**Abstract**

现有的研究忽略了位置信息在影响中可以发挥重要作用的事实，本文研究位置感知的影响阻塞最大化问题

**Introduction**

随着科学技术的发展，越来越多的个人和公司使用社交网络来促进他们的业务，传播新闻、思想和观点，社交网络有许多研究应用，如链接预测、谣言控制、影响力最大化。然而，大多数研究工作只考虑单一影响的传播。事实上，不同的观点甚至相反的观点在社交网络中同时传播的情况经常发生。一个自然的问题是，当社会网络中存在竞争时，影响力如何传播，我们称之为影响力竞争问题。影响力阻塞最大化( IBM )问题是一种影响力竞争问题。IBM问题的目标是尽可能阻止竞争对手的影响力的扩散。它忽略了位置信息在inuence传播中的重要作用。

在一个图中，用户之间的线表示用户的社交关系，虚线表示的矩形R1或R2表示地理区域。有两家竞争激烈的公司，甲公司和乙公司，在这个城市生产同样的商品。用户N是甲方的销售员。现在，乙方想雇佣一个人在给定的区域销售商品，以降低甲方的业绩。有三个候选人P1、P2和P3。我们可以看到，位置信息是选择合适推销员的一个非常重要的因素。如果没有区域限制，将选择P1。然而，如果该区域限于R2，则将选择P2而不是P1

LIBM的目标是将大小为k的最佳正种子集S P分配给nd，从而使阻塞的负inuence扩散最大化。

我们证明了在齐次竞争独立级联模型下，LIBM问题是NP难的，inuence扩散函数是单调的和子模的，因此，基于贪婪的方法可以用来解决LIBM问题。为了克服蒙特卡罗贪婪算法的低efciency，我们设计了一种启发式算法libm - h。libm - h的设计有两个主要挑战。rst是如何识别位于给定查询区域的节点集。我们使用四叉树索引结构来存储节点的位置信息，并且使用depth-rst顺序搜索方法来搜索位于从四叉树的根开始的查询区域中的nd节点。二是如何设计有效的选种方法。我们使用Chen等人提出的最大inuence树状结构。在[ 7 ]和类似于[ 8 ]的动态编程算法，以预先计算网络中任何节点的负激活概率。

据我们所知，我们是第一个研究位置感知社交网络下的IBM问题。总之，我们的贡献如下:

1. 我们第一个提出了LIBM问题，并证明了它是NP难的，影响力扩散函数是单调的和子模的。2.我们提出两个启发式的算法LIBM-H和LIBM-c来解决LIBM问题。3. 在真实数据集上的大量实验结果表明，我们的算法能够达到与贪婪算法的匹配效果，并且通常比其他启发式算法更好，同时比贪婪算法快4个数量级。

**2.相关工作**

模型：Kempe提出的IC模型和LT模型，在竞争的社交网络中，也有许多基于IC或LT模型的扩展模型。巴拉蒂等人提出了多个影响力竞争的IC模型，是原来的推广。Chen等人[ 11 ]考虑网络中的负面意见，提出了IC-N模型，该模型有一个质量因子q来模拟意见变成负面的行为。Carnes等人[ 3 ]提出了竞争社交网络中基于距离的模型和波传播模型

算法：1.基于仿真：CELF 2.基于启发式：MIA 3.基于采样的：RIS

IM扩展：李等人提出的基于位置感知影响力最大化问题。在[ 27 ]。LIM问题的目标是选择k-best种子，该种子能够在给定的查询区域中inuence最大数量的节点。Wang等人[ 28 ]考虑位置感知提升问题，提出一种基于锚点的估计算法，zhou等人[ 29 ]设计了一个两阶段( TP )模型，并提出了两种启发式算法来解决位置感知的inuence最大化问题。

