AODV仿真实验报告

小组成员：

姓名：

学号：

姓名：

学号：

日期：

**一、实验要求和目的**

* 掌握无线自组织网络的组网方式
* 掌握 AODV 路由协议的工作过程
* 利用 NS2 仿真实现AODV 路由协议

**二、实验环境**

Xshell + Ubuntu + Xmanager

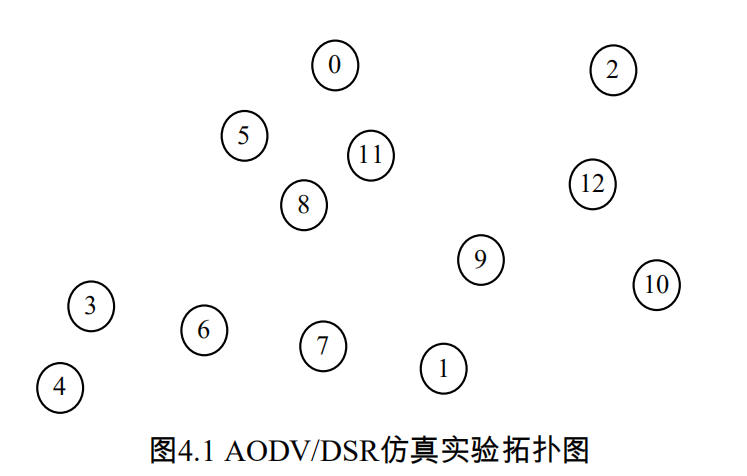
**三、实验内容**

1.AODV协议

AODV(Ad hoc On-demand Distance Vector Routing)是一种按需路由协议。当一个节点需要给网络中的其他节点传送信息时，如果没有到达目标节点的路由，则必须先以多播的形式发出RREQ(路由请求)报文。RREQ报文中记录着发起节点和目标节点的网络层地址，邻近节点收到RREQ，首先判断目标节点是否为自己。如果是，则向发起节点发送RREP(路由回应);如果不是，则首先在路由表中查找是否有到达目标节点的路由，如果有，则向源节点单播RREP，否则继续转发RREQ进行查找。

