

# 基于独立级联模型的社会网络信息传播问题

## 1. 引言

在给定社会网络结构的情况下，我们希望利用独立级联模型来确定最少的起始节点，以确保信息传播能够覆盖800个以上的节点。本报告旨在提供一个逻辑清晰的解决方案，通过独立级联模型及提出的算法来解决该问题。

## 2. 问题分析

### 2.1 社会网络的结构

我们使用开源库 `networkx` 对网络进行分析。其拓扑结构如下。

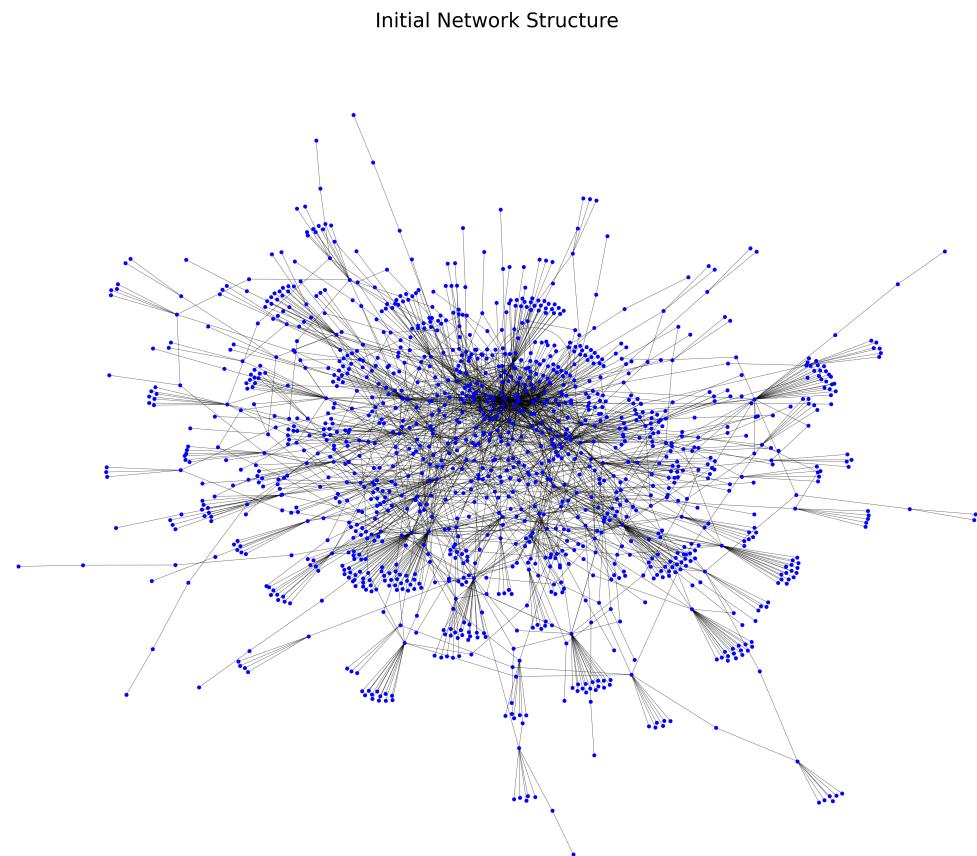


图1. 网络结构

对于社会网络中的信息传输，我们认为应该着重关注整体平均度数、集聚系数、网络密度和强连通分量数量，因为以上参数表征了网络的整体特性，即网络中结点之间的连接程度、网络的分散性及网络的划分数量。

运行结果图如下。

```
Average Degree: 3.3100944081336237  
Global Clustering Coefficient: 0.035248134549330803  
Average Clustering Coefficient: 0.0328791648995902  
Network Density: 0.00120279593318809  
Number of Strongly Connected Components: 1376  
Number of Connected Components: 1
```

图2. 网络参数

1. 平均度数 (Average Degree) 为3.3101，这表明社会网络中的节点平均有3个连接。该值可以作为度量节点连接程度的指标，考虑到网络中的结点规模超过1000，我们认为该网络的平均度数较小，这意味着节点之间的信息传播困难、传播范围狭窄。
2. 整体集聚系数 (Global Clustering Coefficient) 为0.0352，平均集聚系数 (Average Clustering Coefficient) 为0.0329，二者较低。这表示社会网络中的节点之间形成的密集子图较少，节点的朋友之间的连接程度相对较低。因此该社会网络中存在一定的分散性，节点之间的信息传播可能更多地通过长程传播而非邻近节点之间的传播。
3. 网络密度 (Network Density) 为0.0012，这个值极低。网络密度表示实际边数与可能的最大边数之比，是衡量网络紧密程度的指标。极低的网络密度表明社会网络中的节点连接稀疏，信息传播的路径可能长。
4. 强连通分量数量 (Number of Strongly Connected Components) 为1376，弱连通分量 (Number of Connected Components) 为1。强连通分量表示网络中相互可达的节点集合，较多的强连通分量表示社会网络中存在多个相对独立的子群体，信息传播在这些子群体之间可能存在一定的障碍。

