**Scalable Influence Maximization for prevalent viral marketing in large-scale social networks**

利用社会关系原因：

It is based on trust among individuals’ close social circle of families, friends, and co-workers. Research shows that people trust the information obtained from their close social circle far more than the information obtained from general advertisement channels such as TV, newspaper and online advertisements [16].

The problem is whom to select as the initial users so that they eventually influence the largest number of people in the network.

The main idea of our heuristic scheme is to use local arborescence2 structures of each node to approximate the influence propagation. We first compute maximum influence paths (MIP) between every pair of nodes in the network via a Dijkstra shortest-path algorithm, and ignore MIPs with probability smaller than an influence threshold θ, effectively restricting influence to a local region.

Notations

S ⊆ V be a seed set

*S*t *⊆ V* be the set of nodes that are activatedat step *t ≥ 0*

MIPG(u, v) is simply the shortest path from *u* to *v* in the weighted graph G

MIIA(MaximumInfluence In-Arborescence): the union of the maximum influence paths to *v* . It can be done using efficient implementations of Dijkstra’s shortest path algorithm.

MIIA(v, θ) = ∪u∈V;pp(MIPG(u;v))≥θMIPG(u, v)

MIOA(MaximuminfluenceOut-Arborescence): the union of the maximum influence of *v* to other nodes.When *MIIA*(*v, θ*)’s for all node *v* ∈ *V* are available, *MIOA*(*v, θ*)’s can be derived from *MIIA*(*v, θ*)’s, therefore no extra running time for *MIOA*(*v, θ*)’sis needed.

MIOA(v, θ) = ∪u∈V;pp(MIPG(v;u))≥θMIPG(v, u)

NOTE: MIIA(v, θ) and MIOA(v, θ) give the local influence regions of v, and different values of θ controls the size of these local influence regions.

pp(u, v) ：Propagation probability of the edge, which is the probability  
that *v* is activated by *u* through the edge in the next step after *u* is  
activated. *u*激活*v*的概率。

Note that for each edge (u, v) in the graph, if we translate the propagation probability pp(u, v) to a distance weight- log pp(u, v) on the edge, then MIPG(u, v) is simply the shortest path from u to v in the weighted graph G.

σI(S) ：The **influence spread** of *S*, which is the expected number of activated nodes given seed set *S.* In[10], We define the influence of a set of nodes A, denoted *σ*(*A*), to be **the expected number of active nodes at the end of the process**, given that A is this initial active set *A*0.

*N*in(*u*, MIIA(*v, θ*)) ： be the set of in-neighbors of *u* in MIIA(*v, θ*)

MIIA(*v, θ*)节点集合中 节点u 的入邻居

*ap*(u, S, *MIIA*(v, θ)) : the activation probability of any node *u* in

MIIA(*v, θ*) 集合MIIA(*v, θ*)中 *u*节点被 种子集合中节点 激活的概率。

σ*M*(*S*): the influence spread of *S* in our MIA model

σM(*S*)= ∑v∈Vap(v, S, MIIA(v, θ))

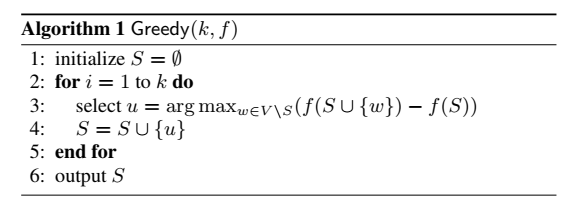
*IncInf* (*v*)：the incremental influence spread

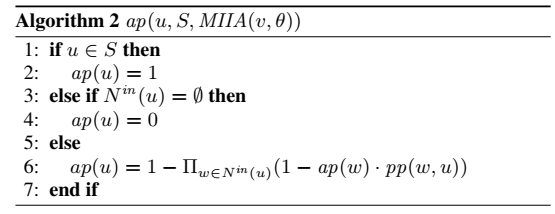
α(v, w): linear coefficient. Since ap(w) and ap(v) have a linear relationship with the linear coefficient α(v, w), the incremental influence of w on v is given by α(v, w) · (1 - ap(w)). w对v的影响系数。如果选择w作为下一次种子节点，ap(w)=1.

