多社交网络的影响力最大化分析

背景

社交网络不断普及，各种社交平台涌现，用户开始分布在多个社交网络上，这种变化将直接影响基于病毒营销的应用。单一社交网络推广产品可能满足不了需求，为了获取更大的利润，需要在多社交网络寻找种子用户最大化传播范围。

解决问题

- 解决了现有影响力最大化研究只针对单一网络进行分析，而不是针对多社交网络分析的问题。
- 为多社交网络建立了影响传播模型，并提出了计算最大传播范围的算法。

解决方法

影响传播模型——自传播性

多社交网络中的节点因为存在自传播的性质所以导致了多社交网络之间的连接。

给定n个网络$G_1(V_1,E_1),G_2(V_2,E_2),\dots,G_n(V_n,E_n)$，对于每个网络$G_i(V_i,E_i)$，$V_i$表示该网络节点的集合，$E_i$表示该网络边集合，$ET_i$表示该网络实体集合。不同网络中不同的节点可能指代同一个实体。

在给定网络G_i和G_j中的实体e_u，分别表示为$u(e_u,G_i)$和$u(e_u,G_j)$，如果实体e_u将信息从网络G_i传到了G_j，这种行为就是自传播。

该文在选择关键节点以最大化影响范围时按照节点进行选择，计算影响传播范围时按照实体进行度量。

传播模型采用独立级联模型。问题定义为：给定n个社交网络和一个整数k，多社交网络最大化旨在找到一个包含k个节点的种子集合S，使得对于任何包含k个节点的集合K，都有$\sigma(S)\geq\sigma(K)$。

影响计算模型

节点间的近似影响计算模型

1. Node 2 Node

   考虑节点u沿着路径$P=(u=n_1,n_2,\dots,v=n_m)$对节点$v$的影响传播强度$pp(P)$:

   $$pp(P)=\prod_{i=1}^{m-1}p(n_i,n_{i+1})$$

由于节点间路径有多条，为了减少计算量，将最大传播概率的路径$MPP(u(e_u,G_i),v(e_v,G_j))$来衡量两点之间的传播概率：

$$pp(u(e_u,G_i),v(e_v,G_j)) \approx pp(MPP(u(e_u,G_i),v(e_v,G_j)))$$

2. Node 2 Entity

在多数情况下，需要衡量节点对实体的影响强度，假设n个网络上有m个表明名称指代实体e_v。因此由节点$u(e_u,G_i)$到实体e_v可以得到以节点u为根叶节点为指代实体e_v的节点的多叉树。则u对实体e_v的影响：

$$pp(u(e_u,G_i),e_v) = 1-\prod_{j=1}^{m}(1-pp(u(e_u,G_i),v(e_v,G_j)))$$

基于树的算法模型

根据上述，将节点的的影响传播范围定义为：

$$\sigma(u(e_u,G_i)) = \sum_{e_v \in E} pp(u(e_u,G_i),e_v)$$

其实种子集合的影响范围不能直接是每个节点累加，因为会有节点影响重复现象。所以考虑计算影响范围的边际增益$gain(v|S)$:

$$gain(u|S) = \sigma(S \cup u) - \sigma(S)$$

$\sigma(S)$的计算方式为$\sigma(S) = \sum_{e_v \in E} pp(S,e_v)$，而$pp_(S,e_v)$可以通过下式计算：

$$pp(S,e_v) = 1-\prod_{j=1}^{m}(1-pp(S,v(e_v,G_j)))$$

而计算$pp(S,v(e_v,G_j)$不能通过计算集合S中每个节点对v的影响概率求和，因为会有路径重叠。

为此考虑基于树的算法模型，将节点v视为根，所有能够到达v的起始节点作为树的叶子节点，逆向构建一棵树。从而计算：

$$pp(S,v) = \begin{cases} 1 & if\ v \in S, \\ 1-\prod_{c \in C_v}(1-pp(S,c)p(c,v)) & if\ v \notin S \end{cases}$$

C_v表示v儿子节点的集合。

基于上界的优化算法

1. $gain(u,S,EM_s,NM_s)$

算法 1. $gain(u,S,EM_S,NM_S)$.

