基于两阶段启发的社交网络影响最大化算法

杨书新,刘成辉,鲁纪华

(江西理工大学,江西 赣州 341000)
E-mail:yimuyunlang@ sina. com

摘 要: 社交网络中影响最大化问题的研究一直是社交网络分析的重点之一,其技术在人们生活的很多领域中具有应用价值. 针对现有影响最大化算法存在时间复杂度高、算法精度低和不稳定的问题,文中利用线性阈值模型的能够将影响力累积的特性,提出一种基于度和影响力的混合启发式算法—DIH (Degree and Influence Heuristic)算法. 该算法综合考虑网络的传播特性和结构特性,基于线性阈值模型将整个影响最大化计算分为两个启发阶段:首先进行度折启发,在激活节点的同时将节点的影响力积累,然后进行影响力启发,将度折启发期间积累的影响力爆发,从而激活更多的节点. 为保证算法的效率,本文在影响力启发阶段设计了一种近似估计节点影响力的计算方法. 最后,本文在三个不同的真实网络中验证了算法的有效性.

关 键 词: 社交网络;影响最大化;两阶段启发;线性阈值模型
中图分类号: TP399 **文献标识码:** A **文 章 编 号:** 1000-1220(2017)10-2268-07

Two Stages of Heuristic Based Algorithm for Influence Maximization in Social Network

YANG Shu-xin, LIU Cheng-hui, LU Ji-hua
(School of Information Engineering, Jiangxi University of Science and Technology, Ganzhou 341000, China)

Abstract: The study of influence maximization is the focus of the social network analysis, and its theory has a great significance in our lives. There still exist some problems to be solved in its application, such as high time complexity, low precision and instability. To solve these problems, this paper proposes a hybrid heuristic algorithm named Degree and Influence Heuristic (DIH). According to the accumulation caused by using linear threshold model features, the computation of influence maximization is broken up into two stages: the degree discount heuristics and influence-based heuristics. We first employ the degree discount heuristics to active nodes and accumulate influence of nodes. Then, we present an approximate evaluation method to activate more nodes during the stage of influence-based heuristics. The experimental results on three different real datasets show the effectiveness of the proposed algorithm.

Key words: social network; influence maximization; two-stage heuristics; linear threshold model

1 引 言

影响最大化问题自从被 Domingos 和 Richardson 等[1,2]提出以来,就一直是研究人员的关注的热点,并取得了大量的研究成果,这些成果主要集中于一些贪心算法和启发式算法[3,4]. 其中,启发式算法的效率优于贪心算法,但现有的启发式算法大都专注于单个启发因素,如度、介数等,没有考虑网络的整体结构,使得算法的精度较低且适应性差. 针对以上问题,本文基于线性阈值(Linear threshold, LT)模型,综合网络中度中心性和影响力中心性的优点,提出 DIH(Degree and Influence Heuristic)算法.

本文在第二部分介绍影响最大化问题的相关工作;第三部分介绍 DIH 算法用到的影响最大化传播模型;第四部分给出 DIH 算法的思想;第五部分给出了本文的实验设计与结果分析;第六部分为总结部分及未来工作.

2 相关工作

影响最大化问题就是如何在大规模网络中挖掘一个 k 个节点组成的集合,使得该集合在社会网络中的影响最大化. 长期以来,研究人员对于影响最大化问题的研究主要集中在算法的精度和效率上. 基于此,现有的影响最大化算法主要可以分为基于贪心策略的算法和基于启发式策略的算法两大类.

基于贪心策略的算法主要解决的是算法精度问题,这类算法能够保证算法可以获得良好的影响效果,但算法的效率比较低. 例如,Kempe 等[5]提出的爬山贪心算法,这是用来解决影响最大化问题的原始算法,该算法的影响效果能够达到最优解的$(1-1/e)$以上,但算法每次选取最大影响力的种子节点时需遍历整个网络,算法效率低. 此后,为了改进贪心算法的效率,Leskovec 等[6]利用函数的子模特性,提出了优化的贪心算法 CELF(Cost-Effective Lazy Forward),算法在保持原始贪心算法效果的同时,将效率提高了近 700 倍. Chen 等[7]提出了两种快速的贪心算法 NewGreedy 算法和 MixGreedy 算法. NewGreedy 算法利用每轮次缩减样本的策略来提升算法的效率,而 MixGreedy 算法则是 NewGreedy 算法与 CELF 算法相结合的产物,实验表明两种改进的贪心算

