggling these goods, so they will not take any second thought to kill the people who takes the medicine. Secondly the police will be after these criminals who are carrying illegal medicine.

The way I thought this will be close to the algorithm is that the hero need to get all the doses to be alive so the nodes are places in which the medicine is available. The hero cannot visit the nodes twice because the polices will arrest him and the criminals will kill him if he visits the place again. SO this is same as the algorithm as to visiting all nodes without visiting any node twice.

Second algorithm is based on finding minimum number of nodes using which we need to go to all the nodes such that there exist exactly one edge through which we can go to any node from the node in set. Let u be any node then we say that v can be reached when there exist some u such that u-v edge exist. The idea I though is that we can assume that the nodes to be people and there are some particles which are essential to go someplace but there are not many of the particles. So the hero plans to allote the particles such that the persons will be able to carry the people surrounding him to that all the people will be able to go to the required place.

Segment Tree Algorithm Research:

> As part of my research I listened to the tutorial by SecondThread. I plan to listen to two more tutorials by the same author. I did made some effort to find an algorithm on segment tree but as of now that stands to be incomplete. I will finish listening to rest of the tutorials by tomorrow. And hopefully can come up with an idea.

5Hrs

Past Two Days --Intro to CO**D??(Episode-3)

07 November 2020 14:28

I thought about the idea for EPISODE-3 , this is one of the idea which made me excited. But that is early call, but the basic idea is as follows:

FOR BETTER EXPERIENCE DONOT READ THE FOLLOWING CONTENT BEFORE THE EPISODE GETS READY AS PLOTS SPOILERS WILL BE REVEALED IN THE UPCOMING TEXT.

The villian is fed up with trying to make fun out hero with some really easy (atleast for him) , now he is after something really big. Trust me when I say BIG, it is BIG. He in the Episode one tried to bombard a city which kinda failed he would say, then he tried to test our hero intellectual power. But this time he is after something big.(Guess what??). Firstly his plan has started 14days back. Buts its consequence of which are seen after a fortnight. What is it all about? . This time he used animals. What ? How on earth can he use animals. But he did? As of now I don't what to reveal how he used animals. But as of now I am thinking it as the spread of a virus which is lethal and is spread through animals. In a span of weeks it spread through out the nation. The country is decrared to be in EMERGENCY. As usual Jekyll was informed about the same. He will try to think about the various ideas as to what he can do. But one fairly good idea would be give antidotes to them. But how that too at this pace. Maybe drones, that would be too late to manufacture them. How will I get the medine firstly.

PLot I thought and would make it s thriller is that I assumed that the villian has the antidote with him. Rather he has hudge amount of it. The major problem is that how to get into it. I didn't think much about what should happen here. But as always I have made sure that a thirllar can be setup here. May be something like some levels of defence security which should be shackled before entering the final level of the process. But is story for another day.

But the final solution for the proposed problem as of now will be through the flowing water. What ?, Yeah you are read it right it is water, drinking water. How? , When? , Why? Go through this EPISODE to make moset of it.

Algorithms planning as of now are shortest distance, max flow. Algo Team will be ready latest by tomorrow as to what the algorithm will be.

5Hr

Idea that didn’t Launch ☹

09 November 2020 09:35

In episode two what seems to be a cakewalk now I am challenged to find a real life situation. I thought about the same while thinking about the core idea of the algorithm-2 which I proposed. My idea didn’t gain enough strength to take of. But nevertheless I am here to discuss it here as part of the effort made. I got the idea pretty quickly but explaining different team mates at different depths and instants of time took around 1hr of work.

That idea was that instead of worrying too much about proof I thought I will illustrate using the graphs and images as part of the reactions brain. Like thunders in brain and so on. That idea was to show something like the code which we get sometimes when we boot our PC. But this idea didn’t gain ground in the group. I wish this idea was continued to next level but for now they requested me to search for one more idea for which I need to explore the algorithms in detail. But since the majority didn’t like that idea I am trying to find a real life situation.

There are few more ideas that were rejected by our team. The goal for next few hours would be to find the proof of it and connect it to real life.

1Hr 30 min

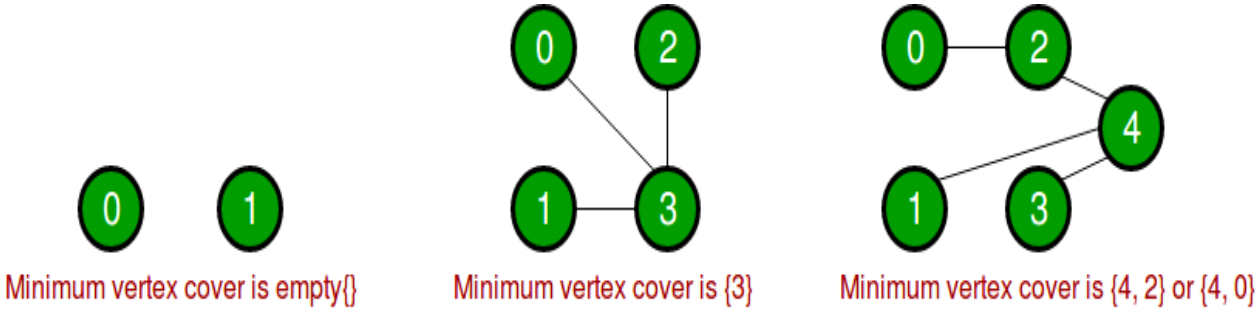
Exploring Proof Of Approximate Vertex Cover.

