

" UM SIMULADOR DE REDES ABERTAS DE UMA CLASSE "

Autor : SÉRGIO DE FIGUEIRÊDO BRITO

ÍNDICE

1) INTRODUÇÃO.....	
2) CARACTERÍSTICAS DOS MODELOS A SEREM SIMULADOS.....	
3) ESTRUTURA DE DADOS IMPLEMENTADA.....	
4) ALGORITIMO DO SIMULADOR	
4.1) DESCRIÇÃO DO SIMULADOR EM BLOCOS.....	
4.2) DETALHAMENTO DO ALGORITIMO.....	
4.3) DESCRIÇÃO DOS ARQUIVOS FONTES.....	
5) OPERAÇÃO DO SIMULADOR	
5.1) ENTRADA DE DADOS.....	
5.2) EXECUÇÃO DE UMA SIMULAÇÃO.....	
5.3) APRESENTAÇÃO DOS RESULTADOS.....	
6) APLICAÇÕES.....	
7) ANÁLISE DOS RESULTADOS.....	
8) EXTENSÃO PARA IMPLEMENTAR DIVERSAS CLASSES.....	

ANEXOS

ANEXO [1] ==> MODELO 1 arquivo de entrada de dados e resultados da simulação

ANEXO [2] ==> FORMULÁRIO utilizado para os cálculos teóricos e análise dos resultados

1) INTRODUÇÃO

Este trabalho visa descrever o Simulador de Redes Abertas de uma classe implementado em linguagem de programação C.

A metodologia para implementação do simulador e teste de conformidade foi a seguinte :

Definição da estrutura de dados que suportasse um sistema de redes de filas abertas de apenas uma classe.

Especificação do algoritmo a ser implementado.

Implementação do algoritmo em código C.

Depuração do código e testes de execução.

Elaboração de tres modelos exemplo para simulação.

Obtenção dos resultados de simulação dos tres exemplos.

Desenvolvimento dos cálculos teóricos destes tres modelos através da teoria das filas.

Análise dos resultados através da obtenção do erro relativo entre os resultados do simulador e dos cálculos teóricos

Sucintamente o simulador opera da seguinte forma :

1. Obtem o modelo a ser simulado de um arquivo de entrada de dados, com isto não é necessário a cada simulação se recompilar o programa.
2. Executa a simulação do modelo, permitindo o término da simulação via o número máximo de clientes gerados ou tempo máximo de simulação.
3. E ao término apresenta as seguintes informações :

. DO PROCESSAMENTO GERAL

- . Tipo de Término - se por tempo simulado máximo
- se por número de clientes máximo
- . Tempo Simulado Total
- . Número de Clientes Gerados durante a Simulação

. DE CADA FONTE

- . Número de Clientes Gerados
- . Tempo de Geração de cada Cliente dos dados de entrada

. DE CADA SERVIDOR

- . Número de Clientes atendidos
- . Utilização
- . Taxa de Serviço de Clientes (vazão)
- . Tempo de Serviço de Clientes dos dados de entrada

. DE CADA FILA

- . Taxa de Chegada de Clientes na Fila
- . Tamanho Médio da Fila
- . Tempo Médio que o Cliente espera em Fila
- . Número Máximo de Clientes ocorrido
- . Número de Clientes ao término da simulação

2) CARACTERÍSTICAS DOS MODELOS QUE PODEM SER IMPLEMENTADOS

O programa tem por objetivo geral simular o comportamento e coletar estatísticas de modelos de redes de filas abertas. As características específicas dos modelos considerados pelo simulador são as seguintes :

- i. Possui "m" NODOS. Onde cada nodo é composto por uma FILA e um servidor.
- ii. Possui uma ou mais FONTES.
- iii. Possui um dreno, representado por uma FILA sem SERVIDOR.
- iv. As FONTES podem ter geração de CLIENTES com t_g (tempo de geração) exponencial ou uniforme.
- v. O SERVIDOR pode servir os CLIENTES com x (tempo de serviço) exponencial ou uniforme.