收稿日期:2016-08-08 收修改稿日期:2016-10-18 基金项目:国家自然科学基金项目(41362015,41562019)资助;江西省教育厅科技项目(GJJ14431, GJJ14432, GJJ14458)资助. 作者简介:**杨书新**,男, 1978 年生,博士,副教授,CCF 会员,研究方向为图数据管理与生物信息学;**刘成辉**,男,1991 年生,硕士研究生,研究方向为社交网络影响分析;**鲁纪华**,女,1989 年生,硕士研究生,研究方向为生物信息学.

法的效率都优于 CELF 算法. Zhou 等[8]提出了 LNG 算法,该算法将影响力的全局计算近似为局部计算,并提出了将惰性节点作为种子节点的策略. 为了进一步加快问题的求解速度,Zhou 等[9]以 CELF 算法为基础,通过设定上限的策略提出了 UBLF(Upper Bound based Lazy Forward)算法,Goyal 等[10]提出了 CELF + + 算法. 尽管上述改进的贪心算法在效率上有着数百倍的提高,但在求解大规模网络的问题上仍有所不足.

基于启发式策略的算法主要关注于算法的效率问题,该类算法能够有效的解决效率问题,但算法的影响效果比较差且不稳定. 近年来随着网络规模的增长,出现了大量的启发式算法. 例如,Chen 等[8]对 Degree 算法[11]进行了改进,提出了 DegreeDiscount 算法,该算法能获得更好的影响效果且效率只是稍差于 Degree 算法. Pablo 等[12]为了解决选取最大度时节点的覆盖问题,提出了 SCG(Set Covering Greedy)算法. 曹玖新等[4]提出了一种基于核数的启发式算法 CCA(Core Covering Algorithm)算法. Jiang 等人[13]采用模拟退火算法来挖掘种子节点,该策略的时间复杂度较低且能够获得较好的影响效果,但该策略的稳定性没有理论保证. Chen 等[14]提出了一种在 LT 模型下针对有向无环图进行影响最大化的 LDAG 算法,该算法能获得良好的影响效果,但只适用于线性阈值模型.

上述两类算法各有优缺点,贪心算法能够保证算法的精度,但算法效率低;启发式算法在效率方面相较于贪心算法通常能够提高数个数量级,而在影响效果方面则远不及贪心算法,且在不同的网络结构上表现不稳定. 因为影响最大化算法是以某一特定模型为前提而展开的,所以本文在第 3 部分具体介绍 DIH 算法中使用的 LT 模型.

为了便于对读者的理解,我们将本文用到的重要符号定义列在表 1 中,具体描述如表 1 所示.

表 1　符号定义

Table 1　Symbol definition

符　号	含　义
$G(V,E)$	表示整个网络,其中 V 表示顶点,E 表示边
A	激活节点集
S	种子节点集
D	两节点间的距离
$D_{w,v}$	节点 w 从自身出发至节点 v 经过的节点数(不包括本身),即从 w 至 v 的距离
D_{max}	节点间互相影响的最大距离.
d_v^{out}	节点 v 的出度
d_v^{in}	节点 v 的入度
$N(v)$	节点 v 的入边邻居组成的集合
$H_D(v)$	与 v 距离为 D 的节点组成的集合
$influ_D(v)$	节点 v 对与其距离为 D 的节点的影响力
$Influ_D(v)$	节点 v 对与其距离为 D 的所有节点的影响力之和
$SInflu(v)$	节点 v 在整个网络中的影响力

