Due: 2019/3/15

Homework 2

Student Number: 118033910019
Name: Xingyi Wang

Problem 1. (20 points) Consider the following fragment of a positional index with the format:
word: document:<position, position, . . .>; document: <position, . . .>
Gates: 1: <3>; 2:<6>; 3: <2,17>; 4: <1>;
IBM: 4: <3>; 7: <14>;
Microsoft: 1: <1>; 2: <1,21>; 3: <3>; 5 :<16,22,51>;

The /k operator, word1 /k word2 finds occurrences of word1 within k words of word2 (on either side), where k is a positive integer argument. Thus k = 1 demands that word1 be adjacent to word2.

a. Describe the set of documents that satisfy the query Gates /2 Microsoft.

b. Describe each set of values for k for which the query Gates /k Microsoft returns a different set of documents as the answer.

Solution.

a. Document 1 and 3 satisfy the query Gates /2 Microsoft.

b. For k=$\{1\}$, document 3 satisfies the query; for k=$\{2, 3, 4\}$, document 1 and 3 satisfy the query; for k=$\{i|i \geq 5\}$, document 1, 2, and 3 satisfy the query.

Problem 2. (30 points) Given two strings S_1 and S_2, write down the pseudo-code of computing the edit distance between them.

Solution.

Algorithm 1: Compute edit distance(S_1, S_2)

input : String S_1, S_2
output: Edit Distance d

```
1  len1 ← the length of S1
2  len2 ← the length of S2
3  if len1 = 0 then
4      return len2
5  if len2 = 0 then
6      return len1
7  dp ← [len1 + 1][len2 + 2]
8  for i : 0 → len1 do
9      dp[i][0] ← i
10 for i : 1 → len2 do
11     dp[0][i] ← i
12 for i : 1 → len1 do
13     for j : 1 → len2 do
14         if S1[i − 1] = S2[j − 1] then
15             dp[i][j] = dp[i − 1][j − 1]
16         else
17             dp[i][j] = min(dp[i − 1][j], dp[i][j − 1], dp[i − 1][j − 1]) + 1
18 d ← dp[len1][len2]
19 return d
```

a b d c 0 1 2 3 4 a 1 0 1 2 3 b 2 1 0 1 2 c 3 2 1 1 1

Problem 3. (30 points) If you wanted to search for *s*ng* in a permuterm wildcard index, what key(s) would one do the lookup on?

Solution. *ng$s**.

Problem 4. (20 points) Write the pseudo code showing the details of computing the Jaccard coefficient while scanning the posting of the k-gram index. (Page 49 in the slide)

Solution.

Algorithm 2: Compute Jaccard coefficient(X, Y)

input : Postings of the k-gram index X and Y
output: Jaccard coefficient J

Sort X and Y according to the alphabetical order, then we get the sorted postings X' and Y'.
Let p and q be the pointer to the first index of posting X' and Y'.
$p', q' \leftarrow NIL$
$a, b, c \leftarrow 0$
while $p \neq NIL$ ***and*** $q \neq NIL$ **do**
    if $X'(p) = X'(p')$ **then**
        $p \leftarrow next(p)$
    if $Y'(q) = Y'(q')$ **then**
        $q \leftarrow next(q)$
    if $X'(p) = Y'(q)$ **then**
        $a \leftarrow a + 1$ $b \leftarrow b + 1$ $c \leftarrow c + 1$
        $p' \leftarrow p$ $q' \leftarrow q$ $p \leftarrow next(p)$ $q \leftarrow next(q)$
    else
        if $X'(p) < Y'(q)$ **then**
            $a \leftarrow a + 1$
            $p' \leftarrow p$ $p \leftarrow next(p)$
        else
            $b \leftarrow b + 1$
            $q' \leftarrow q$ $q \leftarrow next(q)$
while $X'(p) \neq NIL$ **do**
    if $X'(p) = X'(p')$ **then**
        $p \leftarrow next(p)$
    else
        $a \leftarrow a + 1$
        $p' \leftarrow p$ $p \leftarrow next(p)$
while $Y'(q) \neq NIL$ **do**
    if $Y'(q) = Y'(q')$ **then**
        $q \leftarrow next(q)$
    else
        $b \leftarrow b + 1$
        $q' \leftarrow q$ $q \leftarrow next(q)$
$J \leftarrow c/(a + b - c)$
return J