09 November 2020 09:46

I am currently reading geeksforgeeks material and will post relavent proofs and statements from that website which seem to be interesting.

Link https://www.geeksforgeeks.org/vertex-cover-problem-set-1-introduction-approximate-algorithm-2/

A vertex cover of an undirected graph is a subset of its vertices such that for every edge (u, v) of the graph, either 'u' or 'v' is in the vertex cover. Although the name is Vertex Cover, the set covers all edges of the given graph. ***Given an undirected graph, the vertex cover problem is to find minimum size vertex cover***.



While Exploring I realized that this problem is NP complete. And fortunately a resource claims that there is some approximate algorithm for this. Looking for that proof. The algorithm is very simple, but I didn't understand why it is working.

Approximate Algorithm:
1) Initialize the result as {}

2) Consider a set of all edges in given graph. Let the set be E.

3) Do following while E is not empty
...a) Pick an arbitrary edge (u, v) from set E and add 'u' and 'v' to result
...b) Remove all edges from E which are either incident on u or v.

4) Return result

The best thing about this approximate algorithm is its runtime is $O(V+E)$.
This is fairly simple but why is it approximate and why is it working.
Link https://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/AproxAlgor/vertexCover.htm

Theorem: *APPROX-VERTEX-COVER is a polynomial-time 2-approximate algorithm i.e., the algorithm has a ration bound of 2*.

Goal: Since this a minimization problem, we are interested in smallest possible c/c^*. Specifically we want to show $c/c^* \leq 2 = p(n)$. In other words, we want to show that APPROX-VERTEX-COVER algorithm returns a vertex-cover that is atmost twice the size of an optimal cover.

Proof: Let the set c and c^* be the sets output by APPROX-VERTEX-COVER and OPTIMAL-VERTEX-COVER respectively. Also, let A be the set of edges selected by line 4.

Because, we have added both vertices, we get $c = 2|A|$ but OPTIMAL-VERTEX-COVER would have add one of two.

=> $c/c^* \leq p(n) = 2$.

Formally, since no two edge in A are covered by the same vertex from c^* (since, once an edge is picked in line 4, all other edges that are incident on its endpoints are deleted from $E`$ in line 6) and we the lower bound:

$|c^*| \geq A$ ---------- 1

On the size of an OPTIMAL-VERTEX-COVER.

In line 4, we picked both en points yielding an upper bound on the size of Vertex-Cover.

$|c| \leq 2|A|$

Since, upper bound is an exact in this case, we have

$|c| = 2|A|$ ---------- 2

Take $|c|/2 = |A|$ and put it in equation 1

$|c^*| \geq |c|/2$

$|c^*|/|c| \geq 1/2$

$|c^*|/|c| \leq 2 = p(n)$

proving the theorem. □

Wow this is really simple.

This algorithm although not very optimal its runtime is very less when compared to the exact solution. But there are many other special cases where polynomial solution is possible for these algorithms. I am looking forward to explore more about this particular algorithm.

1 Hr

Episode two finishing touch. Salesman Problem

09 November 2020 10:10

The idea for the last part of the episode is not yet confirmed. We need to show how our hero illustrates the proof as to how he solved it. I am responsible for the key idea of the episode-2 hence I am the one who is responsible for finishing the job as well.

Need to see the proof of the algorithm.
Need to think about the real life situation.
Need to talk to the dialogue team so that they can finish the job.

Travelling Salesman Problem (TSP): Given a set of cities and distance between every pair of cities, the problem is to find the shortest possible route that visits every city exactly once and returns to the starting point.

Note the difference between Hamiltonian Cycle and TSP. The Hamiltoninan cycle problem is to find if there exist a tour that visits every city exactly once.

Note that one method is Naïve method which says that we need to find all the possible combinations of the vertices and check if there exist a hamiltonian cycle and we will find the cycle which has the least amount of sum of edge waits to be the path we take. But this seems to be too trivial. I am discarding this solution because of its trivial ness.(Nothing much to learn right).

DP solution also exits for this problem, the main idea is that we divide the problem into smaller sets. Here the sets are the vertices. DP state is S , j where S is the set of vertices which are there in the path from 1 to j and are not repeated and are there at least once. More importantly we need to use the smaller subproblems. For using smaller subproblems we need to know the last node which is visited, since we don't know the most optimal node for which we will get the minimum answer we will iterate through all possible options which may be possible, the base cases are that when the set size is one then it is trivial, and the case when the number of elements in the set are equal to number of nodes then we need only add the edge from the last node to the node 1 hence this is the final answer.

I listened to 30min Coursera Algorithm Tutorial the actual time I spent will be 45 min to digest the same algorithm.

Story:

Corrections to be made: In the story we never mentioned that the roads are

between only Antidote places. There can be an issue that there Roads can also be between from Non-Antidote store to an Antidote Store in which case the algorithm fails, So to avoid that make sure to include in the storyline that the path between such places is blocked(maybe because they are long, or may be there is no such road exits.)

