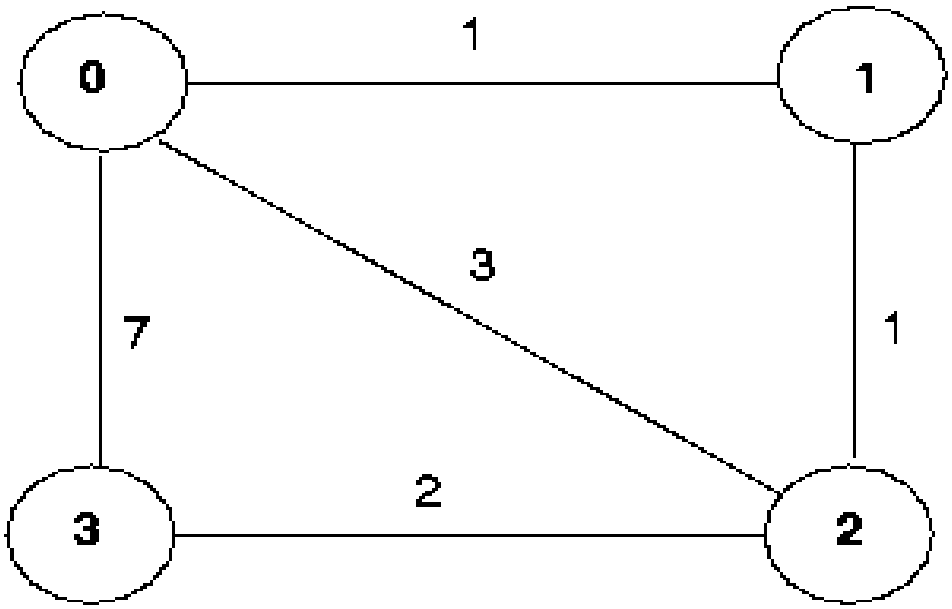
REC

LAB_03 programação: Roteamento Vetor de Distâncias (DV)

Nesta tarefa de programação a equipe escreverá um conjunto distribuído de procedimentos que implementam um roteamento de vetor de distâncias assíncrono distribuído para a rede apresentada a seguir:



A equipe deverá escrever as seguintes rotinas que executarão assincronamente dentro do ambiente emulado fornecido por esta tarefa. Para o nó 0, a equipe escreva as rotinas:

rinti0 (). Esta rotina será chamada uma vez no início da emulação, e não possui argumentos. A equipe deve inicializar a tabela de distâncias no nó 0 para refletir os custos diretos de 1, 3 e 7, até os nós 1,2,e 3, respectivamente. Todos os enlaces da figura são bidimensionais e os custos em ambas as direções são idênticos. Após inicializar a tabela de distâncias e quaisquer outras estruturas de dados necessárias às rotinas de seu nó 0, este deve então enviar a seus vizinhos diretamente ligados (neste caso,1, 2 e 3) o custo de seus caminhos de custo mínimo para todos os outros nós da rede. Esta informação do custo mínimo é enviada aos nós vizinhos em um pacote de atualização de roteamento denominado rotina ***tolayer2 ().***

rtupdate0 (struct rtpkt *rcvdpkt). Esta rotina será chamada quando o nó 0 receber um pacote de roteamento que foi enviado a ele por um de seus vizinhos diretamente conectados. O parâmetro *rcvdpckt é um indicador para o pacote que foi recebido. Esta rotina corresponde ao coração do algoritmo DV. Os valores que ele recebe em um pacote de atualização de roteamento de algum outro nó i contêm os custos correntes do caminho mais curto de i para todos os outros nós da rede. Esta rotina utiliza estes valores recebidos para atualizar a sua tabela de distâncias, conforme especificada pelo algoritmo DV. Se o seu custo mínimo até um outro nó mudar como resultado da atualização, o nó 0 informará esta mudança no custo mínimo a seus vizinhos diretamente conectados enviando a eles um pacote de roteamento. Lembre-se de que no algoritmo de roteamento DC apenas os nós que estão diretamente conectados trocarão pacotes de roteamento. Assim os nós 1 e 2 vão se comunicar, mas não os nós 1 e 3.

Rotinas semelhantes são definidas para os nós 1, 2 e 3. Assim a equipe deverá escrever oito procedimentos ao todo: rtinit0(), rtinit1(), rtinit2(), rtinit3(), rtupdate0( ), rtupdate1 ( ), rtupdate2 ( ), rtupdate3 ( ). Juntas estas rotinas implementam um cálculo assíncrono, distribuído, das tabelas de DV para a topologia e os custos mostrados na figura apresentada.