Redes de Computadores
Projeto de Programação – Vetor de Distâncias

Revisão 1: 11/01/2024.

1. Definição do Sistema

Neste exercício deverá implementar uma versão simplificada do roteamento de vetor de distâncias, baseado na equação de Bellman-Ford. No sistema, cada processo fará o papel de roteador e usará UDP (através do componente Canal desenvolvido) para a comunicação. O sistema deverá ser capaz de atender mudanças nos links e eliminação de roteadores, mas não deve implementar a técnica de reversão envenenada.

Cabe destacar que a comunicação entre dois roteadores somente poderá ser realizada se estes estão conectados por um link.

2. Recomendação Inicial

Assistir o vídeo do link https://www.youtube.com/watch?v=nysfXweTI7o e implementar os exemplos mostrados em Java. Só assistir o vídeo não lhe será de utilidade quando tiver que implementar funcionalidades mais complexas.

3. Funcionalidades do ponto de vista do roteador

1. Implementar o formato (cabeçalho) da mensagem que será transmitida. Crie a classe *DatagramaInfo* com os atributos que considerar necessários para o correto funcionamento.
2. Inicialização.
    a. Deverá ler uma matriz de adjacência de um arquivo txt, onde (i) a primeira linha ou primeira coluna corresponderão ao identificador do roteador, começando em um; (ii) cada célula corresponde ao peso do link. Se o valor da célula for 0, considere como infinito, ou seja, não há um link. Se o valor da célula for maior a 0, considere o identificador como um vizinho.
    b. Crie um processo para cada roteador. Considere o IP o 127.0.0.1. Para a porta, some 10.000 ao identificador capturado.
    c. Mostre a console de um dos roteadores (pode ser qualquer um). Este servirá para apresentar as informações enviadas e recebidas pelo roteador. Os outros roteadores serão executados em background em processos diferentes.
3. Envio de DatagramaInfo, somente quando houver uma alteração no caminho, caso não haja alterações, não deve enviar.
4. Recebimento do DatagramaInfo de um dos vizinhos e cálculo dos atualizações dos caminhos.

5. Finalização. O roteador deve indicar na console quando conseguiu encontrar todos os caminhos com seus respectivos valores (compare o estado atual com o da matriz de adjacência).

4. Mensagens (prints) apresentadas na console

Na console escolhida apresente usando o print (e.g, System.out.println no java):

- As informações do datagrama recebido ou enviado.
- As informações da tabela de roteamento, deixando claro quando e onde tiveram as atualizações/mudanças.
- Apresentar claramente quando o roteador finalizou o processo.

5. Teste a ser realizado

O professor compilará o código o qual deverá abrir automaticamente uma console. Essa console representará um dos roteadores, mostrando as informações mencionadas na seção anterior e executará no background outros processos, representando os outros roteadores.

Na finalização da execução, a console do roteador selecionado deverá mostrar as seguintes informações para cada um dos testes abaixo:

a) Teste para uma comunicação/roteador sem perdas
b) Teste para uma comunicação com perdas
c) Teste para um (1) link que teve seu valor modificado. Mostre claramente qual foi o link, qual foi o valor inicial e qual foi o valor final.
d) Teste para um roteador que não envia nem recebe mais datagramas.

- Média da quantidade de iterações até chegar à finalização.
- Média da quantidade de datagramas enviados.

Mostre gráficos que permitam observar como evolui as médias mencionadas acima para diferentes quantidades de roteadores (1 à 20) e quantidades de links (1%, 5%, 10%, 15%, 20%).

6. Código fonte

- Deverá criar somente as classes Principal, Roteador e DatagramaInfo. A última deverá ser utilizada **obrigatoriamente** para o envio e recebimento de informações (nas requisições e respostas). Caso envie novas classes, serão descontados 3 pontos da nota final.
    - A única exceção é a criação das classes para Threads, mas elas deverão ser criadas dentro de uma das classes acima (e.g, classes aninhadas).

- O código fonte deverá apresentar claramente (usando comentários) os trechos de código que realizam as funcionalidades mencionadas na Seção 3.

- Comente para que serve cada atributo da classe DatagramaInfo

- Independente da linguagem, o uso de bibliotecas que realizem parte das funcionalidades pedidas não será aceito. Caso tenha dúvidas de alguma específica, pergunte ao professor.

7. Entrega

A entrega é individual e consistirá em: (a) um relatório com nome SeuRA.pdf; (b) o código fonte do programa com as pastas. A entrega será como definida no plano de ensino, não sendo aceita por outras formas.

- Caso tenha utilizado uma biblioteca permitida pelo professor, envie-a também.

O relatório deverá ter obrigatoriamente as seguintes seções:

a) Nome e RA do participante
b) Formato da classe DatagramaInfo
c) Link do vídeo do funcionamento (screencast). O vídeo do screencast deverá conter no máximo 5 minutos, mostrando a compilação, o funcionamento do código no teste local e no teste remoto. **Não envie o vídeo do screencast**, envie só o link do vídeo, o qual pode disponibilizar no Youtube ou em outro lugar semelhante (como vimeo). Lembre-se de dar permissão para visualizá-lo.
d) No vídeo do screencast do ponto anterior, deve realizar e mostrar os testes da Seção 5.
e) Cada estrutura de dados criada deve ser explicada (para que serve, o que armazena, etc).
f) Explicação do uso das threads, mencionado as linhas do código fonte que fazem referência.
g) Explicação dos gráficos obtidos na Seção 5.
h) Links dos lugares de onde baseou seu código (caso aplicável). Prefiro que insira os lugares a encontrar na Internet algo similar.

8. Observações importantes sobre a avaliação

A seguir mencionam-se alguns assuntos que descontarão a nota.

- Código fonte não compila ou usa bibliotecas externas sem aprovação do professor (nota zero).
- Não criou e usou a classe DatagramaInfo para a comunicação (menos 3 pontos).
- Não enviou o relatório (menos 4 pontos).
- Não enviou o link do vídeo, o link não está disponível ou o vídeo não mostra o funcionamento, separado nas consoles (menos 4 pontos).
- Enviou outros arquivos do código fonte além dos citados na Seção “Código fonte”, por exemplo os .class ou outras classes (menos 3 pontos).

- Código fonte sem comentários que referenciem as funcionalidades da Seção 3 (menos 2 pontos)

9. Links recomendados (para Java)

- Para baixar a JDK 8
  https://www.oracle.com/br/java/technologies/javase/javase-jdk8-downloads.html

- Vídeo explicativo sobre programação UDP, TCP e Threads:
  https://www.youtube.com/watch?v=nysfXweTI7o

- Informações sobre programação com UDP podem ser encontradas em:
  https://www.baeldung.com/udp-in-java
  https://www.geeksforgeeks.org/working-udp-datagramsockets-java/
  https://docs.oracle.com/javase/tutorial/networking/datagrams/index.html

- Informações sobre Threads, que permitem que o servidor ou peer receba e envie informações de forma simultânea:
  https://www.baeldung.com/a-guide-to-java-sockets (Seção 6: TCP com muitos clientes)
  https://www.tutorialspoint.com/java/java_multithreading.htm

10. Ética

Cola, fraude, ou plágio implicará na nota zero a todos os envolvidos em todas as avaliações e exercícios programáticos da disciplina.