注:在无向网络中,上表定义同样适用,此时节点的出度 = 节点的入度 = 节点的度

3　线性阈值模型

线性阈值模型和独立级联模型是影响最大化问题中的两种基础模型[5],其中线性阈值模型具有将影响力累积的特性,这也是本文 DIH 算法能够将影响最大化算法分为两个阶段的前提. 在线性阈值模型中,每个节点只存在两种状态:激活和未激活. 每个节点只能被激活一次,且节点的状态只能由未激活状态转变为激活状态. 任意节点 v 都会随机分配一个特异性阈值 $\theta_v \in [0,1]$,θ_v 表示 v 被激活的难易程度. 节点 v 对邻居 w 的影响力用 $b(v,w)$ 表示,在本文中将 $b(v,w)$ 的值定义为节点 w 入度的倒数. 一般来说,节点 v 的邻居 w 对 v 的影响力 $\sum b(v,w)$ 小于等于 1. 给定初始激活集合 $A = \varnothing$,线性阈值模型的传播过程如下:在每个离散步骤 T 中,一个处于未激活状态的节点 v,受每一个处于激活状态的邻居节点 u 的影响. 如果节点 v 的所有处于激活状态的邻居节点对 v 的影响力之和大于等于 θ_v,则 v 在步骤 $T+1$ 处于激活状态. 整个过程一直持续到到不再有节点被激活才结束.

4　DIH 算法

4.1　算法描述及伪代码

度数大的节点大多数情况下都处于网络的中心[12]. 在现有的启发式策略中,度数启发式策略的效率是最优的. 但度数只是一个局部最优的表现,没有考虑到网络的整体结构,因而获得的影响效果较差且不稳定. 影响力大的节点在网络中也往往处于中心地位. 度数最大的节点拥有较大的影响力,但并不一定具有最大的影响力. 因为节点的影响力是基于全局的而度数只是局部的. 例,在图 1 中度数最大的节点是 v_1,但影响力最大的节点却是 v_4.

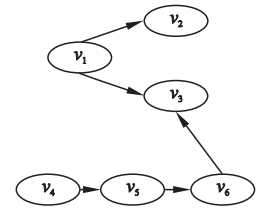

图 1　简单网络

Fig. 1　Simple network

本文综合考虑网络中节点的度中心性和影响力中心性的优势,利用 LT[5]模型能够将节点的影响力积累的特性,提出 DIH 算法. 该算法将整个影响最大化过程分为两个启发阶段:度折启发阶段,影响力启发阶段. 在度折启发阶段中,我们以"度数"为中心快速激活节点,并将激活失败时产生的影响力累积. 在影响力启发阶段中,我们以"影响力"为中心,将度折启发阶段积累的影响力爆发,激活更多的节点. 本文之所以将度和影响力分为两阶段启发而不是整合度和影响力进行启发式计算,是因为在度折启发过程中,处于激活状态的节点因激活失败而产生的影响力会被存储下来,从而提升影响力在网络中的中心性. 因 DegreeDiscount 算法[8]较 Degree 算法[12]能够获得更好的影响效果,且效率只是稍逊,所以本文将 DegreeDiscount 算法中的"度折策略"作为第一启发阶段的启发策略. 由于 LT 模型的特性,算法在完成度折启发阶段后,会将"遗留的影响力"积累起来. 为了能够更好的利用这些影响力,我们在算法的第二阶段提出一种新的影响力启发方法.

在 DIH 算法中,将 k 个种子节点分成两个启发式阶段进行选取,在度折启发式阶段选取 k_1 个"度数最大"的节点作为种子节点. 在影响力启发式阶段选取 k_2 个影响力最大的启发式节点作为种子节点. 其中,$k_1=\lceil \alpha k \rceil$,$k_2=k-k_1$. 启发因子 α 的值在(0,1)之间,这是一个经验值,这个值我们会在本文的 5.3.1 中给出. DIH 算法伪代码详见算法 1.

