Influence Maximization Meets Efficiency and Effectiveness: A Hop-Based Approach

背景

信息以说词的方式通过在线社交网络广泛快速的传播，病毒营销就是一个典型的应用。大量的工作聚焦于病毒营销的影响最大化问题，即选取初始化种子节点集合并使其尽可能的广泛传播，它基于两个传播模型：独立级联和线性阈值。

随后有研究者提出简单贪心爬山算法来解决影响最大化问题，跟进的研究都聚集于解决贪心算法在大型网络上的效率问题：基于蒙特卡洛模拟的算法和一些启发式算法。

基于蒙特卡洛模拟的方法有代表性的是基于抽样的RIS方法，然而这会消耗大量时间和内存来获得样本和存储样本。因此面临效率问题。

启发式算法利用相关特征和提取子图等方式使影响传播计算变得容易，但是这些方法没有理论保证，因此面临有效性问题。

解决的具体问题

解决了在影响最大化问题方面一些方法面临的效率和有效性问题。

解决方法

One Hop of Propagation

算法思想

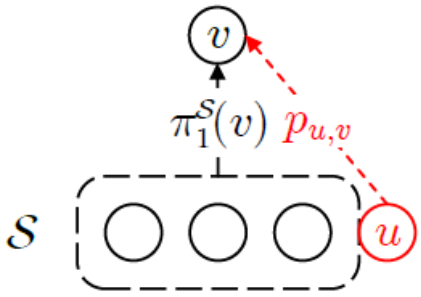
One-Hop表示种子集合S直接影响并以一定概率激活它的出邻居。显然，对于所有节点$v \in S$，它被种子节点激活的概率为$\pi_1^S(v) = 1$。

假设非种子节点$v \notin S$，v可以直接被它在种子集合中的逆邻居I_v激活，而且每个逆邻居激活v都是独立的。

可以得到以上两种情况v被激活的概率：

$$\pi_1^S(v) = \begin{cases} 1, & \text{if } v \in S \\ 1 - \prod_{w \in I_v \cap S}(1 - p_{w,v}), & \text{if } v \notin S \end{cases}$$

当新的种子节点u加入S时，种子集合变成了$S \cup \{u\}$，如图：



此时考虑概率1-$\pi_1^{S\cup\{u\}}$由两部分乘积计算得到，即通过S没有激活v概率和通过u没有激活v概率乘积：

$$\pi_1^{S\cup\{u\}} = 1 - \prod_{w\in(I_v\cap S)\cup\{u\}}(1-p_{w,v}) = 1-(1-\pi_1^S(v))(1-p_{u,v})$$

影响传播可以由下面算法得到：

Algorithm 2: $OneHopIncrement(\mathcal{G},\mathcal{S},u)$

$\pi_1^{\mathcal{S}\cup\{u\}}(u) \leftarrow 1$;
for *each node* $v \in \mathcal{N}_u \setminus \mathcal{S}$ **do**
  $\pi_1^{\mathcal{S}\cup\{u\}}(v) \leftarrow 1-\left(1-\pi_1^{\mathcal{S}}(v)\right)\cdot(1-p_{u,v})$;
return $\sum_{v\in\{u\}\cup(\mathcal{N}_u\setminus\mathcal{S})}\left(\pi_1^{\mathcal{S}\cup\{u\}}(v)-\pi_1^{\mathcal{S}}(v)\right)$;

时间复杂度和空间复杂度

上述算法的时间复杂度为$O(1+|N_u|)$，因此对于所有节点$u\in V$，其时间复杂度为$O(\sum_{u\in V}(1+|N_u|))=O(|V|+|\mathcal{E}|)$，所以总的时间复杂度是$O(k(|V|+|\mathcal{E}|))$。

除了存储图的空间，One-Hop用了$O(|V|)$的空间存储概率$\pi_1^S, v\in V$。

Two Hops of Propagation

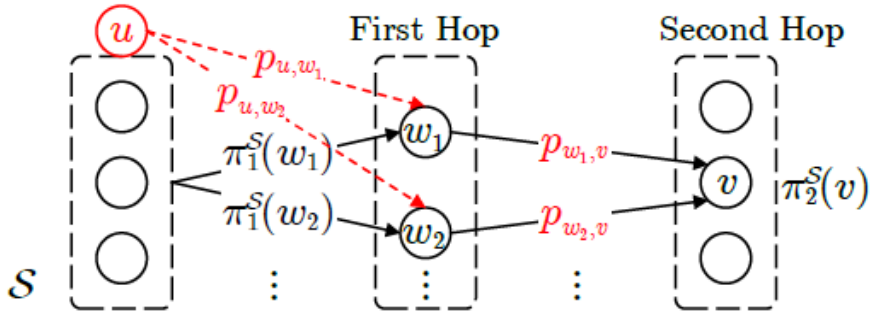
算法思想

Two-Hops表示节点v被经过One-Hop激活的逆邻居节点影响激活，即一个非种子节点v可以直接被种子节点u_i或者间接通过u_i的邻居节点w_j激活。v被u_i激活的概率为$p_{u_i,v}\cdot\pi_1^S(u_i), since\ \pi_1^S(u_i)=1$，而间接激活概率为$p_{w_j,v}\cdot\pi_1^S(w_j)$。