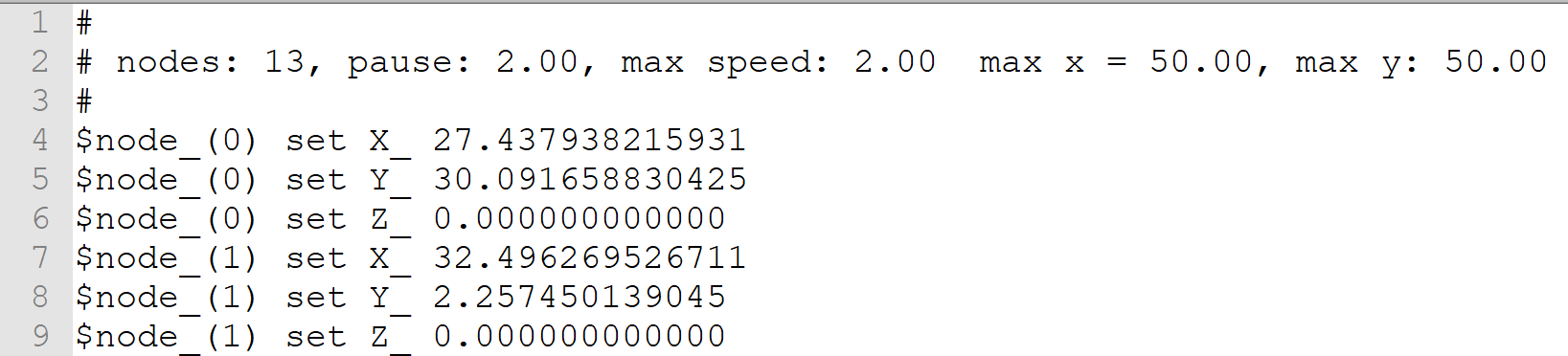
2.仿真拓扑

本实验的仿真拓扑结构如下图所示，共需用到 13 个移动节点，分别是节点 0~12。其中，节点 8 是源节点，结点 2 是目的节点。

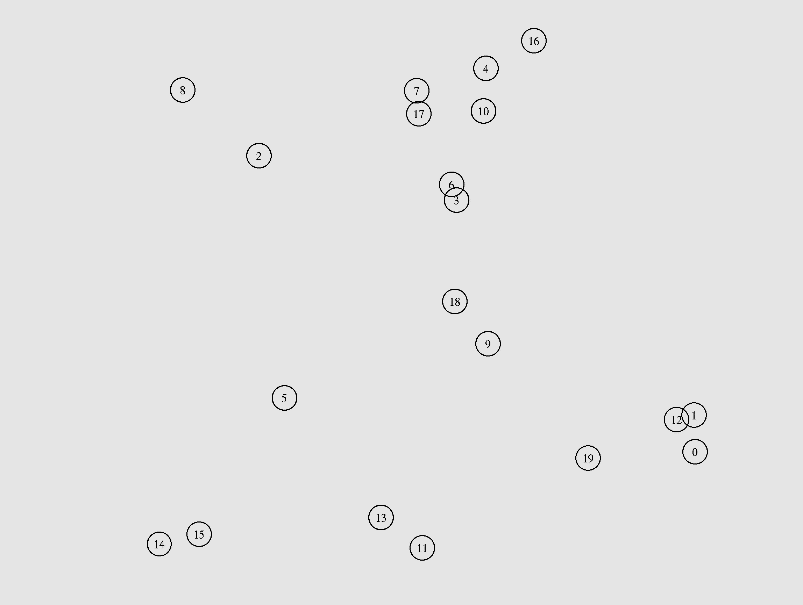


**图1 原实验拓扑图**

我们用scn文件记录所有节点的拓扑信息，形式如下：



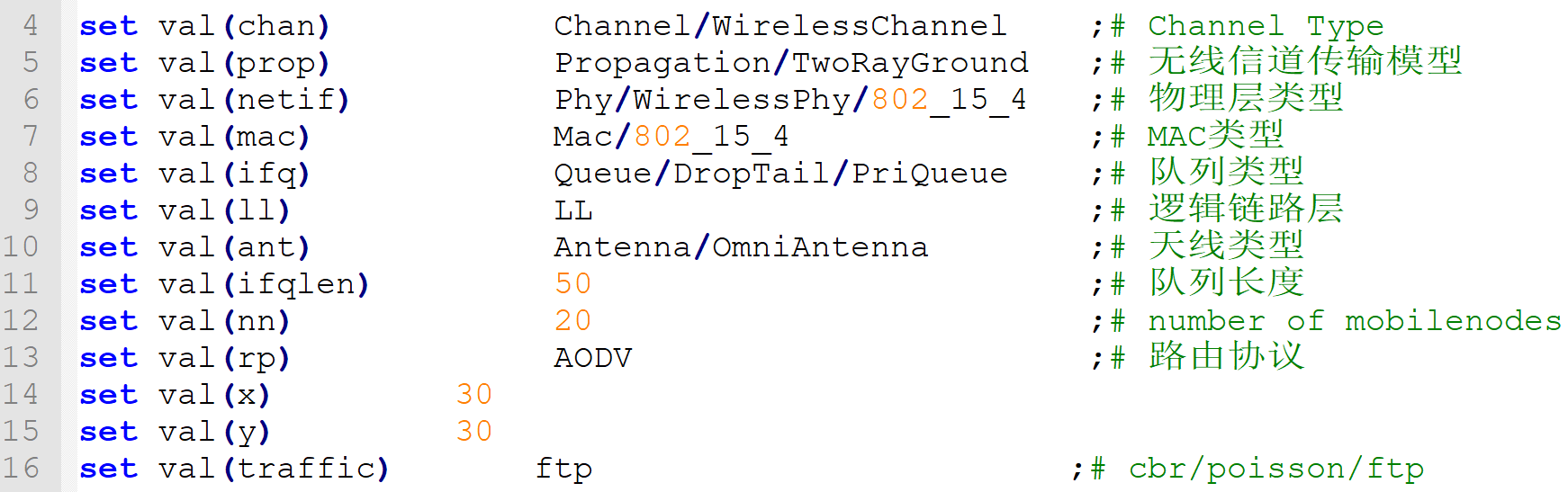
为了实验结果的有效性和多样性，我们还做了20个节点的拓扑图及其对应的实验，拓扑结构如下：