算法 1. DIH 算法伪代码

输入:图 $G(V,E)$,θ,种子节点个数 k,启发因子 α

输出:激活节点集 A

1. initialize
$S=\varnothing, A=\varnothing, k_1=\lceil \alpha k \rceil, k_2=k-k_1, t_v=0$;
2. for each node $v \in V$ do
3. $d_v^{out}=CC(v)$;
4. $dd_v=d_v^{out}$;
5. end for
6. for $i=1$ to k_1
7. $w=\text{argmax}\{dd_v \mid v \in V/S\}$;
8. $S=S \cup w$;
9. $A=A \cup w$;
10. for each node $v \in V/S$ do
11. for each node $w_1, w_1 \in N(v)$
12. if $w_1 \in S$
13. $t_v=t_v+1$;
14. end if
15. end for
16. $dd_v=d_v^{out}-2t_v-(d_v^{out}-t_v)t_v b(w_1,v)$;
//进行度数折扣计算
17. $t_v=0$;
18. end for
19. end for
20. $A=Active(S)$;
21. Refresh $I(v)$
//更新未激活节点受到的影响力
22. initialize $S=\varnothing$;
23. for each node $v \in V/A$ do
24. $SInflu(v)=CI(v)$;
25. end for
26. for $i=1$ to k_2
27 $w=\text{argmax}\{SInflu(v)\}$;
//其中 $v \in V/(S \cup A)$
28. $S=S \cup w$;
29. $A=A \cup w$;
30. end for
31. $A=Active(S)$;

其中,S 表示种子节点集,t_v 表示节点 v 的状态为激活的邻居个数,函数 $CC(v)$ 用来计算节点 v 的出度.

在 DIH 算法伪代码中,第 2 至 5 行计算各节点的出度并保存. 第 6 至 20 行为算法的第一阶段:度折启发阶段,其中,第 6 至 19 行对节点进行度折计算,并将找到的 k_1 个种子节点加入到激活节点集 A 中;第 20 行的 $Active(S)$ 函数用来将种子节点集在 LT 模型下对未激活状态的节点进行激活,并将被激活的节点加入 A 中. 第 21 行利用 $I(v)$ 更新处于未激活状态的节点受到的影响力,将度折启发阶段中因激活失败而产生的影响力积累起来. 第 23 至 31 行为算法的影响力启发阶段,其中,第 23 至 25 行运用 $CI(v)$ 函数计算处于未激活状态节点的影响力,其具体计算方法将在 4.2 中详细介绍;第 26 至 30 行选取影响力最大的 k_2 个节点加入种子集,并将种子节点加入激活节点集 A 中. 第 31 行利用种子节点去激活其它处于未激活状态的节点,并将状态为激活的节点加入 A 中.

4.2 节点的影响力评估

定义 1. (节点的影响力) 节点的影响力 *SInflu* 是指节点对同一网络中的所有其它节点产生影响的量值.

定义 2. (节点间的距离) 节点间的距离 D 是指两个节点间路径的长度.

假设节点 v 的直接邻居与 v 的距离为 1,则 v 与直接邻居的邻居的距离为 2,以此类推,节点 v 和节点 w 间的距离用 $D_{v,w}$(或 $D_{w,v}$)表示.

在 DIH 算法中节点 v 对 w 要产生影响,需满足一个条件:节点 $v \to w$ 路径中所有节点所受的影响力不小于本身的阈值.

假设节点 v 对单个节点的影响力用 $influ(v)$ 表示,节点 v 对自身不产生影响力,节点 v 对直接邻居的影响力定义为 $influ_1(v)$,节点 v 对与其距离为 D 的节点的影响力为 $influ_D(v)$. 节点 v 对与其距离为 D 的节点的影响力的可用如下公式表示:

$$influ_D(v)=\begin{cases} b(u,w)+b(s,w), & \text{当} \dfrac{b(u,w)}{\theta_w} \geqslant 1 \\ 0, & \text{否则} \end{cases} \tag{1}$$

其中,$b(u,w)$ 表示节点 u 和 w 之间的影响力,$u \in (H_{D-1}(v) \cap N(w))$,$w \in H_D(v)$,$u,w \notin A$. $b(s,w)$ 表示度折启发阶段积累的影响力,$s \in N(w)$ 且 $s \in A$,$s \notin H_{D-1}(v)$.

由公式(1)可以得出,节点对与其距离为 D 的所有节点的影响力之和 $Influ_D(v)$ 为

$$Influ_D(v)=\sum influ_D(v) \tag{2}$$

同理,由公式(2)可以得出,节点 v 在整个网络的影响力 $SInflu(v)$ 为

$$SInflu(v)=\sum_{D=1} Influ_D(v) \tag{3}$$

定义 3. (节点间的最大距离) 由于节点间的影响力随着距离增长而减弱,所以节点只会在一定的范围内产生影响. 在 DIH 算法中节点间的能够产生影响的最大距离用 $D_{\max}$ 表示,一旦超过这个阈值,双方之间就不再产生影响.