OBS : Este simulador não implementa diversas classes contudo, em um item posterior do trabalho serão discutidas modificações que permitem a implementação do simulador com diversas classes de atendimento.

3) ESTRUTURA DE DADOS IMPLEMENTADA

Para implementação do simulador em questão utilizamos as seguintes estruturas de dados para armazenamento das informações dos elementos da rede.

- a. Para as FONTES uma estrutura tipo ARRAY onde cada elemento do ARRAY tem os seguintes campos:

 FONTE_i
 . informações sobre a FONTE_i
 *saida

onde: *saida é um ponteiro que indica o elemento da estrutura de FILAS, que está conectado após a FONTE_i.

- b. Para as FILAS uma estrutura tipo ARRAY onde cada elemento do ARRAY tem os seguintes campos:

 FILAI
 . informações sobre a FILAI
 *saida

onde: *saida é um ponteiro que indica o elemento de SERVIDORES, que está conectado após a FILAI.

- c. Para os SERVIDORES também uma estrutura tipo ARRAY, onde cada elemento do ARRAY tem os seguintes campos :

 SERVIDORI
 . informações sobre o SERVIDORI
 *entrada
 *saida

onde: *entrada é um ponteiro que indica o elemento da estrutura de FILAS, que está conectado antes do SERVIDORI e,
 *saida é um ponteiro que indica o elemento da estrutura de FILAS, que está conectado após o SERVIDORI.

- d. Para a cadeia de EVENTOS foi implementada uma LISTA DUPLAMENTE ENCADEADA, onde cada elemento tem os seguintes campos :

TEMPO - TIPO - *servidor - *fonte - *next - *prec

TEMPO : identifica o tempo simulado de ocorrência do próximo evento (geração de novo cliente ou serviço de um cliente).

TIPO: identifica o tipo do elemento que efetuará o próximo evento : FONTE ou SERVIDOR.

*servidor : ponteiro indicando o SERVIDOR que efetuará o serviço do CLIENTE.

*fonte : ponteiro indicando a FONTE que efetuará a geração de um novo cliente.

*next : ponteiro indicando o elemento posterior da lista de eventos.

*prec : ponteiro indicando o elemento anterior da cadeia de eventos.

4) ALGORITIMO DO SIMULADOR

4.1) DESCRIÇÃO DO SIMULADOR EM BLOCOS :

O simulador é composto dos seguintes blocos mais importantes :

- i. *main()* --> corpo do programa. Efetua o processamento da cadeia de eventos, analisando o próximo evento que deve ocorrer.

.chama as seguintes subrotinas:

```
init()
proc_fonte()
proc_serv()
mostra_result()
apag_event()
```

- ii. *proc_fonte()* --> subrotina responsável pela geração de novos clientes e pela manutenção das informações das fontes e filas.

.chama as seguintes subrotinas:

```
lanc_freg()
gera_tempo()
init_event()
insere_event()
```

- iii. *proc_serv()* -->. subrotina responsável pelo serviço dos clientes e modificação das informações dos servidores e filas.

.chama as seguintes subrotinas:

```
lanc_freg()
gera_tempo()
init_event()
insere_event()
```

- iv. *lanc_freg()* -->.subrotina que lança um cliente na fila posterior ao elemento em questão (FONTE ou SERVIDOR), também modofoca as informações de filas e servidores.

.chama as seguintes subrotinas:

```
gera_tempo()
init_event()
insere_event()
```

- v. *init()* -->.subrotina responsável pela leitura do arquivo de entrada de dados com o modelo a ser simulado e inicialização das estruturas de dados.

- vi. *mostra_result()* -->.subrotina responsável pela apresentação dos resultados da simulação.

- vii.*init_event()* -->.subrotina responsável pela inicialização de mais um elemento na cadeia de EVENTOS.

