Simulação por Eventos Discretos

Aplicação à simulação de tráfego

Sistemas de Telecomunicações
MIEEC - Área de Telecomunicações

4º Ano - 2º Semestre
FEUP – 2018-19

MJL, MPR, RLC

Intencionalmente em branco

Simulação por eventos discretos – princípios

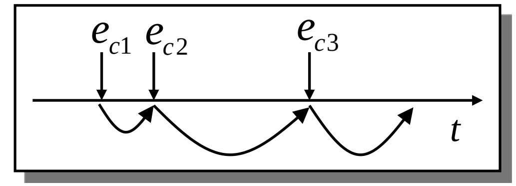
- Aplicável a sistemas de estados discretos
  - as mudanças de estado são determinadas por eventos
  - anima-se o sistema analisando o efeito dos sucessivos eventos no estado do sistema
- Processo de simulação
  - **inicialização**
    - relógio, contadores e outras variáveis a 0
    - determinam-se os instantes de eventos futuros
  - **avanço do tempo**
    - avança-se o tempo para o próximo evento
    - processa-se o evento, actualizando o estado do sistema
    - actualizam-se os contadores e outras variáveis
    - determina-se a ocorrência de futuros eventos
  - **fim da simulação**
    - guardam-se os contadores e outras variáveis para análise
  - **repetição da simulação**
    - novas amostras independentes aumentam a confiança dos resultados

fila de eventos

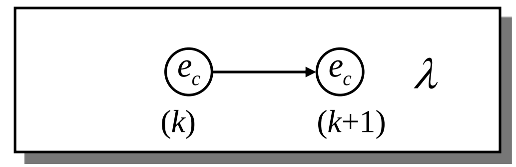
- (instanteDeOcorrência, tipoDeEvento, ...)
- **<u>ordenada por ordem crescente do tempo</u>**

Simulação do processo de Poisson

- Chegada de clientes
  - λ – taxa de chegada de clientes (s^{-1})



  - eventos $\boldsymbol{e_c}$ – chegada de um cliente
  - **processamento de um evento** $\boldsymbol{k}$ $\rightarrow$ determina-se a ocorrência de um novo evento (neste caso, o seguinte k+1)



  - intervalo entre chegada de clientes (c) com distribuição exponencial
    - » geração de uma distribuição exponencial x a partir de uma distribuição uniforme u

$$x = -\ln(u)$$

u gerado aleatoriamente no intervalo]0,1[(função *random*)

$$\mathrm{E}(x) = 1$$

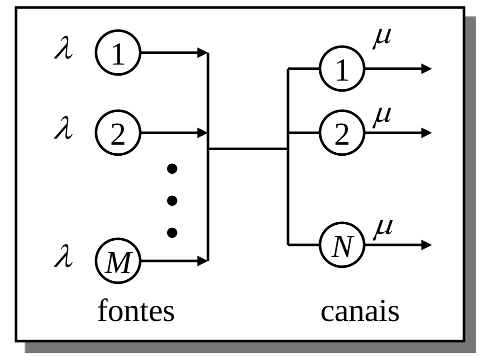
    - » **normalização para a média de intervalos entre chamadas E(c)=1/λ**

$$\boxed{c = -\frac{1}{\lambda}\ln(u)}$$

Simulação de tráfego

- Sistema de perda: caso geral
  - M fontes de tráfego independentes
    - » taxa de chegada de chamadas de cada fonte livre λ
    - » intervalo entre inícios de chamadas c

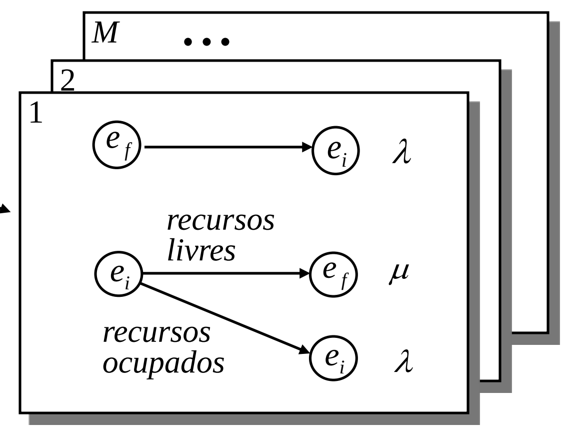
$$\boxed{c = -\frac{1}{\lambda}\ln(u)}$$