在 DIH 算法中,$D_{\max}$ 每增加 1,如,$D_{\max}$ 由 1 变成 2,算法的时间复杂度就会至少增 $O(n \times \max(d^{out}))$. 不仅如此,当节点间的距离超过 2 时,节点间就有可能产生回路. 这时在计算节点的影响力前,必须将整个网络进行去除回路的处理,否则,就会导致节点间的影响力被重复计算,同样这也增加了算法的复杂度. 为保证算法的效率,DIH 算法将节点间的最大距离 $D_{\max}$ 定义为 2,即,$D_{\max}=2$. 因此,节点 v 对整个网络的影响力 $SInflu(v)$ 可近似计算为

$$SInflu=\sum_{D=1}^{D_{\max}} Influ_D(v) \tag{4}$$

为了更好的理解本文提出的节点影响力计算方法,我们以图 2 的网络来演示节点影响力的计算过程. 其中矩形表示处于激活状态的节点,椭圆为未激活,边表示节点间的影响力. 在本例中,我们将每个节点的阈值定义为 $\theta=0.5$,节点受到邻居的影响力为节点入度的倒数. 从图中可看出,节点 v_7 处于激活状态(假设 v_7 为度折启发阶段被激活的节点),其它节点都处于未激活状态. 在这里我们以计算节点 v_1 的影响力为例进行阐述.

利用公式(1),可得节点 v_1 对它的直接邻居(与 v_1 距离为 1 的节点)v_2 和 v_3 产生的影响力分别为

$$\underset{v_1\to v_2}{influ_1}(v_1)=b(\underset{v_1\to v_2}{v_1,v_2})=1 \tag{5}$$

$$\underset{v_1\to v_3}{influ_1}(v_1)=b(\underset{v_1\to v_3}{v_1,v_3})=1 \tag{6}$$

利用公式(2),可得 v_1 对与其距离为 1 的所有节点的影响力之和

$$Influ_1(v_1)=\underset{v_1\to v_2}{influ_1}(v_1)+\underset{v_1\to v_3}{influ_1}(v_1)=2 \tag{7}$$

同理可得,v_1 对与其距离为 2 的所有节点的影响力之和

$$Influ_2(v_1)=\underset{v_1\to v_4}{influ_2}(v_1)+\underset{v_1\to v_5}{influ_2}(v_1)=\frac{5}{3} \tag{8}$$

因为节点 v_6 与 v_1 的距离 D_{v_1,v_6}($D_{v_1,v_6}=3$)大于 $D_{\max}$($D_{\max}=2$),所以节点 v_1 对 v_6 不产生影响. 节点 v_7 处于激活状态,因为激活节点只能被激活一次,所以节点 v_1 对 v_7 的影响力不被计算在内. 节点 v_1 至 v_8 之间没有路径,所以节点 v_1 对 v_8 不产生影响.

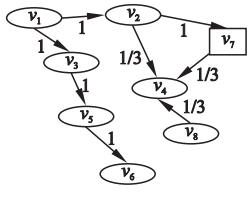

图 2　小型网络

Fig. 2　Small network

综上,节点 v_1 在该小型网络图中的影响力为

$$SInflu(v_1)=\sum_{D=1}^{2}Influ_D(v_1)=\frac{11}{3} \tag{9}$$

5　实　验

本文所有实验都是在计算机上完成的,该计算机的配置如下:主频为 3.70GHz 的 Intel Xeon E5-1620 v2 CPU,运行内存为 16GB. 实验中涉及的代码均为 Matlab 编写.

5.1　实验数据

为了验证实验的有效性,本文选取了三个真实的网络数据集进行仿真,它们均来源于 Stanford Network Analysis Platform(http://snap.stanford.edu/index.html). 具体的数据信息见表 2.

