Comp 305 Spring 2021 | Algorithms and Complexity | Final Project Groups 07-13

1 Introduction

In this document, 7 different projects are given for you. Each of the groups is responsible of selecting one project among these as their course project. **DO NOT FORGET THAT THIS SELECTION IS FIRST-COME-FIRST-SERVED**. In other words, once a project is selected and the group information is written to the Google Sheet Link below, no other groups can select the same project. You can use any of the programming languages you prefer, there is no limitation. However, usage of complex libraries are prohibited. Please request permission if you plan to use any of these (if you are using Python, NumPy is allowed).

Please open the Google Sheet Link, and enter your group number, names of the members and the emails of the members to the cell, where your project is listed. Please keep in mind that the sheet document is divided into different sub-sheets. Thus, select the correct sub-sheet with respect to your group number. All of the test cases for each of the projects can be viewed and downloaded from this Google Drive Link.

Please open a GitHub public repository, include all of your members as contributors and add the repository link to the given Google Sheet document. This step is quite important for us to see your progress and has to be done quickly. Therefore, in the README file please keep a list of completed steps, a TO DO list and the results retrieved if there are any. You will also prepare a presentation and present to the TAs. Therefore, you should also create a Google Slides presentation and include its link to the Google Sheet document.

Please email to *comp305staff-group@ku.edu.tr*, if there is any problem in viewing the drive folder or modifying the document or if you have some troubles with any of the test cases.

Note: If any of these additional steps are not done at most 3 days after assigning yourselves into a project, then your group will be removed from that project and the project will be again available to others.

2 Presentation Details

Each of the presentations should take ∼10 minutes and there will be a 5-minute Q&A session afterwards. If a presentation lasts longer than 10 minutes, then it will be interrupted. During the presentation each of the groups should explain and report:

- The algorithm you designed to solve the problem, the choices of the data structures you used and your reasoning.
- The time complexity of your algorithm (and the space complexity if applicable).
- Your run times for each of the test cases.
- Further improvements that can be done as future works.

This project does not expect from you to come up with just one solution and then test only that solution. For each of the problems you can start with some baseline approaches with more complexity and improve the baseline algorithm step by step. Be as creative as possible. Report different approaches you tested and why did you decide on the final algorithm you present. Your grading will be based on your creativity, your cumulative progress and how well did you present your approach.

3 Deadlines

You can work on your project until the end of ***23th May, 2021***. The project presentations will be held between ***24th-28th of May, 2021***.

In the following pages, you can see the available project(s):

Recursive Lempel-Ziv–Welch Algorithm

The Lempel-Ziv–Welch (LZW) is a data compression algorithm, which is the basis of some of the most popular compressing techniques of today, such as gzip and zip. Given a sequence of characters, the algorithm maps a set of input characters into codes. Moreover, it can reconstruct the original document without any data loss without storing the mapping information. The algorithm has 2 main parts, *Encoder and Decoder respectively.*

Encoder

The Encoder operates with these following steps:

1. To begin with, all characters that may occur in the text file (that is, the alphabet) are assigned a code. Since we will work with only ASCII characters in this project, all of the ASCII characters should be assigned to a code at the initial step.
2. Each of these assignments should be stored in a dictionary, where the key is the subset of input characters (the ASCII characters at the beginning) and the values are the codes assigned to these characters.
3. Beginning with the dictionary initialized as above, the LZW compressor repeatedly finds the longest prefix, p, of the unencoded part of the input file that is in the dictionary and outputs its code.
4. At each step after finding the longest prefix, if there is a next character after this string and concatenation of the prefix and the next character is not in the dictionary, a code is assigned to this concatenated subset and this pair as added to the dictionary.

The following example demonstrates the encoding process of the LZW algorithm. Assume that we have an input string: ***"aaabbbbbbaabaaba"***.

