RYERSON UNIVERSITY

RYERSON UNIVERSITY

Faculty of Engineering, Architecture and Science

Department of Electrical and Computer Engineering

Program: Biomedical Engineering

Course Number	**BME 808**
Course Title	**Computations in Genetic Engineering**
Semester/Year	Winter 2021
Instructor	P. Sidda

Lab No. 3

Report Title **Decoding in Hidden Markov Model**

Section No. 3

Group No. N/A

Submission Date March 19 2021

Due Date March 20 2021

Name Andrew Mullen Student ID 500 787 631 Signature*

(Note: remove the first 4 digits from your student ID)

BME 808 – Lab 3

1.

In [115]:
```
n = 2
trans = ['']*n
trans[0] = [0.99, 0.01]
trans[1] = [0.2, 0.8]

print(trans)
```

```
[[0.99, 0.01], [0.2, 0.8]]
```

In [116]:
```
initial = [0.5, 0.5]
print(initial)
```

```
[0.5, 0.5]
```

In [117]:
```
emit = ['']*n
emit[0] = [1.0/6, 1.0/6, 1.0/6, 1.0/6, 1.0/6, 1.0/6]
emit[1] = [1/10.0, 1/10.0, 1/10.0, 1/10.0, 1/10.0, 1/2.0]
print(emit)
```

```
[[0.16666666666666666, 0.16666666666666666, 0.16666666666666666, 0.16666666666666666, 0.16666666666666666, 0.16666
666666666666], [0.1, 0.1, 0.1, 0.1, 0.1, 0.5]]
```

In [118]:
```
Eseq = [2,5,3]
print(Eseq)
```

```
[2, 5, 3]
```

In [119]:
```
M = len(trans)
N = len(Eseq)
pmax = 0

Sseq = []

# Find Sseq to give highest probability
for i0 in range(M) :
    p0 = initial[i0] * emit[i0][Eseq[0]]

    for i1 in range(M) :
        p1 = emit[i1][Eseq[1]] * trans[i0][i1]

        for i2 in range(M) :
            p2 = emit[i2][Eseq[2]] * trans[i1][i2]
            p = p0*p1*p2

            if p > pmax :
                pmax = p
                Sseq = [i0, i1, i2]


print(pmax)
print(Sseq)
```

```
0.0022687499999999995
[0, 0, 0]
```

2.

In [7]:
```
# Part 2
Eseq = [2,2,2,2,2,2,2,5,5,5,5,5]
print(Eseq)
```

```
[2, 2, 2, 2, 2, 2, 2, 5, 5, 5, 5, 5]
```

In [8]:
```
M = len(trans)
N = len(Eseq)
pmax = 0

Sseq = []

for i0 in range(M) :
    p0 = initial[i0] * emit[i0][Eseq[0]]
    for i1 in range(M) :
        p1 = emit[i1][Eseq[1]] * trans[i0][i1]
        for i2 in range(M) :
            p2 = emit[i2][Eseq[2]] * trans[i1][i2]
            for i3 in range(M) :
                p3 = emit[i3][Eseq[3]] * trans[i2][i3]
                for i4 in range(M) :
                    p4 = emit[i4][Eseq[4]] * trans[i3][i4]
                    for i5 in range(M) :
                        p5 = emit[i5][Eseq[5]] * trans[i4][i5]
                        for i6 in range(M) :
                            p6 = emit[i6][Eseq[6]] * trans[i5][i6]
                            for i7 in range(M) :
                                p7 = emit[i7][Eseq[7]] * trans[i6][i7]
                                for i8 in range(M) :
                                    p8 = emit[i8][Eseq[8]] * trans[i7][i8]
                                    for i9 in range(M) :
                                        p9 = emit[i9][Eseq[9]] * trans[i8][i9]
                                        for i10 in range(M) :
                                            p10 = emit[i10][Eseq[10]] * trans[i9][i10]
                                            for i11 in range(M) :
                                                p11 = emit[i11][Eseq[11]] * trans[i10][i11]
                                                p = p0*p1*p2*p3*p4*p5*p6*p7*p8*p9*p10*p11
                                                if p > pmax :
                                                    pmax = p
                                                    Sseq = [i0,i1,i2,i3,i4,i5,i6,i7,i8,i9,i10,i11]

print(pmax)
print(Sseq)
```

```
2.1524466149999991e-10
[0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1]
```

3.

a.

In [130]:
```
# Find maximum in the kast column and its location
lastcolF = vit[0][len(vit[0])-1]
lastcolL = vit[1][len(vit[1])-1]

if lastcolF > lastcolL :
    pmax = lastcolF
    imax = 0

else :
    pmax = lastcolL
    imax = 1

print(imax)
```

```
0
```

In [131]:
```
Sseq = []
prob = pmax

imaxseq = [imax]

print(imaxseq)

for j in range(N-2, -1, -1) :
    target = prob/emit[imax][Eseq[j+1]]
    candidate = []
    for k in range(M) :
        # Check the value of the reverse equation
        candidate.append(vit[k][j]*trans[k][imax])

    for i in range(len(candidate)) :
        if candidate[i] == target :
            imax = i

    prob = vit[imax][j]

    imaxseq.append(imax)

imaxseq.reverse()
```

```
[0]
```

In [132]:
```
print(imaxseq)
print(pmax)
```

```
[0, 0, 0]
0.0022687499999999995
```

3.

a.

In [141]:
```
# Find maximum in the kast column and its location
lastcolF = vit[0][len(vit[0])-1]
lastcolL = vit[1][len(vit[1])-1]

if lastcolF > lastcolL :
    pmax = lastcolF
    imax = 0

else :
    pmax = lastcolL
    imax = 1

print(imax)
```

```
1
```

In [142]:
```
Sseq = []
prob = pmax

imaxseq = [imax]

print(imaxseq)

for j in range(N-2, -1, -1) :
    target = prob/emit[imax][Eseq[j+1]]
    candidate = []
    for k in range(M) :
        # Check the value of the reverse equation
        candidate.append(vit[k][j]*trans[k][imax])

    for i in range(len(candidate)) :
        if candidate[i] == target :
            imax = i

    prob = vit[imax][j]

    imaxseq.append(imax)

imaxseq.reverse()
```

```
[1]
```

In [143]:
```
print(imaxseq)
print(pmax)
```

```
[0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1]
2.1524466149999991e-10
```