Simulador do algoritmo de roteamento por rumores

PUC
RIO

Professora: Noemi Rodriguez
Alunos: Fernando Homem da Costa
Thiago Sousa Bastos

INF2545 - Sistemas Distribuídos
Departamento de Informática
Pontifícia Universidade Católica do Rio de Janeiro
18 de junho de 2021

1 Introdução

O presente relatório visa descrever o desenvolvimento de um simulador para o algoritmo de roteamento por rumores proposto por Braginsky e Estrin [1].

O simulador em questão utiliza o protocolo MQTT [2] para fazer a comunicação entre os diversos nós do algoritmo utilizando um modelo de publicação/inscrição. Para a visualização do algoritmo, uma implementação em Lua [3] utilizando o *framework* LOVË [4] foi feita, de modo que os nós são caixas com botões e estes botões são responsáveis por disparar eventos e consultas, conforme especificado no artigo do algoritmo.

2 Tecnologias e organização dos diretórios

Para a implementação do algoritmo, a infraestrutura básica do código consistem em utilizar o *framework* LOVË 11.3, o protocolo Mosquitto 1.6.9 e Lua 5.1. Além disso, para a serialização das mensagens foi utilizada a biblioteca json-lua 0.1.2.

Dentro da pasta do código existem quatro diretórios: *json-lua*, *mqtt-lua*, *config* e *log*. O diretório *json-lua* contém a biblioteca json-lua. Já o diretório *mqtt-lua* contém os arquivos mqtt_library.lua e utility.lua, fornecidos para a implementação do algoritmo.

Por decisão de projeto, nós optamos por enviar através da linha de comando o nome do arquivo de configuração de cada nó da rede. Estes arquivos de configuração se encontram na pasta *config*. A Listagem 1 mostra um exemplo de arquivo de configuração presente neste diretório.

```
1       config = {
2         id = 1,
3         topic = 'channel1',
4         subscribedTo = {
5           'channel2',
6           'channel3'
7         },
8         numberOfNodes = 4
9       }
```

Listing 1: Exemplo de arquivo de configuração de nó.

O diretório *log* é utilizado para guardar os registros das atividades executadas pelos nós da rede. A atividade de cada nó é registrada em um arquivo de extensão CSV com a data, horário, *id* do nó e a mensagem enviada/recebida. A Listagem 2 ilustra o exemplo de um registro de atividades.

```
2021-06-17 11:15:19, NODE4, {
  "payload": "Chegou ao destino!",
  "hops":0,
  "sender":4,
  "type":"consulta",
  "recipient": 4,
  "path": "134"
}

2021-06-17 11:15:28, NODE4, Encerrando o programa
```

Listing 2: Exemplo de arquivo de *log* de nó.

3 Instruções de uso

A inicialização do programa é feita em uma única etapa: executar o *script* start.sh desenvolvido na linguagem shell. Para isso, executar o seguinte comando em sua linha de comando/terminal:

```
# bash start.sh -l <qtd de linhas> -c <qtd de colunas>
```

onde a quantidade de linhas se refere ao número de nós presentes em cada coluna e a quantidade de colunas se refere ao número de nós presentes em cada linha do grid.

Por esse trabalho ter sido desenvolvido em dupla, optamos por utilizar uma instância do Mosquitto *broker* no Google Cloud. Desta maneira, foi possível desenvolver e testar a aplicação simultaneamente. O endereço TCP/IP deste *broker* é **34.145.30.230:1883**. É importante mencionar que cada nó da rede é representado por um desenho feito no LÖVE, conforme ilustrado na Figura 1.

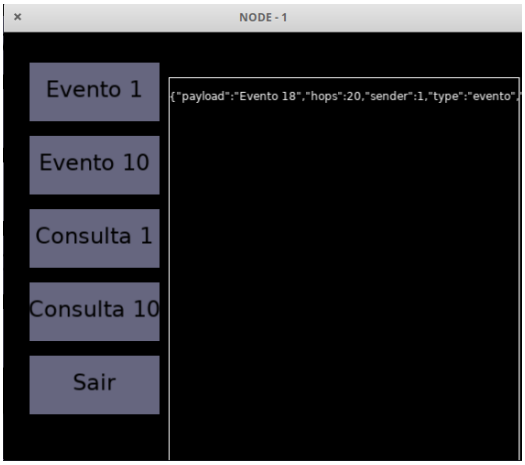

Figura 1: Interface de um nó da rede.