- **Step 1:** Create a dictionary having all ASCII characters as keys and numbers from 0 to 255 as their codes. So, the character "a" has the code 97 and character "b" has the code 98.
- **Step 2:** Starting from the first character find the longest prefix, which is "a". Then print its code: 97.
- **Step 3:** Then combine the next character with the longest prefix, assign a code to this new subset and add to the dictionary. So, the concatenation gives "aa" and it is added to the dictionary with the code 256.
- **Step 4:** Continue to read the input text from the second character and again find the longest prefix, which is "aa". Then print its code: 256.
- **Step 5:** Get the next character and add the concatenation to the dictionary. So, add "aab" to the dictionary with the code 257.
- **Step 6:** Read starting from the first unencoded character, which is "b" (the 4th character). Get the longest prefix, which is again "b" and print its code: 98.
- **Step 7:** Concatenate the longest prefix with the next character and add to the dictionary. So, add "bb" to the dictionary with the code 258.
- ...
- **Last Step:** Since all the characters in the input are read and their codes are printed, the compressed version of this input sequence becomes: *"97 256 98 258 259 257 261"*.

A pseudo code for this process can be viewed below:

Algorithm 1: LZW Encoder

```
Result: Compressed string SEQ
Initialize the dictionary;
SEQ = empty string;
P = first input character;
while not the end of file do
    C = next input character;
    if P + C is in dictionary then
        P = P + C;
    else
        SEQ = SEQ + P;
        add P + C to the dictionary;
        P = C;
    end
end
```

Decoder

The decompression algorithm proceeds as follows:

1. As we did in the compression, the dictionary is initialized with the ASCII codes and their corresponding characters, but in this case the codes are the keys and the characters are the values.
2. We read each of the codes one-by-one and we will either have the code we read in the dictionary or not. For these 2 cases, the following actions below should be taken:
    - **When the code x is ALREADY in the dictionary:** The corresponding text, text(x), is extracted from the dictionary and output. Also, from the working of the compressor, we know that if the code that precedes x in the compressed file is q and text(q) is the corresponding text, then the compressor would have created a new code for the text(q) followed by the first character (that we will denote by fc(x)), of text(x) (Understand why this is the case!) So, we enter the pair (next code, text(q) fc(p))into the dictionary.
    - **When the code x is NOT in the dictionary:** This case arises only when the current text segment has the form "text(q)text(q)fc(q)" and "text(x)" = "text(q)fc(q)" (Understand why this is the case!) The corresponding compressed file segment is "q x".During compression, "text(q)fc(q)" is assigned the code x, and the code x is output for the text "text(q)fc(q)". During decompression, after q is replaced by text(q), we encounter the code x. However, there is no code-to-text mapping for x in our table. We are able to decode x using the fact that this situation arises only when the decompressed text segment is "text(q)text(q)fc(q)". When we encounter a code x for which the code-to-text mapping is undefined, the code-to-text mapping for x is "text(q)fc(q)", where q is the code that precedes x in the file.

A pseudo code for this process can be viewed below:

Algorithm 2: LZW Decoder

```
Result: Original string SEQ
Initialize the dictionary;
OLD = first input code;
SEQ = empty string;
while not the end of file do
    X = next input code;
    if X is in dictionary then
        SEQ = SEQ + string for X;
        C = first character of the string for X;
        P = string for OLD;
        P + C is added to the dictionary;
    else
        P = string of OLD;
        C = first character of P;
        add P + C to dictionary with the code X;
        SEQ = SEQ + P + C;
    end
    OLD = X;
end
```

(a) Implement the basic LZW structure, measure its runtime and the compressed file size for each of the sample files provided.

(b) In the given example, we have used the numbers with the base 10 for the character code representations. However, different bases or different representations can be selected to run this algorithm. Extend your implementation to support different coding representations and report the runtime and the compressed file size for each of the sample files for 3 other coding representations. Be creative as much as possible to improve your compressed file size. The grading will also be based on your creativity, not only by checking if your algorithm is working without any error.

(c) Can we decrease the size of the compressed file if we recursively encode the original file (encode the encoded file again)? Extend your implementation to support recursive encoding and decoding calls (brute-force calls one after the other is not allowed) and report the time and file size results for base 10 coding representation and the best representation you have found in part b. **Do not forget that after the first encoding, number of characters in the input file will be limited only with the number of characters you used in your coding representation. If your representation is base 10 representation, then the encoded file will consist of digit and space, so no need to keep all the ASCII characters in your dictionary during the recursive encoding and decoding processes.**

Whom Should I Be Friends With?