输入：候选种子节点 u，种子集合 S，EM_S，NM_S

输出：候选种子节点 u 的影响增益 $gain(u|S)$

```
初始化 EM_{S∪u}=EM_S;
初始化 gain(u|S)=0.0;
FOR(EACH v IN O(u)){
IF(v NOTIN NM_S.keyset()){
  IF(v.entity NOTIN EM_S.keyset()){
  EM_{S∪u}[v.entity]=pp(u,v);
   }
  ELSE{
  EM_{S∪u}[v.entity]=1-(1-EM_S[v.entity])×
                     (1-pp(u,v));
   }
  }
  ELSE{
   npp(u,v)=pp(S⋃u,v);
  nval=1-(1-EM_S[v.entity])×(1-npp(u,v))/
       (1-NM_S[v]);
    EM_{S∪u}[v.entity]=nval;
  }
}
FOR(e IN EM_{S∪u}.keyset()){
  IF(e IN EM_S) THEN{
   gain(u|S)+= EM_{S∪u}[e]-EM_S[e];
  }
  ELSE{
  gain(u|S)+= EM_{S∪u}[e];
  }
}
RETURN gain(u|S);
```

该算法计算了节点u在集合S上的边际增益。用EM_s和NM_s分别表示实体和节点影响概率的缓存。算法表示对于节点u所有每个出邻居v，如果v没有记录在NM_s里：

- v的实体没有记录在EM_s里，那么就更新$EM_{S\cup u}[v.entity] = pp(u,v)$
- v的实体记录在EM_s里，那么更新$EM_s[v.entity] = 1-(1-EM_s[v.entity]) \times (1-pp(u,v))$

如果v记录在NM_s里，先计算$npp(u,v) = pp(S \cup u, v)$，然后计算
$nval = 1-(1-$

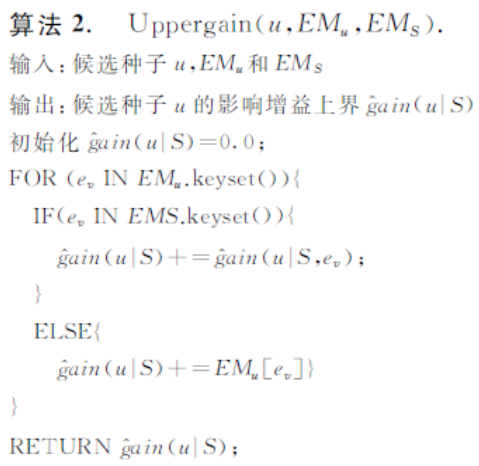
$EM_s(v.entity) \times (1-npp(u,v))/(1-NM_s[v])$，更新
$EM_{S\cup u}[v.entity] = nval$

。$nval$计算中有一个条件概率，即在v未被种子集合影响的条件下其实体也未被影响的概率，而$nval$就表示了种子集合加节点u之后v的实体被影响的概率值。

然后对于所有的实体$e \in EM_{S\cup u}$，更新边际增益$gain(u|S)$:

- e在EM_S中，即e之前已经被S影响，则更新为 $gain(u|S)+= EM_{S\cup u}[e] - EM_S[e]$.
- e不在EM_s中，即e是后来被u影响的，更新边际增益为 $gain(u|S)+= EM_{S\cup u}[e]$.