数据集 1 是来自 Wikipedia 的投票历史网络,其中点表示 Wikipedia 用户,有向边 $v\to w$ 表示用户 v 将选票投给 w,这是一个有向网络. 数据集 2 是来自 HepTh 的高能物理理论合作者网络,点代表作者,边表示两个作者间存在合作,这是一个无向网络. 数据集 3 是来自 AstroPh 的合作者网络,点代表作者,边表示两个作者间存在合作,这是一个无向网络.

表 2　实验数据集

Table 2　Experimental data sets

	数据集 1	数据集 2	数据集 3
数据名称	Wikipedia	HepTh	AstroPh
节点数	7115	9877	18872
边数	103689	25998	396160
平均度	14.994	1.5120	2.0404
平均聚类系数	0.1409	0.4714	0.6306

5.2　实验设置

为验证本文所提方法的性能,本文选择 DegreeDiscount[8],SCG[13],LDAG[14] 作为对比算法. 这几个算法都是比较经典的启发式算法. 其中 DegreeDiscount 算法较好地衡量了一个节点在当前邻域的影响力,并在多数实验中有可观的表现. 基于种子节点覆盖的 SCG 算法具有较好的影响效果. LDAG 算法是在 LT 模型中表现较好的启发式算法,它具备良好的影响效果和运行时间等特征. 为了避免数据的随机性,实验以随机选取种子节点的 Random 算法作为基准比较.

DIH 算法对于影响因子 α 的取值有一定的依赖性,不同的 α 取值会产生不同的结果. 因此需要将 DIH 算法在不同 α 值时的表现进行分析对比找出最佳的 α 值. 在 LT 模型中,节点的阈值 θ 是一个在[0,1]之间的随机数. 为了方便对比,本文中的算法都是在节点阈值 $\theta=0.5$ 的情况下进行的[5].

在本实验中,各算法将以种子节点 $k\in\{10,20,30,40,50,60,70,80,90,100\}$ 时的表现进行对比分析.

5.3　结果分析

影响最大化算法的性能一般通过两种指标评判:

1)影响效果:算法在同一传播模型和相同种子节点个数的条件下能够影响整个网络的节点数. 影响的节点个数越多表示算法影响效果越好.

2)运行效率:算法基于同一传播模型和相同种子节点数下的运行时间. 耗费的时间越短表示算法的效率越高.

5.3.1　DIH 算法结果分析

为了确定最优的 α 值,我们在三个网络上对不同 α 值时的 DIH 算法进行了实验.

在下页图 3 至图 5 中,我们给出了在三个网络中不同 α($\alpha\in\{0.1,0.4,0.5,0.6,0.9\}$)下的 DIH 算法随 k 值变化的影响效果. 从图中可以明显看出,在这三个网络中,当 k 相同时,不同 α 值的 DIH 算法的影响效果不同. 其中,在数据集 1 中,α 的值越趋于中心,算法的影响效果越好;$\alpha=0.5$ 时,算法的影响效果最好. 在数据集 2 中,α 的值越大,算法的影响效果越差;$\alpha=0.1$ 时,算法的影响效果最好. 在数据集 3 中,算法在面对不同 α 值的表现与在数据集 2 中的趋势一致,α 越大,效果越差;$\alpha=0.1$ 时,算法的影响效果最好.

通过上述分析,我们可知,在这三个网络中,DIH 算法的最佳 α 值不同. 那么面对不同的数据类型,我们不可能对每一个 α 值进行实验,然后选取最优 α 值. 但从实验结果中可以看出算法在选择不同 α 值时,$\alpha=0.4$ 时的 DIH 算法表现最稳定的,且其在三个网络中的影响效果始终保持在前两名. 尽管 $\alpha=0.1$ 时的 DIH 算法在数据集 2 和 3 中都获得了最佳的

影响效果,但其在数据集 1 中的表现十分糟糕,不够稳定. 同理,$\alpha=0.5$ 时的 DIH 算法虽然在数据集 1 中的影响效果优于 $\alpha=0.4$ 时的 DIH 算法,但只是略优,并没有明显的优势;而在其它两个数据集中,$\alpha=0.4$ 时的 DIH 算法相较于 $\alpha=0.5$ 时的 DIH 算法有明显的优势. 同时,$\alpha=0.4$ 是一个接近中心的取值,无论 α 偏大或偏小,都不会出现影响效果极差的表现. 因此,本文把 $\alpha=0.4$ 作为算法的参数.