Barış is a teenager, who is using social media very actively and tries to be close friends with the most popular guys which are added to his connections (like friends list). In order to build some friendship with the right person, he decides to design an algorithm which finds the most popular person as follows:

1. He extracts all of his connections.
2. He also finds all of the connections that the persons on his connections list have.
3. He filters the overall information such that only both sides of the all connections consists of the persons from his personal connections.
4. Lastly, he finds the shortest paths between each pair of his friends and decides that the most popular person in his connection list is the person who has the most occurrence in these shortest paths.

Please note that if a person A is connected to a person B, then B is also connected to A. Each connection works in both ways (bi-directional).

(a) Given the data with already processed with the first 3 steps, implement Barış's algorithm and find the most popular person's id for each of the test cases provided. Also analyze the time and space complexity of your algorithm and report the overall running time.

(b) What other algorithms can be used for selecting the most popular person? Propose different algorithms that run with the same data.

Note: No libraries are allowed to use other than standard libraries of your preferred language and most basic libraries such as NumPy in Python. For faster computation time, languages like C++ are suggested but you are free to use any language you want. Please ask for a permission if you plan to use any library that do not apply to this condition.

Input Format. The first line includes number of people V in Barış's friends list and the second line indicates total number of connections E between people in Barış's friends list. The next E lines have 2 numbers and each of the distinct numbers indicate one of Barış's connections. For instance if a line is *"2 5"*, then it means that Barış's friend with the ID 2 is connected with Barış's friend with the ID 5 and vice versa.

Best Place for the New Hospitals

Although 1 and a half year is passed since the beginning of the pandemic, the number of cases are still increasing in the Gotham City. The current hospitals in the city are too small and not sufficient to handle this amount of patients. Therefore, the governor decides to build 2 really huge hospitals that will only take care of the patients of the pandemic disease. Since time really matters for early treatment, it is important to decide the most suitable place in the city, where the total distance from patients districts to the hospitals are minimum.

In this assignment, you are given:

- the number of districts and their ids,
- the number of persons living at each of these districts,
- the distances between the districts if there is a direct connection between them.

The hospitals will be built at the selected districts and the distance with the hospitals will be 0 for people that live in these districts. Every person will only go to the hospital that is nearest to him-/herself. A figure below is a demonstration of a simple case:

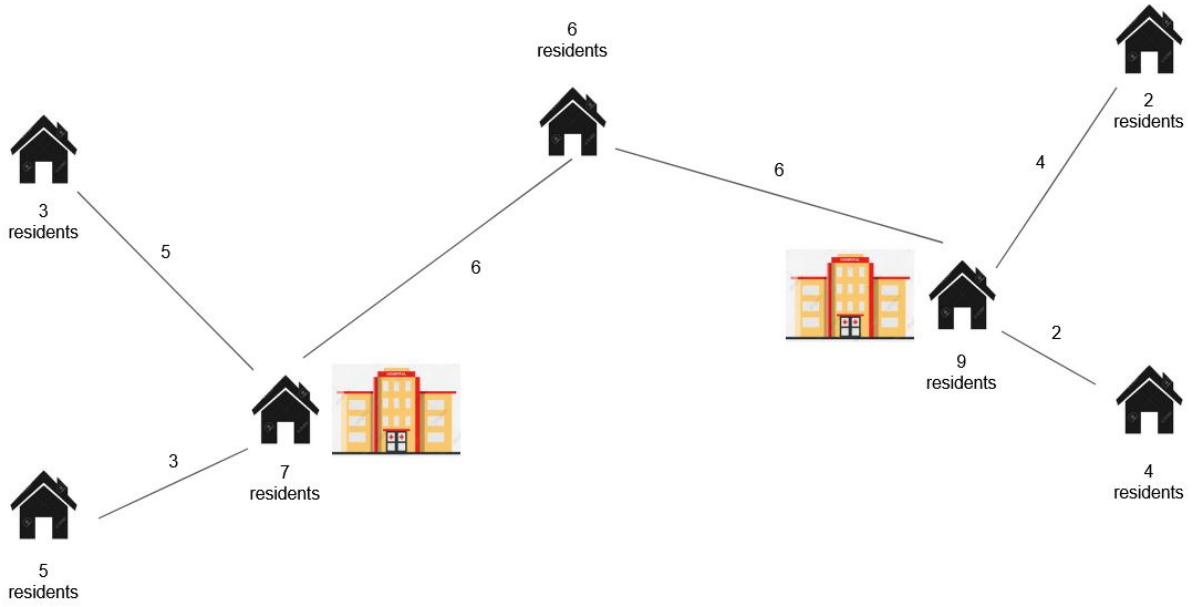

Figure 1: A sample city and hospital demonstration

If a district is equally distant to each of the hospitals, then you can select either one of them.

