Assignment 7, Amogha Sekhar, Question 1

In [2]:
```
#import necessary libraries

import numpy as np
from math import log
import string
```

In [3]:
```
#function to read in the input files
def read_files():
    pi = list()
    with open('initialStateDistribution.txt', 'r') as f:
        for line in f:
            pi.append(float(line.strip()))

    a = list()
    with open('transitionMatrix.txt', 'r') as f:
        for line in f:
            a.append([float(entry) for entry in line.strip().split()])

    b = list()
    with open('emissionMatrix.txt', 'r') as f:
        for line in f:
            b.append([float(entry) for entry in line.strip().split()])

    O = list()
    with open('observations.txt', 'r') as f:
        O = [int(entry) for entry in f.readline().strip().split()]

    return pi, a, b, O
```

In [4]:
```
#function to calculate base case

def calc_base_case(pi, b, O, list_alpha):
    l = list()
    for i in range(len(list_alpha)):
        l.append(log(pi[i]) + log(b[i][O[0]]))
    return l
```

In [5]:
```
#function for all cases

def calc_all_cases(a, b, O, l, phi, list_alpha):
    for t in range(1, len(O)):
        l_temp = list()
        phi_temp = list()

        for j in range(len(list_alpha)):
            max_sum = np.NINF
            max_i = -1
            for i in range(len(list_alpha)):
                i_sum = l[t-1][i] + log(a[i][j])
                if i_sum > max_sum:
                    max_sum = i_sum
                    max_i = i
            l_temp.append(max_sum + log(b[j][O[t]]))
            phi_temp.append(max_i)
        l.append(l_temp)
        phi.append(phi_temp)
    return l, phi
```

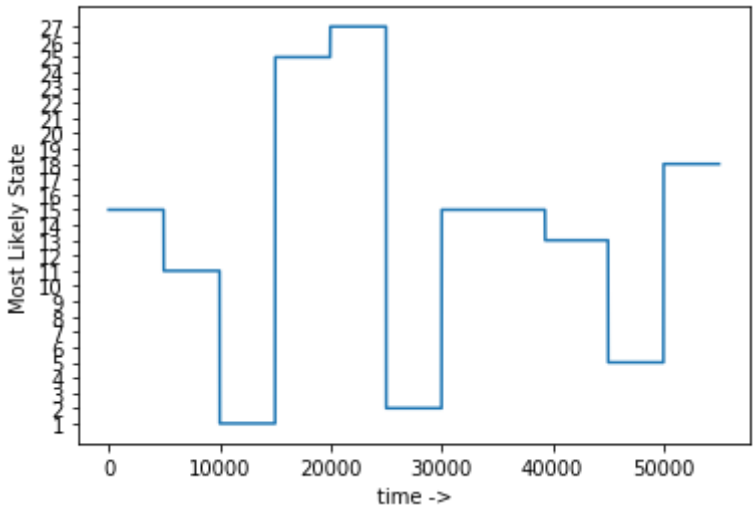
In [6]:
```
#function to plot the graph

def plot_graph(S, O):
    import matplotlib.pyplot as plt
    plt.plot(range(len(O)), [s+1 for s in S])

    plt.xlabel('time ->')
    plt.ylabel('Most Likely State')

    plt.yticks(range(1, 28))

    plt.show()
```

In [12]:

```
#main function
%matplotlib inline

list_alpha = list(string.ascii_lowercase)
list_alpha.append(" ")

pi, a, b, O = read_files()

l = [calc_base_case(pi, b, O, list_alpha)]
phi = [[-1]*27]

l, phi = calc_all_cases(a, b, O, l, phi, list_alpha)

S = [-1 for i in range(len(O))]

S[-1] = l[-1].index(max(l[-1])) #argmax step for last state

for t in range(len(O)-2, -1, -1): # Back pointer step
    S[t] = phi[t+1][S[t+1]]

sentence = ""
for t in S:
    sentence += list_alpha[t]

print(sentence)
```

```
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
```

In [13]:
```
plot_graph(S, O)
```



In [ ]:
```

```