Hua Yang
Alexio Mota
Erik Kamp

Homework 4

Searching for Paths on Graphs

Deadline: April 20, 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: [25 points] After the departure of Storm Sandy, the small town of New Town suffered an electricity power shortage. After this experience, the local electricity agent decided to create plans to better handle future events. The idea is to send out immediately after the power failure a truck to investigate the damage along every street in the town. In order to recover the electricity network as soon as possible, they want to find an efficient tour so that the truck visits each street exactly once and then goes back to the town headquarters. Suppose each street in New Town is a two-way street. The task is to find whether there exists a tour that visits each street only once and then goes back to the starting point.

A. Show that the problem has a solution if and only if there are no dead ends and no n-way junctions in New Town, where n is odd, i.e., each intersection in the town has an even degree.

Answer

Using a Eulerian path algorithm such as Hierholzer's , we can visit each node or in this case each street exactly once, given that the graph or in this case New Town, has even degree intersections or in this case does not contain a dead end or odd number of connecting streets. This algorithm will start a tour from the headquarters continuing to visit unvisited streets until returning to the headquarters. Due to the fact that it is an even intersection degree we will not get stuck at an intersection and will always have an unvisited street to go to. If it was an odd degree we could visit a street/intersection and either have no where to go after (dead-end) or come to realize we cannot go further because all connected streets have been visited already.

B. If such tour exists, please describe an efficient algorithm to find the tour in linear time, i.e., in at most $O(|V| + |E|)$ time, where V is the set of junctions and E is the set of streets.

Answer

(Hierholzer's algorithm):

- Visit any sequence of roads, visiting a new road each time.
- Because it is guaranteed that intersections have an even degree we will never get stuck because there will always be an unvisited road r leaving the current road (vertex)
- Each time visiting a street keep track of the street and mark as visited
- After returning to headquarters if there exist roads in which contain intersections that have unvisited roads basically repeat the main process by following all unvisited roads on that path until returning to the original intersection.

Problem 2: [25 points] In a chemistry class, the professor asks you to synthesize products using existing chemicals. For instance, 2 parts of chemical A produces 1 part of chemical B, or 3 parts of chemical C can give us 1 part of chemical D. Lets assume that we have n different chemicals that we can mix. The production ratio given two chemicals is fixed and does not change. These ratios are given in an $n \times n$ matrix, between all the chemicals. In particular, using x parts of chemical i you will receive $R_{ij} \cdot x$ parts of chemical j, or using y parts of chemical j you will receive $R_{ji} \cdot y$ parts of chemical i. Note that R_{ij} is not necessarily an integer, it can be a real number.

Because of the principle of mass conservation the ratios have to satisfy the following requirement: $R_{ij} \cdot R_{ji} < 1$. This means that converting an amount of chemical i to chemical j and then back to chemical i should result in a smaller quantity of chemical i relatively to the original one.

A. The Chemistry professor asks from you the following. Given the $n \times n$ matrix, find the best sequence of chemical reactions to produce the most of chemical D given an available quantity of chemical A. As mentioned before there are n different chemicals you can use and the matrix specifies the production ratios of all corresponding reactions. Provide an algorithm for finding the best sequence. Argue about the running time of your algorithm.

Answer

IDK

B. One of the teams found out that using chemical i_1 to produce chemical i_2, then chemical i_3, ..., then chemical i_k and back to i_1 results in a bigger quantity of chemical i_1 than the original one, i.e., $R_{i_1,i_2} \cdot R_{i_2,i_3} \cdot \ldots \cdot R_{i_k,i_1} \geq 1$. Is this a problem in the ratio matrix or the students did something wrong in their calculations? Provide an efficient algorithm that can detect if such an impossible sequence of chemical reactions seems feasible given the $n \times n$ production ratio matrix. Argue about the running time of your algorithm.

Answer

IDK

Problem 3: [25 points] Consider a courier company with a big network of transfer stations and trucks. You are responsible for coordinating the routing of packages through the network. You have access to and the database indicating the departure/arrival locations and times for m trucks among n transfer locations.

You have to design an algorithm that receives a package's origin and destination location (the location always corresponds to one of the transfer stations), then output the list of trucks that can allow the package to arrive at the destination as early as possible. Note that in every intermediate transfer station, it takes about an hour for a package to move from a truck to another. You need to take this time into account.

Give an algorithm as efficient as possible to solve this problem, argue its running time, and prove its correctness.

Answer

- First we say that C is the cost to get from one transfer station to the other C_{ij} is the cost to get to location j from location i
- Ri is the route for truck i
- Assuming that each truck has one route
- If C_{ij} for all locations are the same then the problem is said to be symmetric, and we can simply just have an edge set.
- With each transfer station containing a quantity qi of some goods to be delivered. Then the problem is just determining the set of m truck routes of minimal cost, ending and starting at a depot.
- Since it takes an hour to transfer we can say our transfer cost or $\theta = 1hr$
- Therefore we can say that the cost of one route will take : $R_i = \sum_{i=0}^{m} C_{ij} + 1 + \sum_{i=1}^{m} \theta_i$
- Therefore the total cost of the routes will be $C_{VRP} = \sum_{i=1}^{m} C(R_i)$

Problem 4: [35 points] The clash of kings has come to an end in the land of Westeros and the winds of winter have passed... The Grand Maester has proposed a series of jousting tournaments to celebrate the events. These tournaments will be used to decide who are the finest knights in the Seven Kingdoms that should be selected as new members of the Kingsguard.

