

Universidade Federal da Fronteira Sul
Curso de Ciência da Computação, Chapecó/SC
CCR: Redes de Computadores – 2018.1
Prof. Marco Aurélio Spohn

Descrição trabalhos T1 e T2: Programação Socket e Roteamento

Descrição geral	<p>Nesses dois trabalhos práticos, você praticará programação com <i>sockets</i> e simulação de um protocolo de roteamento em redes. Você executará processos representando os roteadores (nós) da rede, os quais trocarão pacotes de roteamento via <i>sockets</i> UDP. Na versão final do trabalho, os nós executarão o algoritmo <u>Bellman-Ford distribuído</u> para computar as suas tabelas de roteamento.</p> <p>→ LEIA ATENTAMENTE A DESCRIÇÃO DO PROJETO.</p>
Observações:	<ul style="list-style-type: none">• O projeto deve ser implementado em linguagem C.• O projeto deve ser desenvolvido SOMENTE na plataforma Linux!• Utilizar somente <i>sockets</i> UDP!• Cada NÓ deve executar como um processo individual!!!! (obs.: no entanto cada processo pode, caso desejável, ser <i>multithread</i>)• Submeter todos os arquivos necessários para a compilação e execução do sistema. Isto inclui um arquivo do tipo LEIAME com instruções de como utilizar o seu sistema.
Modalidade:	<ul style="list-style-type: none">• Individual OU dupla (a avaliação será sempre individual!)
Descrição:	<p>Desenvolva um programa que simule os roteadores de uma rede. O programa deve obter as informações de configuração via arquivos. Cada roteador deve ser capaz de se comunicar com outros roteadores (<i>i.e.</i>, mesmo programa instanciado múltiplas vezes) através de <i>sockets</i> UDP.</p> <p>O projeto está dividido em duas etapas: a) Etapla 1: correspondente à componente avaliativa T1; b) Etapla 2: correspondente à componente avaliativa T2.</p> <p style="text-align: center;"><u>Informações Gerais</u></p> <p>O programa lê pelo menos um parâmetro da linha de comando, sendo este o identificador (ID) do roteador instanciado. Informações sobre os enlaces existentes entre os roteadores são obtidas do arquivo de configuração “<i>enlaces.config</i>”. Informações sobre em quais portas UDP cada roteador está se comunicando com os demais roteadores são obtidas do arquivo de configuração “<i>roteador.config</i>”.</p>

Etapa 1 (T1)

Nessa etapa, assume-se que os roteadores conhecem a **topologia completa** da rede. Utilizando-se soluções de teoria dos grafos, cada nó computa o caminho completo para todos os destinos possíveis na rede. Nessa etapa, deve-se assumir que a topologia permanece constante (i.e., não sofre nenhuma mudança) ao longo da vida da rede.

Implementar uma aplicação de transmissão confiável de mensagens de texto, limitadas a 100 caracteres. A mensagem deve ser roteada da origem até o destino segundo a rota computada pelos roteadores. Da origem até o destino, qualquer roteador encaminhando o pacote deve apresentar uma mensagem na tela (e.g., “Roteador X encaminhando mensagem com # sequência N para o destino Y”).

Etapa 2 (T2)

Inicialmente, cada roteador conhece apenas os seus vizinhos imediatos e a distância (i.e., custo dos enlaces) até os mesmos. Os roteadores trocam informações de roteamento utilizando o algoritmo **Bellman-Ford distribuído**.

Os roteadores devem trocar informações de roteamento periodicamente para manter as rotas atualizadas. Assumindo uma rede conectada, as tabelas de roteamento convergem após um determinado tempo.

A qualquer momento, os roteadores podem ser “ligados” ou “desligados” (e.g., criando ou matando os processos correspondentes). O roteamento deve se adaptar a estas situações. Você também deve tratar o problema de contagem ao infinito (*count to infinity*). **Observe que você não vai resolver o problema da contagem ao infinito: apenas defina um valor finito (i.e., valor maior que o diâmetro da rede) para parar a contagem, caso ela ocorra!**

Após cada atualização da tabela de rotas, o roteador deve retornar a tabela no console com o *timestamp* da mudança. Ele também deve apresentar uma mensagem quando recebe ou envia pacotes.

Cada roteador se comunica somente com os seus vizinhos imediatos (adjacentes), utilizando os seus respectivos endereços de *sockets* (informação obtida do arquivo de configuração **roteador.config** para enviar e receber mensagens.

Dicas

Vocês podem discutir com os demais colegas, mas **em hipótese alguma compartilhem código!!!**

Teste o seu programa com cenários diferentes. Desative e ative roteadores para observar como a topologia da rede muda. Você pode tentar verificar o resultado primeiro no papel para comparar com os resultados obtidos com o seu programa em execução.

Não espere até o último minuto para começar a trabalhar no seu projeto (mesmo que você seja um programador experiente!).

Arquivos exemplo	<p style="text-align: center;">roteador.config</p> <p>O formato do arquivo é (por linha): identificador do roteador (inteiro), número da porta e número IP (espaçamento livre entre parâmetros).</p> <p>O programa recebe na entrada o identificador do roteador em questão, basta então ler do arquivo de entrada o seu IP e qual a porta do seu <i>socket</i>.</p> <p>Exemplo:</p> <pre> 1 25001 127.0.0.1 2 25002 127.0.0.1 3 25003 127.0.0.1 4 25004 127.0.0.1 5 25005 127.0.0.1 6 25006 127.0.0.1 </pre> <hr/> <p style="text-align: center;">enlaces.config</p> <p>O formato de cada linha do arquivo é:</p> <p>ID ID custo (espaçamento livre entre parâmetros).</p> <p>Ou seja, o identificador dos dois roteadores conectados e o custo do enlace. Assume-se que os enlaces são simétricos (bidirecionais).</p> <p>Exemplo:</p> <pre> 1 2 10 1 3 15 2 4 2 2 5 5 3 4 2 4 6 10 5 6 5 </pre>
Documentação	O código fonte deve ser acompanhado de documentação. Também deve incluir um arquivo com instruções DETALHADAS de como operar o programa.
O que e quando submeter	<p><u>O que:</u> para cada etapa (T1 e T2), submeter (via <i>moodle</i>) um arquivo compactado com todos os arquivos necessários para a execução do sistema.</p> <p><u>Quando:</u></p> <ul style="list-style-type: none"> • Etapa 1: 07/05/2018 • Etapa 2: 27/06/2018 <p>Após o prazo de entrega de cada trabalho, definir-se-á uma data para apresentação!!</p>