Análise de desempenho de sistemas 30/08/2019

Lista 2: Exercício - Cadeias de *Markov*

Aluno: Paulo Fylippe Sell Professor: Eraldo Silveira e Silva

1. Considere a matriz de estocástica que representa uma DTMC:

$$P = \begin{vmatrix} 0,2 & 0 & 0,8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0,2 & 0,3 & 0,3 & 0,2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0,1 & 0 & 0 & 0,9 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0,3 & 0,7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0,2 & 0 & 0,8 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.8 & 0,2 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0,2 & 0,6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0,2 \end{vmatrix}$$

(a) Use um simulador para computar os resultados em regime permanente. Utilize cem mil passos. Qual estado possui o maior e menor ocupação?

Solução: Em regime permanente o vetor de estados fica aproximadamente da seguinte maneira:

$$\pi = \begin{vmatrix} 0,071 & 0,21 & 0,132 & 0,064 & 0,043 & 0,131 & 0,094 & 0,026 & 0,075 & 0,155 \end{vmatrix}$$

Portanto, o estado 1 (ou segundo estado) possui o maior tempo de ocupação, ocupando aproximadamente 21% do tempo. O estado com menor ocupação é o estado número 7 (ou oitavo estado), com aproximadamente 2,6% do tempo.

(b) Compute por simulação o tempo médio de recorrência para ir do estado 0 para o estado 5.

Solução: O tempo médio de recorrência é dado a partir da seguinte equação:

$$f_{ij}^{(n)} = p_{ij}^{(n)} - \sum_{l=1}^{n-1} f_{ij}^{(l)} p_{jj}^{(n-1)},\ n > 1$$

Desta forma, o tempo médio de recorrência entre os estados 0 e 5 é de aproximadamente 2,378 épocas.

(c) Suponha que a DTMC representa uma entidade de protocolo que somente transmite no estado 2. Suponha que um pacote usa exatamente uma época *T* para ser transmitido e que pacotes possuem tamanho fixo de 1000 *bytes*. Assumindo que a energia gasta para transmitir um pacote de tempo T = 10*ms* é de 0,05J, qual a potência média gasta para transmissão em dBm?

Solução: A potência média para transmissão, assumindo que a transmissão ocorre apenas no estado 2, se da a partir da seguinte equação:

$$P = \frac{0,05}{0,01} * 0,132 = 0,66\ W$$

Logo, a potência em dBm é de:

$$P_{dBm} = 10log\left(\frac{0,66}{0,001}\right) = 28,19\ dBm$$

(d) Qual seria a vazão em bps?

Solução: É sabido que cada pacote transmite 8000 *bits*. Sabe-se também que cada pacote dura 10*ms* e a transmissão sempre ocorre no estado 2. Portanto, a vazão em bps é:

$$V_{bps} = \frac{8000}{0,01} * 0,132 = 105,6\ kbps$$

1 Apêndice

1.1 Códigos

Cálculos realizados com o auxílio do módulo *numpy* implementado em *Python3*:

```
'''
    27/08/2019
    Autor: Paulo Fylippe Sell
    Disciplina: ADS29009
                .>.>.> Cadeias de Markov <.<.<.
    Função PMFdata adaptada de Steven Kay Springer, 2006
    Função dtmcfpt adaptada do pacote queueing, do software Octave

'''

import numpy as np
import random

def dtmcfpt(P):
    N = 100000
    p0 = [1,0,0,0,0,0,0,0,0,0]
    m = P
    m = np.matrix(m)
    xi = np.arange(m.shape[0]) + 1
    X0 = PMFdata(1,xi,p0)
    i = X0
    X = []
    X.append(PMFdata(1,xi,P[i]))
    i = X[0]+1

    for n in range(1,N-1):
        i = X[n-1]-1
        X.append(PMFdata(1,xi,P[i]))

    states = [0,0,0,0,0,0,0,0,0,0]

    for state in X:
        states[state-1] = states[state-1] + 1
    P = np.matrix(P)

    matrix = np.zeros(shape=(P.shape[0],P.shape[1]))

    for x in range(P.shape[0]):
        for y in range(P.shape[1]):
            counter = 0
            flag = 0
            array = []
            for state in X:
                if state - 1  == x and flag == 0:
                    counter = 0
                    flag = 1
                elif state - 1 == y and flag == 1:
                    counter = counter + 1
```

```
                        array.append(counter)
                        counter = 0
                        flag = 0
                    elif state != y:
                        if flag == 1:
                            counter = counter + 1
                matrix[x][y] = sum(array)/len(array)

    return matrix

def PMFdata(N,xi,pX):
    bi = []
    x = 0
    M = len(xi)
    for k in range(M):
        if k == 0:
            bi.append(pX[k])
        else:
            bi.append(bi[k-1]+pX[k])
    u = np.random.random()
    if (u > 0) and u <= bi[0]:
        x = xi[0]
    for k in range(1,M):
        if u > bi[k-1] and u <= bi[k]:
            x = xi[k]
    return x


def dtmc(P):
    N = 100000
    p0 = [1,0,0,0,0,0,0,0,0,0]
    xi = [1,2,3,4,5,6,7,8,9,10]
    X0 = PMFdata(1,xi,p0)
    i = X0
    X = []
    X.append(PMFdata(1,xi,P[i]))
    i = X[0]+1

    for n in range(1,N-1):
        i = X[n-1]-1
        X.append(PMFdata(1,xi,P[i]))

    states = [0,0,0,0,0,0,0,0,0,0]

    for state in X:
        states[state-1] = states[state-1] + 1
    states = np.array(states)
    return states/N



P =           [[0.2, 0.0, 0.8, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
        [0.0, 0.2, 0.3, 0.3, 0.2, 0.0, 0.0, 0.0, 0.0, 0.0],
        [0.0, 0.0, 0.1, 0.0, 0.0, 0.9, 0.0, 0.0, 0.0, 0.0],
```

```
        [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0],
        [0.0, 0.0, 0.0, 0.0, 0.0, 0.3, 0.7, 0.0, 0.0, 0.0],
        [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2, 0.0, 0.8],
        [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.8, 0.2],
        [1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
        [0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
        [0.2, 0.6, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2]]

states = dtmc(P)
matrix = dtmcfpt(P)

print('Vetor pi de estados em regime permanente:', np.round(states,3))
print()
print('Estado com maior ocupação: {}, com {} %  do tempo'.format(np.where(states==np.amax(states))[0], round(max(states)*100,3)))
print()
print('Estado com menor ocupação: {}, com {} %  do tempo'.format(np.where(states==np.amin(states))[0], round(min(states)*100,3)))
print()
print('Tempo médio de recorrência entre os estados 0 e 5 é de {} épocas'.format(round(matrix[0][5],3)))
print()

#print(np.round(matrix,3)) #descomente esta linha para imprimir a matriz de recorrencia
```