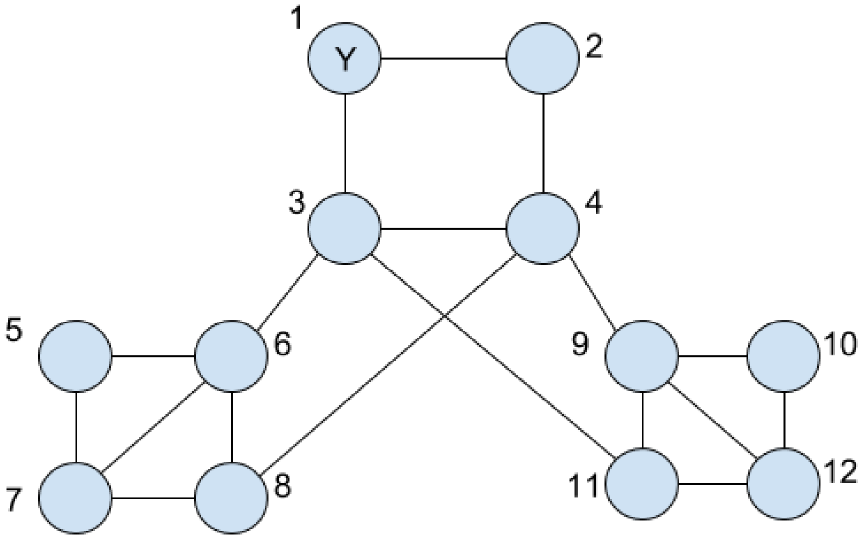
Big Data Analysis (Practice Final Exam)

Question 1

The parliament has organized a voting scheme for a new bill this summer. You are a strategic advisor in charge of vote forecasting and voter acquisition tactics. You have the following social graph of voters, which is undirected.



The undecided voters will go through a 3-day decision period where they choose a candidate based on the majority of their friends. The decision period works as follows:

1. The graphs are initialized with every voter's initial state as the above figure. (yes (Y), no (N), or undecided)
2. In each day, every undecided voter decides on a vote 'yes' or 'no'. Voters are processed in an increasing order of node ID. For every undecided voter, if the majority of their friends (>=50%) vote 'yes', they now vote 'yes'. Otherwise, they vote 'no'.
3. When processing the updates, use the values from the current day. For example, when update the votes for node 2, you should use the updated votes for nodes 1 and 4 from the current day.
4. There are 3 days of the process described above.
5. On the 4th day, the votes are counted.

a) Perform iterations of the voting process. How many votes each option has?

Hints:

The winning result is 'no'.
Yes: 2 votes (node 1 and 2)
No: 10 votes (remaining nodes)

b) You have a public relation idea to increase the ‘yes’ voters by organizing a very classy $1000 per plate dinner event. Assume everyone that comes to your dinner is instantly persuaded to vote ‘yes’ regardless of his/her previous decision. This event will happen before the decision period.

Choose a minimum number of voters to invite for dinners such that all the voters in the graph vote ‘yes’. Justify your strategy and compute the voting result.

Hints:

The minimum number of voters to invite is 2. We can invite node 6, and node 9.
+ After inviting node 6: ‘yes’ voters will include node 1, 2, 3, 4, 5, 6, 8
+ After inviting node 9: all nodes will vote ‘yes’

c) You have another idea to increase the ‘yes’ voters by spending $1000 to make any two voters in the network become friends.

Choose a minimum number of connections you want to create such that all the voters in the graph vote ‘yes’. Justify your strategy and compute the voting result.

Hints:

The is no exisiting solution.

Question 2

Schema reuse is a new trend in creating database schemas by allowing users to copy and adapt existing ones. The motivation behind schema reuse is the slight differences between schemas in the same domain; thus making schema design more efficient. Reusing existing schemas supports reducing not only the effort of creating a new schema but also the heterogeneity between schemas.

Finding related schemas is one of the core problems of schema reuse. You work as a data engineer at Oracle. Oracle has a large repository of schemas. Each database schema has a set of attributes. Some attributes are common among schemas, while others are not. Your task is to support database designers to create new schemas via the schema reuse paradigm. For instance, when a database designer wants to create a new schema, he wants to query the schema repository for references:

- He can start with a few attributes and query the schema repository for hints to finish his design.
- Alternatively, he can complete a schema and query the schema repository to check his design.

