Hua Yang
Alexio Mota
Erik Kamp

Homework 5

Minimum Spanning Trees, NP-Complete Problems and Approximation Algorithms

Deadline: May 6, 11:59pm.
Available points: 110. Perfect score: 100.

Homework Instructions:

Teams: Homeworks should be completed by teams of students - three at most. No additional credit will be given for students that complete a homework individually. Please inform Athanasios Krontiris **if your team has changed from the previous assignments.** (email: ak979 AT cs.rutgers.edu).

Submission Rules: Submit your solutions electronically as a PDF document through `sakai.rutgers.edu`. Do not submit Word documents, raw text, or hardcopies etc. Make sure to generate and submit a PDF instead. Each team of students should submit only a single copy of their solutions and indicate all team members on their submission. Failure to follow these rules will result in lower grade in the assignment.

Late Submissions: No late submission is allowed. If you don't submit a homework on time, you get 0 points for that homework.

Extra Credit for LaTeX: You will receive 10% extra credit points if you submit your answers as a typeset PDF (using LaTeX, in which case you should also submit electronically your source code). Resources on how to use LaTeX are available on the course's website. There will be a 5% bonus for electronically prepared answers (e.g., on MS Word, etc.) that are not typeset.

25% Rule: For any homework problem (same will hold for exam questions), you can either attempt to answer the question, in which case you will receive between 0 and 100% credit for that question (i.e., you can get partial credit), or you can write "I don't know", in which case you receive 25% credit for that question. Leaving the question blank is the same as writing "I don't know." You may get less than 25% credit for a question that you answer erroneously.

Handwritten Reports: If you want to submit a handwritten report, scan it and submit a PDF via Sakai. We will not accept hardcopies. If you choose to submit handwritten answers and we are not able to read them, you will not be awarded any points for the part of the solution that is unreadable.

Precision: Try to be precise. Have in mind that you are trying to convince a very skeptical reader (and computer scientists are the worst kind...) that your answers are correct.

Collusion, Plagiarism, etc.: Each team of students must prepare its solutions independently from other teams, i.e., without using common notes or worksheets with other students or trying to solve problems in collaboration with other teams. You must indicate any external sources you have used in the preparation of your solution. Do not plagiarize online sources and in general make sure you do not violate any of the academic standards of the course, the department or of the university (the standards are available through the course's website). Failure to follow these rules may result in failure in the course.

Problem 1 (20 points): You are a programmer in an electronic commerce company, where you are asked to design a new recommendation system. The idea is to divide all users into k groups based on their purchase history. After the grouping, your system can provide recommendations to users based on the purchases of other users in the same group.

To perform the grouping, we define a function $s(u_i, u_j)$ to reflect the similarity of two users u_i and u_j. The similarity function is defined as:

$$s(u_i, u_j) = \begin{cases} \text{1 + the number of common purchases } u_i \text{ and } u_j \text{ have made} & (i \neq j) \\ 0 & (i = j) \end{cases}$$

To optimize the grouping, we need to minimize the similarity of the most similar pair of users that have been assigned to different groups.

(a) Provide an efficient algorithm that groups all users into k groups and satisfies the optimality requirement mentioned above (i.e., minimizes the similarity of users in different groups).

Answer

Let L be a list of n users. Define a graph with n vertices corresponding to the n users. Draw edges between the nodes such that the weight on the edge is $s(u_i, u_j)$, Now we'll apply Kruskal's algorithm on this graph until there are k clusters (groups) left.

(b) What data structures do you assume for the implementation of your algorithm? What is the running time of your approach given the data structures you employ?

Answer

We assume the use of a disjoint-set data structure which would provide a running time of $O(ElogE)$.

(c) Prove the correctness of your algorithm.

Answer

Let $G = (V, E)$ be a graph with weights on its edges, and let X be a subset of edges of G such that $X \subseteq T$ where T is a minimum spanning tree of G.

Use Induction.