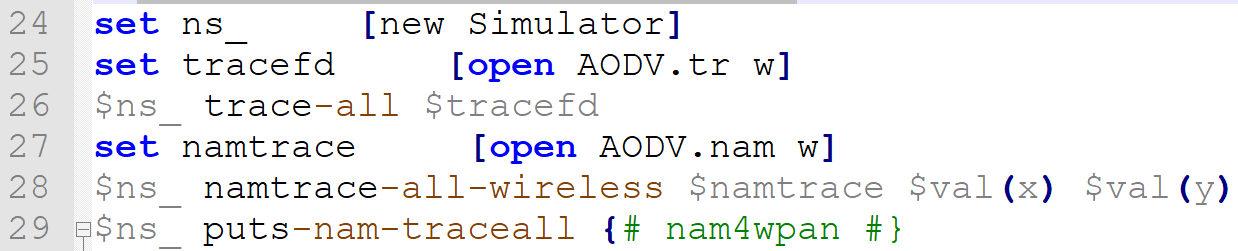
**图2 20个节点拓扑图**

**四、实验代码**

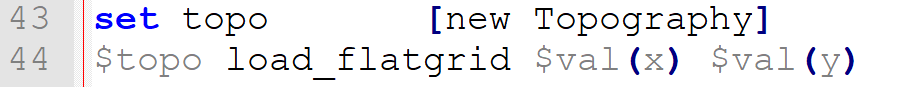
1.定义与网络层相关的各项参数



2. 产生一个ns实例，并设置记录文件



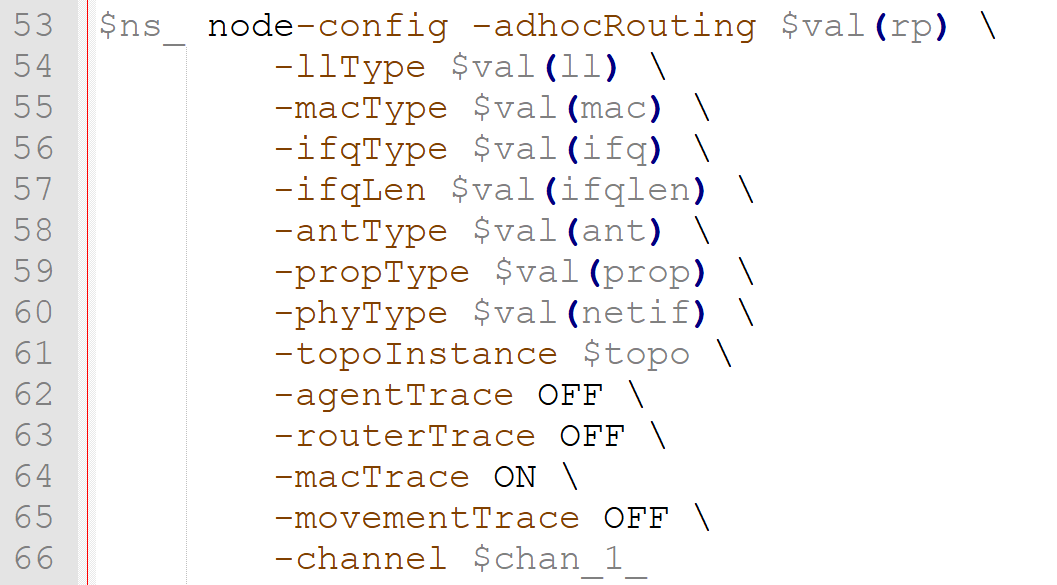
3. 产生一个拓扑实例



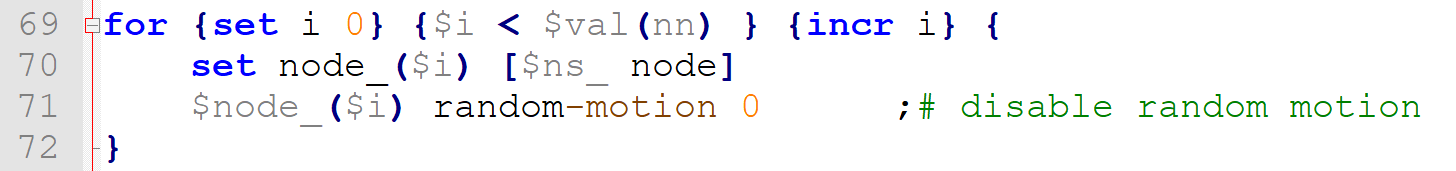
4.调用仿真场景文件



