Amanda de Souza Caetano

Antônio Angelo Passini Júnior

Trabalho de Programação

Parte 1: Implementação do Modelo M/M/1

Professor Carlos Lopes

Uberlândia  
2019

1. Instruções

1.1. Download e Execução

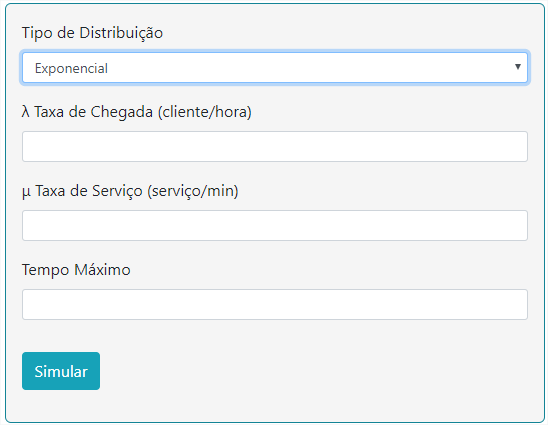
Primeiramente, para ter acesso ao programa desenvolvido, é necessário baixar e descompactar seus arquivos. Tais arquivos estão disponíveis no moodle e também no link <https://drive.google.com/open?id=1l3STQ4rd7g10n1tOgk1UB_x7fXHXhZJ9>.

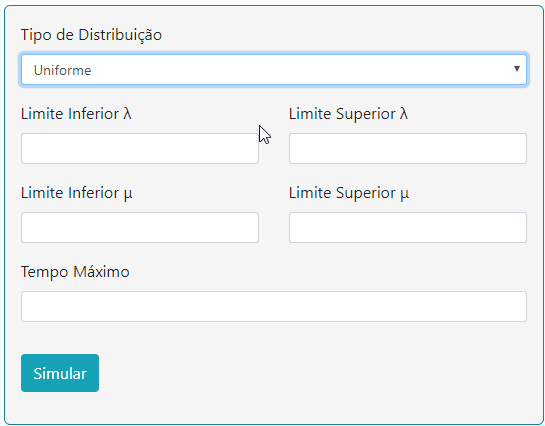
Uma vez baixado e descompactado os arquivos, basta abrir o arquivo *index.html* em um navegador web[[1]](#footnote-1), como Google Chrome ou Mozilla Firefox.

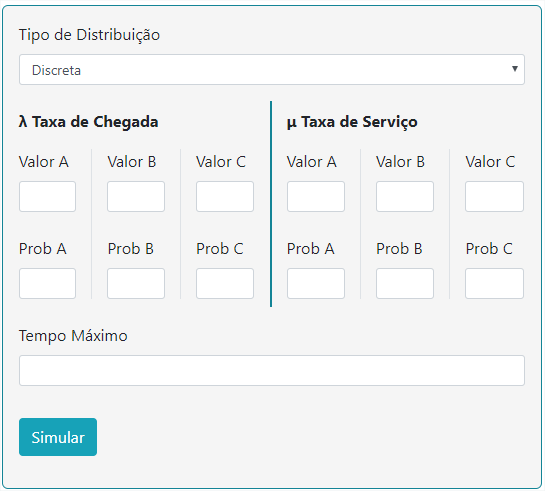
1.1. Como Simular

A tela principal[[2]](#footnote-2) do programa contém somente um formulário para obtenção dos dados necessários para a simulação. O primeiro passo é escolher qual tipo de distribuição você pretende utilizar, sendo elas: exponencial, uniforme e discreta.

