Mark Stnozahi

CptS 350

Homework 6

1. Design an algorithm that finds such a $\gamma$ for any given $\alpha$ and $\beta$. Also, analyze its complexity.

Table definition.

Let DP[i][j][k] be the length of the longest subsequence of strings $\alpha[1...i]$ and $\beta[1...j]$, k can represent an additional statate

Base case:

For $i=0$ or $j=0$

DP[i][j][k] = 0 for all k

Recurrence relation

For $i>0$ and $j>0$

if $\alpha[i] == \beta[j]$:

if $\alpha[i]$ != 'b' or k != 1:

DP[i][j][0] = DP[i-1][j-1][0] + 1

DP[i][j][1] = DP[i-1][j-1][1] + 1 (if $\alpha[i] == 'b'$ and $\alpha[i-1] == 'a'$)

else

DP[i][j][0] = max(DP[i-1][j-1][0], DP[i-1][j-1][1])

else

DP[i][j][0] = max(DP[i-1][j][0], DP[i][j-1][0])

DP[i][j][1] = max(DP[i-1][j][1], DP[i][j-1][1]), if $\alpha[i]$ or $\beta[j]$ != 'b')

Starting from DP[n][m][k], 'abb' pattern is never completed

| | 0 | 1 | 2 | ... | m |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | ... | 0 |
| 1 | 0 | | | | |
| 2 | 0 | | | | |
| : | | | | | |
| n | 0 | | | | |

* The Table has dimensions $n \times m \times 2$ which is for each entry DP[i][j][k] since a constant # of operations is performed the time complexity is $O(nm)$.

2. Design an algorithm that compute the number $D$ with $D = \max_{\alpha \in L_1, \beta \in L_2} d(\alpha, \beta)$

- Here a high level way of explaining how to conceptualize an algorithm.

- pseudocode could be

```
N1 ← construct NFA(L1)
N2 ← construct NFA(L2)
P ← construct Product Automaton (N1, N2)

IF has cycle(P)
    return "D is infinite"

D ← 0
    for each state pair (S1, S2) in P:
        IF IsAccepting (S1, N1) and IsAccepting (S2, N2):
            length ← Find Longest Path to (P(S1, S2))
            D ← max (D, length)

return D.
```

constructing an NFA from a regular expression has a complexity of $O(r)$ for a regular expression of size 'r'. The automation P can be $O(n \cdot m)$ where n and m are values N1 and N2,
The has cycle check is $O(V+E)$ for # of vertices V and edges E. Lastly computing the longest path can be $O(V+E)$ also.

3. Is there any locality sensitive hash scheme for strings?

Both MH-Hashing and Sim Hash can be considered locality-sensitive hashing schemes because they tend to ensure that similar items hash to the same bucket more often than dissimilar items. Overall, yes there are locality-sensitive hashing LSH scheme for strings.