IBM问题也是IM问题的扩展

**3.准备工作**

A IM：

给一个社会网络，一个影响传播模型，一个正整数k，影响力最大化问题就是要找到一个种子集合S，它包含k个种子，使得它在这个图中的影响范围最广。单调性和子模性

B 独立竞争级联模型

在我们的工作中，用同构的CIC模型去模拟影响力的传播，CIC模型是魏等人提出的模型。。我们将其表述如下：

**定义2**：竞争的独立级联模型：一个社交网络G，每个节点有三种状态，未激活，正向状态，反向状态。每个边有正向传播概率PP（u,v）和负向传播概率PN（u,v）一开始有两个种子集合，一个是正向种子集合一个是负向种子集合。两种传播方式各自独立。在t步时，一个未激活的节点u，变为负向的，他有一次机会对他的未激活邻居节点进行负向传播，反之亦然。如果一个节点被影响到了了，它在t+1时刻被激活。U只有一次机会激活别人。如果在某一步时，没有新的节点被激活，传播结束。

两种平局决胜（当一个节点同时被positive和negative影响）：TB-FP：给一个固定概率。TB-FP（0）代表negative。TB-PP利用一个比例概率。

我们称一个CIC模型是同构的如果每个边上的正向传播和负向传播概率是一样的。

**C 影响阻塞最大化**

IBM的目标是通过选择多个正向种子来尽可能多地阻止一组负种子的影响传播。

**定义**：

**影响阻塞集合**（IBS）：给一个社交网络G，一个竞争影响传播模型，一个负向种子集合Sn，一个正向种子集合Sp的影响阻塞集合为：如果正向种子集合为空会被负影响，如果正向种子集合为Sp，则不会被负影响。

**封锁负面影响**（BNI）：给一个社交网络G，一个竞争影响传播模型，一个负向种子集合Sn，一个正向种子集合Sp的封锁负面影响为IBS的期望

**影响力阻塞最大化**（IBM）：通过一个正向集合Sp让BNI最大

**4.位置感知的影响力最大化问题**

在位置感知的竞争社交网络中，每个节点v都有一个地理位置( x，y )，其中x是经度，y是纬度。给定一个查询 Q=(R,k)，其中R是一个地理区域，k是正种子集的大小，我们正式将LIBM问题定义为: 给定位置感知的社交网络G、竞争影响力传播模型、负种子集Sn和查询Q ( R；k )，位置感知的inuence阻塞最大化是找到大小为k的正种子集的问题，使得R中的阻塞负影响最大化。

定理：在cic模型汇总，LIBM是NP难的，Wei等人[ 12 ]证明了IBM问题CIC模型中的影响力扩散函数是单调的。不幸的是，他们证明了影响力扩散函数对于IBM问题不具有子模性。然后，他们提出了以下定理：带阻塞的影响力传播具有子模性当在同构的CIC模型并且竞争规则是TB-FP（0）或1和TB-pp。

更准确地说，如果t节点v同时被正节点和负节点激活，我们使用TB-FP(0 )作为平局决胜规则。这符合消极观点不太可能改变的常识。

**5.LIBM-H算法**

我们提出了一种基于整体的算法LIBM-H来解决LIBM问题。该算法基于四叉树和MIA。

**A 四叉树索引结构**

LIBM的第一个挑战是如何识别位于给定查询区域的节点集。为了解决这个问题，我们使用四叉树索引结构来存储节点的位置信息，并使用depth-first顺序搜索方法从四叉树的根到位于查询区域的节点进行搜索。