Example: we have a repository of schemas:

- *S1: {a1, a3, a7}*
- *S2: {a1, a4, a8}*
- *S3: {a2, a6, a9}*
- *S4: {a1, a5, a10}*

Given a query Q = {a1, a2}, we should rank these schemas as S3 > S1=S2=S4. S3 has the highest rank since attribute a1 occurs frequently in many schemas and thus has less discriminatory power (i.e. the more schemas contain an attribute, less information it provides).

Design an algorithm to find related schemas ranked by their similarity to the query.

a) How do you model the problem (input, output, etc.)? Justify your model.

Hints:

Input: A schema repository D={S1,S2,…} and a query schema Q
Output: a ranking score f: D→[0,1]
Model: TD-IDF. We consider each attribute as a term.

b) What steps should be involved? Provide a quantitative measure for each step if needed. Justify the design choice for each step.

Hints:

Step 1. Build a term vocabulary of the current schema repository

Step 2. Compute term frequency and normalize term frequency by tf-idf weights

Step 3. Transform all schemas and the query into a vector of term frequencies

Step 4. Compute the similarity between the query and each schema by cosine similarity.

c) Apply your approach to the above example and calculate the quantitative results.

Solution:

Term vocabulary: {a1, a2, a3, a4, a5, a6, a7, a8, a9, a10}

Document-term matrix:

	S1	S2	S3	S4
a1	1	1	0	1
a2	0	0	1	0
a3	1	0	0	0
a4	0	1	0	0
a5	0	0	0	1
a6	0	0	1	0
a7	1	0	0	0
a8	0	1	0	0
a9	0	0	1	0
a10	0	0	0	1

TF-IDF normalization:

idf(a1,D) = ln(4/3) ~ 0.29

idf(a2,D) = idf(a3,D) = idf(a4,D) = idf(a5,D) = idf(a6,D) = idf(a7,D) = idf(a8,D) = idf(a9,D) = idf(a10,D) = ln(4/1) ~ 1.39

S1 = [0.29, 0, 1.39, 0, 0, 0, 1.39, 0, 0, 0]

S2 = [0.29, 0, 0, 1.39, 0, 0, 0, 1.39, 0, 0]

S3 = [0, 1.39, 0, 0, 0, 1.39, 0, 0, 1.39, 0]

S4 = [0.29, 0, 0, 0, 1.39, 0, 0, 0, 0, 1.39]

Q = [0.29, 1.39, 0, 0, 0, 0, 0, 0, 0, 0]

Cosine similarity:

Sim(Q,S1) = (0.29*0.29)/(sqrt(0.29^2 + 1.39^2 + 1.39^2) * sqrt(0.29^2 + 1.39^2)) ~ 0.029

Sim(Q,S2) = (0.29*0.29)/(sqrt(0.29^2 + 1.39^2 + 1.39^2) * sqrt(0.29^2 + 1.39^2)) ~ 0.029

Sim(Q,S3) = (1.39*1.39)/(sqrt(1.39^2 + 1.39^2 + 1.39^2) * sqrt(0.29^2 + 1.39^2)) ~ 0.565

Sim(Q,S4) = (0.29*0.29)/(sqrt(0.29^2 + 1.39^2 + 1.39^2) * sqrt(0.29^2 + 1.39^2)) ~ 0.029

Question 3

For the following problem describe how you would solve it using MapReduce. The input is a list of documents (ID, text). The output should be the count of each word over all documents. You are given only two machines.

ID	Text
1	*Peter Piper picked a peck of pickled peppers*
2	*A peck of pickled peppers Peter Piper picked*
3	*If Peter Piper picked a peck of pickled peppers*
4	*Where's the peck of pickled peppers Peter Piper picked*

1) You should explain how the input is mapped into (key, value) pairs by the map stage, i.e., specify what is the key and what is the associated value in each pair, and, if needed, how the key(s) and value(s) are computed. Then you should explain how the (key, value) pairs produced by the map stage are processed by the reduce stage to get the final answer(s). If the job cannot be done in a single MapReduce pass, describe how it would be structured into two or more map - reduce jobs with the output of the first job becoming input to the next one(s).

Hints:

Key, value mapping: word -> key; count of word -> value.
Map Stage:
+ Input: documents (e.g., "Peter Piper…").
+ Output: list of (word, count). For example: (Peter, 1), (Piper, 1).
Reduce Stage:
+ Input: list of (word, count)
+ Ouput: list of (word, totalCount)

2) If there are 5 documents, how do you distribute them into two machines? Explain your criteria for this distribution.

Hints:

We simply distribute two machines with an equal workload. An example could be assigning 2 longest documents to the first machine, and assigning the remaining documents to the second machine.