根据上述分析，该社会网络中的各结点通过有向边连接成一个弱联通图，具有相当强的分散性和稀疏性，信息传播可能通过较长的路径进行，并且存在多个相对独立的子群体。这些特征会对信息传播的速度、范围和效果产生影响。而由于整体集聚系数较低，节点之间的信息传播更倾向于通过一些重要的中心节点或者跨越子群体的桥接节点。

## 2.2 算法设计

针对社会网络中的信息传播问题，我们提出了两个算法，其基本逻辑类似，但针对模拟的准确性和速度各有侧重。

在模拟中，节点之间信息传播的成功率 $p$ 为恒定值。

### 2.2.1 算法1

若在本问题中使用蒙特卡洛模拟，考虑到结点规模较大，且每次模拟信息传播使用的独立级联模型的计算成本较高，因此我们在此基础上优化。

首先定义一些名词：

信息传播模拟：利用BFS算法，每两个之间传播成功的概率为 $p$

seedSet：即最终选出的初始节点的集合（源节点集），初始为空

结点集的信息传播范围：即对结点集进行信息传播模拟后，成功被传播的点的个数

节点的边际收益：对于现有源节点集而言，增加此节点带来的信息传播范围增长

具体算法过程如下。

当seedSet的信息传播范围小于预设值800时：

step 1：将每个未被激活结点依次加入seedSet，进行信息传播模拟，记录每个结点传播的边际收益，并进行从大到小的排序，其中第 $k$ 大的结点记作 $n_k$ ，其传播范围记作 $S_k$

step 2：选取 $n_1$ 进入seedSet（即最终选出的初始节点的集合），并将其激活

step 3: 选取 $n_i$  (初始时 $i=1$ ) , 与seedSet一起进行一次信息传播模拟, 记此时传播边际收益为 $S'_i$   
 step 4: 若 $S'_i$ 大于等于 $S'_{i+1}$ , 则将 $n_i$ 加入seedSet, 并将其激活, 重新回到step 1; 若 $S'_i$ 小于 $S'_{i+1}$ , 则  
 i++并回到step 3  
 step 5: 继续比较 $S'_i$ 与 $S'_{i+1}$ , 若 $S'_i$ 大于等于 $S'_{i+1}$ , 则将 $n_i$ 加入seedSet, 并将其激活, 重新回到step  
 1; 若 $S'_i$ 小于 $S'_{i+1}$ , 则i++并回到step 3

## 2.2.2 算法2

为了充分利用给定网络的稀疏性, 加快模拟速度, 可以对上述算法进行改进。

当seedSet的信息传播范围小于预设值800时:

step 1: 将每个未被激活结点依次加入seedSet, 进行信息传播模拟, 记录每个结点传播的边际收益, 并  
 进行从大到小的排序, 记为序列 $L$ , 其中第k大的结点记作 $n_k$ , 其传播范围记作 $S_k$   
 step 2: 选取 $n_1$ 进入seedSet (即最终选出的初始节点的集合), 并将其从 $L$ 中删除  
 step 3: 选取 $n_i$  (初始时 $i=1$ ), 与seedSet一起进行一次信息传播模拟, 记此时边际收益为 $S'_i$   
 step 4: 若 $S'_i$ 大于等于 $S'_{i+1}$ , 则将 $n_i$ 加入seedSet, 并将其从 $L$ 中删除, 令 $i = 1$ 重新回到step 3; 若 $S'_i$   
 小于 $S'_{i+1}$ , 则i++并回到step 3

改进后的算法只用对所有节点模拟信息传播遍历一次, 由于网络的分散与稀疏, 每多一个节点被选入源  
 节点后, 对剩余结点的边际收益的影响是平均的, 因此可以充分信任第一次模拟信息传播的边际收益排  
 序结果, 因此每一轮增加都可以依据此排序结果, 顺序比较两点的边际收益, 从而最终选出满足条件的  
 源节点集

## 3. 实验过程

实验中, 对于算法1, 考虑到其运行所需时间较长, 我们选择迭代150次, 每轮迭代产生所需源节点的编  
 号集和数量, 最终计算出可以将信息传播到800个节点所需源节点的平均个数n, 并根据每轮迭代的结  
 果, 选择被选中为源节点次数前n大的节点作为最终的源节点集。