因此，这种情况下，对于任意$v\in V$，激活概率为：

$$\pi_2^S(v)=\begin{cases}1, & if\ v\in S\\ 1-\prod_{w\in I_v}(1-p_{w,v}\cdot\pi_1^S(w)) & \end{cases}$$

当新的种子节点u加入S时，仅在u的two hops内的节点被影响，将这些节点表示为$N_u^2=N_u\cup(\bigcup_{w\in N_u}N_w)\setminus\{u\}$。如图：



假设$A = \{S\text{未直接和间接激活}v\}$，$B = \{S\text{加入}u\text{之后为直接和间接激活}v\}$，则：

$$\pi_2^{S\cup\{u\}} = 1 - p(B|A) = 1 - \frac{p(A\cap B)}{p(A)} = 1 - (1-\pi_2^S(v)) \prod_{w\in(M_{u,v}\cup\{u\})} \frac{1-p_{w,v}\cdot\pi_1^{S\cup\{u\}}(w)}{1-p_{w,v}\cdot\pi_1^S(w)}$$

$$M_{u,v} = \{w : (u,w)\in\varepsilon \; and \; (w,v)\in\varepsilon\}$$

影响传播可以由下面算法得到：

Algorithm 3: $TwoHopsIncrement(\mathcal{G},\mathcal{S},u)$

$\pi_1^{\mathcal{S}\cup\{u\}}(u) \leftarrow 1;$
$\pi_2^{\mathcal{S}\cup\{u\}}(u) \leftarrow 1;$
for *each node* $v \in \mathcal{N}_u^2 \setminus \mathcal{S}$ **do**
  $\pi_2^{\mathcal{S}\cup\{u\}}(v) \leftarrow \pi_2^{\mathcal{S}}(v);$
for *each node* $w \in \mathcal{N}_u \setminus \mathcal{S}$ **do**
  $\pi_1^{\mathcal{S}\cup\{u\}}(w) \leftarrow 1 - \big(1-\pi_1^{\mathcal{S}}(w)\big)\cdot(1-p_{u,w});$
  $\pi_2^{\mathcal{S}\cup\{u\}}(w) \leftarrow 1 - \big(1-\pi_2^{\mathcal{S}\cup\{u\}}(w)\big)\cdot\frac{1-p_{u,w}\cdot\pi_1^{\mathcal{S}\cup\{u\}}(u)}{1-p_{u,w}\cdot\pi_1^{\mathcal{S}}(u)};$
  for *each node* $v \in \mathcal{N}_w \setminus \mathcal{S}$ **do**
    $\pi_2^{\mathcal{S}\cup\{u\}}(v) \leftarrow 1 - \big(1-\pi_2^{\mathcal{S}\cup\{u\}}(v)\big)\cdot\frac{1-p_{w,v}\cdot\pi_1^{\mathcal{S}\cup\{u\}}(w)}{1-p_{w,v}\cdot\pi_1^{\mathcal{S}}(w)};$
return $\sum_{v\in\{u\}\cup(\mathcal{N}_u^2\setminus\mathcal{S})}\big(\pi_2^{\mathcal{S}\cup\{u\}}(v) - \pi_2^{\mathcal{S}}(v)\big);$

时间复杂度和空间复杂度

上述算法的时间复杂度是$O(1+|\mathcal{N}_u| + \sum_{w\in N_u}|\mathcal{N}_w|)$，因此对于$u\in V$，时间复杂度是$O(\sum_{u\in V}(1+|\mathcal{N}_u| + \sum_{w\in N_u}|\mathcal{N}_w|)) = O(|V| + |\mathcal{E}| + \sum_{w\in V}(|\mathcal{I}_w|\cdot|\mathcal{N}_w|))$，所以总的时间复杂度是$O(k(|V| + |\mathcal{E}| + \sum_{w\in V}(|\mathcal{I}_w|\cdot|\mathcal{N}_w|)))$。

Two-Hop除了存储图的空间外，需要$O(|\mathcal{V}|)$的空间去存储激活概率$\pi_1^S(v)\; and \; \pi_2^S(v), v\in V$。