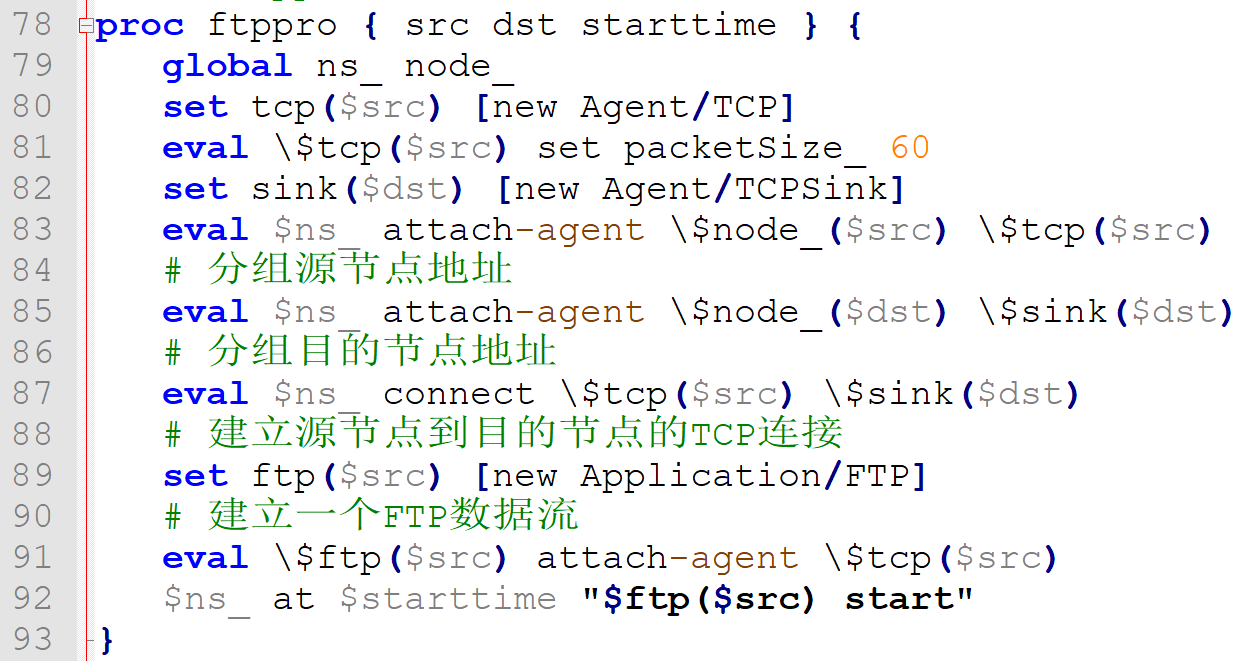
5. 利用上述定义的各项参数配置移动节点参数



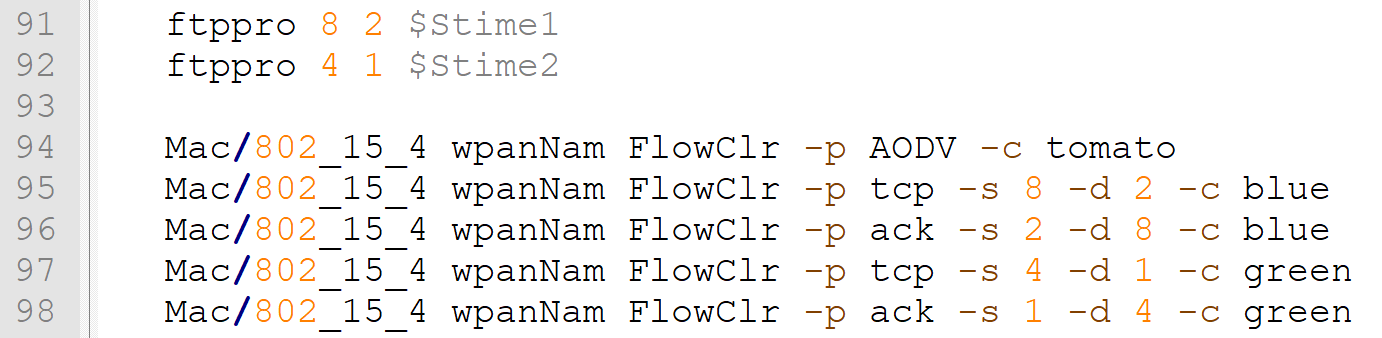
6.产生所有的节点



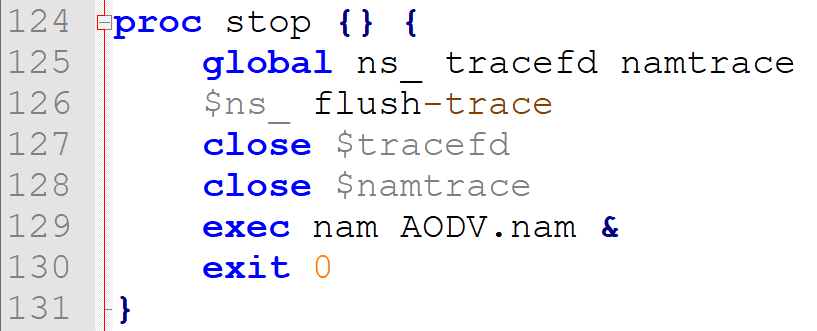
7.定义ftppro过程



8.调用ftppro过程，在节点8和2、1和4之间进行文件传输

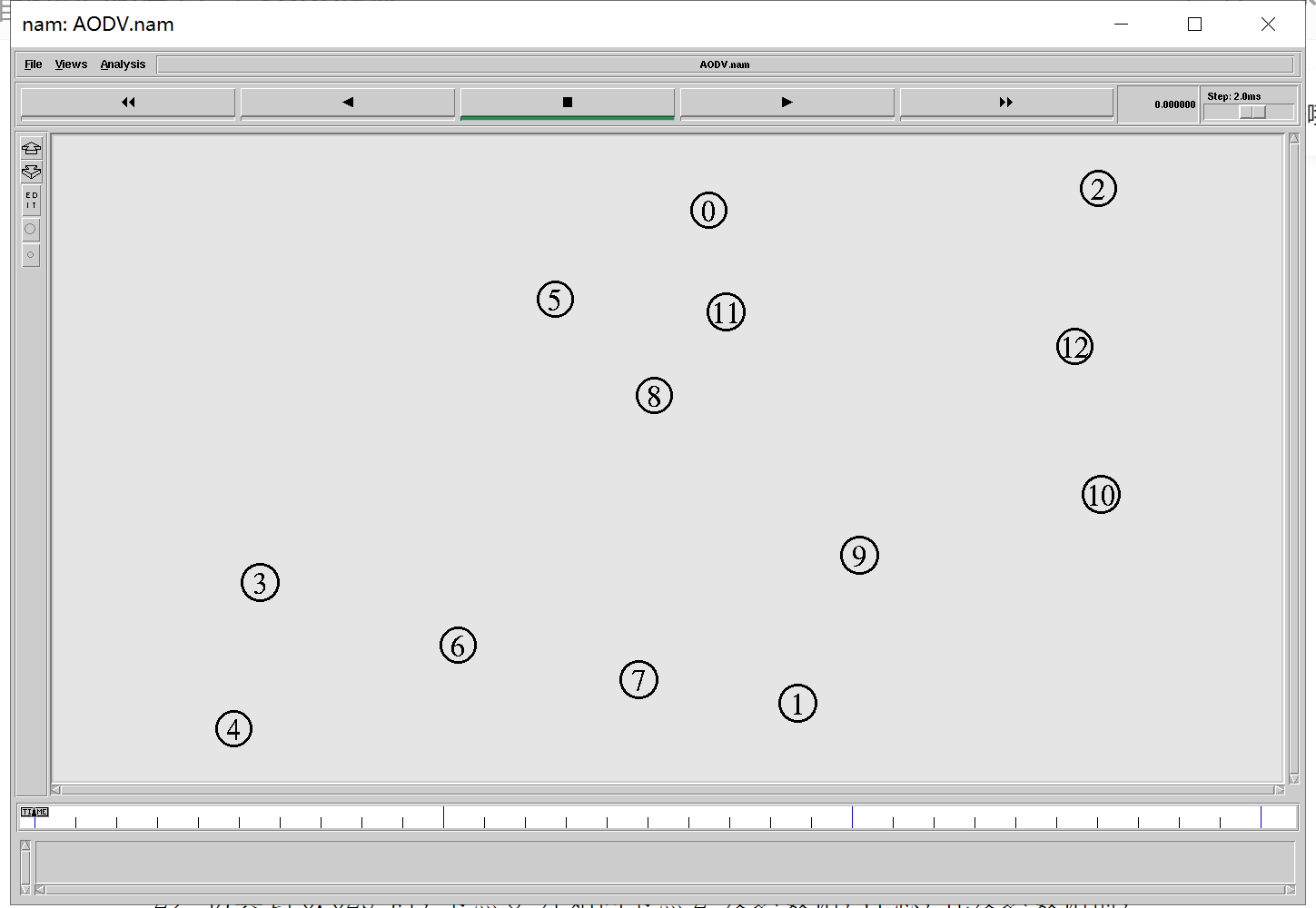


9.定义停止情况，并展示仿真文件