对于算法2, 考虑到每次迭代的时间成本低于算法1, 在尝试后我们选择迭代500次, 同理生成最终的源  
 节点集。

在节点间信息传播成功率p值的设定上, 我们分别选择0.2、0.4、0.6、0.8、1.0进行试验, 得到源节点  
 集, 并且利用matplotlib库, 画出随着源节点集的扩大, 可以被源节点集传播到信息的点的个数的变化  
 趋势。同时, 我们利用networkx库可视化该社会网络和最终选择的源节点集。

## 4. 实验结果分析

**迭代分析** 首先为了确保迭代次数选择的合理性, 我们选用 $p=0.5$ 进行了迭代分析, 由下图可知, 算法1和  
 算法2的试验中, 源节点集中节点数的均值随着迭代次数的增加, 逐渐趋于稳定。因此可以判断, 迭代次  
 数的选择是合理的。

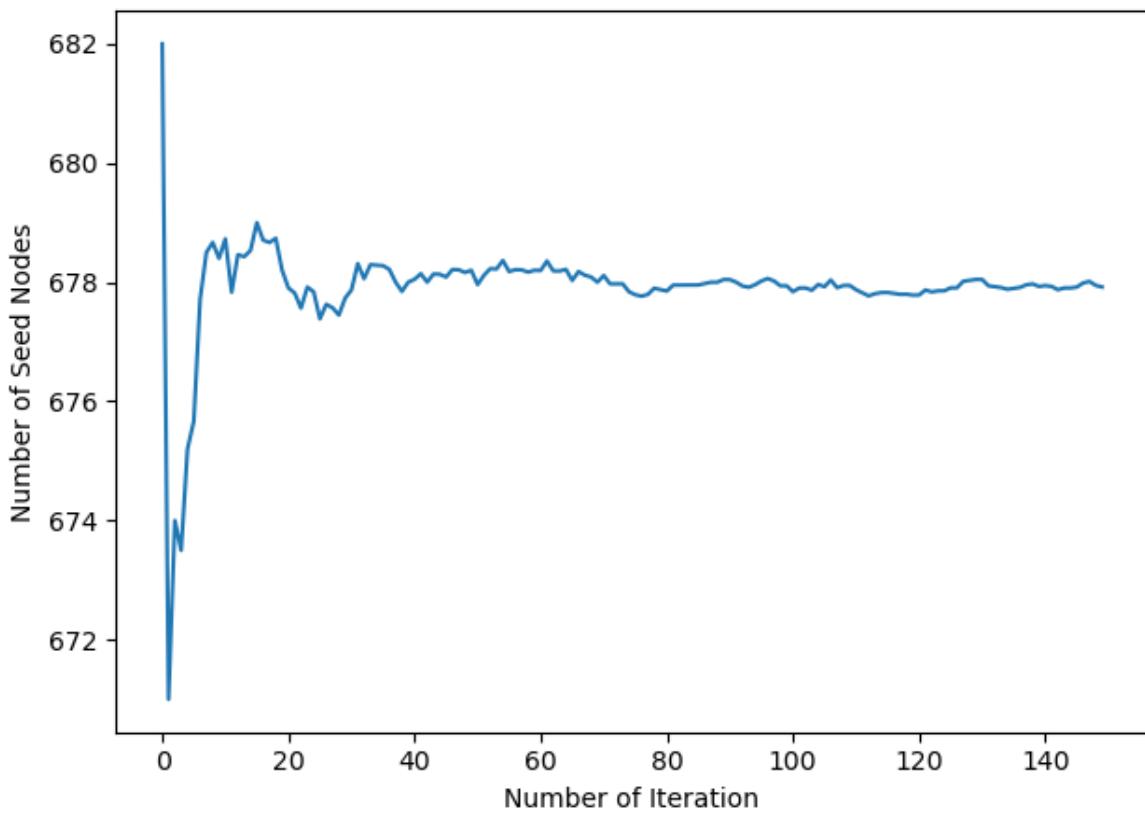


图3. 算法1迭代分析

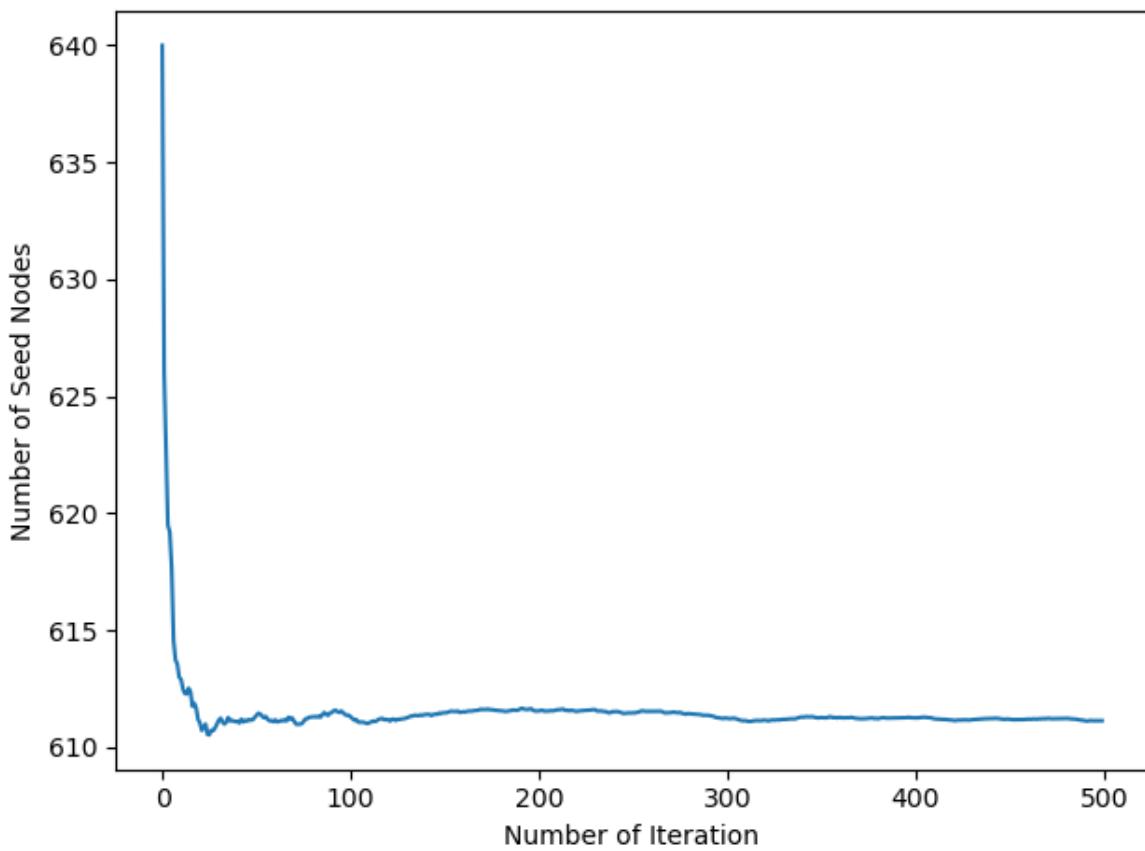


图4. 算法2迭代分析

**模拟过程分析** 下图的横坐标为源节点集的个数，纵坐标为信息传播到的点的个数，代表在一次迭代的过程中，源节点数量增加，信息传播到的点的个数也随之增加。但这种增加是趋势性的，而非严格单调递增，这是因为虽然源节点集的个数在增加，但每次信息传播的模拟是独立的，有可能发生更多的节点将信息传播给更少的节点的情况。

