ECE368 folks,

Here are some clarifications of project 3 that address most of the questions I got from you:

1. About Lab and shortest path

   Many of you asked questions regarding the semantic understanding of Lab. According to the definition of Lab, Lab is more like a similarity metric of the users, i.e. larger Lab indicates closer relationship between the users. Therefore it makes little sense to compute the shortest distance using Lab as the weight of edges. However, the purpose of this project is to test your understanding of the graph algorithms learned from the class. Once the graph is created, the problem is totally separated from the application scenario. Users are just vertices and Labs are just used to specify the connection and weights of edges. No need to pursue the applicability of the queries in real-world circumstances. Just implement the algorithm on the graph you obtain by exactly following the steps in the document.

2. About IDs

   The IDs (first column of the user profile) are all in increasing order from 1 to the maximum number of users with step 1. In Query 1, 3, 4, you're asked to print a list of sorted Node IDs. That is to say, if you found multiple nodes meeting the requirement of the query, you should print them in sorted order with respect to their IDs. For example, in Query 3, if you found that three nodes 2,4,5 are directly connected to the given source node, you should print

   3,2,4,5

   Rather than

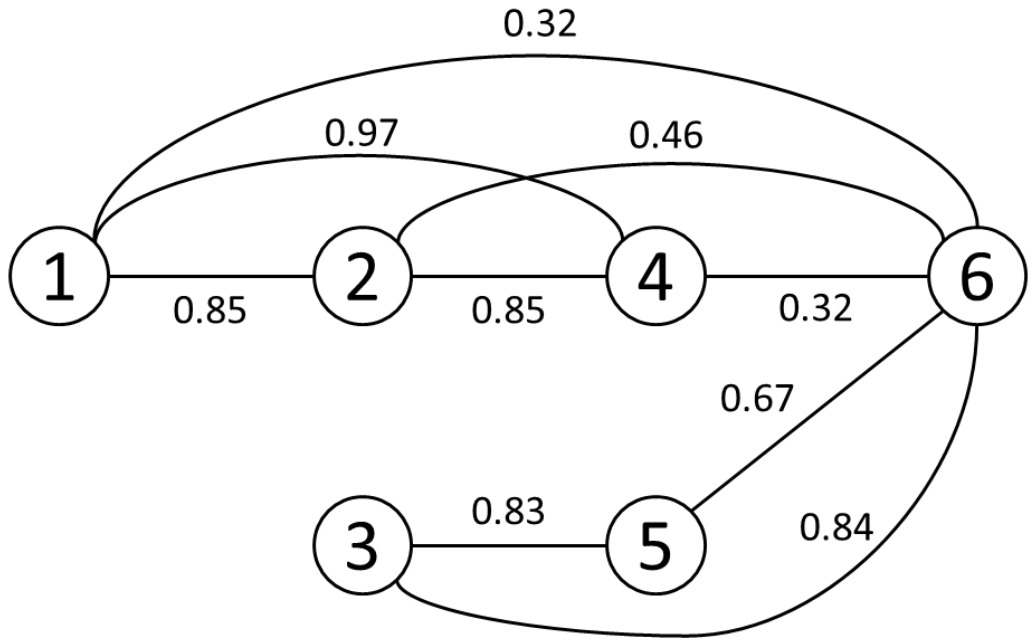
   3,5,2,4 or 3,4,2,5

3. Data Precision and truncation

   The user attributes you read from the input file are all integers. As is mentioned in the document, you're asked to truncate Lab to two decimal points after it is computed from ULab. Many of you don't get what that really means. Correct way to do it is to multiply the double/float number by 100 and convert it to integer: Lab_int = (int)(Lab_double*100) without converting it back to double/float. Here Lab_int is a Q100 representation of the truncated Lab. Lab_int is in the range from 0 to 100, where 0 corresponds to 0 and 100 corresponds to 1. Q100 notation represents fractional number by an integer obtained by multiplying the fractional number by 100 and truncating it to the nearest integer. For example,

Lab_int =32 stands for 0.32, and Lab_int=64 stands for 0.64 in the original fractional number representation. Thus, in the final graph your representation of the weights should be INTEGERS, not double or float, although conceptually they are fractional numbers. Of course, in order to compare the weights with the input arguments you will need to use Q100 representation for delta1, delta2 and alpha as well, although they're read in as double/float number. So, for delta1_double=0.3, delta2_double=0.5, alpha=1.5. You're really representing them as delta1_int=30, delta2_int=50, alpha_int=150 in your program.

There are several mistakes in the example graph shown in the document. If computed correctly, your graph should be like this:



4. 2-hop away neighbors

   2-hop away neighbors should INCLUDE all vertices that are reachable from the source vertex in exactly 2 hops, including those that are also reachable in 1 hop. For example, in the above graph, all 2,3,4,5,6 are counted as 2-hop away neighbors of 1.

5. Sample Input and Output

   Sample input will be provided with this document. Sample output will be provided later. One thing that's different from the previous clarification is that, between two numbers the delimiter is only a comma, with NO SPACE.