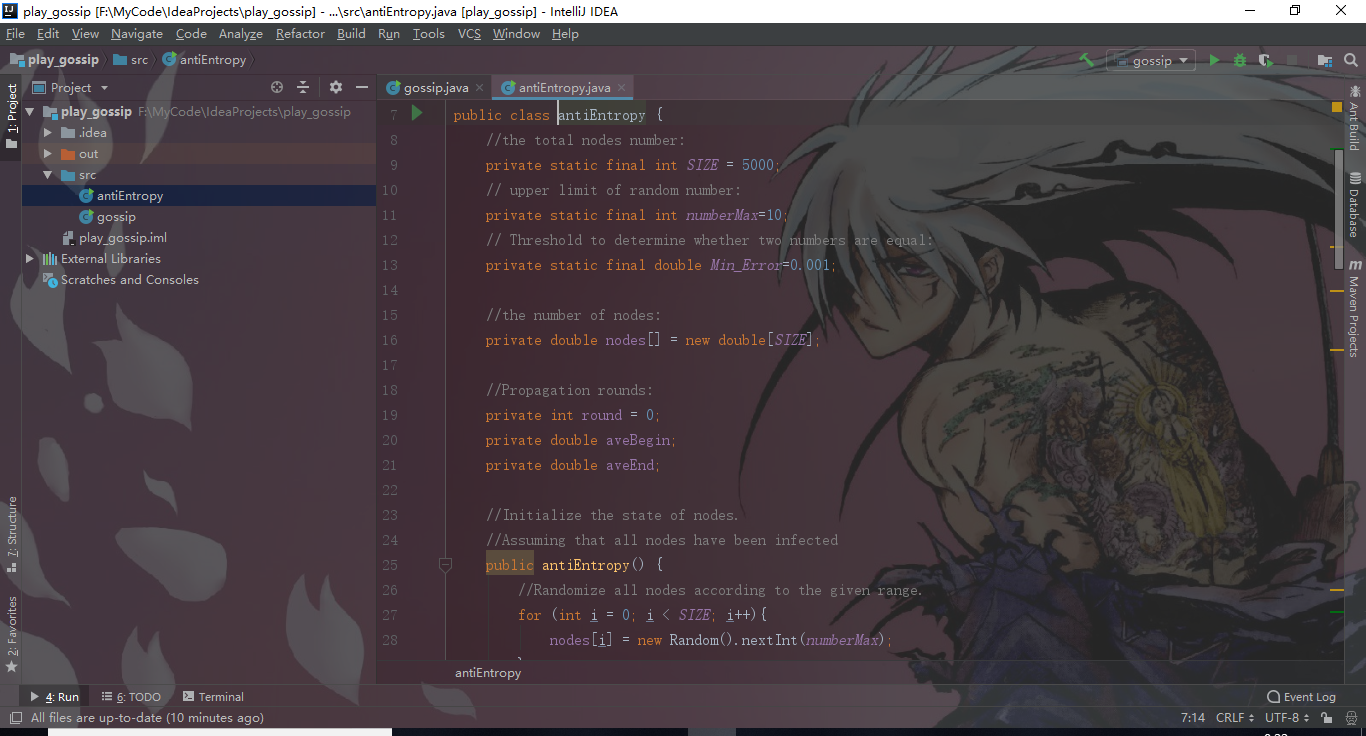
**实验一——反熵算法模拟实现**

1. 程序说明

本程序模拟实现了反熵算法，其传播以求平均数的方式呈现。

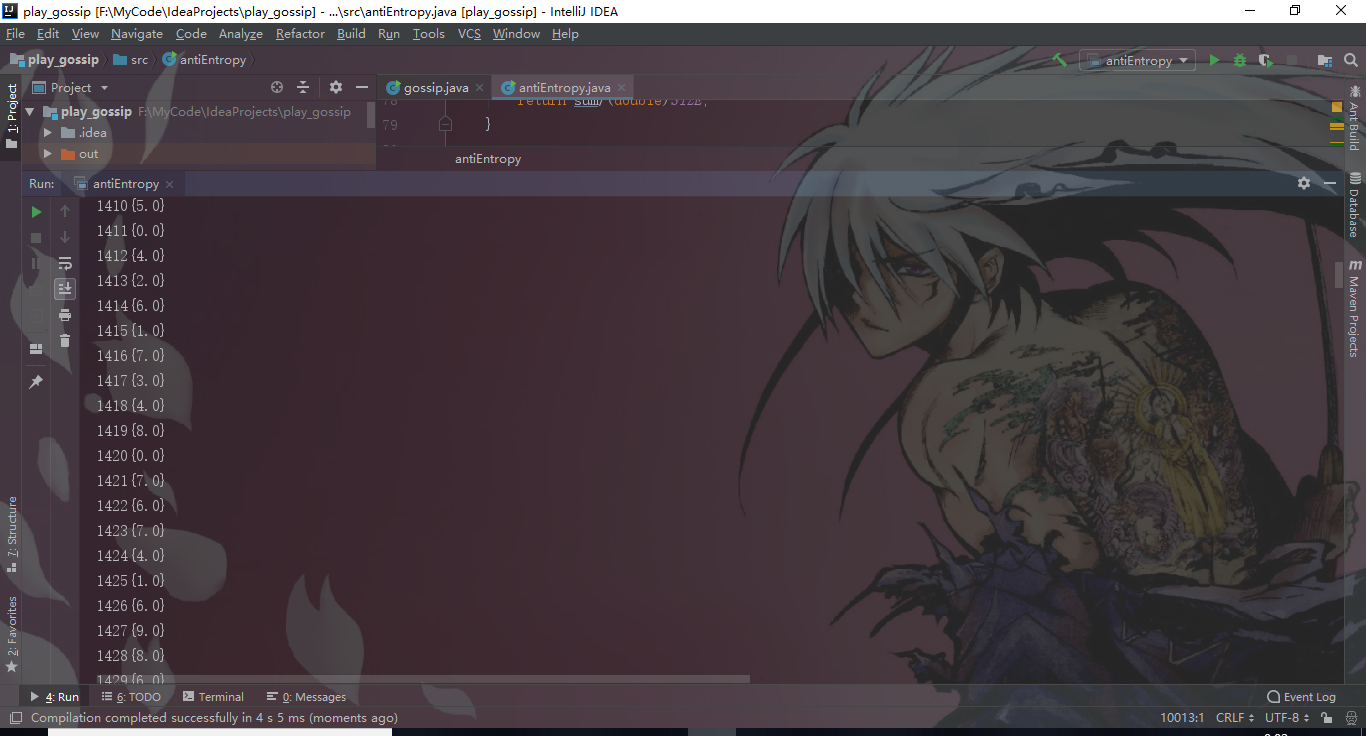