Propagation Analysis

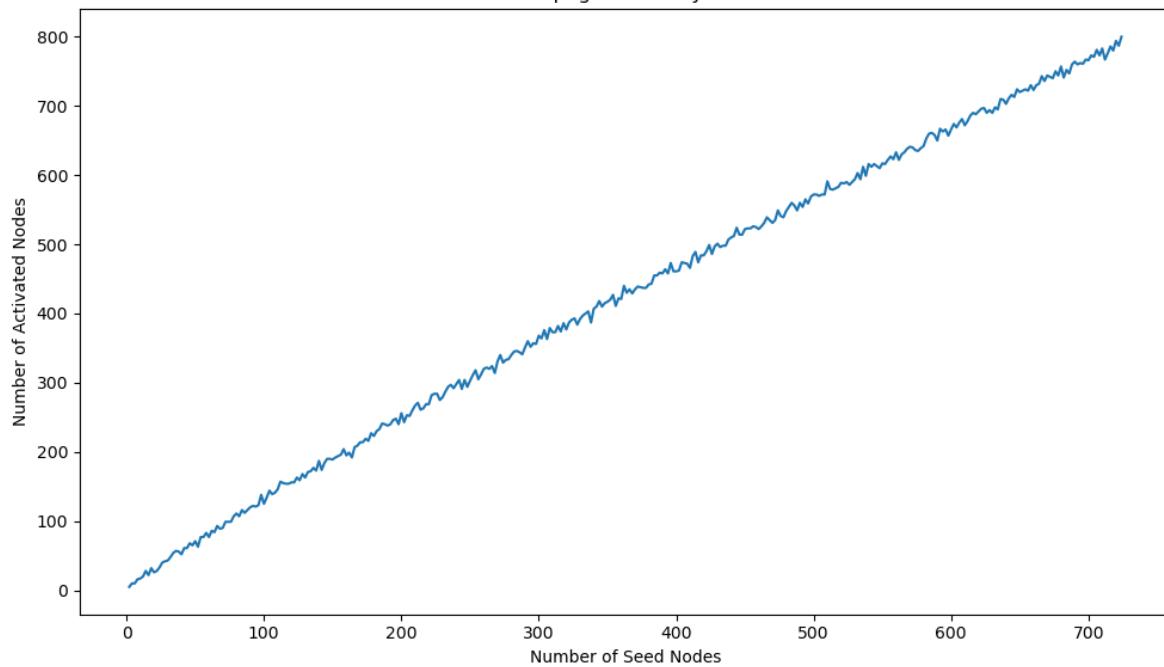


图5. 算法1传播模拟( $p=0.2$ )

Propagation Analysis

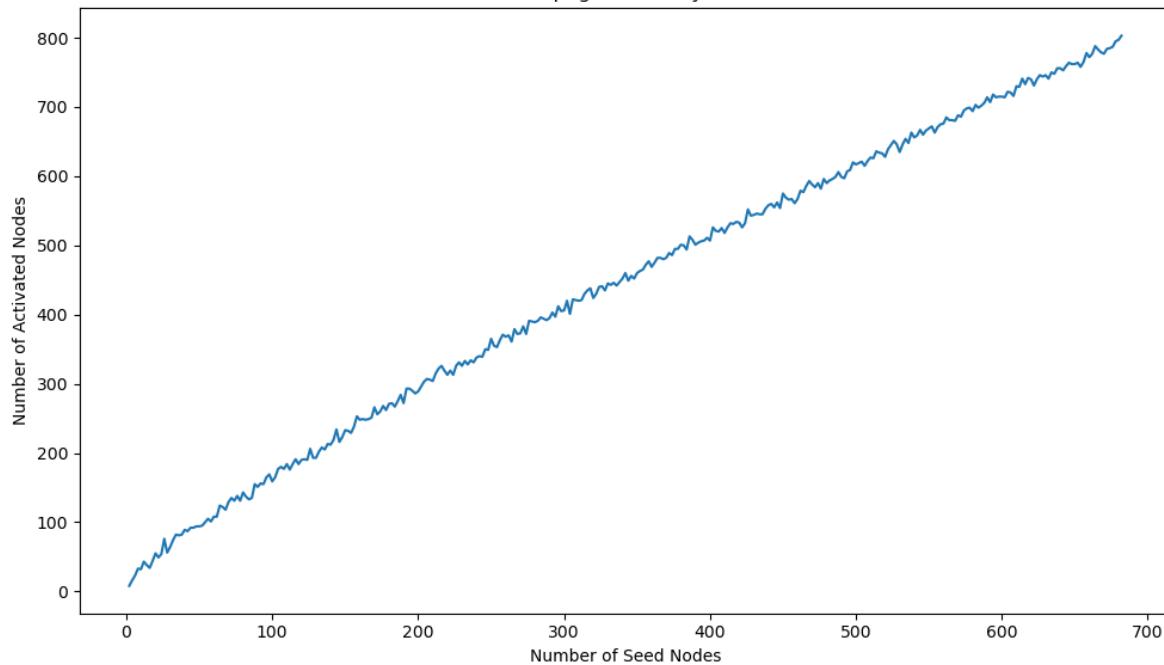


图6. 算法1传播模拟( $p=0.4$ )

