Feedback — Week7A Advanced

Help Center

You submitted this quiz on **Sun 22 Mar 2015 2:43 PM PDT**. You got a score of **4.00** out of **4.00**.

Question 1

Suppose we have an LSH family h of (d_1,d_2,.6,.4) hash functions. We can use three functions from h and the AND-construction to form a (d_1,d_2,w,x) family, and we can use two functions from h and the OR-construction to form a (d_1,d_2,y,z) family. Calculate w, x, y, and z, and then identify the correct value of one of these in the list below.

Your Answer		Score	Explanation
○ z=.16			
● w=.216	✓	1.00	
○ y=.936			
○ w=.064			
Total		1.00 / 1.00	

Question Explanation

When we use the AND-construction with three hash functions, we cube the probabilities associated with h. Thus, w=.216 and x=.064. To get the probabilities associated with the OR-construction on two hash functions, we take each probability associated with h, subtract it from 1, square the result, and subtract that from 1. Thus, .6 becomes 1-(1-.6)^2 = .84, and .4 becomes 1-(1-.4)^2 = .64.

Question 2

Here are eight strings that represent sets:

s_1 = abcef

s_2 = acdeg

s_3 = bcdefg

s_4 = adfg

s_5 = bcdfgh

s_6 = bceg

s_7 = cdfg

s_8 = abcd

Suppose our upper limit on Jaccard distance is 0.2, and we use the indexing scheme of Section 3.9.4 based on symbols appearing in the prefix (no position or length information). For each of s_1, s_3, and s_6, determine how many *other* strings that string will be compared with, if it is used as the probe string. Then, identify the true count from the list below.

Your Answer		Score	Explanation
○ s3 is compared with 4 other strings.			
● s6 is compared with 3 other strings.	✓	1.00	
○ s6 is compared with 6 other strings.			
○ s3 is compared with 2 other strings.			
Total		1.00 / 1.00	

Question Explanation

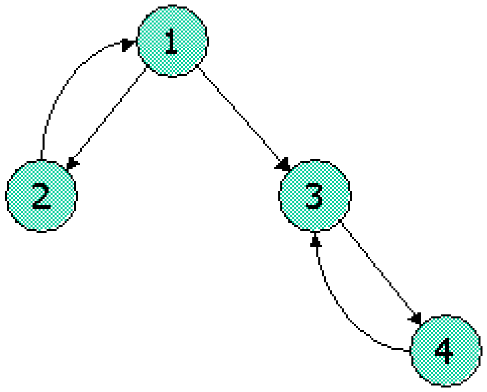
First, we index a string of length L on the symbols appearing in its prefix of length floor(0.2L+1). Thus, strings of length 5 and 6 are indexed on their first two symbols, while strings of length 4 are indexed on their first symbol only. Thus, the index for *a* consists of {s_1, s_2, s_4, s_8}; the index for *b* consists of {s_1, s_3, s_5, s_6}, the index for *c* consists of {s_2, s_3, s_5, s_7}, and no other symbol is indexed at all.
For s_1, we examine the indexes for *a* and *b*, which contains all strings but s_7. Thus, s_1 is compared with 6 other strings.

For s_3, we examine the indexes for *b* and *c*, which together contain s_1, s_2, s_3, s_5, s_6, and s_7. Thus, s_3 is compared with five other strings.

For s_6, we examine only the index for *b*. Thus, s_6 is compared only with the three other strings s_1, s_3, and s_5.

Question 3

Consider the link graph



First, construct the L, the link matrix, as discussed in Section 5.5 on the HITS algorithm. Then do the following:

1. Start by assuming the hubbiness of each node is 1; that is, the vector h is (the transpose of) [1,1,1,1].
2. Compute an estimate of the authority vector **a**=L^T**h**.
3. Normalize **a** by dividing all values so the largest value is 1.
4. Compute an estimate of the hubbiness vector **h**=L**a**.
5. Normalize **h** by dividing all values so the largest value is 1.
6. Repeat steps 2-5.

Now, identify in the list below the true statement about the final estimates.

Your Answer		Score	Explanation
○ The final estimate of the hubbiness of 1 is 3/5.			
● The final estimate of the hubbiness of 1 is 1.	✓	1.00	
○ The final estimate of the authority of 2 is 1/5.			
○ The final estimate of the authority of 2 is 1/8.			
Total		1.00 / 1.00	

Question Explanation

Here is the matrix L:

$$\begin{matrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{matrix}$$

In what follows, all vectors will be written as rows, i.e., in transposed form. We start with **h** = [1,1,1,1] and compute L^T**h** = [1,1,2,1]. Since the largest value is 2, we divide all values by 2, giving us the first estimate **a** = [1/2,1/2,1,1/2].

Next, we compute L**a** = [3/2,1/2,1/2,1] and normalize by multiplying by 2/3 to get **h** = [1,1/3,1/3,2/3].

The next calculation of **a** from the estimate of **h** gives L^T**h** = [1/3,1,5/3,1/3], and normalizing gives **a** = [1/5,3/5,1,1/5].

For the final estimate of **h** we compute L**a** = [8/5,1/5,1/5,1], which after normalizing gives **h** = [1,1/8,1/8,5/8].

Question 4

Consider an implementation of the Block-Stripe Algorithm discussed in Section 5.2 to compute page rank on a graph of N nodes (i.e., Web pages). Suppose each page has, on average, 20 links, and we divide the new rank vector into k blocks (and correspondingly, the matrix M into k stripes). Each stripe of M has one line per "source" web page, in the format:

[source_id, degree, m, dest_1, ..., dest_m]

Notice that we had to add an additional entry, m, to denote the number of destination nodes in this stripe, which of course is no more than the degree of the node. Assume that all entries (scores, degrees, identifiers,...) are encoded using 4 bytes.

There is an additional detail we need to account for, namely, **locality** of links. As a very simple model, assume that we divide web pages into two disjoint sets:

1. **Introvert** pages, which link only to other pages within the same host as themselves.
2. **Extrovert** pages, which have links to pages across several hosts.

Assume a fraction x of pages (0 Construct a formula that counts the amount of I/O per page rank iteration in terms of N, x, and k. The 4-tuples below list combinations of N, k, x, and I/O (in bytes). Pick the correct combination.

Note. There are some additional optimizations one can think of, such as striping the old score vector, encoding introvert and extrovert pages using different schemes, etc. For the purposes of working this problem, assume we don't do any optimizations beyond the block-stripe algorithm discussed in class.

Your Answer		Score	Explanation
○ N = 1 billion, k = 2, x = 0.5, 116GB			
● N = 1 billion, k = 2, x = 0.75, 107GB	✓	1.00	
○ N = 1 billion, k = 2, x = 0.5, 112GB			
○ N = 1 billion, k = 3, x = 0.75, 124GB			
Total		1.00 / 1.00	

Question Explanation

The number of bytes involved in reading the old pagerank vector and writing the new pagerank vector to disk = 4 (k+1) N For the M matrix: - The introvert pages will appear xN times and each row will have on average 23 entries (3 metadata and 20 destination links). Total number of bytes read = 4*23 xN - The extrovert pages will appear (1-x) kN times and each row will have 3 (metadata) + 20/k (destination links) entries on average. Total number of bytes read = 4 * (3+20/k) * (1-x) kN Total I/O per pagerank iteration (in GB, where 1GB ~ 10^9 = N bytes) = 4 [(k+1) N + 23 xN + (3k + 20) (1-x) N] / N = 4 [(k+1) + 23 x + (3k + 20) (1-x)] = 4 [21 + k + 3 (x + (1-x) k)]