α(v（v是考查节点）, w（MIIA(*v, θ*)求得的集合中的节点)）

注意：

For every node v ∈ V, our algorithm stores MIIA(v, θ), MIOA(v, θ), and for every u ∈ MIIA(v, θ), (u, S, MIIA(v, θ)) and α(v, u) are stored (note that ap(u, S, MIIA(v, θ)) can reuse the same entry for different seed set S). We also use a max-heap （最大值堆）to store and update IncInf (v) for all v ∈ V.





*算法2*

3：MIIA(*v, θ*) 中的每一个节点*u* ，如果（指向u的节点是空集）且（u不属于种子节点），则*ap(u)*=0,即*u* 被 种子节点 激活的概率是0。

如图1所示，MIIA(C*,*0*.*07)={C*,*A*,*L}，其中u=L，指向L的节点是空集，L被种子节点激活的概率为0。

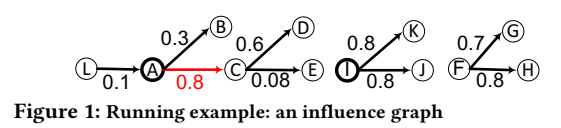
6：MIIA(*v, θ*) 中的每一个节点*u* ，如果（指向*u*的节点不是空集，有节点w指向u）且（u不属于种子节点），则*ap(u)*=0，即*u* 被种子节点激活的概率是

种子节点 （通过w）激活*u*的概率=（种子节点 *激活* 节点*w* 的概率）\*（节点*w* *再激活 u*的概率）

种子节点 激活u**失败**的概率=1-种子节点 激活*u*的概率

种子节点激活u的**成功**概率 =1- 种子节点 激活u**失败**的概率

举例：



对图1中每一个节点计算MIIA MIOA: 给定*θ*=0.07，其中*pp*(*S*, *u*) ：节点*u*被{}中节点激活的概率

如图1所示，若计算节点C的激活概率

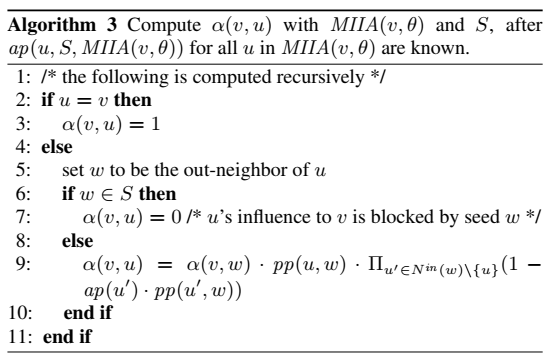
*pp*({C}, C)= 1>*θ*  , pp({A},C) = 0.8>*θ*, and pp({L},C) = 0.8 × 0.1 = 0.08 < *θ*

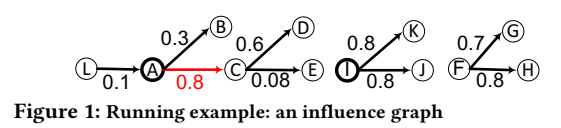
因此，MIIA(C,0.07) ={C,A,L}. 同样的, MIOA( C,0.07) = {C,D,E}.

