**Trabalho 1: Simulador de Filas em Rede**

Disciplina: Simulação e Métodos Analíticos

Professor: Afonso Henrique Correa de Sales

Aluno: Diego Osmarin Basso

<https://github.com/DiegoOsmarinBasso/simulador-de-filas-em-rede>

**1. Definição dos componentes do estado das filas**

Junto com esta entrega encontram-se dois exemplos de arquivos (*ExemploDeArquivo\_1.txt* e *ExemploDeArquivo\_2.txt*) de entrada para o programa. Para testá-los deve ter instalado na máquina Java versão 1.8 ou maior e chamar o programa conforme exemplo: **java -jar SimuladorDeFilasEmRede.jar ExemploDeArquivo\_2.txt**.

O usuário também pode montar seu arquivo para executar, basta montar um arquivo conforme especificações a seguir, mantendo a ordem apresentada.

**INFINITE\_QUEUE\_SIZE (opcional):** as filas podem ser especificadas múltiplos servidores, diferentes tempos de chegada e atendimento e diferentes capacidades (finita ou infinita). No entanto, para efeitos de simulação, filas infinitas são consideradas com capacidade igual a . O tamanho a ser considerado para uma fila infinita, também pode ser especificado através do parâmetro *INFINITE\_QUEUE\_SIZE*.

**QUEUES (obrigatório):** as filas podem ser definidas utilizando-se de quatro a sete parâmetros:

* Quatro parâmetros: fila INFINITA que NÃO possui entrada externa;
* Cinco parâmetros: fila FINITA que NÃO possui entrada externa;
* Seis parâmetros: fila INFINITA que possui entrada externa;
* Sete parâmetros: fila FINITA que possui entrada externa;

De modo geral as filas são definidas conforme sintaxe abaixo, onde os colchetes representam parâmetros opcionais:

***Nome\_Fila Qtd\_Servidores [Capacidade] [Min\_Chegada Max\_Chegada] Min\_Servico Max\_Servico***

**FIRST\_ARRIVAL (obrigatório):** utilizado para definir o tempo de primeira chegada na fila conforme sintaxe abaixo.

***Nome\_Fila Primeira\_Chegada***

**NETWORK (opcional):** utilizado para o roteamento entre filas e saída do sistema. A probabilidade de roteamento deve ser informada com um número entre e . Se a soma das probabilidades de roteamento de uma determinada fila ultrapassar , o programa apresentará um erro e não executará. Se a soma das probabilidades de roteamento de uma determinada for menor que , a probabilidade restante para atingir os será a de saída do sistema. Para simular uma única fila, não incluir este parâmetro. Utilização conforme sintaxe abaixo.

***Nome\_Fila\_Origem Nome\_Fila\_Destino Probabilidade\_De\_Roteamento***

**RAND\_PER\_SEED (opcional):** quantidade de número aleatórios gerados para cada execução. Após a utilização do último aleatório o programa para de simular entradas e saídas e contabiliza os tempos totais das filas. Se não informado utiliza o valor padrão de .

**SEEDS (opcional):** sementes a serem utilizadas pelo gerador congruente linear. A quantidade de sementes informadas será a quantidade de vezes que o sistema de filas será simulado. As sementes devem ser informadas uma abaixo da outra e devem ser um número positivo inteiro que não podem ultrapassar , ou seja, o máximo permitido para inteiros em Java.

**NUMBER\_RANDOM\_SEEDS (opcional):** quantidade de sementes obtidas a partir do relógio do sistema. Se o parâmetro *SEEDS* for informado, estas serão utilizadas, e este parâmetro não terá efeito.

O ExemploDeArquivo\_2.txt exemplifica o de uso do parâmetro *SEEDS*; e o *ExemploDeArquivo\_1.txt* exemplifica o de uso do parâmetro *NUMBER\_RANDOM\_SEEDS*.

**2. Rotina de inicialização do simulador**

Uma classe chamada *Queue* foi definida e cada instância dessa classe guarda as informações de uma determinada fila. Estas instâncias são criadas conforme as informações passadas no arquivo de entrada como nome, quantidade de servidores, capacidade, tempos de chegada e de saída, tempo da primeira chegada e probabilidades de roteamento.

Para utilização no simulador, estas filas são colocadas em um vetor (*ArrayList<Queue>*) e as demais propriedades de cada fila, tamanho atual e perda de clientes, são inicializadas com .

O simulador também é inicializado com informações passadas no arquivo de entrada se estas forem informadas. Senão ele utiliza os seguintes valores padrão, capacidade de uma fila infinita: ; número de execuções: ; e quantidade de números pseudoaleatórios gerados: .

**3. Rotinas de tratamento de eventos**

No simulador, três tipos de eventos são definidos: CHEGADA, SAÍDA e PASSAGEM (ARRIVAL, DEPARTURE e PASSAGE respectivamente). Inicialmente são agendadas as chegadas definidas pelo parâmetro obrigatório FIRST\_ARRIVAL.

Para tanto foi definida a classe *Event*, que possui apenas quatro atributos: tipo de evento, tempo de evento, fila do evento e fila de destino (utilizado apenas para passagem de uma fila à outra). O algoritmo desenvolvido seguiu os modelos dos slides de aula onde tratou-se os eventos conforme abaixo.

|  |  |
| --- | --- |
| CHEGADA:  contabiliza os tempos em todas filas  se há espaço na fila  incrementa tamanho da fila  se há servidores disponíveis  verifica as probabilidades de roteamento  se roteamento  agenda PASSAGEM  senão  agenda SAÍDA  senão  contabiliza perda  agenda próxima CHEGADA | PASSAGEM (da filaX para a filaY):  contabiliza os tempos em todas filas  decrementa tamanho da filaX  se há clientes esperando atendimento na filaX  verifica as probabilidades de roteamento filaX  se roteamento  agenda PASSAGEM  senão  agenda SAÍDA  se há espaço na filaY  incrementa tamanho da filaY  se há servidores disponíveis na filaY  verifica as probabilid. de roteamento filaY  se roteamento  agenda PASSAGEM  senão  agenda SAÍDA  senão  contabiliza perda filaY |
| SAÍDA:  contabiliza os tempos em todas filas  decrementa tamanho da fila  se há clientes esperando atendimento na fila  agenda SAÍDA |