CS205: Design and Analysis of Algorithms **Spring 2022**

Assignment 2

IIT Dharwad *Computer Science and Engineering*

Instructions:

- You are expected to solve this assignment manually (i.e., not with the help of a computer) and submit the relevant files in the required format. Solving this manually MAY help you in the end-semester examination.
- There are 2 questions in this document. Each carries 5 marks.
- Due on 16th March. With 25% penalty, you can submit up to 17th March.
- There will be a penalty of 2% if you deviate from the naming scheme for any question - read the questions for more details.
- Do not zip the two answer files. Submit them as individual files in Moodle.

Q1. You are given with a flow network. The task is to find the maximum flow in it.

Input data

The input is given in a file named <roll-no.>-mf-data.txt, where <roll-no.> is your roll no. - See flow-networks.zip. The input file specifies a flow network with ten vertices, where the vertex with label `s` is the source and the vertex with label `t` is the sink. The first line of the file contains two numbers - the number of vertices and the number of edges in the input flow network. The rest of the lines give the details of the weighted directed edges. Each of such line is of the form `u v w`, which specifies a directed arc from `u` to `v` with weight `w`.

Answer

The answer should be written in a file named <roll-no.>-mf-answer.txt, where <roll-no.> is your roll no. Each line of the answer file must specify an augmenting path. The format of such a line is `s .. .. .. t val`, where the list of vertices in the augmenting path is specified by starting with `s` and ending with `t` (blank space must be there between two vertex labels). The rest of the vertices in the augmenting path must be written between `s` and `t` in order. The last entry (`val`) is a number which specifies the value of the flow that can be augmented using this augmenting path. In short, the sum of last numbers in all lines must give the maximum flow value - you don't have to specify the sum. There is no further constraints on the augmenting paths - at each step you can come up with any augmenting path of your choice and update the answer-file. Some examples with eight vertices are attached (see mf-sample-input-answer.zip).

Q2.

"It has been a long semester, a really long one! There is one more week of intense evaluations. Then I can relax!", Adir murmured. But, then he thought: "Why should I wait to relax? Why can't I just go outside and get some fresh air?" Putting on his face-mask, he walked to the nearby park. The nth wave of the pandemic had deserted the park. There were none. Fortunately, the gate was open. Adir went inside. He laid down on the grass, which was covered with white champaca petals. It seemed to him that the pandemic had imparted a serene look to the park. There were everyone except the most intelligent mammals. A pair of red-whiskered bulbul sat on the white champaca tree. Adir always liked them - the song of the bulbul and the frangrance of the champaca. Adir had never written a poem in his life. But, suddenly, he had an urge to write something - something very short and nice! Something very musical and fragrant! "If only I could get a pen and a paper!", he wished. Suddenly, from nowhere, a small piece of paper came flying. Slowly, it landed on his side. He took it and saw something interesting. On one side of the paper, there was a sequence of zeros and ones and on the other side, there was numbers assigned to some letters in the English alphabet. Since Adir had learned Huffman coding, he could easily relate it to the same. "What could be this string of zeros and ones? -- a message? or a poem? Alas! there is no pen to do the calculations." But his intense wish showed him a platform neatly covered with sand. He went to that corner and worked his calculations on the sand. It was very refreshing for him to translate the zeros and ones to a string of beauty. Repeating what he just read, he walked back to his home.

Put yourselves in the shoes of Adir and find out what he read. You will be given with the string of zeros and ones and the frequency of each letter in the original string. Please note that only small letters and white spaces appear in the string. You can think the process of obtaining the binary code for characters as a binary tree where the vertices are labelled with a string (known as label) and a number, which corresponds to the frequency. To make sure that you decode the encoded string correctly, please follow the following rules:

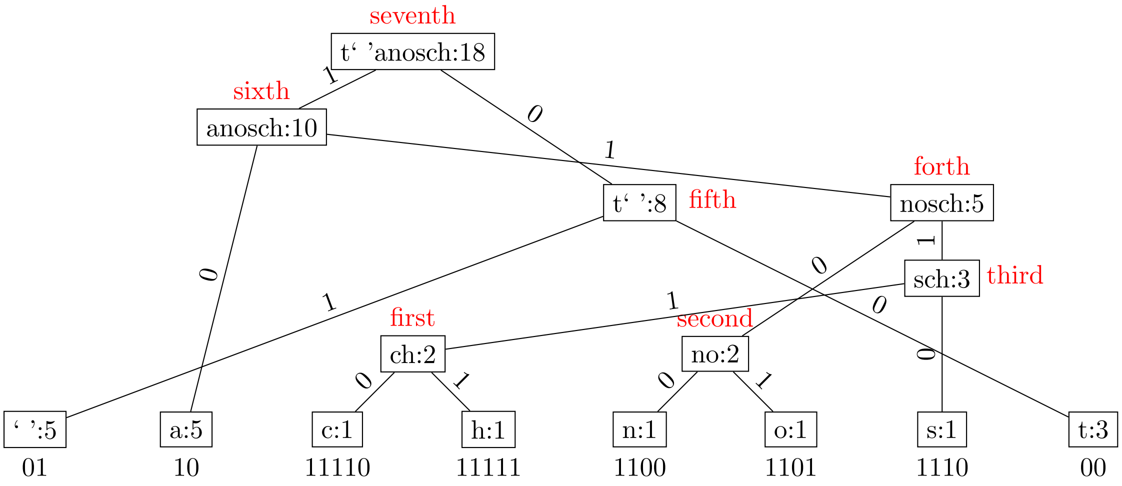
- Whenever you combine two vertices, one with label string X (having frequency f_x) and another with label string Y (with frequency f_y), then the combined vertex will have label string XY if $f_x < f_y$ or ($f_x = f_y$ and X is lexicographically smaller than Y). In that case, the code of the edge from X to XY will be 0 and that of the edge from Y to XY will be 1. Please note that ‘ ’ (white space) is lexicographically smaller than a, and a is lexicographically smaller than b, and so on. Also note that X is lexicographically smaller than Y if the first letter of X is lexicographically smaller than Y.
- As we learned, at every step you choose two vertices of two lowest frequencies and merge them. Assume that there is a tie between two vertices, i.e., with two labels X and Y having the same frequency, then the preference is given to the string which is lexicographically smaller than the other.

Input data
The name of the input data file is <roll-no>-hc-data.txt, where <roll-no> is your roll no. - See huffman-coding-input.zip. An example file is given below.

```
1001111101000011110100001110111000110011111111000
‘ ’:5
a:5
c:1
h:1
n:1
o:1
s:1
t:3
```

The first line of the data file shows the binary string that you have to decode and the rest of the lines specifies the frequencies of the characters.

Answer
Let us try to solve the above sample question. If you work on this data with the rules given above, you will obtain the binary tree and code for each character as shown below. The order in which merging happens is shown in red.



Now, let us decode the binary string.

$\underset{a}{\underline{10}}\ \underset{‘\ ’}{\underline{01}}\ \underset{c}{\underline{11110}}\ \underset{a}{\underline{10}}\ \underset{t}{\underline{00}}\ \underset{‘\ ’}{\underline{01}}\ \underset{s}{\underline{1110}}\ \underset{a}{\underline{10}}\ \underset{t}{\underline{00}}\ \underset{‘\ ’}{\underline{01}}\ \underset{o}{\underline{1101}}\ \underset{n}{\underline{1100}}\ \underset{‘\ ’}{\underline{01}}\ \underset{a}{\underline{10}}\ \underset{‘\ ’}{\underline{01}}\ \underset{h}{\underline{11111}}\ \underset{a}{\underline{10}}\ \underset{t}{\underline{00}}$

I.e., the binary string translates to “`a cat sat on a hat`” (without quotes).

The format of the answer file is described below.

- The answer file must be named <roll-no.>-hc-answer.txt, where <roll-no.> is your roll no.
- The file must contain exactly n lines, where n is the number of unique characters in the original message. The first $n-1$ lines corresponds to the $n-1$ merges in order (i.e., first line corresponds to first merge and so on) that happen while solving the problem. The last line is the decoded string.
- The first $n-1$ line will have the format `X:freq`, where `X` is the label of the new vertex obtained as the parent of two vertices with least frequencies, and `freq` is the sum of frequencies of the child vertices being merged.

The answer file corresponding to the example is given below. Two more sample files are attached (see huffman-coding-sample.zip).

```
ch:2
no:2
sch:3
nosch:5
t‘ ’:8
anosch:10
t‘ ’anosch:18
a cat sat on a hat
```