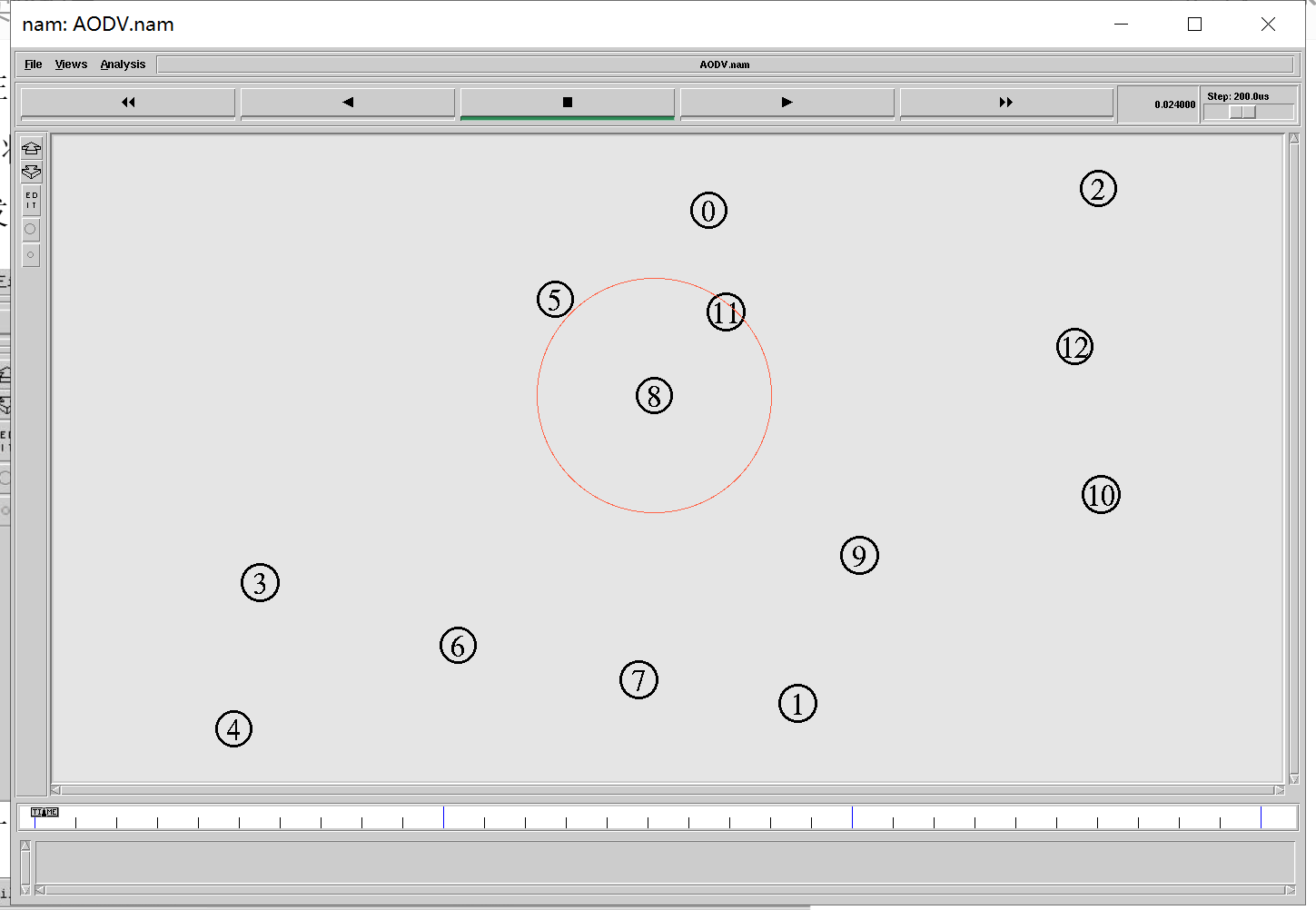


**五、运行仿真**

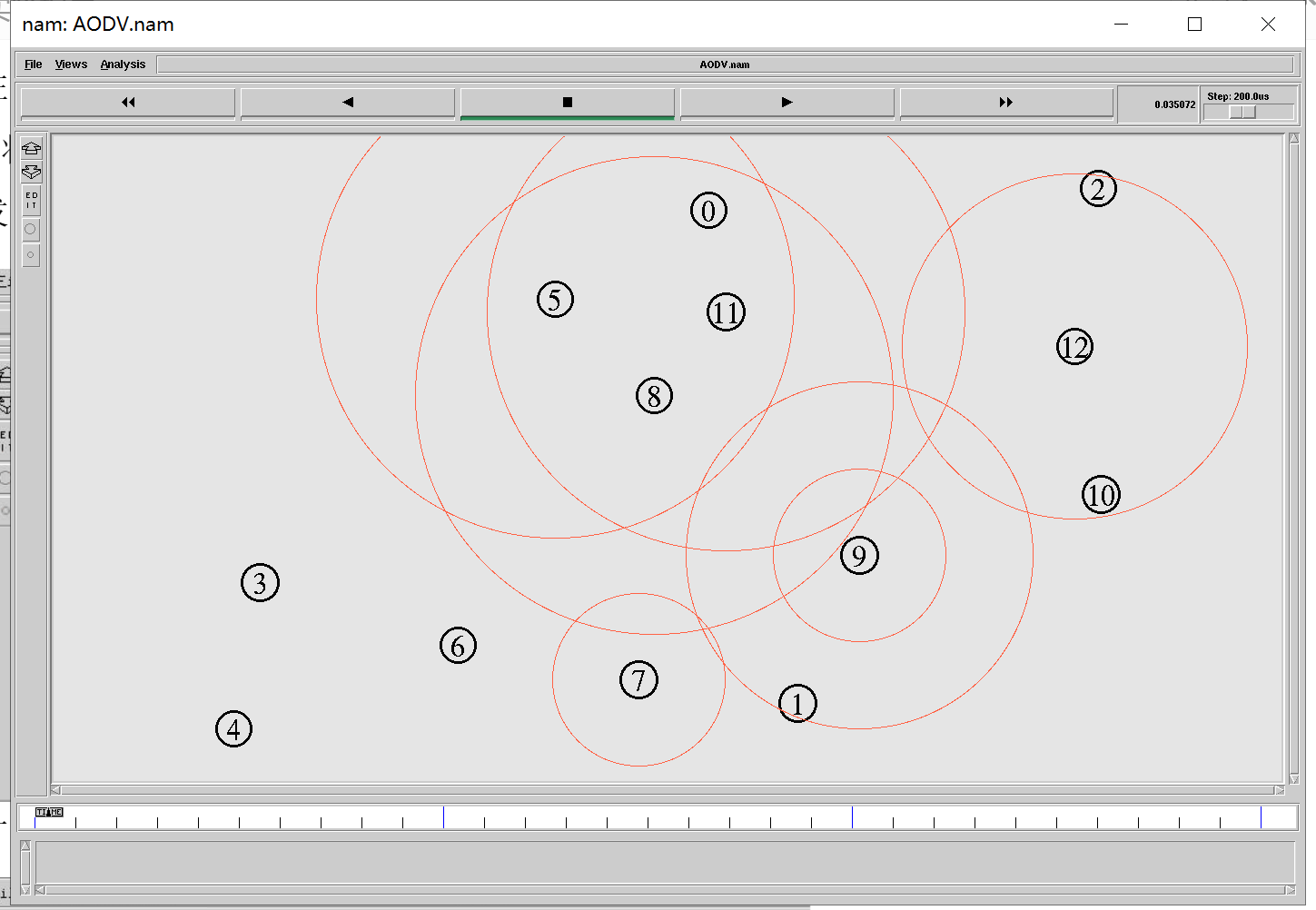
1.创建13个节点，节点间未交换路由信息，路由表尚未建立。



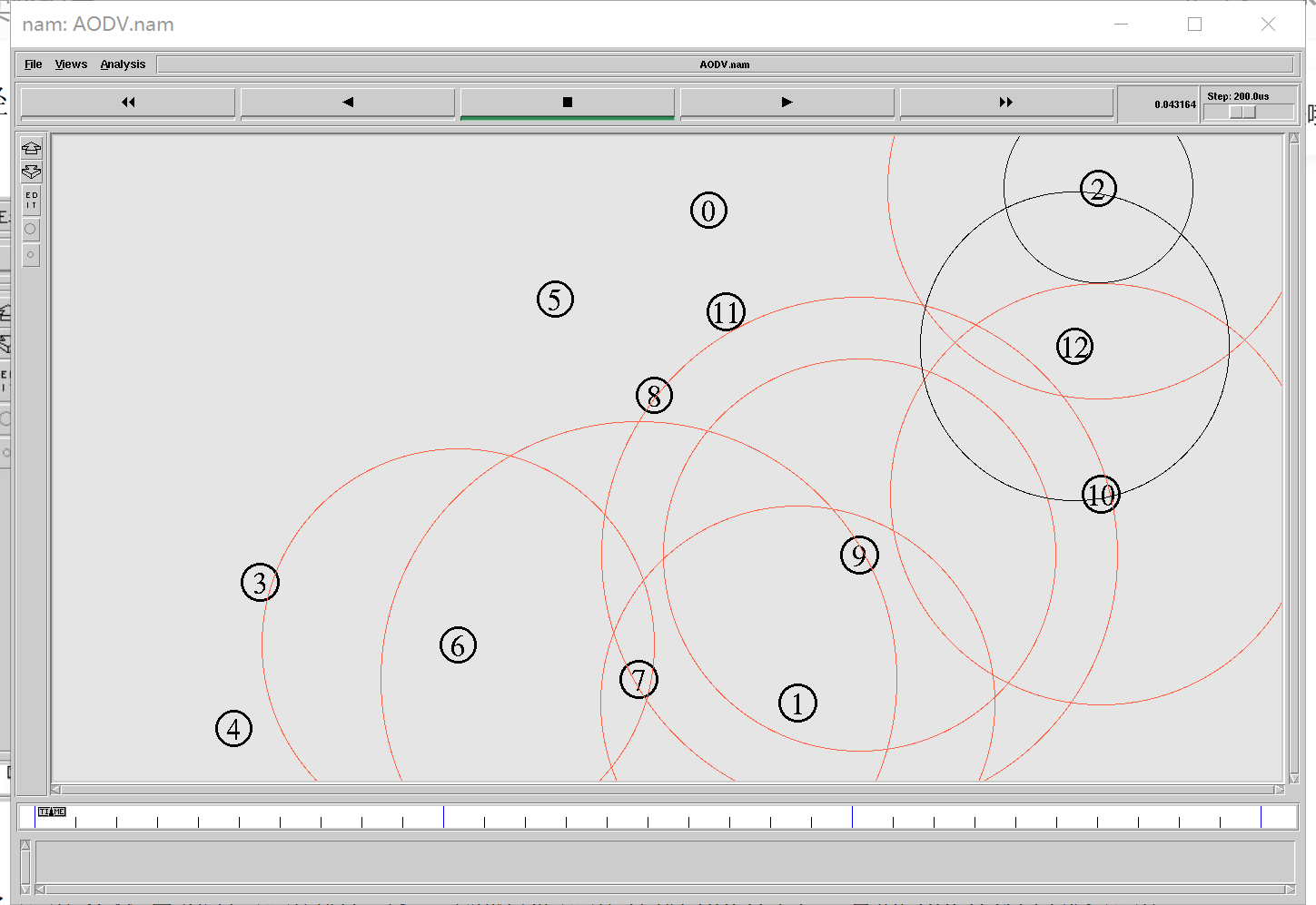
2.仿真到0.02s时，节点8开始向节点2发送数据，在发送数据前，节点8需要通过RREQ/RREP与节点2建立路由。所以节点8开始向周围的节点即邻居广播RREQ信息。