具体实现逻辑，请参见源代码中的注释。

注释内容详尽，这里不进行赘述。



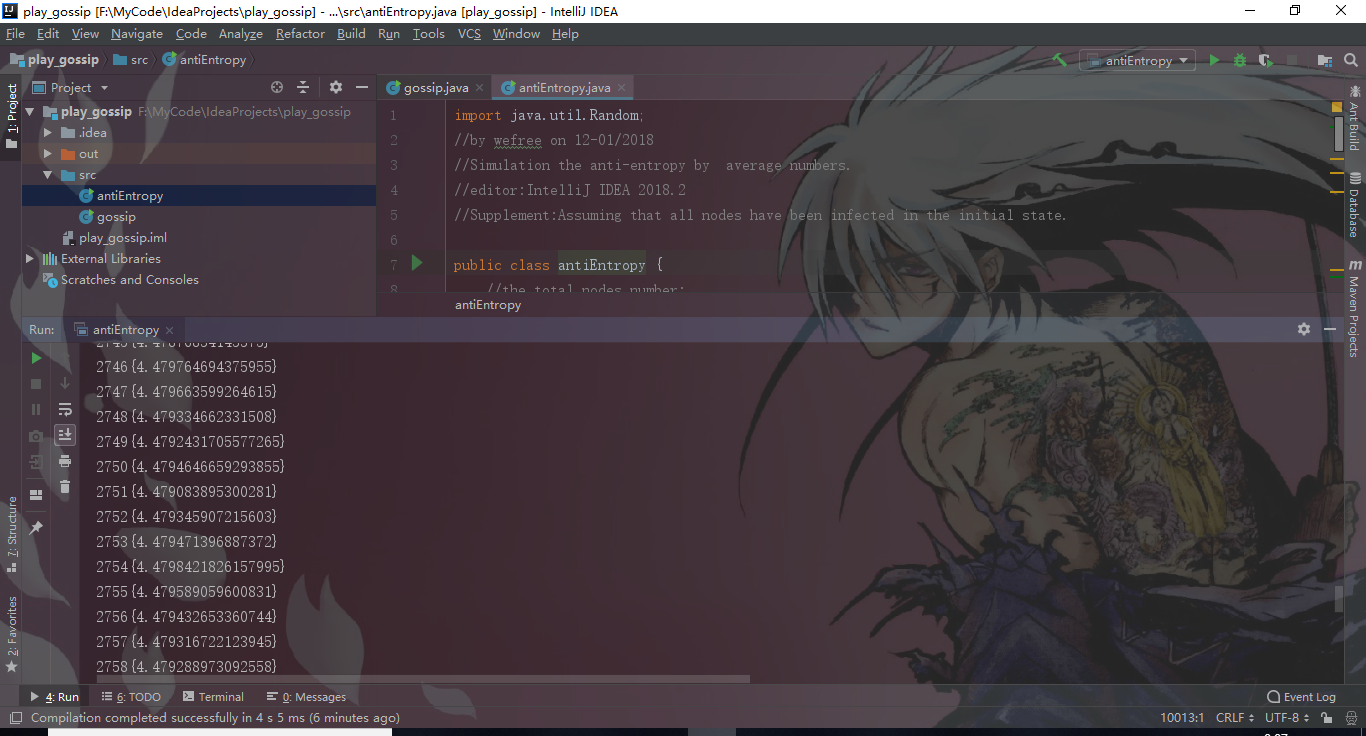
<反熵算法中定义的变量>

1. 运行结果及说明



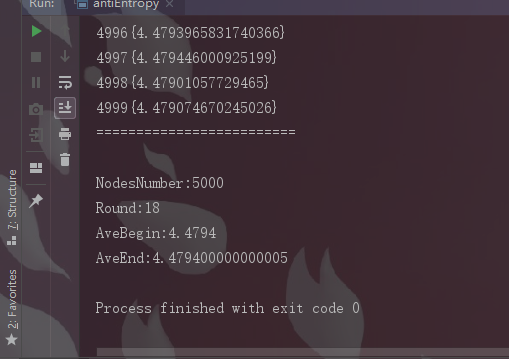
<图一>

图一说明：这里是节点的初始状态。一共有5000个节点，并假设所有节点都已经被感染，其数值为随机产生的0-10之间的数字。如图，左边为标号，右边括号内为初始化的节点数值。