Continuation of story: After listening to the story our hero decides that he will try to solve it. Meanwhile our side character Stephan from his side tries to suggest an idea which may help our hero in recovering. His idea is naïve solution. He tries to explain it on the paper to hero, where our hero doesn't pay much attention to, After a seconds of time Stephan gets impatient that he was not getting proper respect he deserves. (Refer to Naïve algorithm for its method, you should illustrate thast he was trying to explain Hero the logic in which ever way you please.) After a while the hero wont even move a bit. (This is because he was thinking intensely. Do you Know Alien X just like that). Just like that he will think about the problem and begins to talk to his conciousness, This is upto you the extent you go to explain. He will repeatedly try to convince his conciousness about the proof.

Firtly he will imagine about the map and the places he need to visit and then he will think about the obstacles he may be facing in due course. He will try to convince his conciousness that DP is the best. Rest of the explainations are sone in the brain. Call me for more detailed idea.
(3Hr)

Hamiltonian Cycle.

09 November 2020 10:16

While seeing the proof of the salesman problem which Sri Ram proposed I came across this name. Its always good to see new algorithms. I heard about the salesman problem through third person. But now I am exploring what it really is. The first term I encountered is Hamiltonian cycle. Will see its proof and will try to describe it here as well 😊

Resource1 : https://www.geeksforgeeks.org/hamiltonian-cycle-backtracking-6/

Hamiltonian Path in an undirected graph is a path that visits each vertex exactly once. A Hamiltonian cycle (or Hamiltonian circuit) is a Hamiltonian Path such that there is an edge (in the graph) from the last vertex to the first vertex of the Hamiltonian Path.

One stupid Naïve way even I got this idea is as following:

Iterate through all possible arrangements of the vertices if there exists atleast one configuration which satisifies the condition that the adjacent vertices in the list contains an edge in the graph then there exists hamiltonian cycle for the given graph.
Time Complexity: O(n*n!)

Backtracking Algorithm
Create an empty path array and add vertex 0 to it. Add other vertices, starting from the vertex 1. Before adding a vertex, check for whether it is adjacent to the previously added vertex and not already added. If we find such a vertex, we add the vertex as part of the solution. If we do not find a vertex then we return false.

What is time complexity of this? I feel that this is same as the before described algorithm except that we used recursion to solve the same.

Doubt araised while exploring:

Why can't we say that there exist a cycle which contains all the vertices. This is true right since we get all the vertices visiting exactly once and we can make sure that we can come back to the node where we started.

Oh the complexity of these algorithms is huge and just now realised that there are special case proofs for some specific problems. Although it may not help the episode much I may explore depending on the time constraint of the project deadline window.

1Hr 15min

Session 2.1 Explaining Episode-2 finishing content

10 November 2020 12:49

I explained my fellow team mates regarding the Idea which I got, luckily this time most of them seems to be excited, which motivates me. We discussed a bug in the description in the story of episode-2 and asked the person concerned to correct the same. Also I gave an explanation what the final finishing of algorithm should be like,There was a good discussion regarding the complexity of Salesman algorithm with some tweak where eventual winner was Sri Ram. The discussion is basically regarding the Salesman problem where the default algorithm just gives the minimum cost. After some struggle everybody accepted that we need the path to slove the problem given in the episode-2. Soon we were perplexed by the complexity raise which may happen because of this addition to the alogorithm. Some of us even belived that the algorithm is so worst that the Naïve way is better. But soon we arrived at the conclusion that the increase in the complexity is very insignificant. The meeting ended in happy note (atleast for few people).