2. $Uppergain(u, EM_S, EM_S)$



假设节点u和S对节点v的影响独立，那么上述计算$pp(S \cup u, v)$可以表示为$1 - pp(S, v) \times pp(u, v)$。而计算$pp(S, v(e_v, G_i))$:

$$\hat{pp}(S, v) = \begin{cases} 1, & if\ v \in S \\ 1 - \prod_{u \in S}(1 - pp(u, v)), & if\ v \notin S \end{cases}$$

边际增益计算为：
$\hat{g}ain(u|S, e_v) = \hat{p}p(S \cup u, e_v) - pp(S, e_v) = (1 - pp(S, e_v)) \times pp(u.e_v)$

则u在集合S上的影响增益上界$\hat{g}ain(u|S) = \sum_{e_v \in E} \hat{g}ain(u|S, e_v).$

对于每个节点$u \in V$都将其对每个实体的影响概率缓存在EM_u中。算法2计算了影响增益上界。

3. BlendedIMMS

该算法用来寻找$top - k$个影响力范围最大的种子节点。初始根据所有节点初始影响范围制作大顶堆，每次选择大顶堆根元素，通过Uppergain计算节点边际增益上界值，然后重新排序，再次选择顶端节点，通过gain计算边际增益，将节点并入S。最后返回S。

4. BoundBasedIMMS

该算法直接将增益上界作为评估种子节点增益方法。每次选择增益上界最大的节点加入S。

数据集

数据

- DBLP：科研作者网络
- Citeseer： 科研作者网络
- Linkedin：文章
- Aminer：文章

参数设置

- DBLP and Citeseer

  节点间影响概率为$p(a,b)=\frac{N(a,b)}{N(a)}$，$N(a,b)$表示$a$和$b$的合作次数，$N(a)$表示$a$总共的著作数。

- Aminer and Linkedin

  节点影响概率为$p=1/d(v)$，$d(v)$为节点的入度。

数据信息

数据集	点数	边数	平均出度	节点总数	共同点数
DBLP	1 436 596	12 311 706	8.57	1 574 809	73 131
Citeseer	138 368	754 316	5.45		
Aminer	1 056 941	7 859 752	7.44	7 783 231	3041
Linkedin	6 726 290	38 721 380	5.76		

实验分析

通过运行时间和影响范围衡量算法性能。

影响范围

最大传播路径阈值θ设置为0.01，0.02，0.03，0.04和0.05，比较IMMS，BlendedIMMS，BoundBasedIMMS，和PMIA四种方法的影响范围。