  - N servidores (canais)
    - » duração média das chamadas – $\boldsymbol{d_m}$
    - » taxa de libertação de chamadas – $\boldsymbol{\mu}$
    - » duração das chamadas **– *d***

$$\boxed{d = -\frac{1}{\mu}\ln(u) = -d_m \ln(u)}$$

  - geração e processamento de eventos
    - » M processos de geração de tráfego (um por cada fonte)
      - $\boldsymbol{e_i}$ – início de chamada
      - $\boldsymbol{e_f}$ – fim de chamada de uma dada fonte para um dado canal

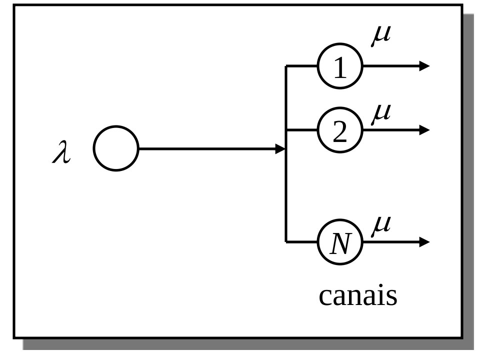


    - » fila única de eventos (nº constante = M)
      - *evento_início*=(*tempo, tipo de evento, 0, 0*)
      - *evento_fim*=(*tempo, tipo de evento, fonte, canal*)

Simulação de tráfego

- Sistema de perda: situação de Erlang-B
  - tráfego oferecido constante (número de fontes $M \to \infty$)
    - » taxa (total) de chegada de chamadas – $\boldsymbol{\lambda}$
    - » intervalo entre inícios de chamadas – ***c***

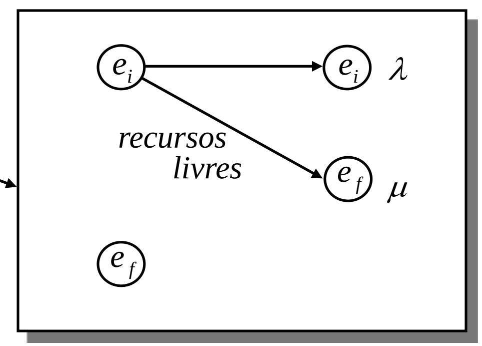
$$\boxed{c = -\frac{1}{\lambda}\ln(u)}$$



  - *N* servidores (canais)
    - » duração média das chamadas – $\boldsymbol{d_m}$
    - » taxa de libertação de chamadas – $\boldsymbol{\mu}$
    - » duração das chamadas **–** ***d***

$$\boxed{d = -\frac{1}{\mu}\ln(u) = -d_m \ln(u)}$$

  - geração e processamento de eventos
    - » **um único processo de geração de tráfego**
      - $\boldsymbol{e_i}$ – início de chamada para uma dada saída
      - $\boldsymbol{e_f}$ – fim de chamada

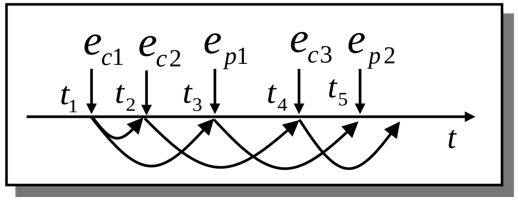


    - » fila única de eventos (nº variável $\leq N+1$)
      - *evento_início*=(*tempo*, *tipo de evento*, *0*)
      - *evento_fim*=(*tempo*, *tipo de evento*, *canal*)

Simulação de tráfego

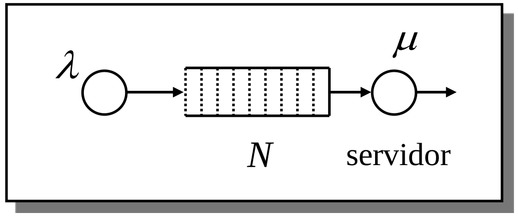
- Fila de espera M/M/1 de comprimento N (situação de Erlang-C quando $N \to \infty$)
  - chegada de clientes
    - taxa de chegada de clientes – $\boldsymbol{\lambda}$
    - intervalo entre chegada de clientes – $\boldsymbol{c}$

