Geração de Tráfego Baseada em Cadeia de Markov

Jéssica Gomes Carrico e Leonardo Ludvig

Objetivo

- Implementar um gerador de tráfego baseado em uma cadeia de Markov de 3 estados.
- Utilizar o `iperf` para gerar tráfego de acordo com a cadeia.
- Comparar resultados práticos e teóricos.

Descrição do Problema

- Estado 0: Ocioso (0 Mbps)
- Estado 1: Tráfego Moderado (10 Mbps)
- Estado 2: Tráfego Alto (50 Mbps)
- Transições definidas por uma matriz P

Matriz de Transição e Geração de Estados

$$P = \begin{bmatrix} 0.6 & 0.4 & 0.0 \\ 0.2 & 0.5 & 0.3 \\ 0.2 & 0.4 & 0.4 \end{bmatrix}$$

- Simulação de 1000 passos
- Estados armazenados em `cadeia_markov.csv`

Matriz de Transição e Geração de Estados

Visualização do arquivo gerado:

Passo	Estado
0	0
1	1
2	2
3	1
4	2
5	2
6	1
7	1
8	1
9	1
10	2

Execução Prática

- Leitura de estados do arquivo
- Execução do `iperf` conforme o estado:
  - Estado 0: `sleep(10)`
  - Estado 1: `iperf -c -b 10M`
  - Estado 2: `iperf - c -b 50M`
- Resultados salvos em `resultadosIPERF.csv`

Execução Prática

Visualização do arquivo gerado:

Passo	Estado	saida_iperf
0	0	Ocioso
1	1	-- Client connecting to 191.36.15.1, TCP port 5001 TCP window size: 16.0 KByte (default) -- [1] local 191.36.15.21 port 37560 connected with 191.36.15.1 port 5001 (icwnd/mss/irtt=14/1448/363) [ID] Interval Transfer Bandwidth [1] 0.0000-10.1175 sec 12.7 MBytes 10.6 Mbits/sec
2	2	-- Client connecting to 191.36.15.1, TCP port 5001 TCP window size: 16.0 KByte (default) -- [1] local 191.36.15.21 port 51438 connected with 191.36.15.1 port 5001 (icwnd/mss/irtt=14/1448/435) [ID] Interval Transfer Bandwidth [1] 0.0000-10.0320 sec 62.7 MBytes 52.4 Mbits/sec
3	1	-- Client connecting to 191.36.15.1, TCP port 5001 TCP window size: 16.0 KByte (default) -- [1] local 191.36.15.21 port 60328 connected with 191.36.15.1 port 5001 (icwnd/mss/irtt=14/1448/357) [ID] Interval Transfer Bandwidth [1] 0.0000-10.1174 sec 12.7 MBytes 10.6 Mbits/sec

Extração de Vazão

- Utilizado `pandas` para extrair Mbps da saída do `iperf`
- Cálculo da vazão média prática

Vazão:

Análise Teórica (Octave)

- Cálculo teórico do estado estacionário:

$$\pi = [0.3333, 0.4444, 0.2222]$$

- Vazão teórica:

$$0.3333 \times 0 + 0.4444 \times 10 + 0.22222 \times 50 = \boxed{15.556 \text{ Mbps}}$$

Comparação de Resultados

- Vazão média prática: **16.06 Mbps**
- Vazão média teórica: **15.55 Mbps**
- Proporção de estados:

Estado	Proporção Prática	Proporção Teórica
0 (Ocioso)	34.45%	33.33%
1 (Moderado)	43.70%	44.44%
2 (Alto)	21.85%	22.22%

Table: Comparação entre proporção de tempo medida e esperada por estado

Discussão

- Diferenças entre valores práticos e teóricos podem ser atribuídas a:
  - Overhead de sistema
  - Condições reais da rede
  - Variações no processamento de tráfego
- Resultados práticos seguem tendência esperada

Conclusão

- A modelagem por cadeia de Markov mostrou-se eficaz para simulação de tráfego.
- O comportamento prático foi comparável ao teórico.
- Potencial para uso em modelagem e simulação de redes reais.

Códigos Utilizados - Parte 1

1. Geração da Cadeia de Markov (Python):

```
P = np.array([
    [0.6, 0.4, 0.0],
    [0.2, 0.5, 0.3],
    [0.2, 0.4, 0.4]
])

states = [0]
for _ in range(n_steps - 1):
    current_state = states[-1]
    next_state = np.random.choice([0, 1, 2], p=P[current_state
    states.append(next_state)
```

Códigos Utilizados - Parte 2

2. Execução com iperf (Python):

```
if estado == 1:
    cmd = f"iperf -c {DESTINO} -b 10M -t 10"
elif estado == 2:
    cmd = f"iperf -c {DESTINO} -b 50M -t 10"
else:
    time.sleep(10)
```

Códigos Utilizados - Parte 3

2. Cálculo média e Proporção de tempo em cada estado (Python):

```
# Calcula a vazão média
vazao_media_pratica = df['Vazao'].mean()
print(f"Vazão média prática: {vazao_media_pratica} Mbps")

# Calcula a proporção de tempo em cada estado
estado_counts = df['Estado'].value_counts(normalize=True)
print("Proporção de tempo em cada estado:")
print(estado_counts)
```

Cálculo do Estado Estacionário (Octave)

3. Cálculo do Estado Estacionário e vazão média:

```
A = [P' - eye(3); ones(1,3)];
b = [zeros(3,1); 1];
steady_state = A \ b;


disp("Vetor de estado estacionário:");
disp(steady_state');

rates = [0, 10, 50];
mean_rate = steady_state' * rates';  % produto escalar


disp("Vazão média:");
disp(mean_rate);
```