1.5Hr

Graph for Episode-2

11 November 2020 09:46

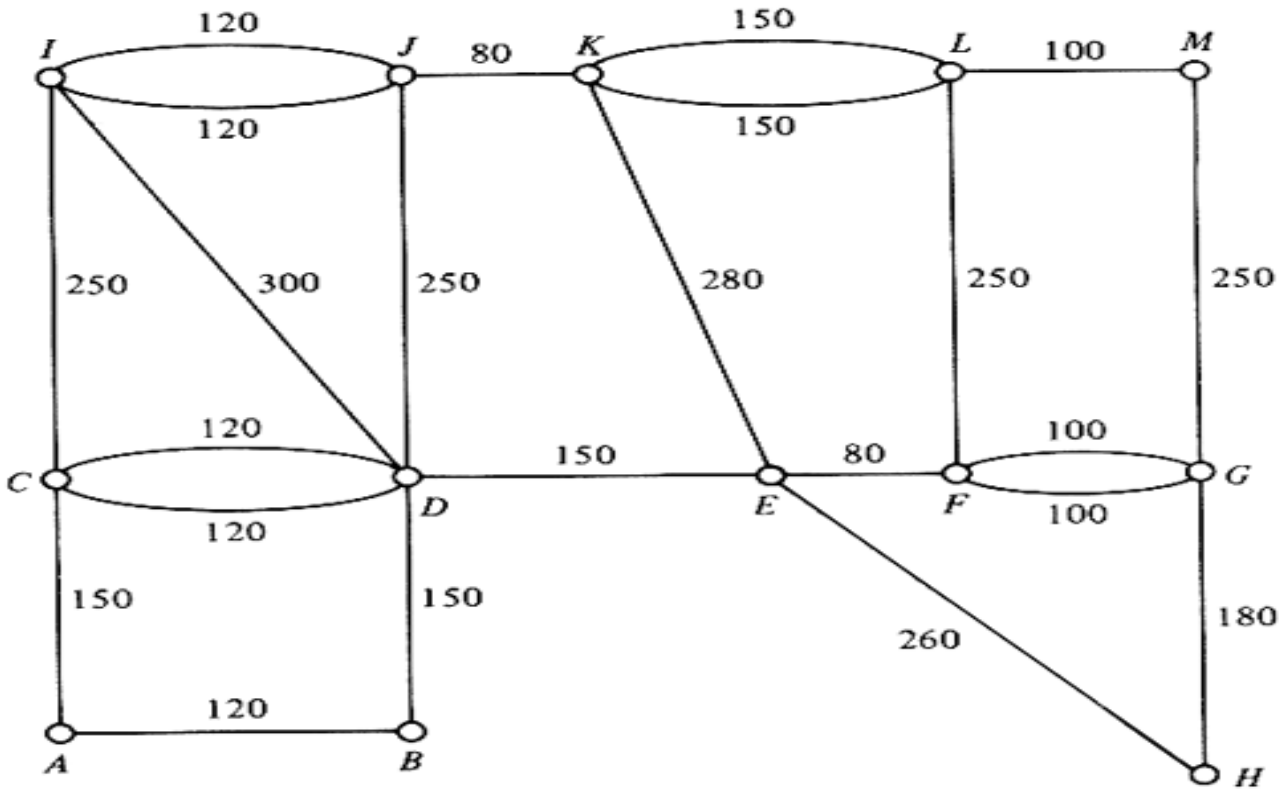


FIGURE 6.18 Optimal solution to the CPP of Figure 6.12. Edges (*C*, *D*), (*F*, *G*), (*K*, *L*), and (*I*, *J*) must be transversed twice, as indicated.

This is the graph I suggested for episode-2. Arjun and Harsha asked me to find a graph for salesman problem . This consists for various cycles and it does have very trivial path where one path which covers all the nodes exist. (this may not be optimal).

Branch and Bound Algorithms

11 November 2020 20:25

Explored Brach and Bound algorithms in search for a story plot for episode-3, given below is the explanation of what I understood and I have quoted few lines from the resource which I felt that they good and simple enough to be understood.

Source: https://www.geeksforgeeks.org/branch-and-bound-algorithm/
Branch and bound is an algorithm design paradigm which is generally used for solving combinatorial optimization problems. These problems are typically exponential in terms of time complexity and may require exploring all possible permutations in worst case. The Branch and Bound Algorithm technique solves these problems relatively quickly.

Knapsack problem:

1) Greedy Approach for Fractional Knapsack.

We know that knapsack problem given by problem statement, given weights and values of n items we need to put these items in a Knapsack of capacity W to get the maximum total value in the knapsack. Note that this is not 0-1 Knapasack i.e where the objects cannot be broken, it will be too easy for solving where we will always choose the object greedily. Not a big deal right.

2) DP solution:

This is also not that much fun where we will store dp[w][i] which says the maximum value which can be possible if the maximum weight allowed is w and there are 1 , 2, … I are still available, So the final answer will be dp[W][N] where W is initial weight of the sack and N is the total number of iteams available, Note that at each step we will consider two possiblities where this particular item is taken and else not taken , if the object is taken the total weight allowed in future iteration si decreased in case of not taken then we will not increase the value and the weight allowed is also not changed. We take the maximum of these two possibilities. Note that without memoization the complexity will be exponential since we will compute the same subproblem agin and again . Once we store the dp value and use the value then the complexity reduces to N*W since these are the only values allowed and hence the complexity.

1Hr

Reviewed the final version of Episode-2

12 November 2020 07:38

Went through the complete story and made suggestions I felt like we need to change. I don’t want to reveal the changes we thought and we discarded as this may spoil the twists if there is one. Basically I only saw the story and saw if it sounded correctly or not. As far a I am concerned everything was fine except one dialogues, we will review it once we complete all the four episodes. We ALSO HAVE LIBERTY IN CHANGING THE DIALOGUES AFTER ALL THE FOUR EPISODES ARE DONE.

Completed WEEK-3 first half in COURSERA course based on greedy Heuristics.

12 November 2020 08:03

Watched Coursera videos in quest to learn about the Knapsack algorithm , completed week-3 all videos based on greedy topic, I listed the proofs and the methods which I learnt in that video. I learnt about greedy their limitations and their accuracies in closeness of answers. About 40 min were spent in video and rest in writing proofs and analysing and writing report here.

Greedy Heuristics:
We have many options to choose the values greediy one such thing is to choose the objects with maximum value first. Obvioulsy this will fail to give the answer in most cases. The second method is to take the objects which have value / weight ratio to be high this will make sure that we get the maximum value for unit weight we spend. But notice that even in this case we might some space and we might end up in not giving the best allocation possible. One question that we might ask is can we give any guarantee on the percentage of accuracy based on this algorithm.

THE ANSWER IS YES!!!
We just need to add one more step to the algorithm that we will take the maximum of the above mentioned algorithm and the maximum valued bag which can be kept in the bag. This ensures that the value is at least 50 % of the most optimal filling of the bag.