在四叉树中，每个单元都是由<SID, M, NSET, SE, SW, NE, NW>表示

SID：序列号， M是该单元格内位置的最小边框，NSET是位于M中的一组节点，SE, SW,NE,NW是指向子单元格的指针, 除叶节点外，每个节点有四个子节点。

QCdd是在R中的节点集合。

**B MIA结构**

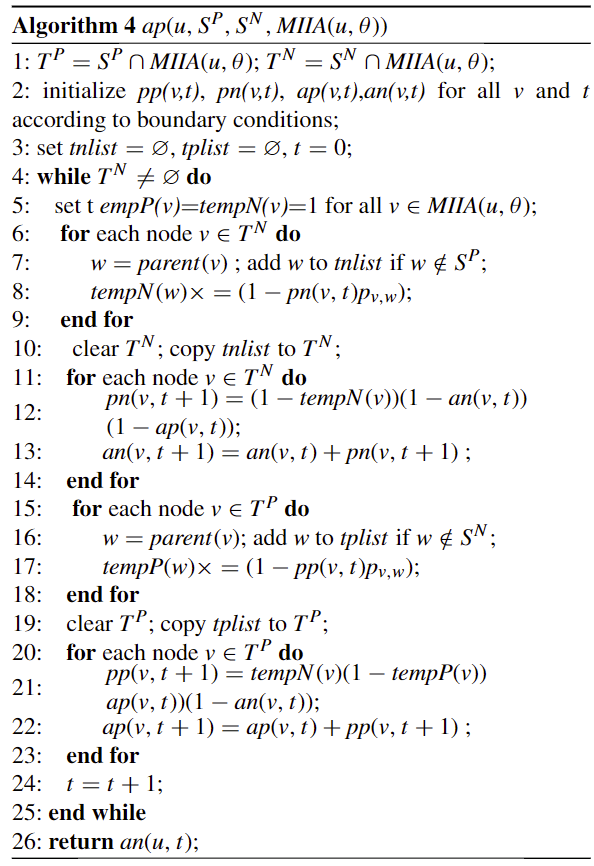
由于贪心算法代价太大，我们使用影响力最大化树状结构来计算影响的传播。在MIA中，节点u只能通过最大影响路径影响到v。u到v有好几条路径，计算每条路径的概率（乘法）然后找到概率最大的那条。Ap（u,S,MIIA(v,o)）的概率为：