- viii.*insere_event()* -->.subrotina responsável pela inserção do elemento inicializado na subrotina anterior na lista da cadeia de EVENTOS.

- ix. *apag_event()* -->.subrotina responsável pela deleção do elemento da lista da cadeia de EVENTOS que já foi processado no tempo simulado atual.

- x. *gera_tempo()* -->.subrotina responsável pela geração do tempo simulado aleatório de ocorrencia do próximo evento.

4.2) DETALHAMENTO DO ALGORITMO DAS SUBROTINAS MAIS IMPORTANTES

i. main()

INICIO . **Faça** enquanto tempo simulado ou numero de clientes não é condição de término de simulação

 .**Retira** o primeiro elemento da lista da cadeia de EVENTOS (pois tem o menor tempo da próxima ação)

 .**Testa** se TEMPO SIMULADO DE EVENTO \neq TEMPO SIMULADO ATUAL

 .Se SIM : faz TEMPO SIMULADO ATUAL = TEMPO SIMULADO DO EVENTO

 .**Se** o EVENTO for GERAÇÃO de cliente;

 .Chama *proc_fonte(*fonte)*

 .Chama *apag_event()*

 .Volta para INICIO

 .**Se** o EVENTO for SERVIÇO de cliente

 .Chama *proc_serv(*servidor)*

 .Chama *apag_event()*

 .Volta ao INICIO

FIM INICIO .Chama *mostra_result()*

ii. proc_fonte(*fonte)

 .**Atualiza** informações da fonte

 .**Lança** o cliente criado na estrutura adiante, chamando a subrotina *lanc_freg(*saida)*

 .**Gera** o tempo de ocorrência do próximo cliente a ser criado por esta fonte, chamando a subrotina *gera_tempo(tipo, t_proc_medio, t_proc_inf, t_proc_sup)*

 .**Cria** o novo evento de geração de cliente chamando a subrotina: *init_event(COD_FONTE, *fonte, t_event)*

 .**Inserere** o novo evento na cadeia de EVENTOS com a subrotina *insere_event(*event)*

iii. proc_serv(*serv)

 .**Atualiza** informações do SERVIDOR

 .**Libera** o servidor

 .**Lança** o cliente servido na estrutura adiante, chamando a subrotina *lanc_freg(*saida)*

 .**Testa** se na estrutura anterior (FILA) tem clientes a serem servidos

 .**Se** SIM

 .**Atualiza** estatísticas da FILA de entrada

 .**Gera** o tempo de término do serviço do cliente adquirido, chamando a subrotina *gera_tempo(tipo, t_proc_medio, t_proc_inf, t_proc_sup)*

 .**Coloca** servidor em serviço

 .**Cria** o novo evento de serviço de cliente chamando a subrotina *init_event(COD_SERVIDOR, *servidor, t_event)*

 .**Inserere** o novo evento na cadeia de EVENTOS com a subrotina *insere_event(*event)*

iv. lanc_freg(*fila)

 .**Testa** se a fila é diferente de sorvedouro

 .**Se** SIM

.**Testa** se o servidor da fila em questão está livre (ou seja não tem nenhum cliente no sistema).

.**Se SIM**

.**Atualiza** estatísticas da FILA de entrada

.**Gera** o tempo de término do serviço do cliente adquirido, chamando a subrotina *gera_tempo(tipo, t_proc_medio, t_proc_inf, t_proc_sup)*

.**Coloca** servidor em serviço

.**Cria** o novo evento de serviço de cliente chamando a subrotina *init_event(COD_SERVIDOR, *servidor, t_event)*

.**Insere** o novo evento na cadeia de EVENTOS com a subrotina *insere_event(*event)*