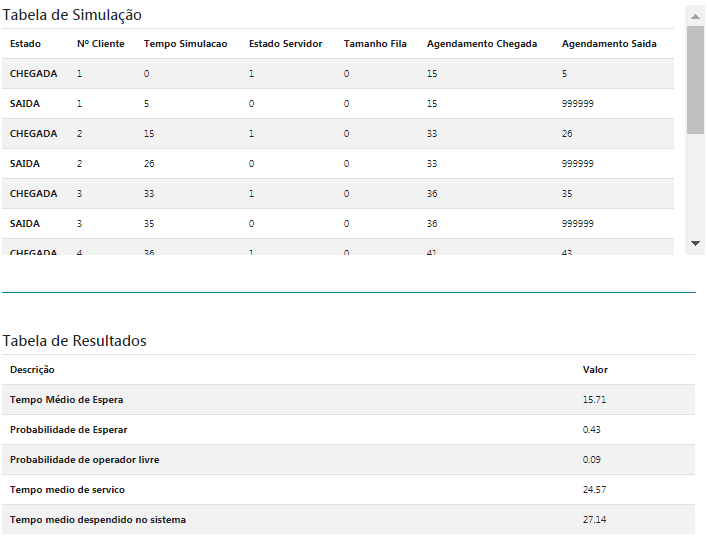
Uma vez selecionado o tipo, novos campos apareceram, de acordo com a escolha anterior, como mostra as figuras abaixo:

  
Figura 1: Formulário para distribuição discreta

  
Figura 2: Formulário para distribuição uniforme

  
Figura 3: Formulário para distribuição discreta

Depois de preenchido o formulário[[3]](#footnote-3), basta clicar no botão **Simular.** Duas tabelas serão mostradas, uma contendo os dados da simulação e outro com análises de tais dados, como exemplificado a seguir.

  
Figura 4: Tabelas de resultados da simulação

É importante notar que para visualizar todas as linhas de uma tabela, pode ser necessário utilizar a barra de rolagem que fica na lateral direta da mesma.

Há também um botão no topo da página que permite fazer nova simulação.

2. Informações Técnicas

Para o desenvolvimento desse projeto foram usadas as linguagens JavaScript, HTML5 e CSS. Além disso, também foram usadas as bibliotecas Bootstrap, JQuery e Popper. Também houve o apoio das ferramentas Visual Studio Code, Google Chrome Developer Tools e GitHub.

O programa possui como programa principal “Simulacao.js” no qual o mesmo possui funções “inEvent()” e “outEvent()” responsáveis por fazer o controle de entrada e saída de clientes dentro do sistema. Como funções auxiliares tem-se “getInEvent()” e “getOutEvent()” responsáveis por fazer o controle de qual distribuição será usada.

No roteiro inicial são setadas todas as variáveis de controle além de criar uma lista de clientes no sistema e clientes que já foram atendidos e um operador para manter o controle do sistema. O programa entra em um while em que enquanto o tempo total for menor do que o desejado e não haver mais ninguém no sistema será feita a simulação.

Se o tempo de chegada previsto for menor que o tempo de saída previsto acontecerá um evento de chegada do cliente no servidor, caso contrário, será acontecerá um evento de saída do cliente no servidor.

Após acabar a simulação (programa sai do while) é gerada uma tabela com os dados de chegadas e saídas e uma tabela com os cálculos estatísticos. Os cálculos foram feitos seguindo as fórmulas propostas nos slides vistos em aula.

Os atores, “cliente” e “operador” são classes criadas para facilitar no controle e armazenamento de dados gerados pelo sistema.

Para gerar um número aleatório é utilizado o “GNA.js” no qual existe uma semente gerada ao criar o objeto, sendo essa semente o tempo em milissegundos do sistema e para gerar um número aleatório usamos a função: (semente = (a \* semente % m))/m

Para a geração das variáveis aleatórias é utilizado o “FGVA.js” possuindo como filho três distribuições: Exponencial; Uniforme; Discreta.

1. Foram testados somente os navegadores citados. [↑](#footnote-ref-1)
2. Para melhor visualização dos dados, sugere-se uma resolução de no mínimo 700 pixels de largura. [↑](#footnote-ref-2)
3. Preencha o formulário corretamente e com atenção, uma vez que não há validação de dados no sistema. Usar como valores zero ou números negativos poderão acarretar em travamento do sistema. [↑](#footnote-ref-3)