4 Desenvolvimento

O desenvolvimento desse projeto é baseado no trabalho *"Rumor Routing Algorthim for Sensor Networks"* [1] e utiliza as seguintes tecnologias:

Como mencionado na Seção 2, o *"script"* **start.sh** é utilizado para inicializar a aplicação. Para isso, ele gera para cada nó, um arquivo de configuração. A estrutura do arquivo de configuração está apresentada no Exemplo 1. Após isso, o **start.sh** realiza "*linhas* × *colunas*" chamadas ao sistema operacional solicitando a execução comando do ***"love"***, passando com parâmetros o diretório local e o arquivo de configuração de cada nó.

5 Testes

Nessa seção executaremos um teste contendo quatro nós, mas esse número pode ser estendido, apenas alterando os parâmetros "-l"e "-c" do *script* bash . A fim de melhor verificar o funcionamento do programa, as ações executadas são registradas em arquivos e exibidas na tela.

A Figura 2 ilustra o caso de teste de rede de 4 nós. Ao apertar o botão "Consulta 8", o evento é propagado para os vizinhos até encontrar o seu destino, que é o nó #4. Após isso, o evento retornará a sua origem, como indica a seta superior vermelha.

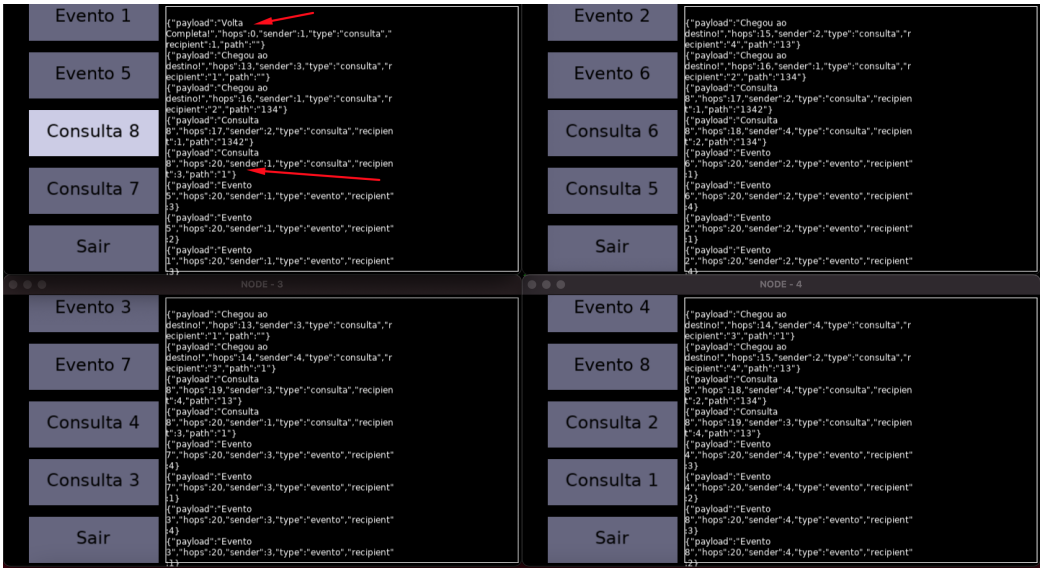

Figura 2: Nó #1 - Efetuando consulta “Consulta 8”

No caso da Figura 3, ela apresenta o caso de teste de rede de 4 nós, porém realizando duas consultas consecutivas. Ao apertar o botão “Consulta 8” do nó #1, o evento é propagado para os vizinhos até encontrar o seu destino, que é o nó #4. Após isso, o evento retornará a sua origem, que é o nó #1. A segunda consulta é a partir do nó #4, que propagará a “Consulta 2” até atingir o seu destino.

Figura 3: Nó #4 - Efetuando Consultado “Consulta 2”

As Figuras 4 e 5 mostram o caso de teste em que há nove nós na rede. Antes de realizarmos uma consulta, todos os eventos são disparados. Após isso, a partir do nó #5 realizamos uma consulta "Consulta 10", que propaga na rede e retorna para sua origem.