.**Sai** da subrotina *lanc_freg()*

.**Atualiza** informações da fila colocando o cliente em fila

4.3) DESCRIÇÃO DOS ARQUIVOS FONTES QUE COMPOEM O SIMULADOR

Para o desenvolvimento deste simulador utilizou-se a filosofia da partição das subrotinas e outras especificações em arquivos diferentes constituindo depois uma compilação do tipo PROJECT utilizando o compilador TURBO C versão 2.0 (ou MAKE do C da MICROSOFT) para gerar o arquivo EXE final. A seguir são descritos os arquivos fontes:

- i. STRUCT.H --> arquivo contendo toda definição das estruturas de dados usada no simulador
- ii. INCLUDE.H --> arquivo contendo os DEFINES que são usados por todas as subrotinas do programa
- iii. GLOBAL.H --> arquivo contendo as variáveis globais usadas nas subrotinas
- iv. MAIN.C --> arquivo contendo a rotina main()
- v. INIT.C --> arquivo contendo a subrotina init()
- vi. FONTE.C --> arquivo contendo a subrotina proc_fonte()
- vii. SERV.C --> arquivo contendo a subrotina proc_serv()
- viii. FILA.C --> arquivo contendo a subrotina lanc_freg()
- ix. LISTA.C --> arquivo contendo as subrotinas apag_event(), insere_event() e init_event()
- x. GERAL.C --> arquivo contendo uma subrotina para reservar espaço em memória para string de caracteres
- xi. RESP.C --> arquivo contendo a subrotina mostr_result()
- xii. GERA.C --> arquivo contendo a subrotina gera_tempo
- xiii. SIMULA.HLP --> arquivo contendo as informações para a formação do arquivo de entrada de dados do modelo a ser simulado
- xiv. SIMULA.EXE --> nome do arquivo com o código executável

5) OPERAÇÃO DO SIMULADOR

5.1) ENTRADA DE DADOS

A implementação do simulador em questão usa, para execução da simulação, um arquivo contendo a descrição do modelo.

Esta técnica para inserção do modelo a ser simulado evita que haja compilação do programa toda vez que se deseje fazer nova simulação além de, evidentemente melhorar apresentação geral do software.

Nas próximas páginas é apresentada uma listagem do arquivo SIMULA.HLP com as especificações do arquivo de entrada de dados. Além disto exemplos de arquivos para simulação podem ser encontrados nos anexos [1,2 e 3].

5.2) EXECUÇÃO DE UMA SIMULAÇÃO

Para efetuar-se uma simulação apenas digita-se :

> **SimRdAb** <nome do arquivo do modelo a simular>

O programa quando estiver efetuando a leitura do arquivo procede a críticas dos campos, visando evitar posteriores erros de processamento. A cada erro encontrado no arquivo de entrada de dados o programa para a execução, dizendo qual tipo de erro ocorrido e a linha do arquivo em que se encontra.

Para facilitar a visualização do processamento do modelo foi colocada a opção **V** <n> (no arquivo de entrada de dados), onde pode-se optar pelos tres seguintes tipos de visualização durante o processamento:

[n=0] --> nada é apresentado durante a simulação. Apenas ao final são apresentados os resultados da simulação. Esta é a forma em que o simulador processa mais rápido.

[n=1] --> é apresentada na tela durante o processamento; o tempo de simulação e o número de clientes gerados. Esta forma aumenta em 15 vezes aproximadamente o processamento do mesmo modelo do item anterior.

[n=2] --> apresenta a cadeia de eventos atual durante o processamento. é a forma mais lenta de processamento.

