type graph = bool [n,n];
chan probe[n](int sender);
chan echo[n](graph topology)    # parts of the topology
chan finalecho(graph topology)  # final topology

process Node[p = 0 to n-1] {
    bool links[n] = neighbors of node p;
    graph newtop, localtop = ([n*n] false);
    int parent;    # node from whom probe is received
    localtop[p,0:n-1] = links;    # initially my links
    receive probe[p](parent);
    # send probe to other neighbors, who are p’s children
    for [q = 0 to n-1 st (links[q] and q != parent)]
        send probe[q](p);
    # receive echoes and union them into localtop
    for [q = 0 to n-1 st (links[q] and q != parent)] {
        receive echo[p](newtop);
        localtop = localtop or newtop;    # logical or
    }
    if (p == S)
        send finalecho(localtop);    # node S is root
    else
        send echo[parent](localtop);
}

process Initiator {
    graph topology;
    send probe[S](S)    # start probe at local node
    receive finalecho(topology);
}

Figure 9.11  Probe/echo algorithm for gathering the topology of a tree.

Copyright © 2000 by Addison Wesley Longman, Inc.