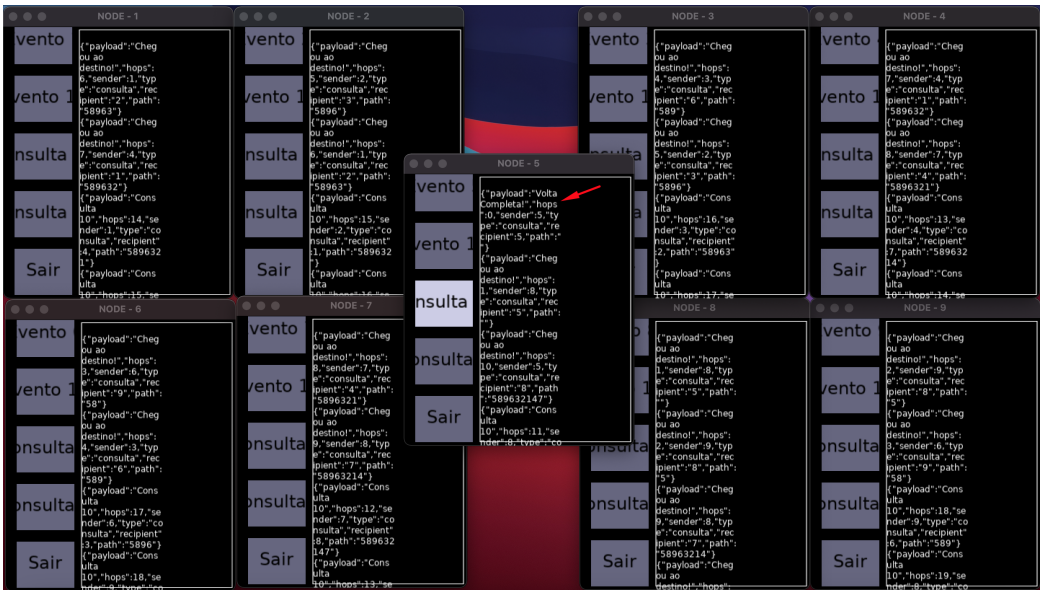

Figura 4: Nó #3 - Efetuando Consulta “Consulta 10”

Devido à limitação de espaço na interface gráfica, não é possível exibir todas as etapas de uma consulta em exemplos com muitos nós. A Figura 5 apresenta o registro de atividades daquele nó, facilitando identificação das ações por data e hora.

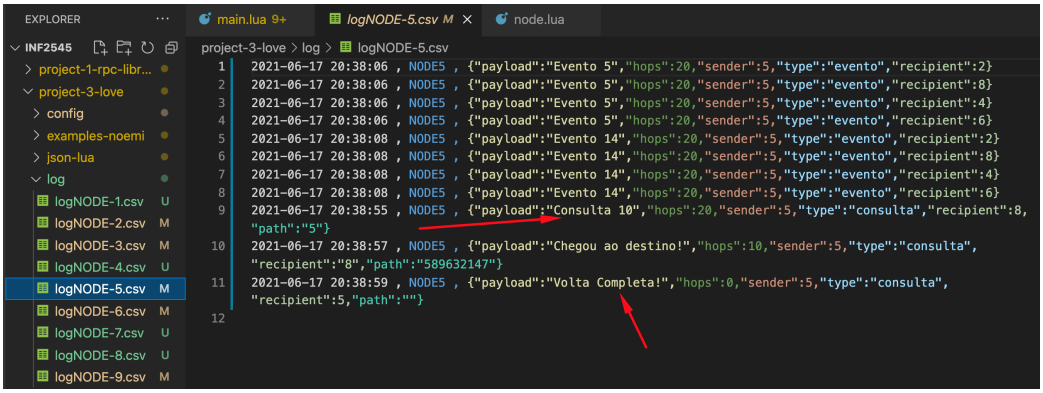

Figura 5: Log Nó #3 - Confirmação dos eventos

Para validar o funcionamento da solução, nós imprimimos a tabela de rotas na linha comando e jogamos essa saída para um arquivo chamado output.txt. Os exemplos 3, 4, 5 e 6 ilustram esse cenário de teste.

```
1 FOR NODE 1 { }
2 FOR NODE 1 { ["Evento 2"] = { ["direction"] = 2,
  ↪ ["distance"] = 1, } ,}
3 FOR NODE 1 { ["Evento 2"] = { ["direction"] = 2,
  ↪ ["distance"] = 1,} , ["Evento 3"] = { ["direction"] =
  ↪ 3, ["distance"] = 1, } ,}
4 FOR NODE 1 { ["Evento 2"] = { ["direction"] = 2,
  ↪ ["distance"] = 1 ,} , ["Evento 3"] = { ["direction"] =
  ↪ 3, ["distance"] = 1, } ,}
5 FOR NODE 1 { ["Evento 2"] = { ["direction"] = 2,
  ↪ ["distance"] = 1 ,} , ["Evento 7"] = { ["direction"] =
  ↪ 3, ["distance"] = 1,} , ["Evento 3"] = { ["direction"]
  ↪ = 3, ["distance"] = 1, } ,}
6 FOR NODE 1 { ["Evento 2"] = { ["direction"] = 2,
  ↪ ["distance"] = 1, } , ["Evento 7"] = { ["direction"] =
  ↪ 3, ["distance"] = 1, } , ["Evento 3"] = {
  ↪ ["direction"] = 3, ["distance"] = 1, } , }
7 FOR NODE 1 { ["Evento 2"] = { ["direction"] = 2,
  ↪ ["distance"] = 1, } , ["Evento 7"] = { ["direction"] =
  ↪ 3, ["distance"] = 1, } , ["Evento 3"] = {
  ↪ ["direction"] = 3, ["distance"] = 1, } , ["Evento 6"]
  ↪ = { ["direction"] = 2, ["distance"] = 1, } , }
```