Analysis:
Notice that if we allow the fractional parts in the algorithm then we can fill the bag optimally using the max ratio algorithm.
3-step algo >= Weight of first k items using max ratio algorithm.
3-step algo >= Weight of (k+1)th item.
2*(3-step algo) >= Greedy solution answer. > Optimal answer.
3-step algo > Optimal answer/2

Hence it is atleast as good as 50% value.
We can further improve this percentage if we know the limitations in weights of each item. If we know that each and every object's weight is 10% atmost the total weight of the sack then using the same above algorithm we will be able to achieve the 90% of accuracy just by step 2 itself. Note that if the percentage is only 1% of the total weight of the bag then the accuracy would be 99% accurate to the optimal one. The proof of this also not something which is not that difficult. Need to follow similar lines to the proof we proved before .

(2 Hr)

Maximum Cut problem.

12 November 2020 12:14

In search of an algorithm for episode – 3 I am going through this algorithm, I am watching week –4 content in coursera course to help me understand this algorithm better, below are the things which I understood and which I felt are worth mentioning. I was looking for the algorithms for local search algorithms , in this process I found this algorithm.

Maximum cut problem states that we need to divide the vertices into two sets of such that the number of edges which cross should be maximized, unfortunately this is NP complete problem. Hence we need to use some trade off for finding the solution in polynomial time. I the graph was polynomial then we can find the number of edges to be the number of edges which cross over, but for finding the edges it is easy because , we just need to iterate over the nodes and divide them if they are even then we keep in one set else the other set. In this way we can get the max cut in bipartitie easily in case of bipartite graphs.

Local Search Algorithms:

Local search algorithm can be used to find the sets such that the number of edges which cross over the sets Is 50 % of the maximum possible. It states that we need to go through the nodes in a set for some distributed set. If we find that the number of edges which dont cross over are more than the number of them which dont in that case we will swap the nodes. In this way it is very evident that the number oof edges which cross over are 50% of the total edges. Note that this is not a great way of doing things because even in random case we will get 50% of the edges to be crossed. Hence this algorithm is not that of significance even though 50% seems to be good it is not actually that good.

1Hr

Knapsack DP Heuristics week –3 Coursera Course done.

12 November 2020 14:15

What an algorithm, the best until now. I am looking for the extension for Knapsack algorithm which possibly will be an idea for Episode – 3. I went through three videos which were there in COURSERA under algorithm section. All the three videos were interesting they basically explained the algorithm and analysed it. I also read a blog on the knapsack Heuristics. More importantly we had a small discussion between Sri Ram and myself regarding the proof and complexity of this algorithm. Overall, this is an algorithm where all forms of learning are done. I learnt myself by self-reading, I watched tutorials, I explained my friend we argued debated. And finally I asked what ever doubts I have to my fellow team mates. The below is the detailed explanation of what I learnt.

Knapsack Heuristic Part – 2:

This is again an algorithm where we compromise accuracy for decreasing complexity, So whats so special, Here we can control the error we will be getting. And the complexity oif the algorithm will increase depending the accuracy we need. Let's dive into the algorithm:

The well-known DP algorithm is that we take subproblems where each subproblem is the maximum value which can be obtained using the items from 0 to I and sum of weights j. The complexity of this with memorization is N*W since we will calculate the value of sub-problem only once and number of distinct problems will be N*W. The problem with this is that the W can be huge in which case the time complexity will blast. So here is the alternate algorithm.

Instead of weights we will use values but with slightly twisted logic, here we will use the logic that we will be trying to find the minimum weigh t for a given value and prefix of items allowed to take. It s complexity is N*max value of total value of the bag possible. Note that the maximum value possible is N* maximum of value. SO the total complexity will be N^2* Vmax. Which is good if the value is small enough. Note that if both value and weight are more then we need to convert those problems in to one of these in some way so that we will be able to solve them quickly.

In the approximated logic we will divide the weights of the items using some number m so that the values of the items will become smaller so that the second dp solution can be applied to them, so depending on the value of m we take the accuracy will depend and accordingly we will get time complexity. If divide by a large number then the complexity will automatically decrease since we are dividing and also the accuracy will also decrease since not all numbers will be divided by m exactly. The time complexity of this algorithm is O(n^3/e) where e is the measure of error which is allowed for this. This is some kind of briefing for this algorithm. The discussion we had was about the complexity in which the eventual winner was Mr. Trinadh finally.

3Hr

Idea Proposed for Episode – 3 !! 😊Impressive Work(Including Discussion made in TEAMS)

12 November 2020 18:11

Finally, … Finally, we have arrived at an idea for algorithm along with story for EPISODE – 3, the idea is mentioned below. Please don't read it if you want to enjoy the story. Please read the story it contains efforts of all our teammates and the algorithm is proposed by the ALGO TEAM (Trinadh, Harsha, Sri Ram) we are proud to present this story and we expect our ever-dependent illustration team. DONOT CONTINUE READING IF YOU WANT TO ENJOY THE TWISTS IN THE STORY. Initial meeting was done to discuss with the algorithm team. That was fun, then I held a meeting for the entire TEAM they raised dome concerns and hopefully we were able to resolve most of them.

THE FOLLOWING IS FOR THE SAKE OF REPORTING DONOT SEE.....