(a) Given this information, design an algorithm to find a good place for these hospitals. Does greedy algorithm give the optimal result? Which techniques you can apply to decrease the complexity and processing time?

Input Format. The first line includes number of districts V and the second line indicates total number of connections E between these districts. The next V lines have 2 numbers indicating number of people living in that district. The format is: *"district_id number_of_people"*. The next E lines indicate the districts that are connected with each other directly and their distances. The format is: *"district_1_id district_2_id distance"*.

Which Course Center Should I Select?

Barış really likes to listen Turkish folk music. He has a collection of CDs and records belonging to most popular artists from this genre. However, the more he becomes into this genre, the less satisfied he becomes by just listening to these songs. Therefore, he makes a major decision, buys himself a Bağlama and searches a suitable course for him to learn this instrument.

Although learning to play an instrument is a fun hobby, it also brings some financial costs with itself. Since Barış's financial status is not too good, he can only select a course that has a fee less than or equal to ***max_fee***. Moreover, the courses with a really low amount of fee may not teach him the instrument too well. Thus, he also decides to sign up to a course, that has a monthly fee greater than or equal to ***min_fee***. Additionally, he does not want the course center to be really far. Hence, he only considers the course centers that have the distance less than or equal to ***min_dist***.

Assume that there is a set of course centers S, that fulfill all of the requirements described above. However, if one of the courses S_1 is both closer and to Barış and cheaper compared to another course S_2, there is no need for Barış to consider S_2 anymore, since there is a course center that is better in each of the aspects compared to S_2.

(a) Given the thresholds ***max_fee***, ***min_fee***, ***min_dist***, and a list of course centers with their distances to Barış and their fees, can you find how many courses are there that Barış should consider to register?

Note: Purely brute-force solutions will receive poor grades. You must show your work, your steps to solve and to improve the problem, your reasoning for the variety of the data structures you used, etc.

Input Format. The first line contains an integer N, indicating the number of course centers. The second line includes ***min_dist***, ***min_fee*** and ***max_fee*** values respectively. Each of the next N many lines include the distances and fees of the course centers.

Neighbors of Our Kingdom

In an imaginary world, there are N many cities governed by K many countries. Some of the cities have direct connection between each other and some do not. Moreover, territorial integrity is not an important issue for the countries.

A person named *Luke*, who lives in one of the cities, is not happy with any of these regimes in the whole world. He/She plans to organize a group of people in arbitrary M many cities (each of the pairs in these cities can be reached by only using the paths between these M many cities, so it has territorial integrity), to start a riot and to found his/her own kingdom with these M cities included. The problem is that the more number of distinct neighbors (number if countries that are connected to the M many cities) that the new kingdom have, the more threats are present for the new country. Thus, Luke wants the number of distinct neighbors as small as possible.

(a) Given the cities, their direct connections (if there are any), their country information, and the number M can you find the optimal set of cities that this new kingdom should be founded? How many number of distinct neighbors does this city have?

Input Format. The first line includes number of cities N and the second line indicates total number of connections E between these cities. The third line contains the number M. The next V lines have 2 numbers indicating the city id and the kingdom id which rules the city. The format is: *"city_id kingdom_id"*. The next E lines indicate the cities that are connected with each other directly. The format is: *"city_1_id city_2_id"*.

A Game Analysis with the Holmes Family

It was the summer of early 1880s and the three *Holmes* children *Sherlock, Mycroft and Enola* were all very little. Since it is the time of summer holiday, there was nothing to do. Suddenly, Sherlock thought of a game and he started to play it with Enola.

First, Sherlock gathered N many saucers and aligned them as a horizontal line. Then he collected N many pieces of paper and wrote all the numbers from 1 to N. Then he explained the game to her with exactly these words: "Look Enola, at each turn you will take one of the pieces of paper and put it on one of the saucers. Then I will do the same thing with the remaining pieces of the paper. If you can put 2 consecutive number on 2 adjacent saucers, you will win. If I can fill all the saucers before you win, then I win." For instance, if there are 5 saucers and 5 numbers, [3, 2, 1, 5, 4] or [1, 2, 3, 4, 5] is a winning state for Enola but [1, 3, 5, 4, 2] is a winning state for Sherlock.