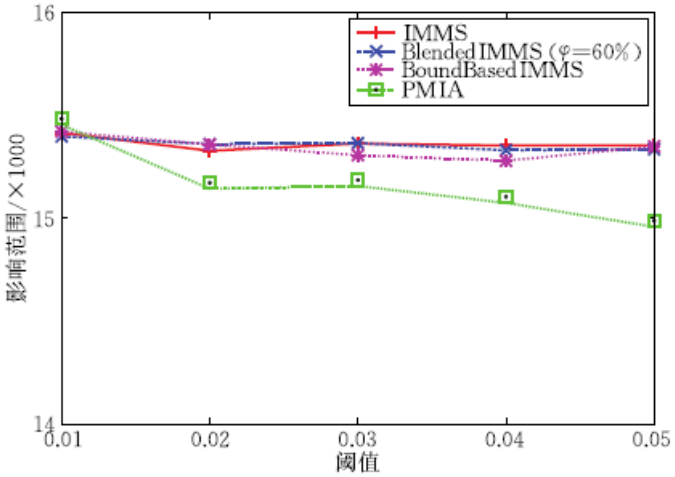


图 3　DBLP&Citeseer 数据集上的影响范围结果
（$k=50$ 时，影响范围与阈值 θ 的关系）

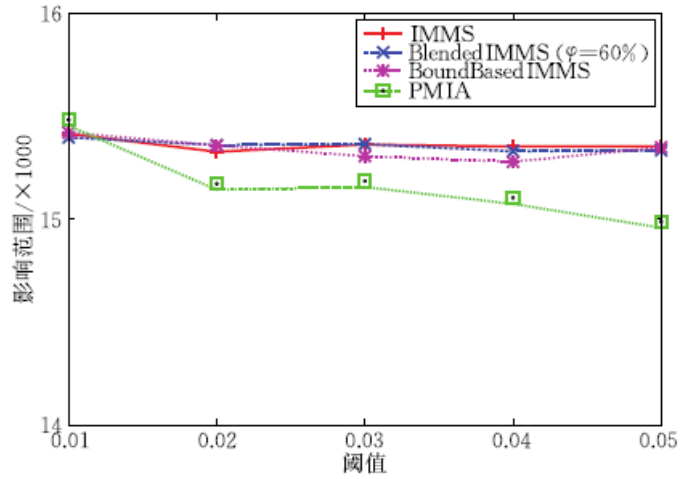


图 3　DBLP&Citeseer 数据集上的影响范围结果
（$k=50$ 时，影响范围与阈值 θ 的关系）

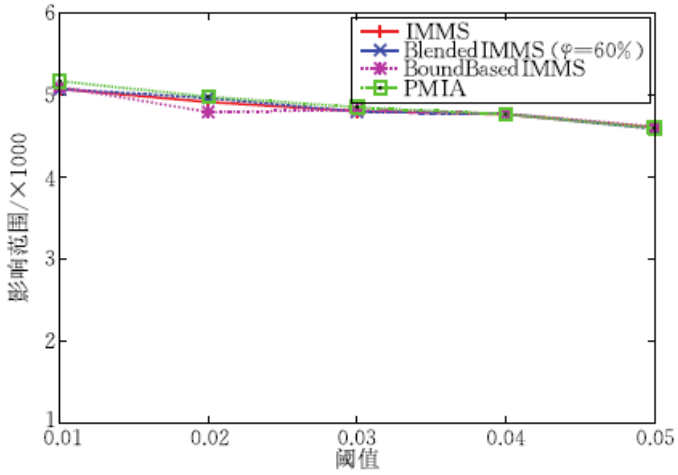


图 5　Aminer&Linkedin 数据集上的影响范围结果
（$k=100$ 时，影响范围与阈值 θ 的关系）

以上三张图表示，随着阈值的增大，算法的影响范围都呈现下降趋势，因为阈值是最大传播路径的阈值，用来控制每个节点的局部传播范围，最大传播路径上概率值小于阈值的路径会被过滤掉。同时可以得出，随着种子数量增加，影响范围也会增加。总体来说，文中方法略优。

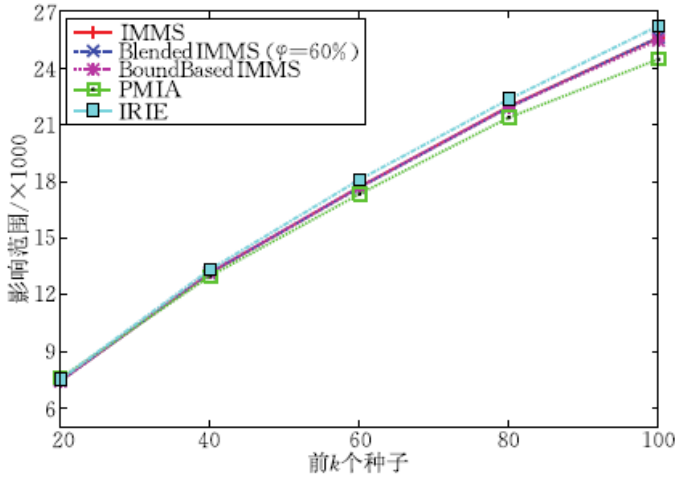
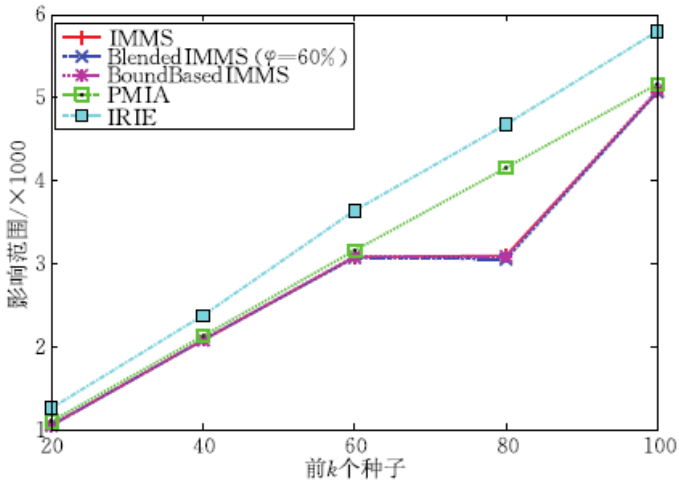