Listing 3: Tabela de rotas - Nó #1

```
1  FOR NODE 3 { }
2  FOR NODE 3 { ["Evento 1"] = { ["direction"] =
   ↪  1,["distance"] = 1,} ,}
3  FOR NODE 3 { ["Evento 1"] = { ["direction"] =
   ↪  1,["distance"] = 1,} ,}
4  FOR NODE 3 { ["Evento 1"] = { ["direction"] =
   ↪  1,["distance"] = 1,} ,["Evento 4"] = { ["direction"] =
   ↪  4,["distance"] = 1,} ,}
5  FOR NODE 3 { ["Evento 1"] = { ["direction"] =
   ↪  1,["distance"] = 1,} ,["Evento 4"] = { ["direction"] =
   ↪  4,["distance"] = 1,} ,}
6  FOR NODE 3 { ["Evento 4"] = { ["direction"] =
   ↪  4,["distance"] = 1,} ,["Evento 1"] = { ["direction"] =
   ↪  1,["distance"] = 1,} ,["Evento 8"] = { ["direction"] =
   ↪  4,["distance"] = 1,} ,}
7  FOR NODE 3 { ["Evento 4"] = { ["direction"] =
   ↪  4,["distance"] = 1,} ,["Evento 1"] = { ["direction"] =
   ↪  1,["distance"] = 1,} ,["Evento 8"] = { ["direction"] =
   ↪  4,["distance"] = 1,} ,}
8  FOR NODE 3 { ["Evento 4"] = { ["direction"] =
   ↪  4,["distance"] = 1,} ,["Evento 5"] = { ["direction"] =
   ↪  1,["distance"] = 1,} ,["Evento 1"] = { ["direction"] =
   ↪  1,["distance"] = 1,} ,["Evento 8"] = { ["direction"] =
   ↪  4,["distance"] = 1,} ,}
9  FOR NODE 3 { ["Evento 4"] = { ["direction"] =
   ↪  4,["distance"] = 1,} ,["Evento 5"] = { ["direction"] =
   ↪  1,["distance"] = 1,} ,["Evento 1"] = { ["direction"] =
   ↪  4,["distance"] = 0,} ,["Evento 8"] = { ["direction"] =
   ↪  4,["distance"] = 1,} ,}
10 FOR NODE 3 { ["Evento 4"] = { ["direction"] =
   ↪  4,["distance"] = 3,} ,["Evento 5"] = { ["direction"] =
   ↪  1,["distance"] = 1,} ,["Evento 1"] = { ["direction"] =
   ↪  4,["distance"] = 0,} ,["Evento 8"] = { ["direction"] =
   ↪  4,["distance"] = 1,} ,}
```