é importante observar que o simulador em questão finaliza o processamento sob duas circunstâncias ; **ou** o tempo simulado é maior do que o tempo simulado máximo especificado **ou** o número de clientes gerado é maior do que o número de clientes especificado. Ou seja, a simulação é finalizada no momento que ocorra primeiro um dos eventos.


```
/*****  
    Sintaxe dos arquivos de entrada de dados do simulador  
*****/
```

CARACTERES RESERVADOS :

V D N E S F e s

PARAMETROS DA SIMULACAO :

V <n>

<n> : integer : Visualizacao durante a simulacao
0 - nada
1 - tempo simulado e numero de clientes atual
2 - cadeia de eventos

D <duracao>

<duracao> : integer : Duracao da simulacao

N <data>

<data> : integer : Numero maximo de clientes a serem gerados

ELEMENTOS DE CIRCUITO :

Fontes :

E <tipo> <nome> <temp1> <temp2>

<tipo> : integer : tipo da geracao 1- exponencial
2- uniforme

<nome> : string : nome da fonte

<temp1> : float: tempo de geracao de um cliente (tg) se;
tipo exponencial => media da distribuicao
tipo uniforme => limite inferior da dist.

<temp2> : float: tempo de geracao de um cliente (tg) se;
tipo exponencial => deve ser colocado zero
tipo uniforme => limite superior da dist.

Servidores :

S <tipo> <nome> <temp1> <temp2>

<tipo> : integer : tipo de servico 1- exponencial
2- uniforme

<nome> : string : nome do servidor

<temp1> : float: tempo de servico de um cliente (x) se;
tipo exponencial => media da distribuicao
tipo uniforme => limite inferior da dist.

<temp2> : float: tempo de servico de um cliente (x) se;
tipo exponencial => deve ser colocado zero
tipo uniforme => limite superior da dist.

Filas :

F <nome>

<nome> : string : nome da fila

PARAMETROS DOS ELEMENTOS :

Entradas :

e <entrada>

<entrada> : integer : No do elemento ligado.

Saidas :

s <saida>

<saida> : integer : No do elemento ligado.
Em caso de fila o sorvedouro deve ter 0.

FORMATOS DE DESCRICAO DOS ELEMENTOS :

Fontes :

E

S

Servidores :

S

e

s

Filas :

F

s

- Uma linha cujo primeiro caractere n...o e reservado sera considerada como linha de comentarios.
- O numero de um elemento e implicitamente a sua ordem na lista.

*/

5.3) APRESENTAÇÃO DOS RESULTADOS

Na introdução foram sucintamente apresentadas as estatísticas de saída do simulador. Para não haver repetição de informações, nos anexos [1,2 e 3], podem ser vistos os documentos de saída para os três modelos exemplificados.

A seguir é apresentado um memorial de cálculo das estatísticas obtidas para cada modelo simulado, fazendo referência aos campos das estruturas de dados e variáveis globais usadas no programa.

ESTATÍSTICAS GERAIS

variáveis

TEMPO SIMULADO TOTAL : t_simul
NÚMERO DE CLIENTES GERADOS : n_tot (FONTEi)

.ESTATÍSTICAS DAS FONTES

NÚMERO DE CLIENTES GERADOS : n_tot (FONTEi)

.ESTATÍSTICAS DOS SERVIDORES

UTILIZAÇÃO :
$$\frac{t_{tot_ocupado} (SERVi)}{t_simul}$$

TAXA DE SERVIÇO (VAZÃO) :
$$\frac{n_tot (SERVi)}{t_simul * UTILIZACAO}$$

TEMPO DE SERVIÇO POR CLIENTE :
$$\frac{1}{TAXA DE SERVIÇO}$$

CLIENTE SENDO SERVIDO NO FINAL DA SIMULAÇÃO : frequês (SERVi)

.ESTATÍSTICAS DAS FILAS

TAXA DE CHEGADA :
$$\frac{n_tot (FILAi)}{t_simul}$$

TAMANHO MÉDIO DA FILA :
$$\frac{tot_tam_acum (FILAi)}{t_simul}$$

TEMPO MEDIO DE UM CLIENTE NA FILA :
$$\frac{TAM. MEDIO DA FILA}{TAXA DE CHEGADA}$$

NÚMERO MÁXIMO DE CLIENTES OCORRIDO : n_max (FILAi)

NÚMERO CLIENTES NA FILA AO TÉRMINO DA SIMULAÇÃO : n_cour (FILAi)

6) APLICAÇÕES

Visando ilustrar o funcionamento do simulador, a seguir, serão apresentados tres modelos.