<图二>

图二说明：这里是经过反熵算法运行之后的节点状态。如图，左边为标号，右边括号内为初始化的节点数值。



<图三>

图三说明：这里是反熵算法的几个参数展示。第一个是节点总数，第二个是传播轮数，第三第四个分别是算法运行前后的前后的平均值。

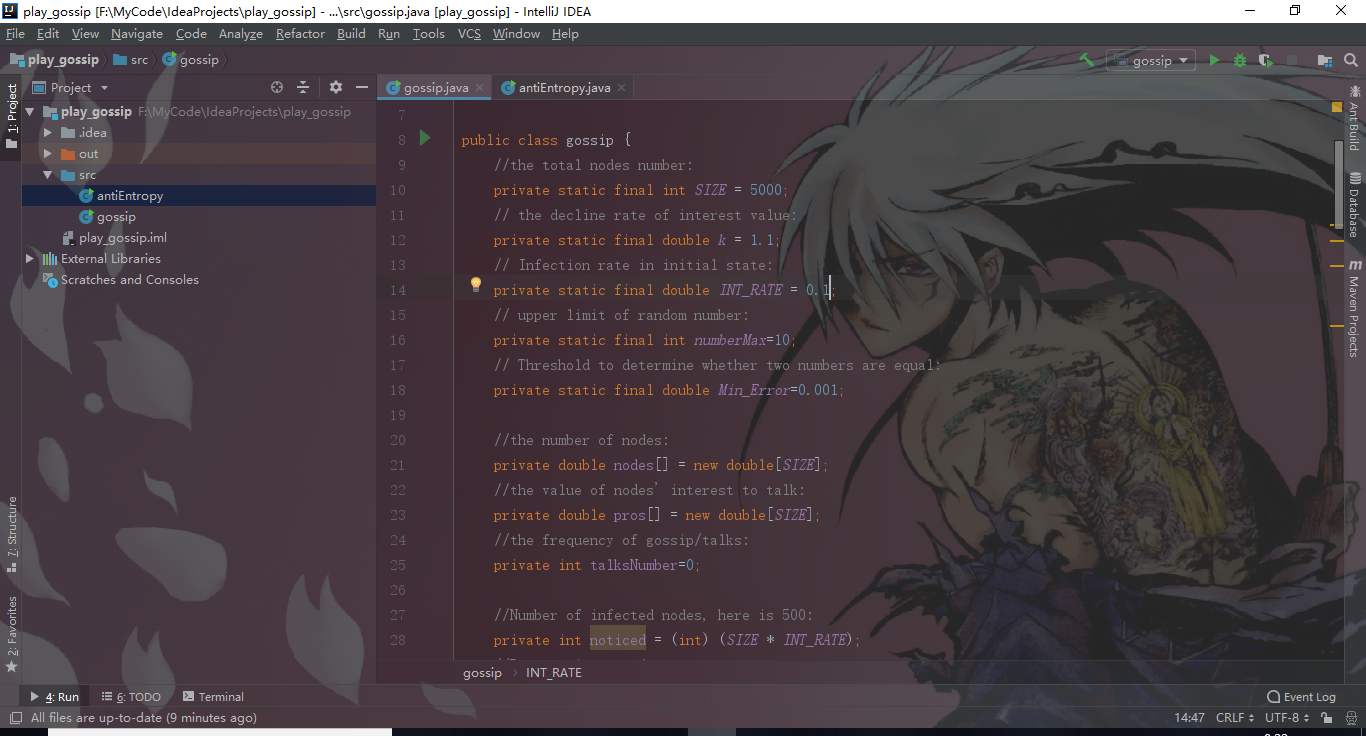
**实验二——gossip算法模拟实现**

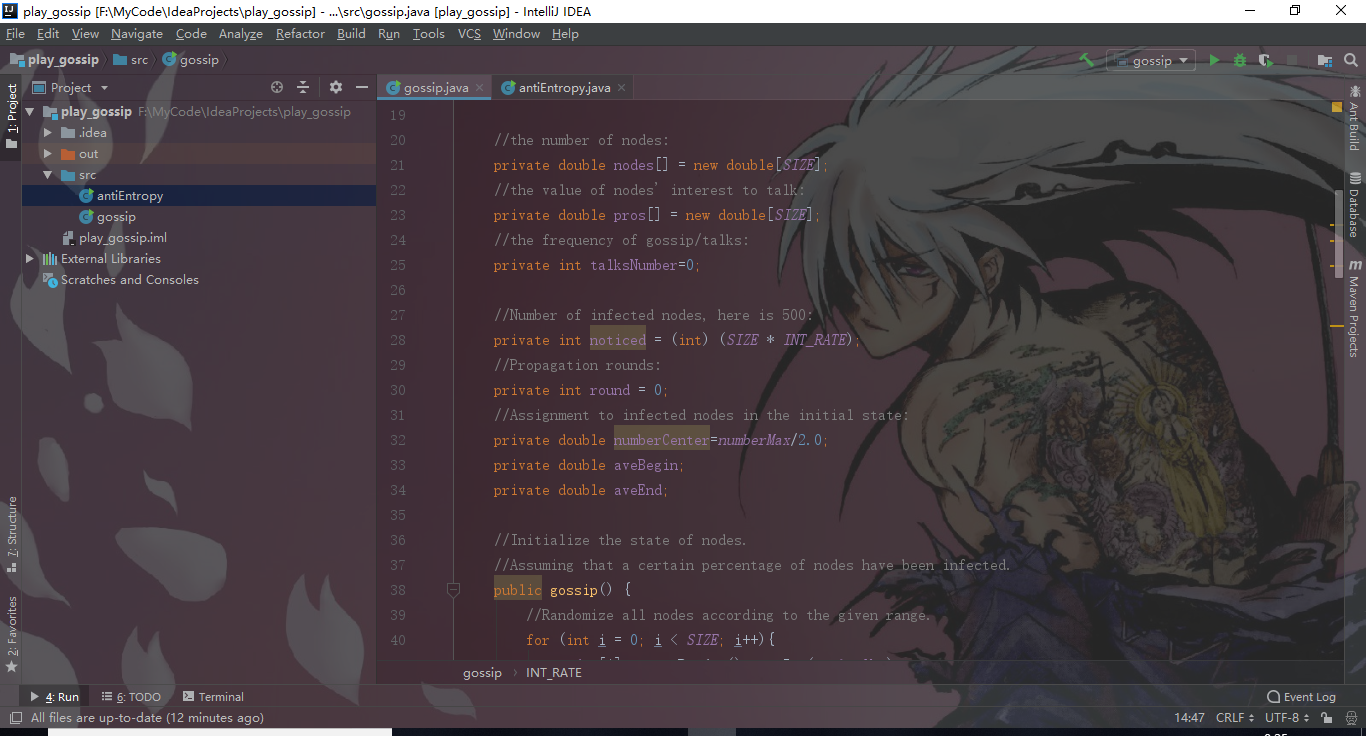
1. 程序说明