Mycroft, the oldest sibling, was also listening to them and he was also bored. So, he decides to watch them while they play. However, it is easy to see that Enola always starts the game with the winning state if she plays optimally. But she is so little that she cannot understand the game well and almost plays randomly. After a while, he is bored of just watching. Hence, he starts to analyze the games by counting number of mistakes both Enola and Sherlock did separately.

A mistake is done when the a player definitely wins with the optimal game before the player plays but after he/she plays, he/she definitely loses when the rest is played optimally.

As an example, the game below demonstrates the whole situation:

1. $[\_, \_, \_, \_, \_]$: all cells are empty, Enola wins if played optimally.
2. Enola plays $[\_, \_, 5, \_, \_]$: still Enola wins in the optimal case.
3. Sherlock plays $[1, \_, 5, \_, \_]$: still Enola wins in the optimal case.
4. Enola plays $[1, \_, 5, 2, \_]$: If Sherlock plays 4 to the rightmost saucer, then he wins. So, this is a mistake made by Enola!
5. Sherlock plays $[1, 4, 5, 2, \_]$: With this move, Sherlock makes Enola the winner. This is a mistake made by Sherlock!

As you can see, if one player gives the winning state to other when played optimally or if Sherlock makes Enola winner by his move, then they are defined as mistakes.

(a) For each of the given games with the moves until one of them is the winner, how many mistakes would Mycroft found? Design an algorithm to solve this problem. Keep in mind that N can be a really big number.

Input Format. The first line includes the number of rounds played in total (If Enola plays once and Sherlock plays once, then the number of rounds is 2), then each line include the actions taken at each round. The first integer is the chosen saucer number and the second integer is the number chosen to put in the selected saucer.

Lighting Edison's Workplace

Although Joseph Swan was the one who first proposed the light bulb in 1860, Edison improved its structure and created a usable light bulb in 1879. To advertise his product and dominate Joseph Swan during this age, Edison planned to lighten his own workplace. However, although electricity is found and used during this period, there were no plug sockets or separate electricity connection for each single room like today. Therefore, he had only one source of electricity and each of light bulbs has to be connected to that single source by cables. Moreover, he had a limited budget for this task. Hence he had to plan a cost-efficient way to lighten as much area as possible. Additionally, each light bulb could only illuminate L x L area, where L is an odd number and the light bulb is at the center of this area.

The workplace has these following structures:

- **Walls:** The light cannot go through a wall. The walls are defined with the character "#"
- **Areas Needed for Lighting:** Not all of the areas require a lighting. However, if a cell need to be lighten, then that cell is represented by the character ".".
- **Areas Not Need any Lighting:** These are shown with the character "-".

So, an example workplace is as follows:

```
- - - - - - - # # # # # -
- # # # # # - # . . . # -
- # . . . # - # . . . # -
- # . . . # - # # # # # -
- # . . . # - - - - - - -
- # . . . # - - - - - - -
- # . . . # - - - - - - -
- # . . . # - - - - - - -
- # # # # # - - - - - - -
- - - - - - - - - - - - -
```

If $L = 3$, the source of electricity is shown with the letter "E" and placed to the up-left-0-indexed coordinate (3, 1), the light bulbs are shown with "X", and the cables are shown with "1", then the optimal placement for the example above is given below:

```
- - - - - - - # # # # # -
- # # E 1 1 1 1 1 X . # -
- # . 1 . # - # . . . # -
- # . X . # - # # # # # -
- # . 1 . # - - - - - - -
- # . 1 . # - - - - - - -
- # . X . # - - - - - - -
- # . . . # - - - - - - -
- # # # # # - - - - - - -
- - - - - - - - - - - - -
```

(a) For each of the test cases, the whole workplace map, the cost of putting cable to a cell, the cost of placing a light bulb, the range L of each light bulb and the total budget is given. Giving this information, what is the maximum number of cells you can illuminate without spending less than the budget?

Input Format. In the first line, the number of rows, columns and the radius R of the illumination is given. To calculate L, you need to do $2R + 1$. In the second line, the cost of putting a cable into one cell, the cost of one light bulb and the budget we have is written. The following lines give the map of the area.