Para cada um dos modelos serão apresentados os seguintes itens :

Apresentação dos modelos graficamente, com as características dos elementos da rede de filas

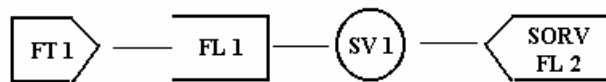
Apresentação de forma condensada dos resultados obtidos através do simulador

Apresentação dos cálculos teóricos, visando análise no item posterior.

Os resultados do simulador da forma completa estão apresentados no anexos [1, 2 e 3] para os modelos 1, 2 e 3 respectivamente. Em cada anexo encontra-se o arquivo de entrada de dados usado para simulação e os resultados.

6.1) MODELO 1 (1 SERVIDOR e 1 FONTE)

6.1.1) APRESENTAÇÃO DO MODELO 1



Dados dos Elementos :

FONTE :

SERVIDOR :

$$tg_1 = 2.5 \Rightarrow \lambda = 0.4 \quad x_1 = 0.5 \Rightarrow \mu = 2$$

onde:

tg => tempo de geração de um cliente [UTS/CLIENTE]

λ => taxa de geração de clientes [CLIENTE/UTS]

x => tempo de serviço de um cliente [UTS/CLIENTE]

μ => taxa de serviço de clientes [CLIENTE/UTS]

6.1.2) RESULTADOS CONDENSADOS DE SIMULAÇÃO DO MODELO 1

É importante lembrar que os resultados de forma completa estão apresentados no anexo [1].

SIMULAÇÃO EFETUADA COM : **10.000** CLIENTES

TEMPO DE SIMULAÇÃO TOTAL : **24.855,75** UTS

SERVIDOR : **1**

UTILIZAÇÃO (ρ) = 0.20

TAXA DE SERVIÇO (μ) 2.00 [CLIENTE/UTS]

TEMPO DE SERVIÇO (x) 0.50 [UTS/CLIENTE]

obs : o TEMPO DE SERVIÇO (x) acima é calculado como (1 / TAXA DE SERVIÇO) simulada e visa compor a soma para obtenção do tempo de sistema do simulador.

FILAS : **1**

TAXA DE CHEGADA (λ) 0.40 [CLIENTE/UTS]

TEMPO MEDIO EM FILA (w) 0.12 [UTS/CLIENTE]

SISTEMA : **1**

TEMPO DE SISTEMA (T) 0.62[UTS] : T = x + w

6.1.3) CÁLCULOS TEÓRICOS DO MODELO 1

Do modelo M/M/1 da teoria das filas apresentado no anexo [4] temos os seguintes resultados :

SERVIDOR : **1**

UTILIZAÇÃO (ρ) 0.20

TAXA DE SERVIÇO (μ) 2.00 [CLIENTE/UTS]

FILAS : **1**

TAXA DE CHEGADA (λ) 0.40 [CLIENTE/UTS]

SISTEMA : **1**

TEMPO DE SISTEMA (T) 0.63 [UTS/CLIENTE]

```
/ ***** EXEMPLO 1 *****
/ SISTEMA UMA FONTE, UMA FILA E UM /SERVIDOR
/ 1 Fonte, 1 fila, 1 servidor
/PARAMETROS DA SIMULACAO :
/-----
V 0
D 10000000.
N 10000
/ELEMENTOS DE CIRCUITO :
/-----
/Fontes :
/-----
/ FONTE
E 1 fonte1 2.5 0.0
s 1
/Filas :
/-----
/ FILA 1
F fila1
s 1
/ FILA SORVEDOURO
F fila2
s 0
/Servidores :
/-----
/SERVIDOR 1
S 1 serv1 0.5 0.0
e 1
s 2
```