本程序模拟实现了gossip算法，其传播以求平均数的方式呈现。

具体实现逻辑，请参见源代码中的注释。

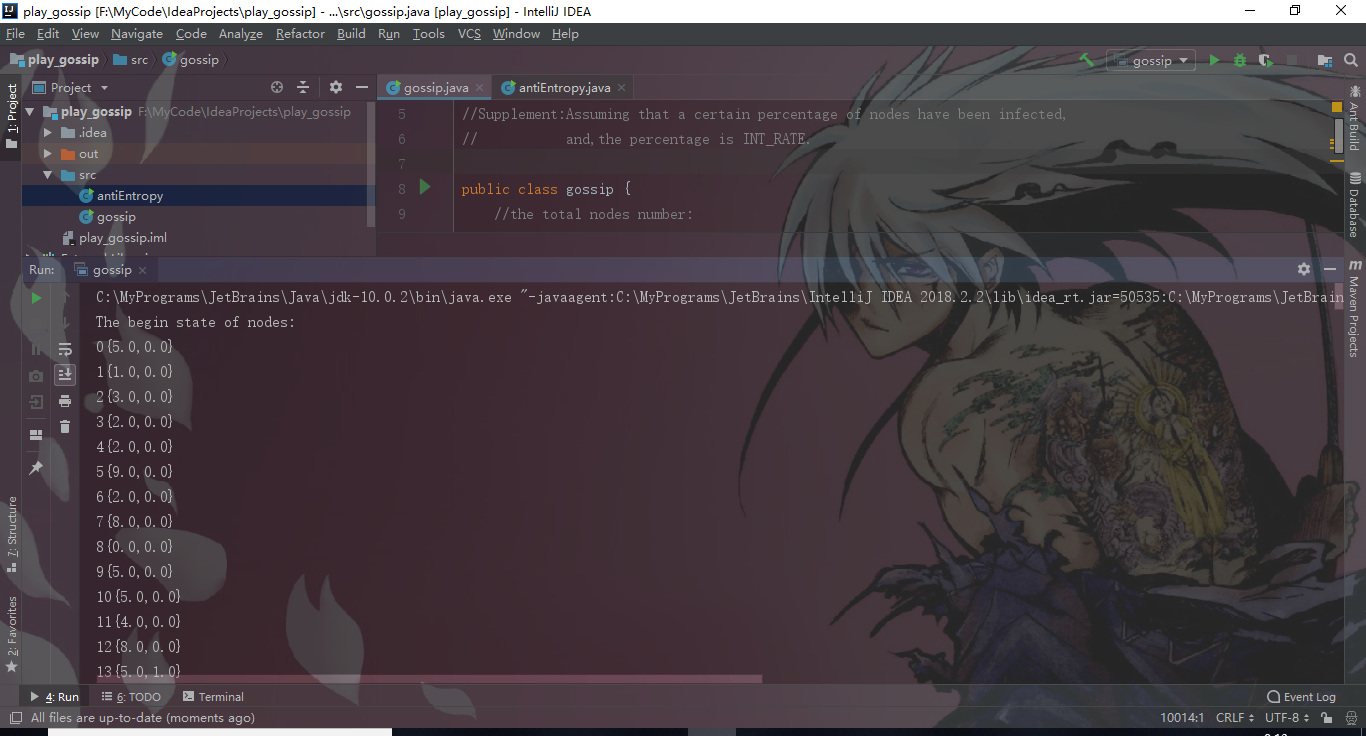
注释内容详尽，这里不进行赘述。





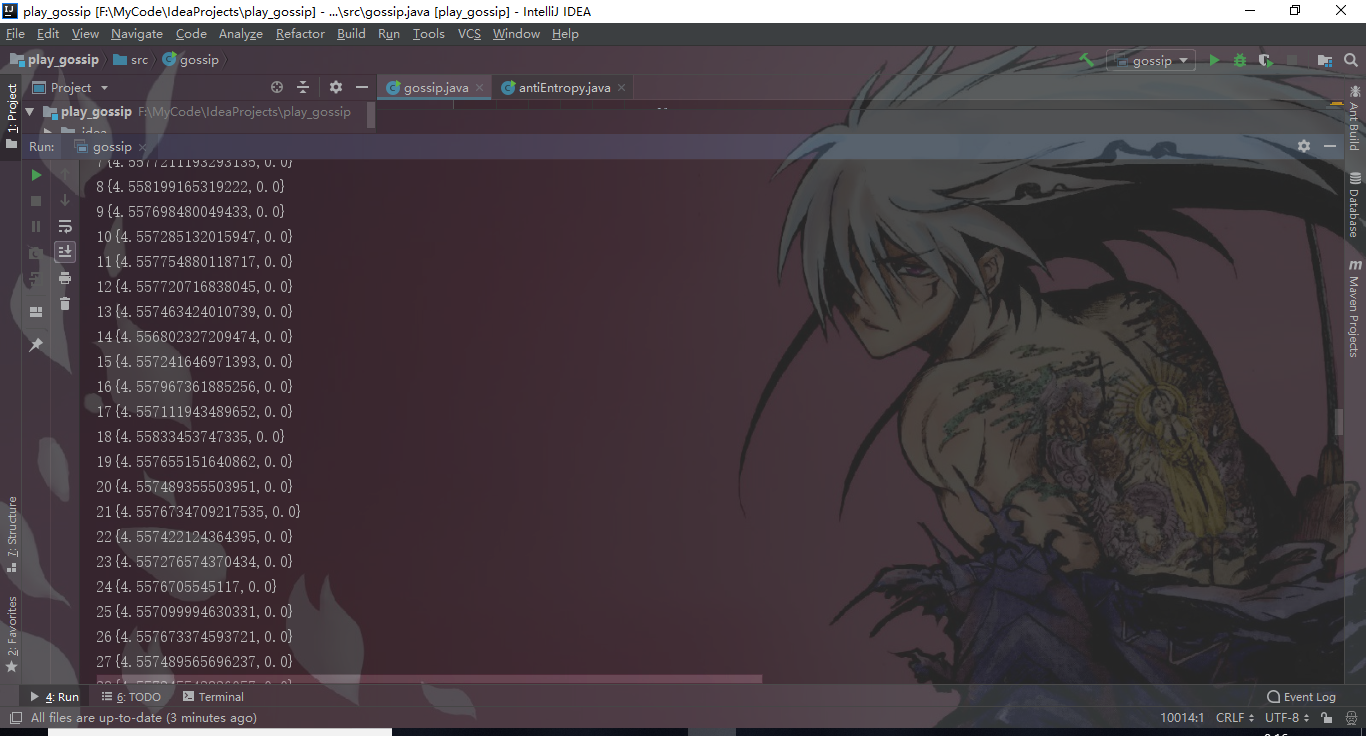
<gossip算法中定义的变量>

1. 运行结果及说明



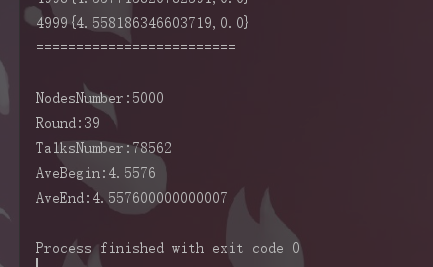
<图四>

图四说明：这里是节点的初始状态。一共有5000个节点，并假设其中500节点都已经被感染，感染节点的数值为5，其它节点的数值为随机产生的0-10之间的数字。如图，左边为标号，右边括号内为初始化的节点数值和该节点的兴趣指数。



<图五>

图五说明：这里是节点的最终状态。如图，左边为标号，右边括号内为节点数值和该节点的兴趣指数。



<图六>

图六说明：这里是gossip节点的若干参数。如图，第一个是节点总数；第二个是轮数；第三个是传染计数总数，或者说，一个节点跟另一个节点进行gossip一次就计数一次，如此得到的总数；第四第五个是算法运行前后的平均数。