Propagation Analysis

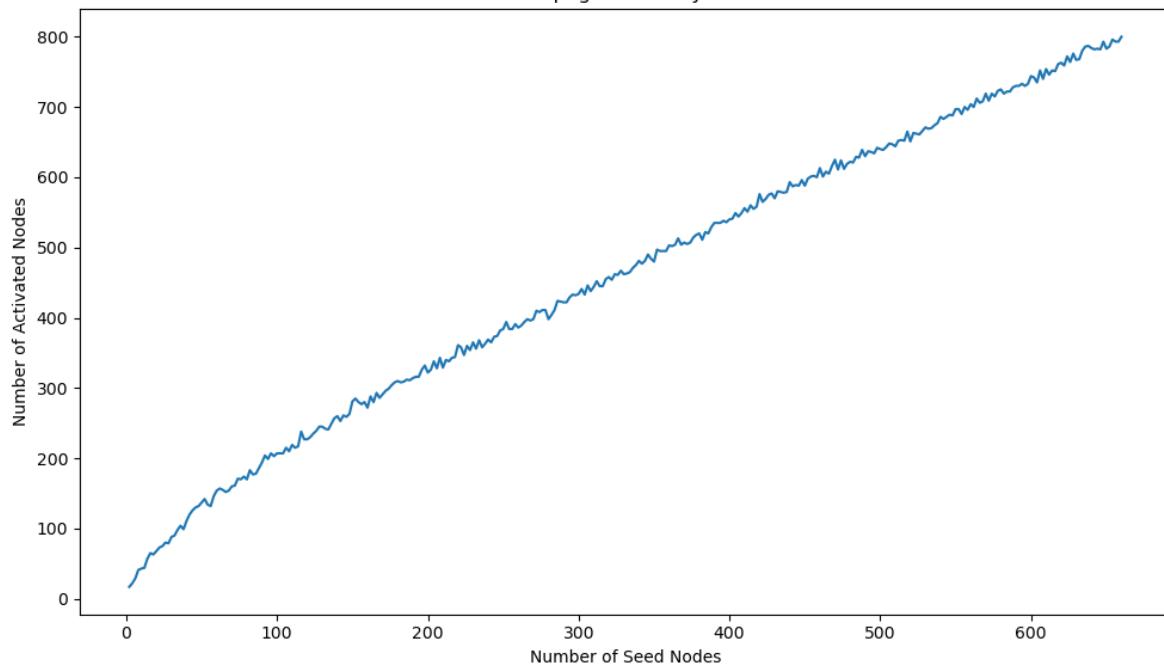


图7. 算法1传播模拟( $p=0.6$ )

Propagation Analysis

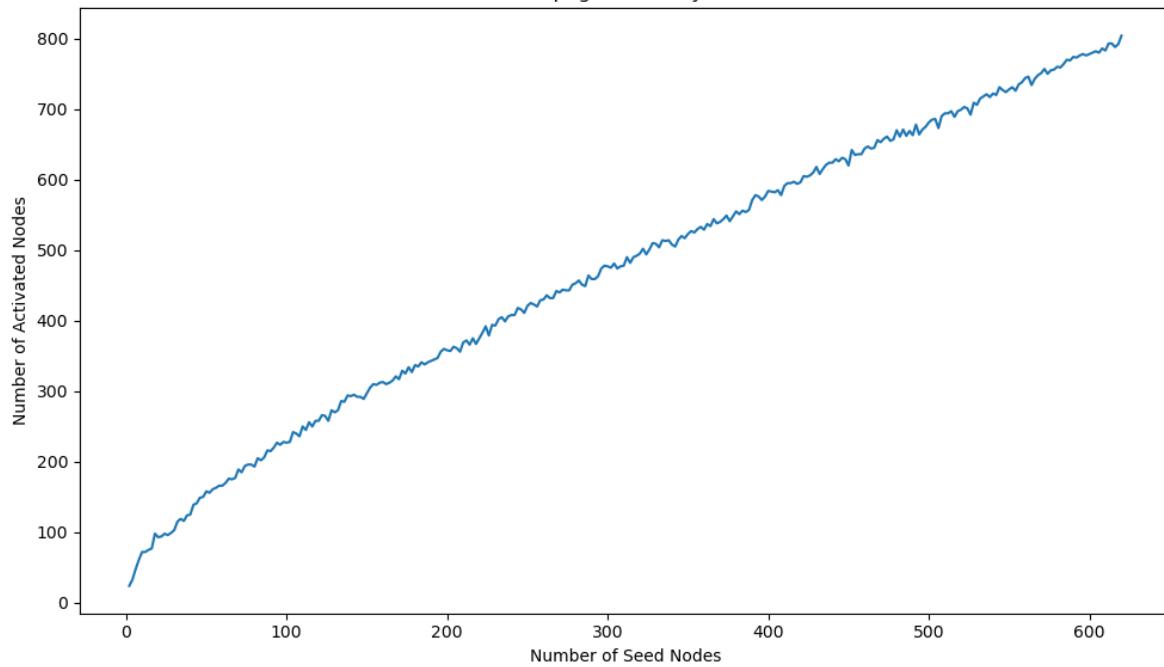


图8. 算法1传播模拟( $p=0.8$ )