Upper Bound

基于CELF的思想，作者得到了Hop-Based算法的一个上界。

$$\sigma_h(\{v\}) \le 1 + \sum_{w\in\mathcal{N}_u}(p_{v,w}\cdot\sigma_{h-1}(\{w\})$$

$h=1$时，上界$\hat{\sigma}_1(\{v\})=1+\sum_{w\in\mathcal{N}_u}p_{v,w}$，这也是$\sigma_1(\{v\})$的准确值。因此计算这个上界时间复杂度同One-Hop算法。而继续计算$\hat{\sigma}_2(\{v\})$由上述公式，可以将时间复杂度缩短到One-Hop算法的复杂度。这样大大提升了计算效率。

Theoretical Analysis

对于$h\geq 1$，作者表示h-hops影响传播是单调且满足子模性的。如果$\sigma_h(S)/\sigma(S)\geq\alpha,\forall|S|=k$，那么通过结合Hop-Based的贪心算法返回的解S_h满足：

$$\sigma(S_h)\geq((1-1/e)\alpha)\cdot\sigma(S^*)$$

因为$h\geq 1$，One-Hop的激活节点期望值为$\sigma_h(S)$的下界，作者通过度的幂律分布函数得出了$\sigma(S)$的上界，因此可以得到$\sigma_h(S)/\sigma(S)$的一个下界α。
$p_{u.v}=p\ for\ (u,v)\in\mathcal{E}$，任意种子集合$S$，任意$h\geq 1$：

$$\frac{\mathbb{E}[\sigma_h(S)]}{\mathbb{E}[\sigma(S)]}\geq\frac{1-(1-k/|\mathcal{V}|)(1-pk/|\mathcal{V}|)}{1-(1-k/|\mathcal{V}|)P_0(1)(1-pA)}$$
$$A=1-(1-k/|\mathcal{V}|)P_1(1),P_1(d)=\frac{d^{1-\gamma}}{\sum_{d_i=1}^{\infty}d_i^{1-\gamma}}$$

Extension to Linear Threshold Model

- One-Hop

  $$\pi_1^S(v)=\begin{cases}1 & v\in S\\ \sum_{u\in\mathcal{I}_v\cap S}b_{u,v} & otherwise\end{cases}$$

  $\pi_1^{S\cup\{u\}}(v)-\pi_1^S(v)=b_{u,v}$，因此总的影响传播增益为:

  $$\sigma_1(S\cup\{u\})-\sigma_1(S)=\sum_{v\in\mathcal{N}_u\backslash\mathcal{S}}b_{u,v}$$

- Two-Hop

  在S中加入一个新的种子节点u包括三部分：

  a. 来自节点u本身：$1-\pi_2^S(u)$
  b. 来自u的非种子邻居：$(1-\pi_1^S(u))\cdot\sum_{v\in\mathcal{N}_u\backslash S}b_{u,v}$
  c. 来自u的邻居的邻居（非种子节点）：
  $\sum_{v\in\mathcal{N}_u\backslash S}(b_{u,v}\cdot\sum_{w\in\mathcal{N}_v\backslash S\ cup\{u\}}b_{v,w})$

  因此，总影响传播增益为

  $$

  \sigma_2(S\cup\{u\})-\sigma_2(S)=1-\pi_2^S(u)+(1-\pi_1^S(u))\cdot\sum_{v\in\mathcal{N}_u\backslash S}b_{u,v}+\sum_{v\in\mathcal{N}_u\backslash S}(b_{u,v}\cdot\sum_{w\in\mathcal{N}_v\backslash S\cup\{u\}}b_{v,w})

$$

$$\sigma_2(S\cup\{u\})-\sigma_2(S)=1-\pi_2^S(u)+(1-\pi_1^S(u))\cdot\sum_{v\in\mathcal{N}_u\setminus S}b_{u,v}+\sum_{v\in\mathcal{N}_u\setminus S}(b_{u,v}\cdot\sum_{w\in\mathcal{N}_v\setminus S\cup\{u\}}b_{v,w})$$

$$\sigma_2(S\cup\{u\})-\sigma_2(S)=1-\pi_2^S(u)+(1-\pi_1^S(u))\cdot\sum_{v\in\mathcal{N}_u\setminus S}b_{u,v}+\sum_{v\in\mathcal{N}_u\setminus S}(b_{u,v}\cdot\sum_{w\in\mathcal{N}_v\setminus S\ cup\{u\}}b_{v,w})$$