Suppose e is a element of T. Adding e into T creates a unique cycle. We will remove a single edge e' from this unique cycle, thus getting $T' = T \cup \{e\} - \{e'\}$. It is easy to see that T' must be a tree – it is connected and has$n - 1$ edges. Furthermore, as we shall show below, it is always possible to select an edge e' in the cycle such that it crosses between S and $V - S$, $w\{e'\} >= \{e\}$. Therefore $w\{T'\} = w\{T\} + w\{e\} - w\{e'\} <= w(T)$. However since T is a MST, it fllows that T' is also a MST and $w(e) = w(e')$. Furthermore, since X has no edge crossing between S and $V - S$, it follows that $X \subseteq T'$ and thus $X \cup \{e\} \subseteq T'$.

Problem 2 (20 points): In wireless sensor networks, it is an important task to periodically collect data from an area of interest for time-sensitive applications. The sensed data must be gathered and transmitted from the sensors to a base station through the most efficient way in terms of energy and speed.

The sensor nodes are deployed randomly in the target field. Then, the network establishment begins with the formation of communication links between the sensor nodes. Consider that $E(a, b)$ is the energy needed to send message from node a to node b, which typically depends on the distance between the two nodes.

An idea in the sensor network literature is to use a minimum spanning tree (MST) as the underlying communication structure in order to shorten the total transmission distance, while guaranteeing that there is a way for a message to reach every node in the network. This means that we need to construct a MST of the nodes in the network.

It is possible, however, that a node in the resulting MST will be connected with a lot of neighboring nodes, e.g., consider the case the MST looks like a star-like structure where a central node is connected to all other nodes. Such nodes with many edges may need to fuse more data collected from its neighbors than other nodes and to consume more energy. This may cause nodes with a lot of neighbors to die earlier because they exhaust their energy. In order to avoid this situation, we limit the number of connections for each node to be at most d in the communication tree, for a given positive integer d.

A. For $d = 2$, describe an algorithm for finding such a communication tree and show that this problem is NP-complete problem. (Hint: consider the relation with finding a RUDRATA-PATH on a graph.)

Answer

k-degree Bounded Spanning Tree

Step 1. Use Rudrata on a k-degree Bounded Spanning Tree

- Given the graph $G = (V, E)$, create an instance of TSP.
- The set of nodes V is the same for both graphs.
- Let weight w_e on an edge $e \in E$ be 1
- The budget is $|V| = 2$
- The Rudrata Cycle exists if TSP(G,n) returns yes in the following cases:
- A TSP tour visits all nodes and is a cycle
- Unable to find a lower weight TSP tour since all edges have weight 1 and the tour requires $|V|$ edges

Step 2. Show that k-degree Bounded Spanning Tree (kBST) is in NP.

- Given a tree T on graph G, check that each node u in T has degree at most 2 in linear time by inspecting each edge $e = (u, v)$.
- Traverse through every node from some starting node s, use DFS to show that the graph contains no cycles.
- Upon successfully traversing the tree T, this proves that it is a 2-degree Bounded Spanning Tree, so kBST is in NP

B. Describe a polynomial time algorithm (in detail) that solves this problem, if you have an algorithm that solves TSP in a polynomial time.

Problem 3 (30 points): Lord Tywin Lannister is organizing the feast for the upcoming wedding of his grandson, King Tommen Baratheon, with Lady Margaery Tyrell. The Queen of Thornes, Olenna Redwyne, and grandmother of Margaery, has purchased two exquisite tables from Bravos for the event for the guests to sit and attend the feast. These tables are big enough to accommodate all n guests but she bought two of them because it is better if some of the guests are separated. Weddings can lead to quite unfortunate events in the land of Westeros...

In particular, the list of n attendees includes members of the Tyrell family of Highgarden, members of the Martell family of Dorne as well as members of the Lannister family of Casterly Rock. Unfortunately, not all attendees have good relationships one with another. For instance, Oberyn Martell accuses Gregor Clegane from the Lannisters for the death of his sister and her children. So, it would not be wise to allow both Oberyn and Gregor to sit on the same table, if possible.