图 6　DBLP&Citeseer 数据集上的影响范围结果
（固定 $\theta=0.01$，种子数目 k 同影响范围的关系）



随着种子节点增加，所有方法影响范围增加具有一致性，在文章数据集上，文中方法表现还是差一些。

运行时间

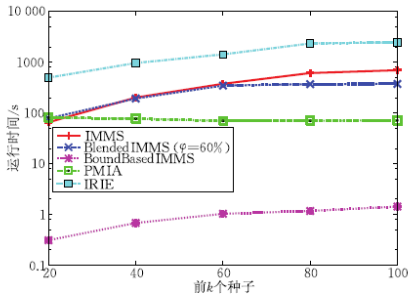


图 11　DBLP&Citeseer 数据集上运行时间结果
（$\theta=0.01$ 时，运行时间与种子数目 k 的关系）

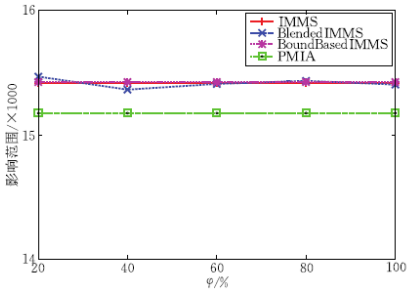


图 13　DBLP&Citeseer 数据集上影响范围结果
（$k=50$，$\theta=0.01$ 时，影响范围与阈值 φ 的关系）

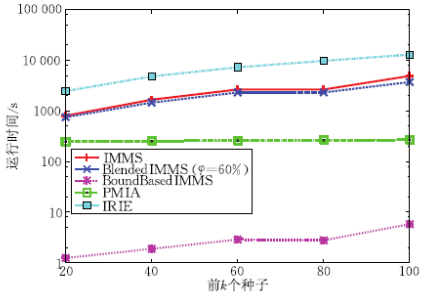


图 12　Aminer&Linkedin 数据集上运行时间结果
（$\theta=0.01$ 时，运行时间与种子数目 k 的关系）

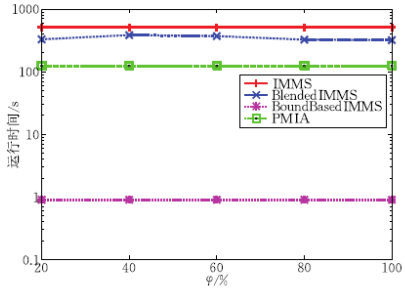


图 14　DBLP&Citeseer 数据集上运行时间结果
（$k=50$，$\theta=0.01$ 时，运行时间与阈值 φ 的关系）

上述四张图片可以看出在两种数据集上，种子数量和阈值作为变量时，各个方法运行时间的快慢。BoundBased IMMS因为只考虑影响增益上界，所有它运行一直最快，IMMS和BoundBased IMMS运行时间都在可接受的范围内。

创新点

- 在已有问题上的创新：提出了新的问题，即在多社交网络下寻找k个种子节点使其影响范围最大化的问题。
- 在已有方法上的创新：提出了基于树的算法模型和基于上界的算法优化来解决提出的问题。