$$\boxed{c = -\frac{1}{\lambda}\ln(u)}$$

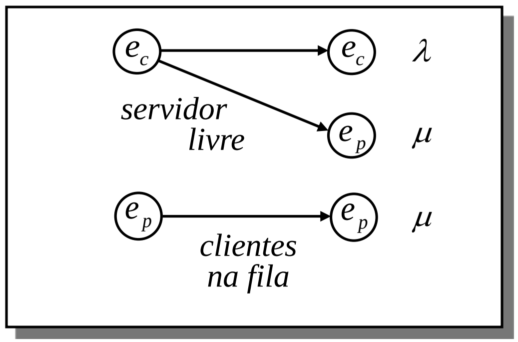


  - serviço de clientes
    - taxa de serviço de clientes – $\boldsymbol{\mu}$
    - tempo de serviço de clientes – $\boldsymbol{s}$

$$\boxed{s = -\frac{1}{\mu}\ln(u)}$$



  - geração e processamento de eventos
    - um único processo de geração de tráfego
      - $\boldsymbol{e_c}$ – chegada de um cliente
      - $\boldsymbol{e_p}$ – partida de um cliente
    - fila única de eventos (nº variável ≤ N+1)
      - *evento_início*=(*tempo, tipo de evento*)
      - *evento_fim*=(*tempo, tipo de evento*)



Simulação de tráfego

- Intervalos de Confiança - exemplo

Estimador do valor médio μ

$$M(n) = \frac{\sum_{i=1}^{n} X_i}{n}$$

Estimador do desvio padrão σ

$$S(n) = \sqrt{\frac{\sum_{i=1}^{n} [X_i - M(n)]^2}{n-1}}$$

Estimador do erro padrão da média σ_M

$$S_M(n) = \frac{S(n)}{\sqrt{n}}$$

Intervalo de confiança para μ a (1-α)×100%

$$I_C(1-\alpha) = M(n) \pm z_{1-\alpha/2} S_M(n)$$

Limite de confiança	$z_{1-\alpha/2}$
80%	1.28
90%	1.65
95%	1.96
98%	2.33
99%	2.58

10 amostras
1.20
1.50
1.68
1.89
0.95
1.49
1.58
1.55
0.50
1.09

idealmente $n \geq 30$

$M(10)=1.34$

$S(10)=0.41$

$S_M(10)=0.13$

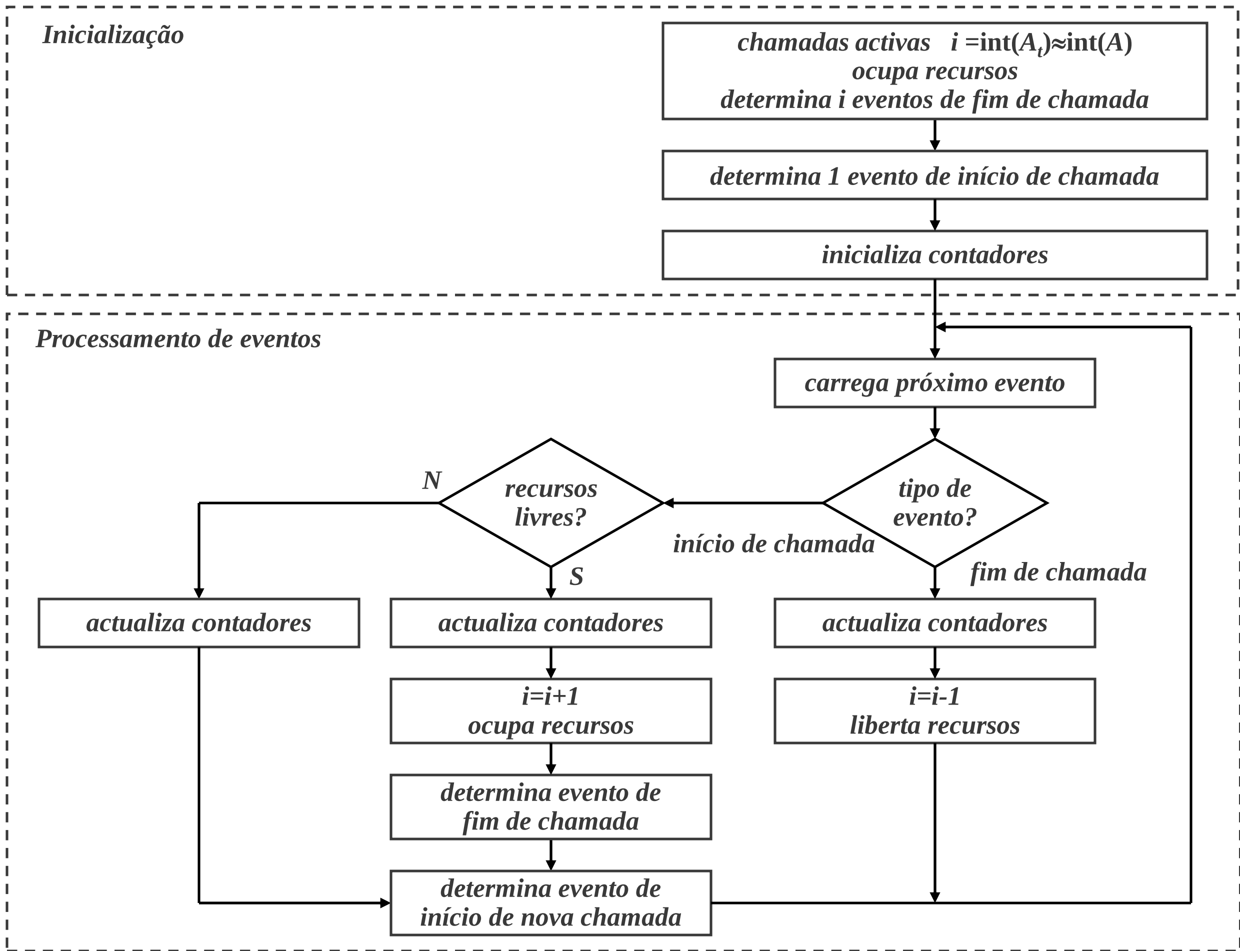
Intervalo de confiança para a média (μ) a 90%