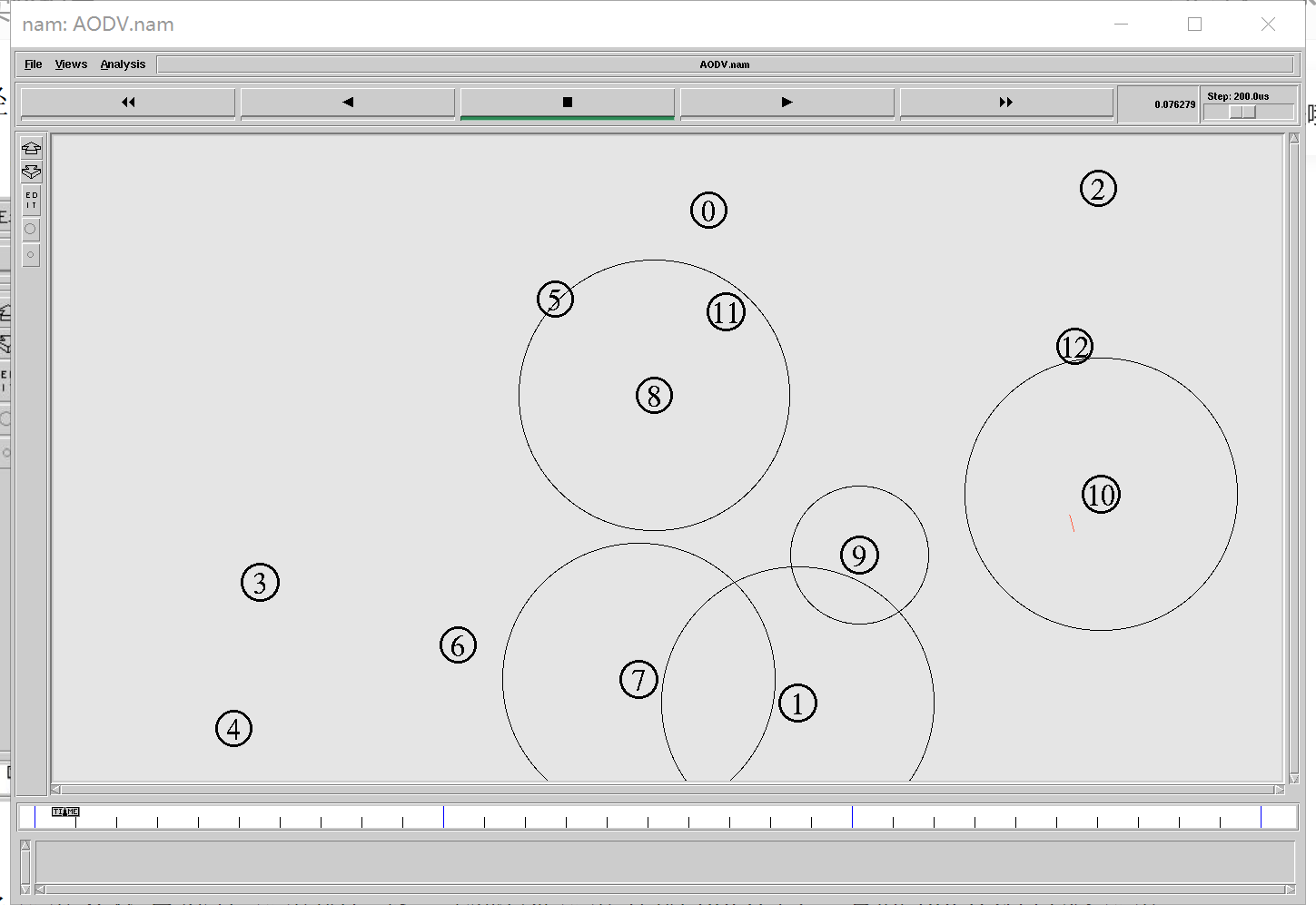
3.广播信息继续扩散，很多节点都在传递广播



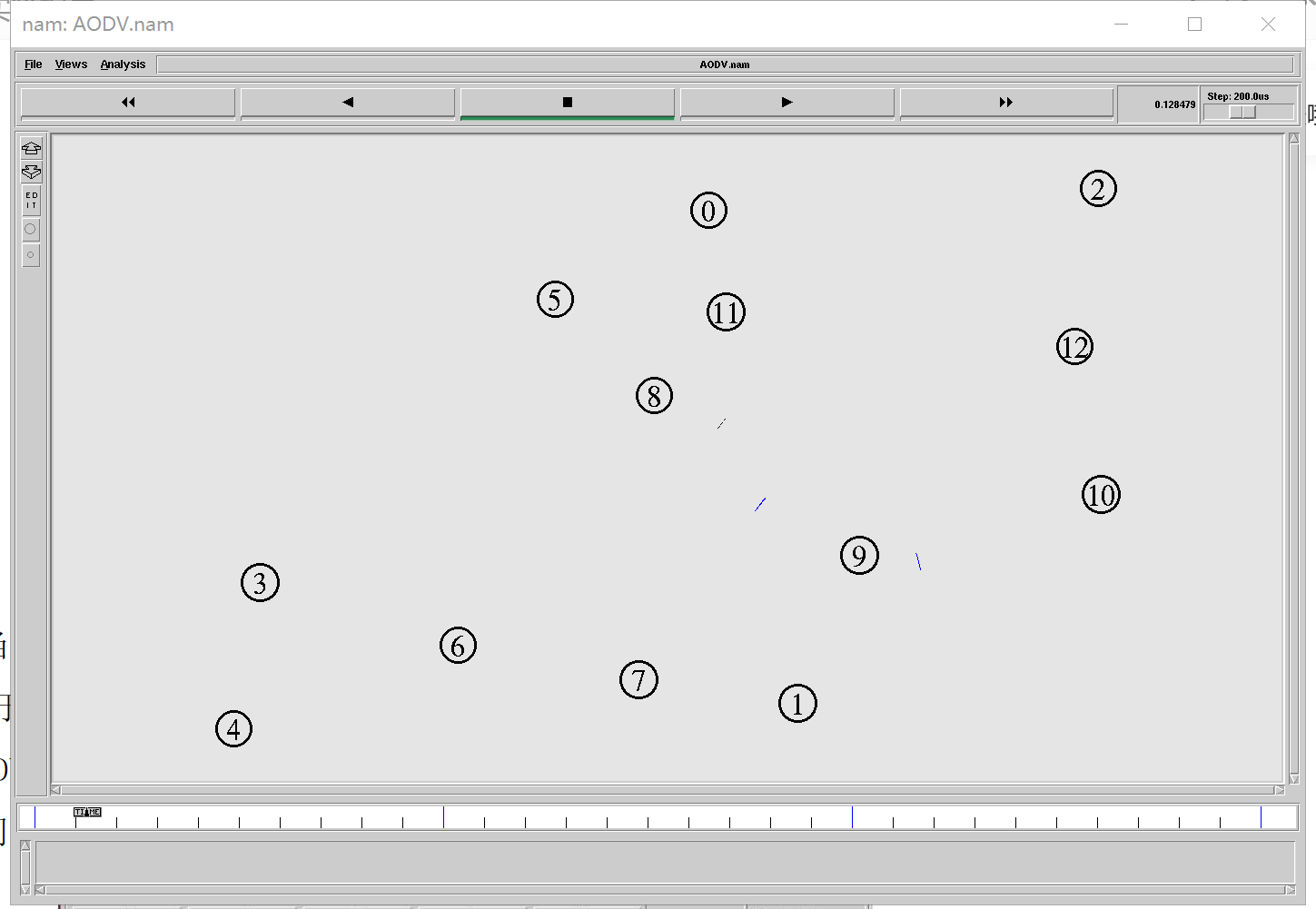
4.经过多节点路由请求广播之后，节点8找到了目标节点2的位置，节点2接收到节点8的RREQ后，返回RREP消息给节点8以建立通信。