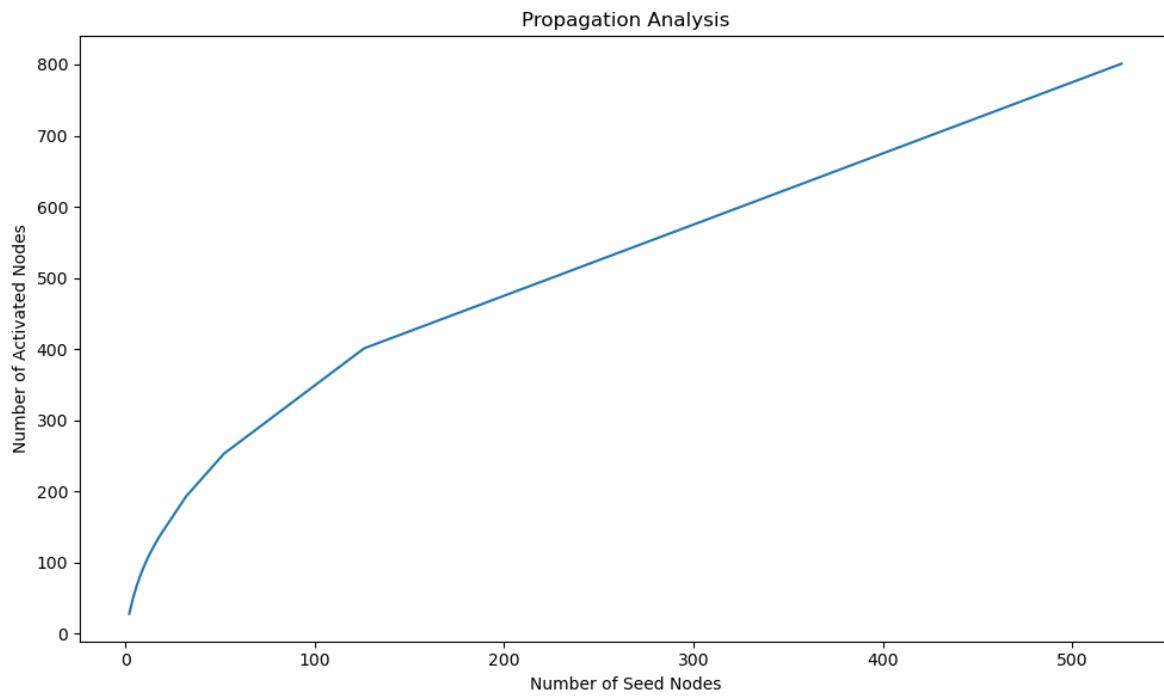


图9. 算法1传播模拟( $p=1.0$ )

以上为不同 $p$ 时，算法1的源节点集中节点数的变化

图10. 算法2传播模拟( $p=0.2$ )

图11. 算法2传播模拟( $p=0.4$ )

图12. 算法2传播模拟( $p=0.6$ )

图13. 算法2传播模拟( $p=0.8$ )

图14. 算法2传播模拟( $p=1.0$ )

以上为不同 $p$ 时，算法2的源节点集中节点数的变化。下表总结了不同 $p$ 值下，两个算法得出的源节点集中节点数。若算法1和算法2的结果分别为 $n_1$ 和 $n_2$ ，若定义差值占比 $\eta$ 为

$$\eta = \frac{|n_1 - n_2|}{(n_1 + n_2)/2} \times 100\%$$

|         | 算法1   | 算法2   | 插值占比  |
|---------|-------|-------|-------|
| $p=0.2$ | 730.8 | 699.8 | 4.33% |
| $p=0.4$ | 693.8 | 644.6 | 7.35% |
| $p=0.6$ | 661.8 | 610.5 | 8.06% |
| $p=0.8$ | 618.6 | 582.7 | 5.98% |

|       | 算法1   | 算法2   | 插值占比 $\eta$ |
|-------|-------|-------|-------------|
| p=1.0 | 526.0 | 552.4 | 4.90%       |

**对比在同一p时，算法1和2的结果的对比** 我们选定p=0.6，通过以下结果可知，算法1得到的源节点集的节点数为661.8，算法2得到的源节点集的节点数为610.5。进一步分析二者得到的源节点集的相似度，若两个节点集相同节点的个数为 $n_s$ ，两个节点集的节点数分别为 $n_1$ 和 $n_2$ ，定义该相似度 $\mu$ 为：

$$\mu = \frac{n_s}{\max\{n_1, n_2\}}$$

根据此定义，相似度为84.13%。因此可以断定，两个算法得到结果是相似的，二者的效果基本一致。同样可以通过对比下面可视化网络，其中红色的节点为源节点，观察可知两个算法结果时相似的。