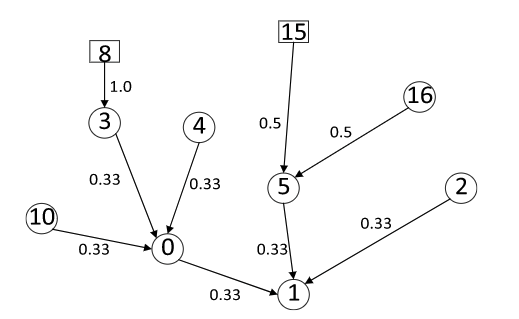
用这个想法无法计算负影响概率如果它的正向集自己不为空。

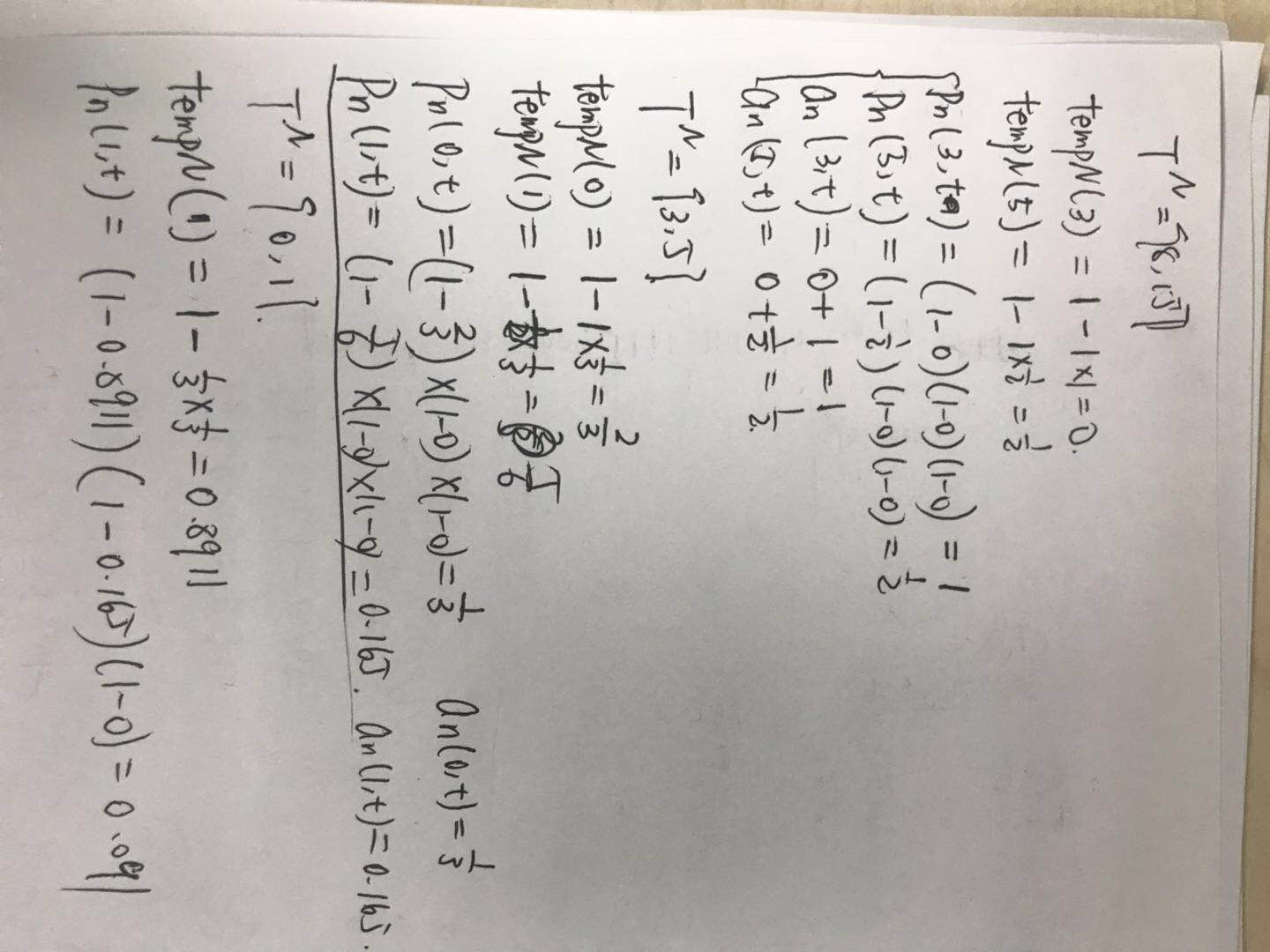
**C LIBM-H 算法**

我们提出一种动态规划的方法去计算MIIA，在一个负向种子集合一个正向种子集中。

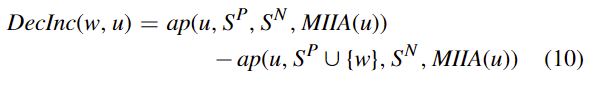
首先我们假设两个种子集是固定的。Pn(v,t) pp(v,t)代表v在t被影响到的概率。







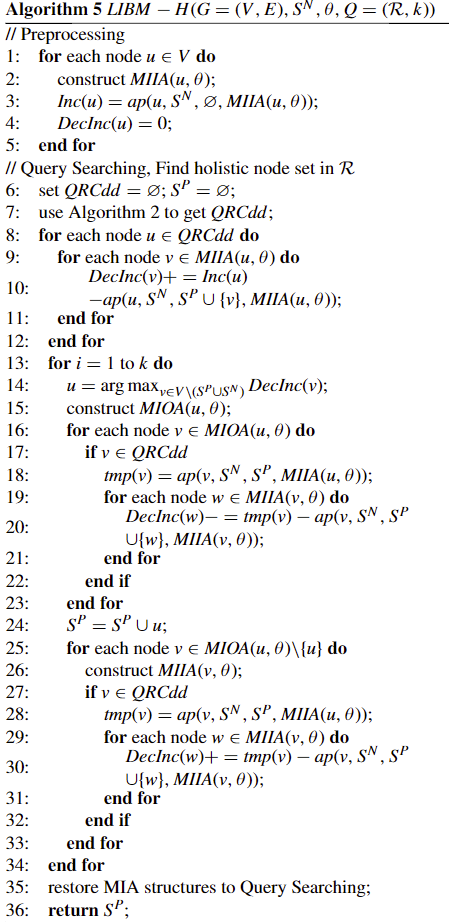
因此，我们使用了DecInc ( w；u)表示在将w加到正种子集Sp之后，节点w到节点u的阻塞负影响。



然后我们用DecInc（w,R）来代表在区域R中的总共的阻塞负影响。就是将节点w加入正向种子集Sp之后w在区域R中的阻塞负影响



随后设计了LIBM-H算法用来选择正向种子



该算法由预处理和查询搜索两个步骤组成。在**预处理步骤**中，由于影响只在MIA结构中传播，我们构造MIIA ( u； )，用于网络中的每个节点u (线路2 )。为了节省查询搜索的时间，我们基于算法4预先计算网络中每个节点u的负激活概率。我们使用一个列表来初始化网络中每个节点u的DecInc(u )，并设置DecInc(u) =0 (第4行)。在**查询**步骤中，我们用QRCdd作为节点集合表示节点在R中，并用算法2计算QRCdd的值。随后基于MIA和QRCdd计算出候选节点。可以看出候选节点就是在R中有阻塞负向影响的节点。每次迭代中都选择阻塞负向影响最大的节点