Every town in Westeros wants to host such a tournament. Even Pike at the Iron Islands wants to host a tournament, although the ironborn are not known to hold any love for the peoples of the mainland...

This posed a challenge for the Grand Maester. How should he rank the knights given a large number of tournaments and different outcomes at each tournament? He asked the advice of the

Citadel and selected the following strategy.

Each local Maester will be responsible to report a ranking of the knights at each individual tournament. Then, it is possible to define a preference score between two knights:

$$d(A,B) = \{ \text{ the number of tournaments knight A was ranked higher than knight B} \}$$

Obviously, if $d(A,B) > d(B,A)$, then knight A is preferred as a member of the Kingsguard over candidate B.

Given this preference score, however, it may not always be possible to select a global ranking for all the knights. For instance, Jamie Lannister may have ranked higher than Barristan Selmy in more tournaments, and Barristan Selmy may have ranked higher than Loras Tyrell in more tournaments but Loras Tyrell may still have ranked higher than Jamie Lannister in more tournaments. For instance, consider the following outcomes for the three knights in three tournaments:

	King's Landing Tourney	Winterfell Tourney	Casterly Rock Tourney
Jamie (J)	1^{st}	2^{nd}	3^{rd}
Barristan (B)	2^{nd}	3^{rd}	1^{st}
Loras (L)	3^{rd}	1^{st}	2^{nd}

In this case, $d(J,B) = d(B,L) = d(L,J) = 2$, and we have a rock/paper/scissors situation in terms of preference: Jamie > Barristan > Loras > Jamie.

To address such issues and provide a global ranking for the knights given their rankings in individual events, the idea of the Grand Maester and the Citadel is the following:

For each pair of knights A and B it is possible to identify a "preference sequence" $s[A,B]$ of knights that might imply that A is more powerful than B. Such a sequence of size m has the following properties:

- $s_1 = A$ and $s_m = B$
- $\forall\ i \in [1, m-1] : d[s_i, s_{i+1}] > d[s_{i+1}, s_i]$

The "strength of a preference sequence is defined as the minimum preference score along the sequence:

$$t(s[A,B]) = \min_{\forall i \in [1,m-1]} d[s_i, s_{i+1}].$$

The "strongest preference sequence" $s^*[A,B]$ is the preference sequence between the two knights A and B that has the maximum strength:

$$s^*[A,B] = argmax_{\forall s[A,B]} t(s[A,B]).$$

The strength of the strongest preference sequence between two candidates is denoted as $t^*[A,B]$. If there is no preference sequence between candidates A and B, then: $t^*[A,B] = 0$. It can be shown that t^* satisfies the transitive property (i.e., if $t^*[X,Y] > t^*[Y,X]$ and $t^*[Y,Z] > t^*[Z,Y]$, then $t^*[X,Z] > t^*[Z,X]$). Then it is possible to define a global preference function between two knights, as follows:

$$T(A,B) = \begin{cases} +1 & if\ t^*[A,B] > t^*[B,A] \\ -1 & if\ t^*[A,B] < t^*[B,A] \\ 0 & otherwise \end{cases}$$

Given the Grand Maester's approach, there is at least one knight X that has better or equal global preference versus all other knights Y, i.e., $\forall\ Y : T(X,Y) = 1$ or $T(X,Y) = 0$. Then knight X is the "global winner" of all the tournaments and becomes the Lord Commander of the Kingsguard. In order to generate the global ranking, the approach of the Grand Maester is to compute the

strength of the strongest preference sequence for every pair of knights and rank the knights by finding first the Lord Commander, then removing him from the list, finding the next winner, etc.

A) The Grand Maester argues that he has an efficient algorithm for computing the reliability of the strongest preference sequences for every pair of knights. What do you believe is this efficient algorithm? In this process, describe the underlying data structure needed to compute these values. What is the running time of the efficient solution?

Answer

IDK

B) Assume that there are 5 knights that participated in the tournaments: Barristan Selmy, Sandor Clegane, Loras Tyrell, Brienne of Tarth and Jamie Lannister. There were 45 different tournaments that took place:

In 8 tournaments the ranking was:
Jamie Lannister > Brienne of Tarth > Sandor Clegane > Loras Tyrell > Barristan Selmy

In 5 tournaments the ranking was:
Loras Tyrell > Barristan Selmy > Jamie Lannister > Brienne of Tarth > Sandor Clegane

In 7 tournaments the ranking was:
Sandor Clegane > Barristan Selmy > Brienne of Tarth > Jamie Lannister > Loras Tyrell

In 2 tournaments the ranking was:
Barristan Selmy > Jamie Lannister > Loras Tyrell > Sandor Clegane > Brienne of Tarth

In 8 tournaments the ranking was:
Brienne of Tarth > Jamie Lannister > Loras Tyrell > Sandor Clegane > Barristan Selmy

In 5 tournaments the ranking was:
Loras Tyrell > Sandor Clegane > Brienne of Tarth > Barristan Selmy > Jamie Lannister

In 7 tournaments the ranking was:
Barristan Selmy > Loras Tyrell > Brienne of Tarth > Jamie Lannister > Sandor Clegane

In 3 tournaments the ranking was:
Barristan Selmy > Loras Tyrell > Jamie Lannister > Brienne of Tarth > Sandor Clegane

i) Compute all the preference values. Does an undisputed winner arise in this case and why yes or no?

ii) Compute the strengths of the strongest preference sequences for all pairs of knights with the efficient algorithm of the Grand Maester.

iii) Provide the final ranking according to the strengths of the strongest preference sequences.

Answer

IDK