$$I_C(1-\alpha) = M(n) \pm z_{1-\alpha/2} S_M(n)$$

$$I_C(90\%) = 1.34 \pm 1.65 \times 0.13 = 1.34 \pm 0.21$$

Modelo de simulação

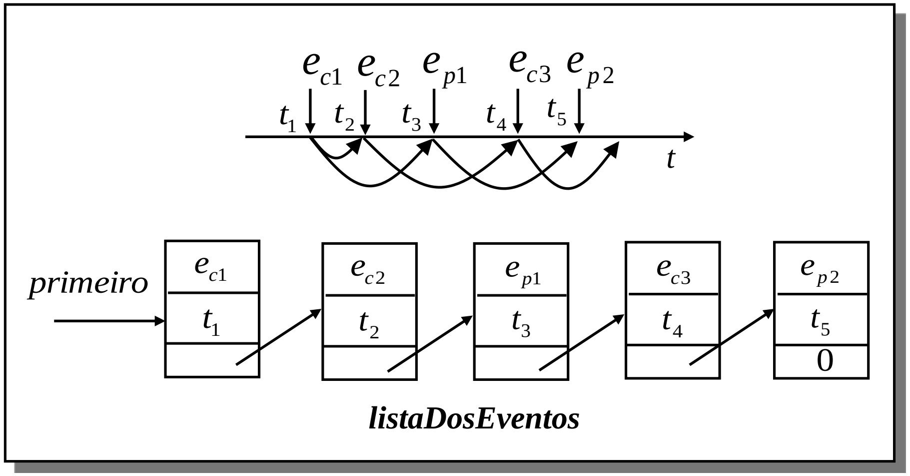
- Sistema de perda: situação de Erlang-B



Modelo de simulação

- Fila de espera M/M/1 de comprimento N (situação de Erlang-C quando $N \to \infty$)

```
main( )
{
inicializa o sistema;
enquanto  simula  /* numero clientes, tempo*/
      {
      selecciona primeiro evento da listaDosEventos;
      actualiza o tempo de simulação t;
      se ( evento = ec ) { chegadaCliente( );  }
      se ( evento = ep ) { partidaCliente( );  }
      }
reporta resultados;
}
```




```
chegadaCliente( )
{
actualiza contadores;
calcula tempo de chegada do próximo cliente (t+c);
insere o novo evento de chegada ec na listaDosEventos;
se  servidor livre
    {
    calcula tempo de partida do cliente (t+s);
    insere o novo evento de partida ep na listaDosEventos;
    }
}
```

```
partidaCliente( )
{
actualiza contadores;
se existem clientes na fila
    {
    calcula tempo de partida do cliente (t+s);
    insere o novo evento de partida ep na listaDosEventos;
    }
}
```