Lord Tywin and the Queen of Thornes have drafted a list of all m pairs of attendees who do not have good relationships and should better be placed on different tables. Their objective is to find an assignment of attendees to the two tables such that the total number of conflicts between any two guests from different tables is maximized.

Based on the above description:

1. Formulate the problem that Lord Tywin and the Queen of Thornes are facing as a graph problem and define the cost function clearly.

   Answer

   Vertex Matching:

   Go down the list of m matches of guests and form subsets of guests that have no vertices in common, in other words are not in conflict with one another. Repeatedly add to the sets until there are no more guests in the n list. Then decide on the table seatings based on each set. The algorithm for this is as follows:

   - Find a maximal matching M from guests / edges E
   - Return S ={all endpoints of edges in M}

   This will return a set with the size or set that is at most twice. This will take an approximation ratio of $\alpha A <= 2$.

2. Propose a local search algorithm that is a 2-approximation polynomial time solution. Show that the algorithm provides a 2-approximation solution.

   Answer

   Local Search :

   To search through the set graph that we had just created and find a particular guest we can use a Local Search algorithm. First off we find an initial solution say s'. Such a solution will exist in a set or neighborhood lets say s. Therefore while there is some solution in this neighborhood that we have not explored explore the neighborhood finding the solutions and compare them to our current optimal solution. After all sets or neighborhoods of guests have been visited go to the next neighborhood until all neighborhoods have been exhausted. When they have been exhausted return your optimal s'. See the algo below :

   - let s be any initial solution
   - while there is some solution s' in the neighborhood/set of s
   - for which cost(s')/path < cost(s)/path: replace s by s
   - return s

3. Propose a greedy algorithm that is a 2-approximation polynomial time solution. Show that the algorithm provides a 2-approximation solution.

   Answer

Greedy Algorithm/Simulated :

First like in local search pick a random solution say s. Randomly choose sets in the graph and see if the cost of the old solution s(first solution if this is the first iteration) is greater than the cost of the new solution s' in our current set/neighborhood. If the old cost is greater then save the new solution s' as s. Otherwise we replace s with s' by looking at the temperature T which is based on finding the low cost solution or in this case finding a guest in the guest list. This temperature will limit where the graph searches optimally taking the lowest solution in this case looking at the problem as greedy. So again we switch s with an s' based on the probability of $e^{(|change\ in\ old\ cost\ -\ change\ in\ new\ cost|)}$/out current T. The algorithm is shown below :

- let s be some random solution encountered in the sets
- repeat {
  - randomly choose a solution s' in the neighborhood of s
  - if Δ = cost(s') - cost(s) is negative:
  - replace s by s'
  - else:
  - replace s by s' with probability $e^{\Delta/T}$
- } end repeat

4. Propose a randomized algorithm that is a 2-approximation polynomial time solution. Show that the algorithm provides a 2-approximation solution.

Answer

Randomized :

When using randomization we can search through the graph by simply using a local search. However, instead of picking one specific start point / start set we can pick a random one to start at. Once the random set has been selected if the local search needs to choose between two paths we can use a randomized approx to choosing which path. This generally leads to the probability of finding an optimal solution on run p of $O(1/p)$. Which in general since were using local search with a simple randomized factor will have an approximation value $<= 2$.

[Note: Though the problem is similar to the problem 4 in homework 3, the assignment here does not require that there is no conflict between guests placed on the same table. You may have a situation where it is impossible to assign the guests so that no conflict arises on a table. For instance, both Oberyn Martell and Willas Tyrell hate Gregor Clegane, but they are also not very fond of one another... Instead, you are required to find an assignment that maximizes the number of conflicts between guests sitting in different tables.]