已知MIIA(C,0.07) = {C,A,L} （见p？），A为种子节点，根据算法2求得

ap（A ）=ap（A, S, MIIA(C, 0.07)）=1, ap（L）= ap（L, S, MIIA(C, 0.07)）=0,

ap（C）=ap（C, S, MIIA(C, 0.07)）=1-1-ap（A）\*PP(A，C)）=1-(1-1\*0.8) = 0.8





对图中每一个节点计算MIIA MIOA: 给定*θ*=0.07，其中*pp*(*S*, *u*) ：节点*u*被{}中节点激活的概率

如图1所示，若计算节点C的激活概率

*pp*({C}, C)= 1>*θ*  , pp({A},C) = 0.8>*θ*, and pp({L},C) = 0.8 × 0.1 = 0.08 < *θ*

因此，MIIA(C,0.07) ={C,A,L}. 同样的, MIOA( C,0.07) = {C,D,E}.

已知MIIA(C,0.07) = {C,A,L} （见p？），A为种子节点，根据**算法2**求得

ap（A ）=ap（A, S, MIIA(C, 0.07)）=1, ap（L）= ap（L, S, MIIA(C, 0.07)）=0,

ap（C）=ap（C, S, MIIA(C, 0.07)）=1-1-ap（A）\*PP(A, C)）=1-(1-1\*0.8) = 0.8

v=C, u= MIIA(C,0.07) ={C,A, L}.

根据**算法3**求得

α(v, u):

u=C ,v=C

α(C, C)=1

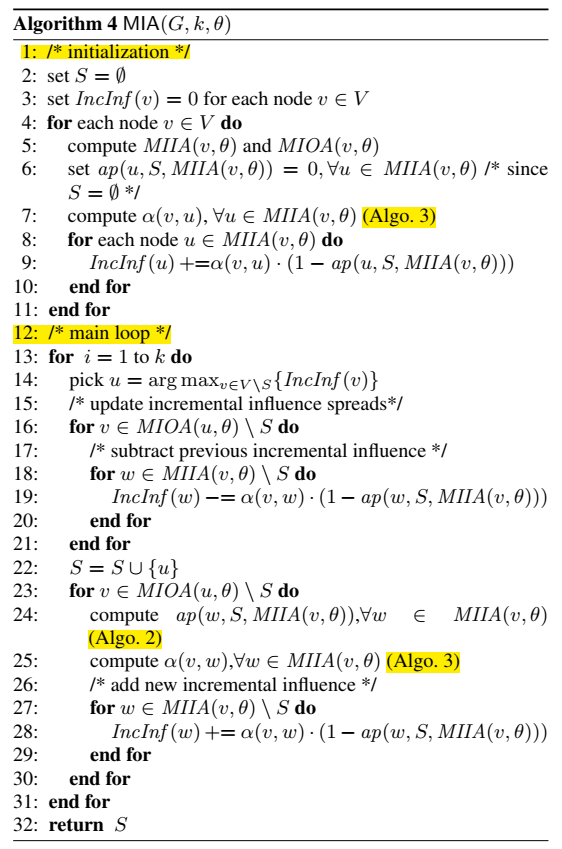
u=L, w=A,

α(C, L)=0

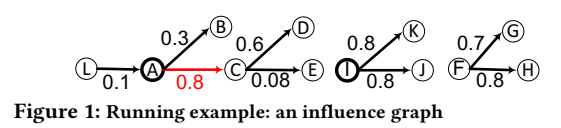
*根据 α*(*v, u*) = *α*(*v, w*) *· pp*(*u, w*) *·* Π*u′∈Nin* (*w*)*\{u}*(1 *-ap*(*u′*) *· pp*(*u′, w*)) , 其中*N*in(*L*, MIIA(*v, θ*)) ={A} (MIIA(*C, θ*)节点集合中 节点L 的入邻居)

*Nin* (*w*)={A}

u=A, v=C, w=L, α(C, A) = α(C, L)\* PP(A, L)）\*（?）=前面两项已经为0，所以为0



举例说明



**1-5：**

对图中每一个节点计算MIIA MIOA: 给定*θ*=0.07，其中*pp*(*S*, *u*) ：节点*u*被{}中节点激活的概率

如图1所示，若计算节点C的激活概率

*pp*({C}, C)= 1>*θ*  , pp({A},C) = 0.8>*θ*, and pp({L},C) = 0.8 × 0.1 = 0.08 < *θ*

因此，MIIA(C,0.07) ={C,A,L}. 同样的, MIOA( C,0.07) = {C,D,E}.