Listing 4: Tabela de rotas - Nó #3

```
1 FOR NODE 3 { ["Evento 5"] = { ["direction"] =
  ↪ 1,["distance"] = 1,} ,["Evento 1"] = { ["direction"] =
  ↪ 4,["distance"] = 0,} ,["Evento 8"] = { ["direction"] =
  ↪ 4,["distance"] = 1,} ,["Evento 4"] = { ["direction"] =
  ↪ 4,["distance"] = 3,} ,["Evento 6"] = { ["direction"] =
  ↪ 4,["distance"] = 1,} ,}
2 FOR NODE 3 { ["Evento 5"] = { ["direction"] =
  ↪ 1,["distance"] = 1,} ,["Evento 1"] = { ["direction"] =
  ↪ 4,["distance"] = 0,} ,["Evento 8"] = { ["direction"] =
  ↪ 4,["distance"] = 2,} ,["Evento 4"] = { ["direction"] =
  ↪ 4,["distance"] = 3,} ,["Evento 6"] = { ["direction"] =
  ↪ 4,["distance"] = 1,} ,}
3 FOR NODE 3 { ["Evento 5"] = { ["direction"] =
  ↪ 1,["distance"] = 1,} ,["Evento 1"] = { ["direction"] =
  ↪ 4,["distance"] = 0,} ,["Evento 8"] = { ["direction"] =
  ↪ 4,["distance"] = 2,} ,["Evento 4"] = { ["direction"] =
  ↪ 4,["distance"] = 3,} ,["Evento 7"] = { ["direction"] =
  ↪ 4,["distance"] = 2,} ,["Evento 6"] = { ["direction"] =
  ↪ 4,["distance"] = 1,} ,}
4 FOR NODE 3 { ["Evento 5"] = { ["direction"] =
  ↪ 1,["distance"] = 1,} ,["Evento 1"] = { ["direction"] =
  ↪ 4,["distance"] = 0,} ,["Evento 8"] = { ["direction"] =
  ↪ 4,["distance"] = 2,} ,["Evento 4"] = { ["direction"] =
  ↪ 4,["distance"] = 3,} ,["Evento 7"] = { ["direction"] =
  ↪ 4,["distance"] = 2,} ,["Evento 6"] = { ["direction"] =
  ↪ 4,["distance"] = 1,} ,}
5 FOR NODE 3 { ["Evento 5"] = { ["direction"] =
  ↪ 4,["distance"] = 1,} ,["Evento 1"] = { ["direction"] =
  ↪ 4,["distance"] = 0,} ,["Evento 8"] = { ["direction"] =
  ↪ 4,["distance"] = 2,} ,["Evento 4"] = { ["direction"] =
  ↪ 4,["distance"] = 3,} ,["Evento 7"] = { ["direction"] =
  ↪ 4,["distance"] = 2,} ,["Evento 6"] = { ["direction"] =
  ↪ 4,["distance"] = 1,} ,}
```

Listing 5: Continuação: Tabela de rotas - Nó #3

```
1 FOR NODE 3 { ["Evento 5"] = { ["direction"] =
  ↪  4,["distance"] = 1,} ,["Evento 1"] = { ["direction"] =
  ↪  4,["distance"] = 0,} ,["Evento 8"] = { ["direction"] =
  ↪  4,["distance"] = 2,} ,["Evento 4"] = { ["direction"] =
  ↪  4,["distance"] = 3,} ,["Evento 7"] = { ["direction"] =
  ↪  4,["distance"] = 2,} ,["Evento 6"] = { ["direction"] =
  ↪  4,["distance"] = 1,} ,}
2 FOR NODE 3 { ["Evento 5"] = { ["direction"] =
  ↪  4,["distance"] = 1,} ,["Evento 1"] = { ["direction"] =
  ↪  4,["distance"] = 0,} ,["Evento 8"] = { ["direction"] =
  ↪  4,["distance"] = 2,} ,["Evento 4"] = { ["direction"] =
  ↪  4,["distance"] = 3,} ,["Evento 7"] = { ["direction"] =
  ↪  4,["distance"] = 2,} ,["Evento 3"] = { ["direction"] =
  ↪  4,["distance"] = 3,} ,["Evento 6"] = { ["direction"] =
  ↪  4,["distance"] = 1,} ,}
```

Listing 6: Continuação 2: Tabela de rotas - Nó #3

6 Conclusão

Durante o projeto foram encontradas algumas dificuldades, tais como: *"debuggar"* o código para encontrar os erros e verificar o funcionamento correto da solução da proposta. Em relação ao processo de *"debuggar"*, utilizamos a biblioteca Penlight [5], permitindo visualizar o conteúdo de uma tabela e contabilizar o número de parâmetros dentro da tabela. Além disso, no trabalho *"Rumor Routing Algorthim for Sensor Networks"* [1] existem alguns pontos que não são esclarecidos e causam confusão no entendimento do pseudocódigo proposto por ele.

Referências

[1] David Braginsky and Deborah Estrin. Rumor routing algorthim for sensor networks. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 22–31, 2002.

[2] Roger A Light. Mosquitto: server and client implementation of the mqtt protocol. *Journal of Open Source Software*, 2(13):265, 2017.

[3] Roberto Ierusalimschy. *Programming in lua*. Roberto Ierusalimschy, 2006.

[4] LÖVE Community. LÖVE - Free 2D Game Engine, 2021.

[5] Lunar Modules. Penlight lua libraries, 2021.