Problem 4 (40 points): Daenerys Targaryen is moving her army of elite warriors, the Unsullied, in the land of Essos trying to liberate as many cities as possible from the slave-owners that rule them. The major cities of Essos, such as Yunkai, Meereen and Astapor, are trying to coordinate their defenses.

The strategy suggested by Daenerys' advisor Ser Barristan Selmy is to try to isolate the major cities one from another so that they cannot communicate and coordinate against Daenerys. Ser Jorah Mormont generated for this purpose a map of Essos that provides the road network connecting the biggest n cities on the continent.

Daenerys agreed with Ser Barristan Selmy and decided to place teams of 100 Unsullied along different road segments so that it would not be possible for a messenger from one of the major cities to reach another through the road network without being detected and... facing the consequences... Daenerys is worried, however, of spending too many of her Unsullied for this purpose and reducing her main military force. So, she wants to identify the minimum number of road segments that her forces will have to guard so as to achieve this objective.

Without knowing how the road network with the *n* cities and the roads connecting them looks like:

a. Provide a polynomial time algorithm that solves Daenerys' problem exactly, when the objective is to separate only Yunkai and Astapor.

Answer

IDK

b. Provide a solution that achieves an approximation ratio of at most 2 when Daenerys wants to separate all three of Yunkai, Meereen and Astapor.

Answer

In order for Daenerys to separate the three cities she must create a maximal matching for each city's road segment sets. In other words she needs to find road segments or sets in which no set of segments contains a path to another city. This will create a vertex cover of road segment sets and can be solved in $\alpha A <= 2$ or in the worst case of 100% error is solved in $\alpha A = 2$ time. Please find the algorithm below :

- Find a maximal matching of road segment sets of one of the cities C within all the road segments
- Then return S which is seen as all the endpoints of the road segments in the maximal matching for a particular city.

From our 3 or more vertex covers S, we can pick the best by first using a maximal matching of size that will give us something close to the optimal vertex cover. By finding a close to optimal maximal matching for each city we can then generate the 3 close to optimal vertex covers that will make sure to separate each city.

c. Propose a local search approach for the general case where Daenerys wants to make sure that at least *m* cities remain isolated one from another.

Answer

In order to make sure that m cities are isolated as stated we can simply use a local search. First we can start off at any one of the m cities and begin a local search starting at a random path or city road segment. Then perform a local search adding new segments or in this case neighborhoods to the search path until a neighborhood outside of the city is hit. Once a foreign / different city is hit exhaust all possible surrounding neighborhoods or segments not leading outside of the city and add them to our search path. Do this for each of the m cities creating a new unique search queue or path for each of the m cities. At the end you should have searched m times and should have m local search queues of all the roads within each of the m cities that do not lead to a foreign city. Please find the general algorithm below.

- for all m cities {
  - let s be an initial path within the city
  - while there is some path within city(m), s' in the neighborhood of s {
    - for which s' does not lead outside of the city(m) {
    - add s' to our local search queue for city m

- replace s with s’
- }

* }

- }

d. The cities of Essos are working on a counter-strategy after they heard the news regarding Daenerys’ strategy from spies.

   They want to place soldiers along the minimum set of roads that need to be protected so that all the cities can still communicate one with another. When a soldier detects the 100 Unsullied along a road segment, they will send a large enough force to take them out. To be able to monitor a road, it is necessary to place a soldier along every mile of the corresponding road. The cities need to minimize the number of soldiers used for this purpose so that they are sufficiently defended in case Daenerys attacks them directly. Consequently, the cost of protecting a longer road is directly related to its length [Hint: Thus, the cost of selecting a road segment to monitor is a metric function and satisfies the triangular inequality.].

   Assume that the road network between the *n* cities of Essos is a complete graph. Consider the problem of identifying the minimum set of roads that need to be monitored so that at least the *m* major cities will remain connected given the above strategy. Provide a polynomial time approximation algorithm for this problem that has an approximation ratio of 2.

   Answer

   IDK