Based on the previous defeat that the Jekyll faced, this time he wants to act smart, So he decides to run an investigation which may lead him any clues. He visits those places in hope of finding any evidence. He finds that the person he is looking for is... yeas you herd it right he finds that Vandal is MR WAYNE who he suspects for long time , he further continues the research and finds all the plans of the Wayne and this time he tries to counter Waynes plans. He decides the plan and works hard to implement that, Alas by the time he implements it is already too late. News everywhere is that a new pandemic is spread everywhere. But there is no clue as to what it is. The government officials concern the mater brain Jekyll to help them, Jekyll behaves as if he didn't listen. But he does pay attention. He looks for all the evidences he need, he always has the doubt that this is evil thought of the Wayne, to his doubt it is true that the Wayne is indeed responsible for this evil act. For the same reason he looks further into the details he got. He finally conforms that there is a laboratory where there is all the vaccines for the same. He goes on to find all the hints which are required. He finds that on a particular day and particular time he finds that most of the people are not there in the laboratory, he finds all the information of a particular person using him he will try to ENTER the lab. To his plan he takes the iris and finger print of that person and enters the lab. REMEMBER that the Jekyll doesn't know how to get back. He goes to the place of vaccines using his laptop.

Finally using his laptop he goes to the place where the vaccine is there. He goes around and gathers all the required data from the lab. He also gatheres all the data required to get back from the lab. This time he uses his laptop to shackle all the security syestems. He calculates the time required to shackle all the syestems. Soon he feels that there is no need of this lab for him, so he sets a timed bomb and tries to act smart. He makies sure that he gives buffer by giving some additional time than wjat is required. Satisfatorily return his weay back, he thinks everything is going to plan now he sees something very unexpected. (ADD THAT GOVT STORY HERE) he sees that easch ogf the govt official is at some distance(contact me in case of doubt) And finally decides on KnapSack Dp improved algo to do the same.

4 Hr 30 min

Reviewed Current Version of Episode – 3 (Dialogues)

13 November 2020 08:59

Reviewed the current version of the story and suggested them some flaws which I found. They are yet to respond but we are likely to conduct a meeting about the same.
After yesterday's meet we thought we will ask the dialogues team to start the necessary part by earliest as it may take illustrator team some effort to complete it. Remarkably the Dialogue Team have completed almost half of the story by the time I woke up, so I thought it would be better for me to have a look.
Suggestions made:
Regarding Wayne intro with Stephan.
Stephans part in current Episode.

Suggested few changes to the Dialogues of Episode – 3.

14 November 2020 17:45

Oh , today I saw that there is considerable gap between what I thought and what the dialogues represent. I suggested 5 6 changes to the dialogues, hope they receive my suggestions in positive sense and change them, I don’t want to tell the changes but they were comments in the doc where we are writing the dialogues. It took some time to understansd the story they wrote and to recollec the story which I though should have been in that place.

1Hr

Added minor changes to story.

14 November 2020 18:08

After seeing the current version of the episode – 3 , I thought few changes will make it better, like the way he hacks the defence syestem. The following it the things which are changed.

Once he sees the place where he found the vaccines or medicine he decides to find his way back to the exit door. He recollects that he doesnt have the acess to go back, SO he searches for clues in the lab where the vaccines are stored which is at -50th floor. He sees that there is a room with DANGER written on it. He some how feels that it not for him, He goes there and sees whats inside it. SUPRISINGLY there are many more cables going somewhere, he thinks that these should be the wires connecting the computers regarding the security. He tries connecting one, for unsuccessfully. He tries few more but finds that they have details regarding something else. Soon he sees that there is no use in trying in this useless place. He thinks of leaving the place, just in time he realizes that why the hell in this world will some one place high security passwords visible to everyone. He thinks hard. He remembers ENIGMA code he thinks in similar lines. He tries to run the algorithm which searches for the initial characters of all the strings which were generasted by these cables. He sees that they are same. OOF he tries to remove the first part of the string and again tries to hack the security, To his surprise it shackles it. He now has the access to the encoded format of the passwords. He runs an algorithm to find the pattern in the known strings which were encoded. (Guys we will discuss in TEAMS it is difficult to write each detail. You can create you brown story but you need to show this part as this is not that easy to hack some laboratory passwords for inspiration see how they decode ENIGMA code the story they build the scene). After that before leaving he sees all the

1 Hr

Searched for Detective's notice board (Illustration Episode - 3)

15 November 2020 20:08

For the sake of the Episode – 3 we have a place where we need to show the notice board of the Hero , the following are the suggestion I made to the illustration team.


SCARS
YELLOW KING
CARCOSA

Dialogues Done for Episode - 3 ?? Had a look at the final version.

15 November 2020 20:24

Today morning and yesterday many of the team members had a discussion as to where the story is leading to , but finally we have decided on the story. And the dialogues team did their job pretty quicky. I as a representative of Algo team took the responsibility of the rechecking of the validation of the story and going through the story. I am glad that all the dialogues were perfect to the semantics, There is one small portion where we need to add algorithm which will be added tomorrow.

Last Three days Nov 13,14,15 of Thinking about FINAL EPISODE – 4

16 November 2020 11:08

In the quest of a good finishing touch of our project we started looking for algorithms in Coursera and other platforms, but haven't found anything very interesting. We have gone through some algorithms but failed to find a good story on the same. As an initial topic I looked at spanning trees but later gave up on it since we already have Salesman problem which some people thought is advance problem than MST. Then I looked at some videos in COURSERA course which I am following for the sake of this project, but never gained interest in those. We held two meetings with our Algorithm team. To my surprise the remaining team members also searched for some algorithms in different domains but failed to get some solid idea. We are looking to find an algorithm latest by Nov 17th and the story should also be ready by the same date.