**6：**

当种子集合为空集时，ap*(u, S,* MIIA*(v, θ)) = 0,*

**7：**

compute α(v, u), ∀u ∈ MIIA(v, θ)

见前

**8-9：**

u={C, A, L}

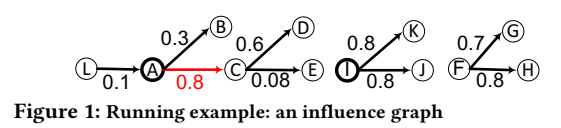
*IncInf* (*u*) +=*α*(*v, u*) · (1 - *ap*(*u, S, MIIA*(*v, θ*)))

*IncInf* (C) +=*α*(*C, C*) · (1 - *ap*(*C, S, MIIA*(*C, θ*)))= 1- ap（C）=0.2

*IncInf* (A) +=*α*(*C, A*) · (1 - *ap*(*u, S, MIIA*(*C, θ*)))=0

*IncInf* (L) +=*α*(*C, L*) · (1 - *ap*(*u, S, MIIA*(*C, θ*)))=0

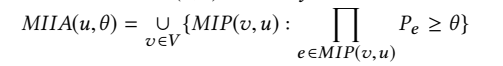
李可完整例子：



按照算法4：

1. 初始化
2. 设定*S* = *∅*
3. 设定*IncInf* (*v*) = 0 ，即*IncInf* (*L*) =0, *IncInf* (*A*) =0 , *IncInf* (*B*) =0 , *IncInf* (*C*) =0 , *IncInf* (*D*) =0 , *IncInf* (*E*) =0;
4. 假定*θ=0.07*
5. 计算*MIIA*(*v, θ*) 、 *MIOA*(*v, θ*)





*MIP*(?, *L*) =*∅*, *MIIA*(*L, θ*)= *∅* ;

*MIP*(*L, A*) ={*L,A*} (0.1>0.07), *MIIA*(*A, θ*)={*A*, *L*};

*MIP*(*L,B*) ={*L,A,B*} (0.03<0.07), *MIP*(*A,B*)={*A,B*}(0.3>0.07), *MIIA*(*B, θ*) ={*A,B*} ;

*MIP*(*L,C*) = {*L,A,C*}(0.08>0.07), *MIP*(*A,C*) = {*A,C*}(0. 8>0.07), *MIIA*(*C, θ*) ={*A,C*};

*MIP*(*L,D*) ={*L,A,C*,*D*}(0.048<0.07), *MIP*(*A,D*) ={ *A,C*, *D*}(0.48>0.07), *MIP*(*C,D*) ={*C,D*}(0.6>0.07), *MIIA*(*D, θ*) ={*A*,*C,D*};

*MIP*(*L,E*) ={ *L,A,C*, *E*}(0.0064<0.07) , *MIP*(*A,E*) ={ *A,C*, *E* }(0.064<0.07), *MIP*(*C,E*) ={*C,E*}(0.08>0.07), *MIIA*(*E, θ*) ={*C,E*};



*MIP*(*L, A*) ={ *L,A* }(0.1> *θ=0.07*), *MIP*(*L,B*) ={*L,A*,*B*}(0.03<*θ=0.07)*, *MIP*(*L,C*) ={*L, A* ,*C*}(0.08>*θ=0.07*)*, MIP*(*L,D*) ={ *L, A*,*C,D*}(0.048<*θ=0.07*)*, MIP*(*L,E*) ={ *L, A* ,*C*,*E*}(0.0064<*θ=0.07*)*, MIOA*(*L, θ)*={ *L, A, C* } ;

