**实验四、从知晓到行动的模拟**

输入：给定任意网络结构，初始节点集合和门槛值

输出：

1. 模拟扩散过程（动态展示），一步步看哪些节点“接受了”新事物，直到不再有扩散的可能。
2. 从知晓到行动的各个节点延时以及平均延时。

相关概念：

一种新事物会如何在网络中得到传播，或者说传播的效果将会如何，在鼓励创新的年代，是一个尤其值得关心的问题。网络级联，即人们通过与有关系的人之间的互动决定是否采纳一项新事物（例如某一款新手机），是学界已经形成的一类基本模型，其形象化的表现就是新事物在网络中从一些点开始，逐步向周边扩散，直至完全覆盖所有节点，或者到某个程度后再也不会扩散了。而每个人从知晓到行动是有延时的。

算法分析:

该算法使用BFS算法模拟了信息在网络中的扩散过程。输入包括任意网络结构、初始节点集合和门槛值。门槛值表示信息扩散需要达到的最小接受节点数量。输出包括模拟扩散过程（动态展示）、从知晓到行动的各个节点延时以及平均延时。

算法流程如下：

1. 初始化初始节点集合以及延时记录map。
2. 将初始节点加入队列并将其延时设置为0。
3. 对于队列中的每个节点，遍历其相邻节点：
   * 如果该相邻节点尚未被遍历过，则将其加入队列中，并将其延时设置为当前节点的延时+1。
   * 如果该相邻节点已经被遍历过，则比较当前节点到达该节点的延时和已有延时记录，保留较小值。
4. 当队列为空时，说明信息扩散已经完成。
5. 计算各个节点的延时和平均延时，并输出结果。

代码实现(java):

import java.util.\*;  
  
public class InformationDiffusionSimulation {  
  
 public static void main(String[] args) {  
 // *TODO: 根据需要修改网络结构、初始节点集合和门槛值* int[][] graph = {{0, 1, 1, 0, 0, 0},  
 {1, 0, 1, 1, 0, 0},  
 {1, 1, 0, 1, 1, 0},  
 {0, 1, 1, 0, 1, 1},  
 {0, 0, 1, 1, 0, 1},  
 {0, 0, 0, 1, 1, 0}};  
 int[] initialNodes = {0, 1};  
 int threshold = 3;  
  
 // 模拟信息扩散过程  
 Map<Integer, Integer> spreadTimeMap = new HashMap<>(); // 记录各个节点的延时  
 Queue<Integer> queue = new LinkedList<>();  
 for (int node : initialNodes) {  
 spreadTimeMap.put(node, 0);  
 queue.offer(node);  
 }  
 while (!queue.isEmpty()) {  
 int currNode = queue.poll();  
 for (int i = 0; i < graph[currNode].length; i++) {  
 if (graph[currNode][i] == 1 && !spreadTimeMap.containsKey(i)) {  
 spreadTimeMap.put(i, spreadTimeMap.get(currNode) + 1);  
 if (spreadTimeMap.get(i) <= threshold) {  
 queue.offer(i);  
 }  
 }  
 }  
 }  
  
 // 计算平均延时  
 double totalSpreadTime = 0;  
 for (int time : spreadTimeMap.values()) {  
 totalSpreadTime += time;  
 }  
 double avgSpreadTime = totalSpreadTime / spreadTimeMap.size();  
  
 // 输出结果  
 System.*out*.println("节点延时：");  
 for (int node : spreadTimeMap.keySet()) {  
 System.*out*.println("节点" + node + "： " + spreadTimeMap.get(node));  
 }  
 System.*out*.println("平均延时： " + avgSpreadTime);  
 }  
}