5.首先按照路由搜索到的路径返回给RREP给节点12，然后收到RREP消息的节点继续按照路由搜索到的路径(2,12,10,9,8)依次将RREP消息返回给节点8。



6.当节点8获得到节点2的路由以后，就按照该路径开始传送FTP信息。而收到源自于节点8的信息后，节点2会向源路径返回ACK信息，此时双方建立连接。在ADOV中这条路由会被一直维护。（图中蓝色的就是传输的信息）



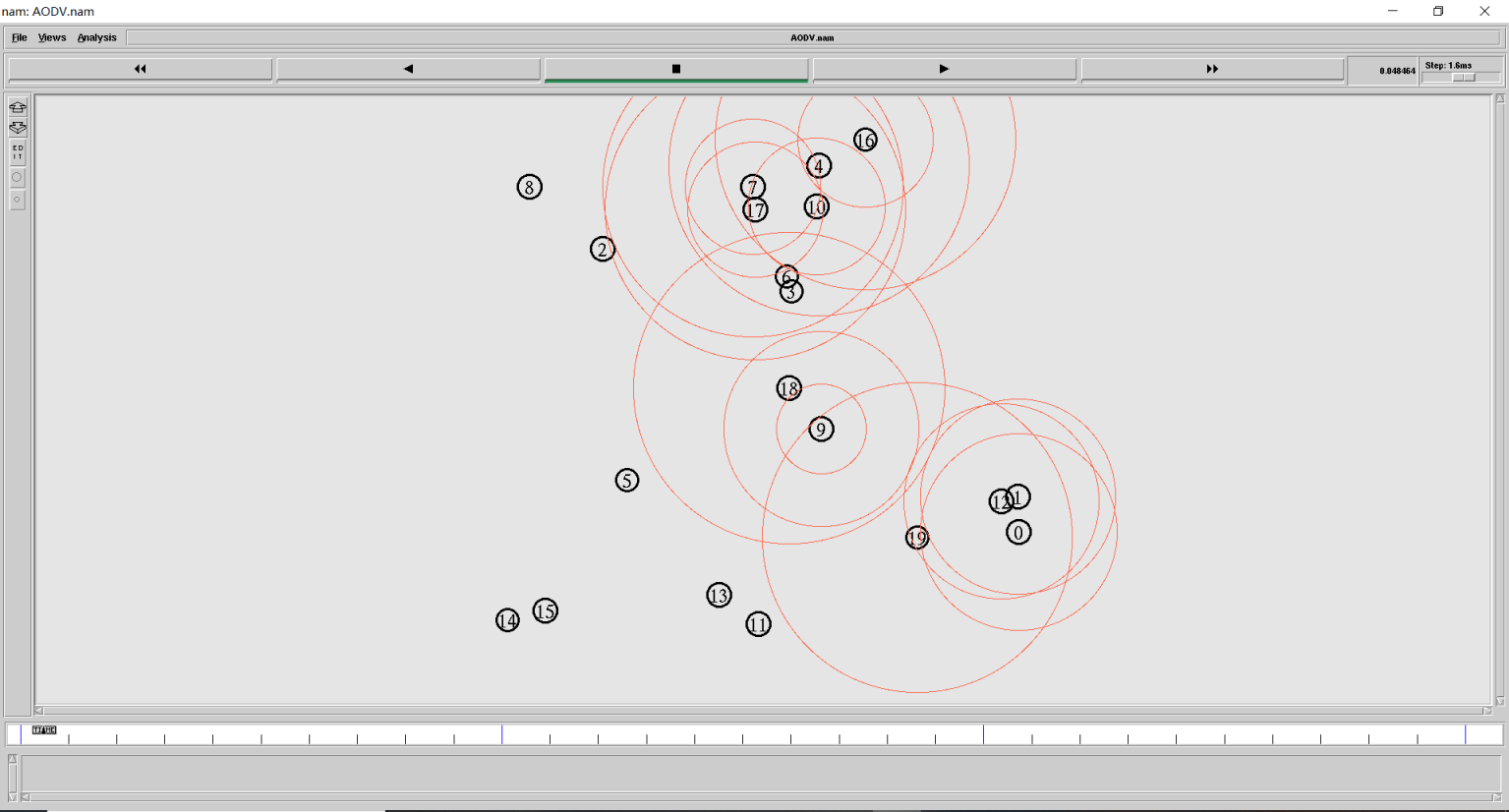
**六、实验拓展**

在最基本的实验的基础上，我们还进行了其他的拓展实验：20个节点的AODV仿真实验、100个节点的AODV仿真实验、DSR对比实验、丢包率分析等数据分析实验等。

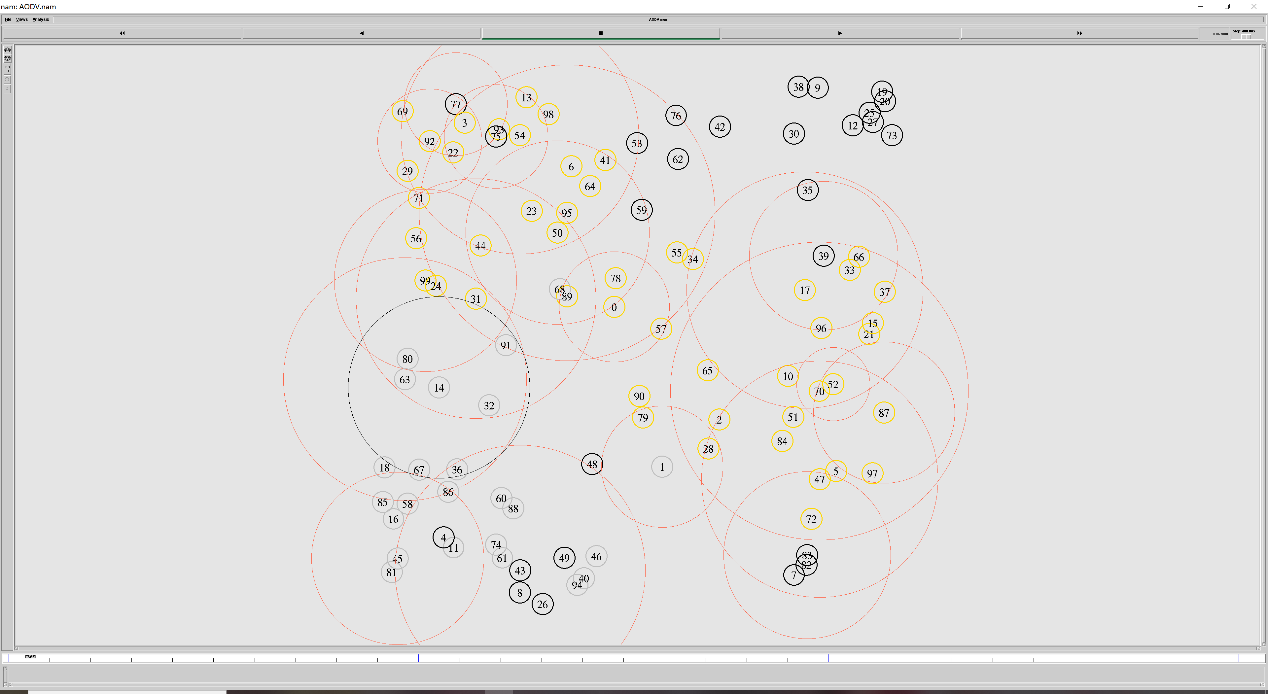
1. 多节点AODV仿真实验

为了体现和展示不同节点数量对于AODV协议的影响，我们还自行增加了节点数量，模拟实际应用中的情况。具体实验运行图如下图所示：

20节点运行结果：

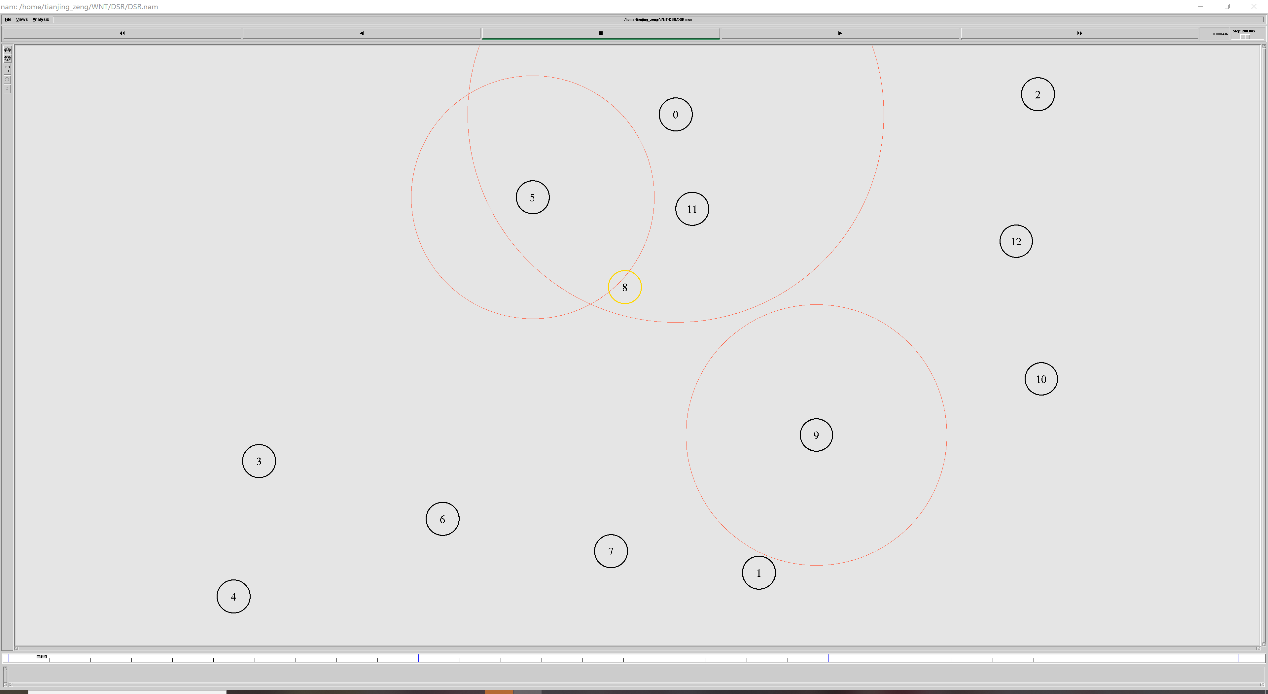


100节点运行结果：



1. DSR对比实验

在与基础实验相同的13节点拓扑图上，我们进行了DSR与AODV的对比实验。DSR运行效果图如下所示：



**区别分析：**

DSR 和AODV 两种协议的路由寻找都是查询和应答往返过程进行的。按照一定的格式将路由信息存储在路由的所有中间节点上。但是两者在动态性上面有差别。

第一DSR协议使用源路由访问的路由明显多于AODV协议。DSR协议中，源节点使用一次路由请求与路由应答过程，除了可以得到目的节点的路由之外，还可以得到到达路由上每个中间节点的路由。

第二，DSR协议对于在单独一个路由请求于路由应答过程中到达目的节点的所有路由请求RREQ分组做应答。因此源节点知道到达目的节点的很多路由。在AODV协议中，目的节点只对第一个到达的路由请求分组RREQ做出应当，而忽略和丢掉随后到达的RREQ

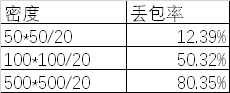