*MIP*(*A,B*) ={*A,B*}(0.3>*θ=0.07*), *MIP*(*A,C*) ={*A,C*}(0.8>*θ=0.07, MIP*(*A,D*) ={*A,C,D*}(0.48>*θ=0.07*)*, MIP*(*A,E*) ={*A,C,E*}(0.064<*θ=0.07*)*,MIOA*(*A, θ)*={*A,B,C,D*};

*MIP*(*B,?*)= *∅*，*MIOA*(*B, θ)*=*∅* ;

*MIP*(*D,?*)= *∅*，*MIOA*(*D, θ)*=*∅* ;

*MIP*(*E,?*)= *∅*，*MIOA*(*E, θ)*=*∅* ;

1. *ap*(u, S, *MIIA*(v, θ)) = 0, ∀u ∈ *MIIA*(v, θ)

按照步骤5结果，且当前*S* = ∅需要设定

*MIIA*(*L, θ*) )= *∅* ;

*MIIA*(*A, θ*)={*A*, *L*}，

*ap*(L )=*ap*(L, S, *MIIA*(A, θ)) =0,

*ap*(A *)=ap*(A, S, *MIIA*(A, θ)) = 1-(1-ap(L)\*pp(L,A))=1-(1-0)=0;

*MIIA*(*B, θ*) ={*A,B*},

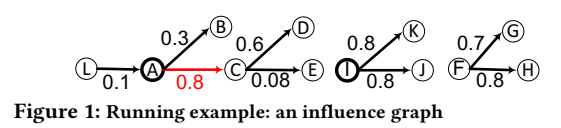
ap(L, S, *MIIA*(B, θ))=0

*ap*(A, S, *MIIA*(B, θ)) = 1-(1-ap(L, S, *MIIA*(B, θ))\*pp(L,A))=0, *ap*(B, S, *MIIA*(B, θ)) = 1-(1-ap(A)\*pp(A,B))=0;

*MIIA*(*C, θ*) ={*A,C*}, *ap*(C, S, *MIIA*(C, θ)) = 0, *ap*(A, S, *MIIA*(C, θ)) = 0;

*MIIA*(*D, θ*) ={*A*,*C,D*}, *ap*(A, S, *MIIA*(D, θ)) = 0, *ap*(C, S, *MIIA*(D, θ)) = 0, *ap*(D, S, *MIIA*(D, θ)) = 0;

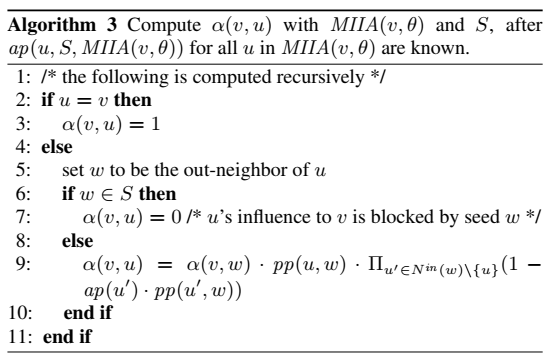
*MIIA*(*E, θ*) ={*C,E*}, *ap*(E, S, *MIIA*(E, θ)) = 0, *ap*(C, S, *MIIA*(E, θ)) = 0.



1 :if u ∈ S then2: ap(u) = 1  
3: else if *N*in(*u*) = ∅ (节点集合中 节点u 的入邻居)*then*4: ap(*u*) = 05: *else*6: *ap*(*u*) = 1 - Π*w*∈*N*in *(u)*(1 - ap(*w*) · pp(*w, u*))