**6.LIBM-C**

在LIBM-H中我们要计算所有候选种子几点，但是很多候选节点不重要不需要计算。所以我们设计算法对那些不重要的进行剪枝操作。为了实现这一目标，我们提出了两种索引来高效存储节点的阻塞负影响，然后设计了一种基于四叉树单元的更有效的算法LIBM-C。

**A indexes**

Cell index: 给一个四叉树单元格Ci，代表节点u对Ci中节点的阻塞负影响之和，按降序排列。

Node index:对于在网络中的节点u，代表了u在单元格Ci中的阻塞负影响。

B 基于单元格的算法

给定一个查询Q=(R,k)，我们从四叉树的根节点开始遍历，并且识别每个单元是否包含区域R，如果包含，则将它们用表示，如果没有用C0表示。

然后对Cq中的每个集合计算单元指针L，最后将它们合并起来作为节点u对Cq的阻塞负影响

然后对于C0，计算

对于每一个候选节点u，我们计算他在R中的阻塞负影响为



这个算法与LIBM-H不同，它不会计算所有候选种子节点，它利用了一个候选链表和一个上线来估计topk个正向种子并且将好多不重要的候选种子删除。

C LIBM-C算法

分为两步：预处理和查询搜索。

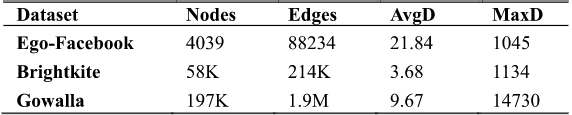
**预处理**：我们构建了用于构建cell索引L和节点索引N的MIA结构，随后我们对于每个几点u计算N（u）并且对每个四叉树单元计算Lci。

**搜索查询**：给定给一个查询Q=（R,k），首先计算那些被R覆盖的四叉树单元的单元列表Lc1。。。然后计算没有被R全部覆盖的四叉树单元列表Lc0。在迭代中，我们使用Bmax去估计还没有访问的候选节点的阻塞负影响的上限，随后我们用Umax表示已访问的最大阻塞负影响。当选择节点时候，我们从每个候选列表中最高的将它们加入到H（maxheap）中，将Umax大于Bmax的才加入。如果u是一个种子，将u从H中弹出，并且加入到Sp中，每次选择一个u之后都要把MIA结构升级，将那些能被u影响到的节点变为1。最后，如果我们找到了k个正向种子或者在候选列表李没有了，就停止。

**7实验**

**A 实验建立**

1.数据集：我们使用三个真实世界的数据集——脸谱网、布莱特凯特和戈瓦拉进行实验。



2.传播概率：在带权重的级联模型下，我们的传播概率为入度分之一，第二个是trivalency模型，我们从{0.1,0.01,0.001}中选择概率

3.参数：为了评估算法的有效性和效率，我们选择200个具有最高度数的节点作为负种子。对于正种子集，大小k从0到200变化，默认值为200。我们生成三种不同区域大小的查询。( 1 )小区域查询:查询区域包含800到1000个节点。( 2 )中等区域查询:查询区域包含4000到5000个节点。( 3 )大区域查询:查询区域包含8000到10000个节点

4.评估算法和指标

LIBN-H, LIBM-C

Greedy

Degree

Random

Proximity

我们根据时间来评估效率，并使用两个关键指标来评估这些算法的有效性。

负激活节点:对于查询请求Q ( R；k ) 在R中的种子集被负激活

阻塞负影响：对于一个查询Q ( R；k )，在R中的种子集合当正向种子集为空时被激活，当正向种子集不为空是不被激活。

5.索引大小和预处理时间

于空间有限，我们只报告戈瓦拉数据集的索引大小和预处理时间，所有被评估的算法都基于四叉树索引，LIBM-H和LIBM-C使用MIA计算阻断的负影响，LIBM-C使用单元索引L和结点索引N