TO DO:
Should have a look at 2-SAT
Inspiration : Recently I saw a question based on the same in codeforces and is fascinated by its name. But I literally don't know what really it means.
Should complete Local Search:
I went through few videos but haven't really understood the importance of these algorithms.
(6Hrs)

Appendix Episode - 4

22 November 2020 09:05

Wrote Appendix for episode – 4 . As part of final algorithm we took KMP, I have written the appendix of the same and haded it to the illustration team.

Session 4.1 conducted on TEAMS yesterday + Chat in Telegram

18 November 2020 15:35

Yesterday we conducted a meet regarding the algorithm of episode – 4 and potential story. The speaker of the session for the majority of the meeting was me. Then Arjun took over, in explaining his opinions. Before the team meet I gathered with my Algo mates we discussed what we did in past few days and after gathering algorithms we have finally assembled for the team meet. We discarded some of the algo proposed by SRI Ram keeping in view of the story we are looking for. We finally ended by taking the algorithm for episode – 4 (suspense) then also planned what we should do in next few days. We also had a small chat in Telegram about the story of the episode – 4.
Meeting attendees : Sri , Harsha , Arjun and myself. Not all three were active atonce because of various issues.

Knuth-Morris-Pratt Algorithm

17 November 2020 23:27

Went through the videos in the COURSERA course based on the same. Also referred some blogs written in gfg. Went through the proof of this algorithm.

Debugged Knapsack code written By HARSHA.

18 November 2020 11:25

As part of Episode – 3, Harsha wrote the code for knapsack. But he is struck with the code. The code he wrote has some minor bugs which I fixed today after he requested me to do. We had a teams chat and I was shown his code. Successfully we were able to debug his code. We also ran few test cases which were running fine. So the code will now be used to be kept in illustrations for Episode – 3 .

Viterbi algorithm (Didn't Understand) 😐

17 November 2020 16:16

I am currently looking for this algorithm, as an initial lead I looked for You tube videos didn't understand anything. Second attempt is trying to understand Wikepedia blog on this algorithm. The following are the statements which I found about Viterbi algorithm from Wikepedia page.

Source : https://en.wikipedia.org/wiki/Viterbi_algorithm

"The **Viterbi algorithm** is a dynamic programming algorithm for finding the most likely sequence of hidden states—called the **Viterbi path**—that results in a sequence of observed events, especially in the context of Markov information sources and hidden Markov models (HMM)."

The following is the example which is given in the same resource. Consider a village where all villagers are either healthy or have a fever and only the village doctor can determine whether each has a fever. The doctor diagnoses fever by asking patients how they feel. The villagers may only answer that they feel normal, dizzy, or cold.

The doctor believes that the health condition of his patients operates as a discrete Markov chain. There are two states, "Healthy" and "Fever", but the doctor cannot observe them directly; they are *hidden* from him. On each day, there is a certain chance that the patient will tell the doctor he is "normal", "cold", or "dizzy", depending on their health condition.

While trying to understand the very first line itself Many unknown terms are there. They are HMM,MIS. So the next aim is to find what those are. Struggled to understand anything regrading this. Finally Arumulla said that this algorithm is found in Machine Language course so gave up on this as the time is peeking up.

Local Search and 2-SAT

16 November 2020 14:08

Watched tutorials in COURSERA about local search. In the tutorial he explains what local search does, and gave an example of max cut. He explained the various questions that we need to ask our self before solving it using Local search. These questions include which optimal solution to take into consideration. Where to start. Is it always better.? For this particular question the answer is sadly no. As there are questions where local search may give very bad optimization. Most of the time we ue it as post optimization method to further improve our solution we got using some other algorithms.

2-SAT:
Going on a high level understanding it is to find assignments of variables to a Boolean expressions such that each small Boolean expression contains only two terms. This is tractable. We can solve using local search. The basic idea is that we flip a variable's truth value of a variable where the boolian expression is failing because of it. Will we get a better solution. May not be the case. The proof I didn't explore yet. But I will explore if time permits. In the first go I didn't understand the algorithm. But with its looks I didn't find it very usefull for the current episode will revisit it depending on how other team mates think about the final episode.

From Genome Sequencing to Pattern Matching

17 November 2020 18:16

What an algorithm !!, I was looking for some string based algorithms , to my luck I found a course in COURSERA which has all the features I was looking for. It is Algorithms on strings offered by UCSnaDiego. I though testing it would be great. As a result saw the introductory video, immediately fascinated by the content and the way the prof was teaching. I gained interest so much so that I completed one whole week in one sitting. It was great experience. I learnt various new algorithms . I also revisited some well known algorithms. I will try to breif the week – 1 content which O learnt.

The basic search is to find if the patterens are found as a substring in a string. The naïve way is always there. In which we will start from each node and try to find the pattern. If we fail we move to next pattern. Note that we have to find this thing for many patterns not just one. So the complexity will be high as TEXT*PATTERN where TEXT is length of Text and Pattern is sum of lengths of Patterns. Which is quite high considering the fact that Patterns sum will be as high as 10^12 and Text will be as high as 10^9.