|  |
| --- |
| 第二次执行该步骤需要先按照算法2计算*ap*(u, S, *MIIA*(v, θ))  *ap*(u, S, *MIIA*(v, θ)) : the activation probability of any node *u* in  MIIA(*v, θ*) 集合MIIA(v, θ)中 *u*节点被 种子集合中节点 激活的概率  若是第一次执行算法，种子集合为空，*ap*(u)=0；    *若是第二次执行算法2, S={？}*  *MIIA*(*L, θ*) )= *∅* ;  *MIIA*(*A, θ*)={*A*, *L*}，*ap*(A, S, *MIIA*(A, θ)) = 0, *ap*(L, S, *MIIA*(A, θ)) = 0;  *MIIA*(*B, θ*) ={*A,B*}, *ap*(B, S, *MIIA*(B, θ)) = 0, *ap*(A, S, *MIIA*(B, θ)) = 0;  *MIIA*(*C, θ*) ={*A,C*}, *ap*(C, S, *MIIA*(C, θ)) = 0, *ap*(A, S, *MIIA*(C, θ)) = 0;  *MIIA*(*D, θ*) ={*A*,*C,D*}, *ap*(A, S, *MIIA*(D, θ)) = 0, *ap*(C, S, *MIIA*(D, θ)) = 0, *ap*(D, S, *MIIA*(D, θ)) = 0;  *MIIA*(*E, θ*) ={*C,E*}, |

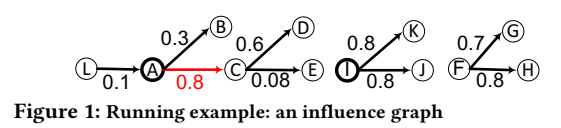
算法3：计算*α*(*v, u*)*,* ∀*u* ∈ *MIIA*(*v, θ*)，



*其中：*

*u*到 *v， w*到*v u*激活*w*

*α*(*v, u*)= α(v, w) · pp(u, w) · Π u′∈Nin (w)\{u}(1 -ap(u′) · pp(u′, w))



*MIIA*(*A, θ*)={*A*, *L*},

*v=A, u=A, α*(*v, u*)=*α*(*A, A*)=1;

*v=A, u=L, w=A*是*u=L*的出邻居节点*，α*(*v, u*)=*α*(*A, L) =？ 第二次计算才可算出*;

*MIIA*(*B, θ*) ={*A,B*},

*v=B, u=B, α*(*v, u*)=*α*(*B, B*)=1;

*v=B, u=A, w=A*是*u=L*的出邻居节点*，α*(*v, u*)=*α*(*A, L*) *第二次计算才可算出*;

*MIIA*(*C, θ*) ={*A,C*},

*v=C, u=C, α*(*v, u*)=*α*(*C, C*)=1;

*v=C, u=A, w=C*是*u=A*的出邻居节点*，α*(*v, u*)=*α*(C, *A*) *第二次计算才可算出*;

*MIIA*(*D, θ*) ={*A*,*C,D*},

*v=D, u=D, α*(*v, u*)=*α*(*D, D*)=1;

*v=D, u=C, w=D*是*u=C*的出邻居节点*，α*(*v, u*)=*α*(*C, D*);

*v=D, u=A, w=C*是*u=A*的出邻居节点*，*u′=A是节点C的入节点但又不是节点u=A，故 *α*(*v, u*)=*α*(*A, D*)= α(D, C) · pp(A, C)· (1- ap(u′) · pp(u′, w))= *第二次计算才可算出*;

*参考：u*到 *v， w*到*v , u*激活*w α*(*v, u*) = α(v, w) · pp(u, w) · Π u′∈Nin (w)\{u}(1 -ap(u′) · pp(u′, w))

*MIIA*(*E, θ*) ={*C,E*},

*v=E, u=E, α*(*v, u*)=*α*(*E, E*)=1;

*v=E, u=C, w=E*是*u=C*的出邻居节点*，α*(*v, u*)=*α*(E*, C*) *第二次计算才可算出*;

*N*in(*u*, MIIA(*v, θ*)) ： be the set of in-neighbors of *u* in MIIA(*v, θ*)

MIIA(*v, θ*)节点集合中 节点u 的入邻居

*实验：*

*对比方法*

*数据集*

*测试指标*