Similarity: 0.84134179

图15. 相似度分析

图16. 算法1结果可视化

图17. 算法2结果可视化

**比较两个算法的运行效率** 进行上述对比时，我们记录每轮迭代的运行时间，结果如下图。

running time: 204.3322479724884 s

图18. 算法1运行时间

running time: 1.2575268745422363 s

图19. 算法2运行时间

可见算法1的每轮迭代时间为204.33秒，算法2的运行时间为1.26秒，算法2的运行时间仅为算法1的0.62%，这是因为其只对所有节点遍历一次，进行模拟信息传播，即遍历一轮选出源节点集。而算法1每遍历一轮，选中2个节点进入源节点集，效率较低但准确度较高。

## 5. 结论

根据以上分析，我们选择使用算法2进行项目问题的求解，问题中参数设定为“传播概率p=20%，传播时间t=8”，因此将p选为0.2，在进行模拟信息传播时，约束深度优先搜索BFS时的搜索深度小于等于8，迭代次数为500，得到源节点集中节点数的平均值，并可视化选择的节点。结果如下。

number of seed node: 700.024  
number of activated node: 802.982

图19. 结果

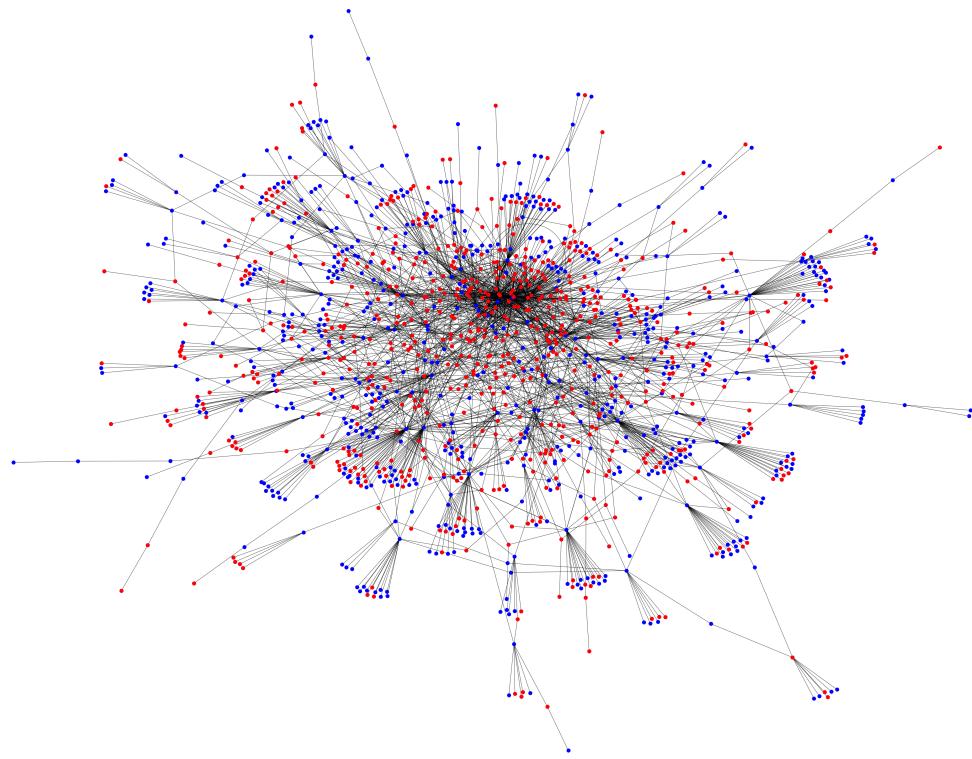


图19. 结果可视化

对于本项目的问题，需要选择700个源节点。

## 6. 讨论

### 6.1 总结研究工作

本项目基本解决了在给定社会网络时，根据独立级联模型分析信息传播的问题，并提出了解决该问题可以使用的两个算法。其中，算法1更为普适，近似程度较低，适用于网络密集程度，并在准确性和高效性之间取得平衡；算法2更多针对较为稀疏的网络，近似程度较高，但可以更高效地得到源节点集。

### 6.2 研究展望

本项目结果具有局限性，没有充分利用网络本身的特性，例如筛选源节点时可以更多利用节点的度、中心性和网络的聚类系数，也可以利用网络本身的社区结构，优化算法设计。同时，在代码层面，有些算法并没有优化到最佳，也没有使用并行操作以提高计算效率，这有待进一步研究。