Sistemas Distribuídos - Trabalho 3

Clara Szwarcman

Guilherme Simas

Funcionamento do Programa

Cada nó possuirá as seguintes informações: um vetor de vizinhos; uma flag para cada tipo de evento, que indica se aquele nó possui uma source para aquele evento; apenas uma possível source para cada tipo de evento (se uma nova source for detectada, apenas a com o menor número de hops prevalece); uma flag para cada tipo de evento que indica se o nó possui alguém interessado naquele evento; uma flag para cada tipo de evento que indica se aquele nó está aguardando a resposta para um request. A estrutura de fila será utilizada para armazenar interesses.

O primeiro passo é o reconhecimento dos vizinhos. Para isso cada um dos nós manda um broadcast a cada 100 milisegundos por 5 segundos. O mesmo nó recebe o broadcast de seus vizinhos e armazena os ids no vetor neighbours. Quando chega uma nova mensagem, é checado se aquele id já está no vetor, se não estiver o novo vizinho é adicionado. O vetor tem tamanho fixo 8, que é o número máximo de vizinhos presente na arquitetura do simulador.

A segunda parte possui dois loops ocorrendo em paralelo. Um que detecta um novo evento ou um novo interesse no próprio nó e outro que aguarda o recebimento e tratamento de uma mensagem. Uma mensagem pode possuir tipos 6 diferentes pré estabelecidos: SOURCE_MIN, SOURCE_MAX, INTEREST_MIN, INTEREST_MAX, REQUEST, INFO.

O loop que detecta um evento começa requisitando a temperatura, e caso ela esteja abaixo ou acima de limites definidos uma flag é setada para que uma mensagem seja enviada. A mensagem terá o tipo "SOURCE_MIN"ou "SOURCE_MAX"e terá como target um vizinho randômico. Quando surge um evento o nó também seta a si próprio como source, e a temperatura lida como informação. O loop de evento é análogo, utilizando o sensor de luz. Quando um interesse é gerado uma flag é setada, indicando que aquele nó está aguardando a resposta para um evento.

O loop que trata as mensagens recebidas foi estruturado tentando generalizar ao máximo esse tratamento. Primeiro ele aguarda a chegada de uma mensagem. Quando uma nova mensagem é recebida, uma mensagem "padrão"(target como vizinho randômico, tipo igual ao tipo recebido,hops igual a hops recebidos acrescido de 1) é montada. Os parâmetros dessa mensagem serão alterados de acordo com as particularidades de cada tipo, e, no final do loop, essa mensagem será enviada caso a flag correspondente esteja setada como *true*.

Ao receber uma mensagem do tipo SOURCE, caso a flag para source daquele tipo de evento seja *false*, ela é trocada para *true* e o id do nó source armazenado. Caso seja *true*, a source com o menor número de hops é mantida. A flag de envio é setada como false caso o número de hops ultrapasse o limite, caso contrário a mensagem será repassada para um vizinho aleatório.

Em uma mensagem do tipo INTEREST é criada uma instância do tipo *interestInfo*, que contém o tipo do interesse e o id do nó interessado. Essa instância é armazenada na fila. A mensagem é reenviada com o memso critério da SOURCE

Ao receber uma REQUEST, além de colocar a mesma variável do tipo *interestInfo* na fila (ou seja, um nó que realiza uma requisição é tratado como um nó interessado naquele tipo de informação), o nó checa se ele possui aquela informação e, em caso positivo, ele dispara uma nova mensagem com o tipo INFO.

No tipo INFO o nó faz um loop do tamalho da fila (qSize()). A cada iteração um elemento da fila é retirado, e se o tipo de evento daquele elemento for igual ao da informação que chegou, a mensagem de INFO é enviada para aquele nó. Se os tipos forem diferentes, o nó é colocado novamente na fila, para aguardar o tipo de

informação correto.

Ao fim desse comportamento particular de cada tipo de mensagem, o nó checa se a flag de envio de mensagem está setada como *true* e encaminha a mensagem em caso positivo. Por fim, o nó checa através de flags se ele possui pelo menos uma source e um interesse no mesmo tipo de evento e ainda não está no aguardo por uma informação daquele mesmo tipo. Caso essas três condições sejam verdadeiras, ele dispara uma nova request para o nó source.

1 Dificuldades

As maiores dificuldades vieram do uso de Terra. Além de todo o debug ser feito utilizando apenas 3 leds, há pouquíssima documentação e informação das funções disponíveis e o que elas fazem.