数据集

DATASETS	NODES	EDGES
NetHEPT	15k	32K
LiveJournal	5M	69M
Twitter	42M	1.5B

实验分析

Algorithm

- HighDegree：选择k个度最高的节点
- DegreeDiscount：启发式算法
- IRIE：最先进的启发式算法
- IMM：先进的基于采样的方法
- D-SSA：相比IMM进一步减少生成样本的方法

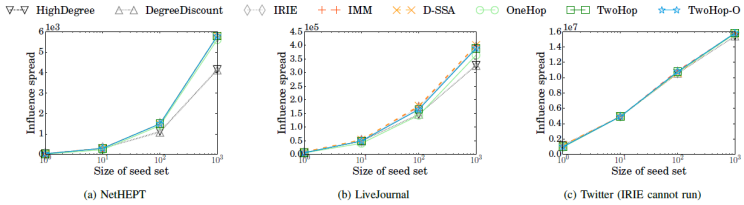
Parameter Setting

对于独立级联模型，通过两种模型来设置概率：

- WC model：$p_{u,v} = 1/|\mathcal{I}_v|$
- TRIVALENCY model：$p_{u,v}\ from\ a\ set\ \{0.1, 0.01, 0.001\}\ at\ random$

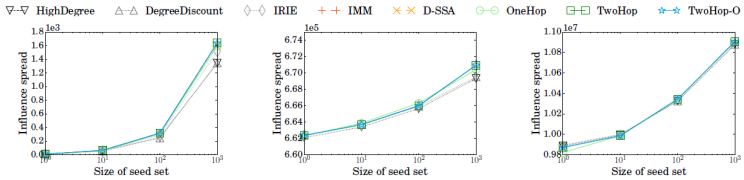
Results

- Influence Spread



(a) NetHEPT (b) LiveJournal (c) Twitter (IRIE cannot run)

Fig. 6. Influence spread on various graphs under the WC model.

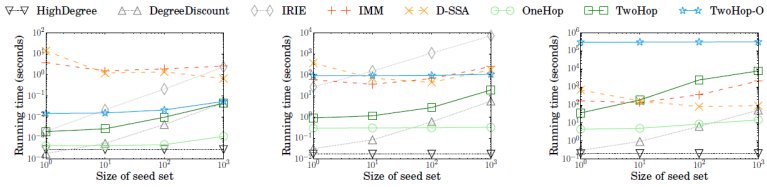


(a) NetHEPT (b) LiveJournal (IMM and D-SSA cannot run) (c) Twitter (IRIE, IMM and D-SSA cannot run)

Fig. 7. Influence spread on various graphs under the TRIVALENCY model.

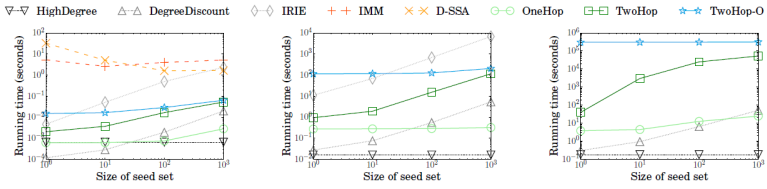
选取不同的k在两个设置概率的模型下的结果如上图，由于内存和时间原因，一些先进方法不能得到结果，文章的方法与IMM和D-SAA同样好，在NetHEPT数据集上表现比HD和DD要好。

- Running Time



(a) NetHEPT (b) LiveJournal (c) Twitter (IRIE cannot run)

Fig. 8. Running time on various graphs under the WC model.



(a) NetHEPT (b) LiveJournal (IMM and D-SSA cannot run) (c) Twitter (IRIE, IMM and D-SSA cannot run)

Fig. 9. Running time on various graphs under the TRIVALENCY model.

根据效率和有效性来说，基于Hop的算法比其他算法要更优一些。Two-Hop耗费时间要高于One-Hop，但性能更好。因此如果应用时对时间敏感度高，One-Hop可以表现的更好，若要求质量，使用Two-Hop要更好，而且它的运行时间是完全可以接受的。

- Memory Usage

  基于Hop的方空间复杂度要比其它方法小很多，因此这些方法在非常大的数据集上不能得到结果。

其它解决方法

首先是$(1-1/e-\epsilon)$的贪心算法，针对其效率问题，有了它的两种改进方式：启发式方法和蒙特卡洛模拟。

其中PMIA贪心方法利用传播路径的独立性去构造树进行粗略影响估计，但它很耗时间和空间。

DegreeDiscount贪心法在one-hop的邻居节点内粗略估计影响传播，和IRIE一样都是改进基于采样的方法。

创新之处

1. 利用传播路径的独立性提出了兼顾效率和有效性的hop-based的算法去计算准确的影响传播。
2. 基于CELF的思想，在影响传播上得出了一个上界去进一步加速hop-based算法。