Can we do better??
Suprisingly, YES.

Trie:

See the main problem in the above method is trying to solve for each string seperately . How about doing all of them at once. But how ? Trie is the solution. Store the patterns in the trie data structure and then it is very easy to compute if a string starting with some node in text has a pattern. So complexity is only TEXT* LONGEST Pattern. So we are done right ? Not yet. Number of edges in trie is PATTERN which is in trillions which is impractical. Oh no!!

Can we do better ??
Yes!!

Suffix Trees:

Now we will try to store each suffix of the text in trie structure. And will just iterate for all patterns in this trie data structure. But wait the number of nodes in trie can be as big as TEXT^2 which is very bad. So give up. Not yet we can store the string to be edge. Note that there can be only 2 nodes where there is a clash. (Refer video for clarity). So the memory came down to O(TEXT) and runtime is also O(TEXT + PATTERN) is it good. Also note that this is after we construct trie , so far for naïve algorithm it takes TEXT^2. But there is a mention in video that there exist linear time algorithm. Hence the above mentioned complexity is after taking that into account. Now the only issue is construction of trie takes some more space. Moreover, we cannot get approximate matching using these methods which is one of our desired result.

Can we do better??
?? (I am yet to find this part 😊)

Burrows-Wheeler Transform

17 November 2020 22:50

Oh !! I have at least watched 4 times the videos, but sadly wasn't able to gra* but now I think I understood the algorithm more or less. I did watch tutorials, course videos, disussion with friends to understand this particular algo. I did understand what it is but was not able to understand the proof of the algorithm. Following is the brief explaination.

Let us try to compress the given string into some string so that we can find the substring question more efficiently. One of the most straight forward thing is that we can read length encoding , where we willl use the number + alphabet format where the number represents the number of time the alphabet repeated in the current string. EG AABBAAA will be represented as 2A2B3C. This is the simplest compressing algorithm. Fair enough.
.The algorithm to get BWT transformation is that generate cyclic rotations for the given string with $ added at the end. Then sort them, considering that $ is having the smallest values than all alphabet. The last column will be the required BTW. But why? Notice that there are more runs in the BTW than in the original string most of the times. (Take the example of and in any newspaper and apply BTW u will notice that the number of times a appears is huge using something similar we will assume that we will get more runs in the BTW transformed string.) Now current goal is apply BTW then compress.

But how will we get the original string from BTW? Is it even possible ?

Suprisingly it is easy to revert it. What we know is the last column => we know the first column as well => we know all 2 mers. We can sort them. => we know all three-mers => Sort them continue DONE we reverted and were able to achievce the string back,

Memory intensive. Can we invert with less space ?

Note that the order in which the alphabet appears in the first column is same as in the last column, (intuition in the video please refer to that in case of confusion). (First Last property). Oh my god after going through zillion times I understood this step. I will try to explain it as clear as possible. Notice that instead of finding all the cyclic rotations and then sorting in which case the complexity will blast we can simply do one thing, is there a way to avoid doing that labour work, can we explore the fact from First Last property. The asnwer is yes. Notice that using the last coulumn and the first column it is easy to retrieve the complete string, not only retrieving the building of the BTW is also easy done, The only thing is the we need to remember that the last column + first colum are 2 mpseudo code.ers. And using the First Last property we will get the complete string. Start with $1 go to corresponding last column check what is there, then go there repeat. So depending on this u should get a string at the end. And one more thing is that we need to build this BTW, that can also be done notice that the First Last property and fill the BTW in the reverse transversal way. This is quite difficult to understand but I at least hope that the after reading this and watching the tut again the things will become clearer.

Great , So we are done right?
No
But why?
Where is Pattern Matching ??

This can be easily understood using the pseudo code.

```
BWMATCHING(FirstColumn, LastColumn, Pattern, LASTTOFIRST)
    top ← 0
    bottom ← |LastColumn| − 1
    while top ≤ bottom
        if Pattern is nonempty
            symbol ← last letter in Pattern
            remove last letter from Pattern
            if positions from top to bottom in LastColumn contain symbol
                topIndex ← first position of symbol among positions from top to bottom
                           in LastColumn
                bottomIndex ← last position of symbol among positions from top to
                              bottom in LastColumn
                top ← LASTTOFIRST(topIndex)
                bottom ← LASTTOFIRST(bottomIndex)
            else
                return 0
        else
            return bottom − top + 1
```

Also given that using cout array its time complexity can bbe reduced but didn't given proof as to how this is done. I am pasting the pseudo code for now but will get to it once I am free from course work.

```
BETTERBWMATCHING(FIRSTOCCURRENCE, LastColumn, Pattern, COUNT)
    top ← 0
    bottom ← |LastColumn| − 1
    while top ≤ bottom
        if Pattern is nonempty
            symbol ← last letter in Pattern
            remove last letter from Pattern
            top ← FIRSTOCCURRENCE(symbol) + COUNT_symbol(top, LastColumn)
            bottom ← FIRSTOCCURRENCE(symbol) + COUNT_symbol(bottom + 1,
                        LastColumn) − 1
        else
            return bottom − top + 1
    return
```

But where is the position of the substring??
Upnext.

5Hr