Question 1: Unstructured P2P Networks
Points: ___

Assume an unstructured P2P network with two types of resources: 1900 non-popular resources each of which is accessed with probability 10^{-4} and n_p popular resources each of which is accessed with a probability p_p. The non-popular resources have only a single copy, whereas the popular resources are found to have 9 copies each as a result of replicating them according to the square root rule. The network has $N=1000$ nodes.

a) Choose the correct answer for the number of popular resources (n_p) and justify your answer

☐ 10 ☐ 100 ☐ 190 ☐ 1900

b) **Message Flooding:** The search is performed using the Message-Flooding strategy. The out-degree of each node is $d=3$. When a node receives a query message it forwards the message to its d neighbors, and then each neighbor forwards the message to its d own neighbors, and so on.

1) For $TTL=4$, compute the probability of finding a particular non-popular resource.
2) For $TTL=4$, compute the probability of finding a particular popular resource.

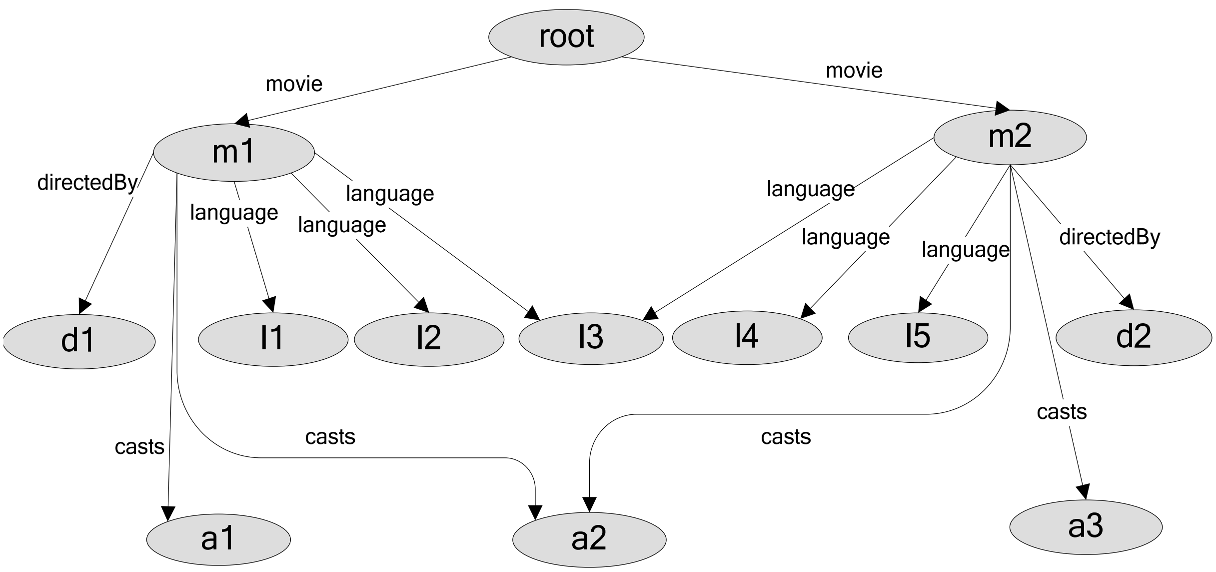
c) **k-Random Walkers:** Assume now, the search is performed using 2-Random Walkers. For the computations make use of the following assumptions: *The probability of a random walker hitting a node is same as the probability of choosing a particular node uniformly at random from the entire set of nodes. The probability of a random walker hitting a node is independent of the previous nodes it hit.*

1) For $TTL=5$, compute the probability of finding a particular non-popular resource.
2) For $TTL=5$, compute the probability of finding a particular popular resource.

d) **Comparison**: Discuss briefly the impact on search latency and network bandwidth consumption for each of the following search strategies: Message Flooding, Expanding Rings, and k-Random Walkers.

Question 2: Graph Databases
Points: ___

The following graph database G_1 is given:



a) Let [*a1*] be the language equivalence class of node *a1*. How many nodes are in this class?