第三，DSR协议的当前版本里面没有任何机制来删除路由存储器中期满过时而失效的路由。也没有任何机制来确定在面俺对多跳路由选择时有限选择哪条。AODV是比DSR协议保守的多的方法。

第四，AODV协议中使用RERR分组的路由删除操作是游标流的。路由错误RERR分组传输到达所有使用中断链到达任何目的节点的路由上的节点。

1. 数据分析实验

在AODV实验中，我们还针对丢包率等数据进行了分析，具体结果如下图所示：





可见随着节点数量的上升，丢包率显著上升，所用时间上升，吞吐量下降。节点密度的上升也会让丢包率显著上升。

**七、问题及解决办法**

1.因为是在服务器上直接配置的NS2,一开始找需要修改的全局环境文件找了好久，后来查阅资料后找到。

2.老师给的方法是在window虚拟机上进行仿真，所以有Xwin运行仿真动画，但linux服务器上并没有Xwin所以找了很久能够适合打开nam仿真的软件，最后找到了Xmanager可以运行NAM。

3.因为前面没有接触过tcl，所以花了很长的时间才明白每行代码都代表什么。

4.一开始看不太明白整个仿真动画的过程，后来通过trace文件就能够很清楚的知道每一步都在干什么了。

**八、实验体会**

经过仿真实验，对于AODV协议有了一个更深刻的理解，觉得AODV协议是一个非常合理的协议，如果没有连接建立的需求，整个网络都是静止的。节点通过周期性的交换路由信息来不断更新自身的路由表，以便能够及时的反映网络拓扑结构和变化，以维护一致的、及时的、准确的路由信息。AODV使用了序列号机制，避免路由环路，且支持中间节点应答，减少了广播数。每个节点只存储需要的路由，减少了内存的要求和不必要的复制。在AODV协议中，节点的路由表中存储着下一跳的节点信息，并且只要该路由保持活性就一直使用它。即是说只要链路不断，路由就会一直保持它的连通性。但是越长的路由耗费的带宽会更多，连不通的可能性就越大。这样的后果是严重影响网络性能并耗费大量的资源，产生长路由现象。在实验里无法体现对资源的占用，但可以直观的看到路由器需要保存下一跳的信息才可以连通。