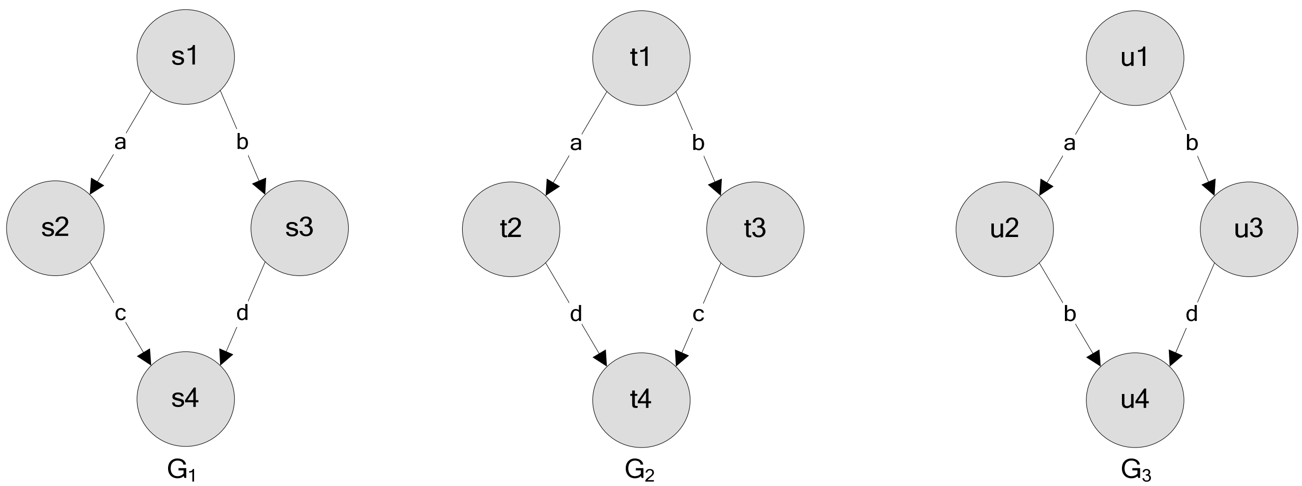
☐ 1 ☐ 2 ☐ 3 ☐ 4

b) Draw the strong data guide SDG_1 for G_1.

c) Add exactly one edge to G_1 such that in the corresponding strong data guide, at least one node identifier appears twice.

d) Add exactly one node to G_1, such that the number of nodes in the corresponding strong data guide is the same as in SDG_1.

e) You are given the following 3 graph databases, identified by G_1, G_2 and G_3 from left to right.



Draw a graph schema for G_2 and G_3 that is not a graph schema for G_1.

ANSWER (Q2):

Question 3: Information Retrieval
Points: ___

We have already seen a **QueryTerm1** *SLOP/x* **QueryTerm2** query. The result of this query is a set of documents where each document contains occurrences of **QueryTerm1** within *x* words of **QueryTerm2** (but not necessarily in that order) and $x > 0$.

Now, let us extend the idea of *SLOP/x* queries to answer a slightly different type of query known as the *SENT/x* query. A *SENT/x* is very similar to the *SLOP/x* query. The only difference is that now each document in the result set of a *SENT/x* query contains occurrences of **QueryTerm1** within *x* words of **QueryTerm2** (but not necessarily in that order) *and* **QueryTerm1** and **QueryTerm2** are in the *same sentence*.

Answer the following questions:

a) As you know, the *term-offset index* addresses terms in documents by their position within the document. How to extend this addressing scheme to address terms by their position within a sentence within the document in order to support a *SENT/x* query?

b) Give the new inverted list I_{ki} for term k_i with the information required for answering a *SENT/x* query.

c) Suppose we have two inverted lists I_{k1} and I_{k2} which are obtained from (b). Give pseudo-codes of algorithms *MergeSent*(I_{k1}, I_{k2}, *x*) and *MergeSlop*(I_{k1}, I_{k2}, *x*) which would be used to "merge" the inverted lists for answering *SENT/x* and *SLOP/x* queries respectively. Maximize the efficiency of your algorithms i.e. minimize the number of steps required to merge the inverted lists.

Question 4: Classification
Points: ___

A company's database has the following attributes:

- *type* is the type of the equipment
- *recommendation* says whether the equipment is recommended for a purchase or not
- *lifetime* is the estimated lifetime of the equipment

A data pre-processing step is performed on the database by *grouping* the tuples with the same values for the attributes *type* and *recommendation* and with similar values for the attributes *lifetime* and *cost*. We consider attribute values as similar if they fall into the same bin (e.g., 31-35). The bins are defined a-priori. The following table is an example of such a group of tuples:

type	***recommendation***	***lifetime***	***cost***
type1	yes	31	46K
type1	yes	32	47K
type1	yes	34	48K
type1	yes	35	50K

The tuples of one group are then aggregated and the number of tuples in a group is stored in a new ***count*** attribute. For the above group this results in the following tuple:

type	***recommendation***	***lifetime***	***cost***	***count***
type1	yes	31-35	46K-50K	4

The value 4 for the *count* attribute indicates that 4 tuples in the original database are aggregated into this single tuple. Using this process, the company database is transformed as shown in *Table 1*. Now we want to construct a decision tree classifier using the data in *Table 1* as training data, with the attribute *recommendation* as the class label attribute.

type	*recommendation*	*lifetime*	*cost*	*count*
type1	yes	31-35	46K-50K	30
type1	no	26-30	26K-30K	40
type1	no	31-35	31K-35K	40
type2	no	21-25	46K-50K	20
type2	yes	31-35	66K-70K	5
type2	no	26-30	46K-50K	3
type2	yes	41-45	66K-70K	3
type3	yes	36-40	46K-50K	10
type3	no	31-35	41K-45K	4
type4	yes	46-50	36K-40K	4
type4	no	26-30	26K-30K	6
type4	no	31-35	46K-50K	2

Table 1: Training Data

Different to the basic decision tree algorithm for categorical attributes, we need to consider the ***count*** value during the construction, as it indicates the number of samples for a specific class present in the original database.

a) Discuss how the values of the *count* attribute need to be considered in the computation of information gain for selecting the optimal attribute for splitting.

b) Use your algorithm to construct a decision tree from the training data given in *Table 1*.

c) Given the test data (*Table 2*) below, what is the accuracy of the classifier you constructed? Justify your answer.

☐ 60% ☐ 70% ☐ 80% ☐ 90%

type	*recommendation*	*lifetime*	*cost*
type3	no	42	49K
type1	no	30	30K
type1	yes	32	47K
type1	yes	34	48K
type3	yes	35	50K
type4	yes	27	47K
type3	yes	50	46K
type4	yes	50	70K
type2	yes	26	32K
type3	no	45	42K

Table 2: Test Data