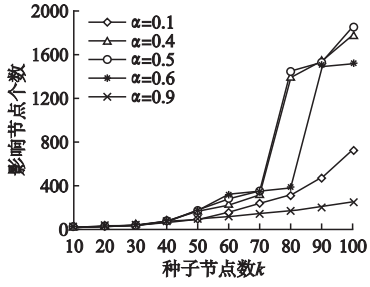

图 3 不同 α 值的 DIH 算法在数据集 1 中的表现

Fig. 3 Performance of DIH algorithm with different α on data set 1

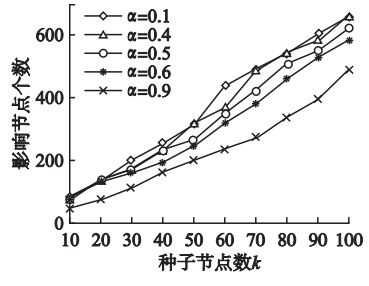

图 4 不同 α 值的 DIH 算法在数据集 2 中的表现

Fig. 4 Performance of DIH algorithm with different α on data set 2

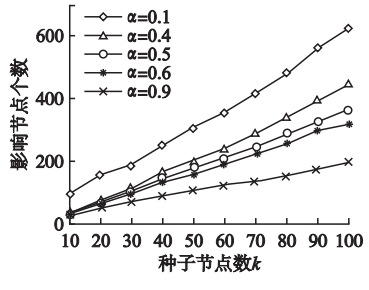

图 5 不同 α 值的 DIH 算法在数据集 3 中的表现

Fig. 5 Performance of DIH algorithm with different α on data set 3

在图 6 至图 8 中我们给出了在 LT 模型下 DIH 算法在 $\alpha\in\{0.1,0.4,0.5,0.6,0.9\}$ 值时,算法的度折启发阶段和影响力启发阶段在不同 k 至下的平均影效果对比. 从图中可明显看出在这三个网络中,除了在 $\alpha=0.9$ 时的 DIH 算法外,其它情况下的 DIH 算法的度折启发阶段激活的节点个数都少于影响力启发阶段. 从实验结果中还可以发现,影响力启发阶段的影响效果与 DIH 算法的最终影响效果成正相关. 此后,在本文中提到的 DIH 算法均指 $\alpha=0.4$ 时的 DIH 算法.

为了验证 DIH 算法在影响力启发阶段之所以能够形成激活节点的爆发是由度折阶段因激活失败而产生的影响力被积累下来而造成的,本实验设计一种 DIH* 算法与 DIH 算法进行对比分析. DIH$*$ 算法基本与 DIH 算法一致,唯一区别在于 DIH 算法在完成度折启发阶段后会利用 LT 模型将影响力积累下来,而 DIH$*$ 算法却不会. 因 DIH 与 DIH$*$ 算法在度折启发阶段完全一样,所以实验只比较影响力启发阶段的影响效果.

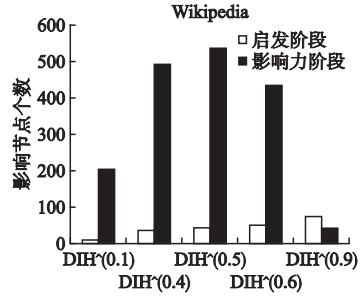

图 6 在数据集 1 中两阶段各激活的节点个数

Fig. 6 Number of activated nodes on data set 1

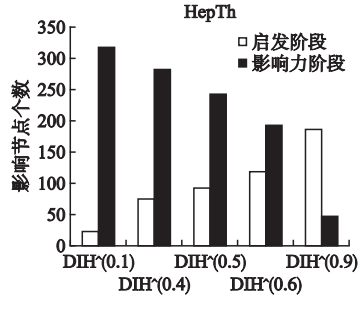

图 7 在数据集 2 中两阶段各激活的节点个数

Fig. 7 Number of activated nodes on data set 2

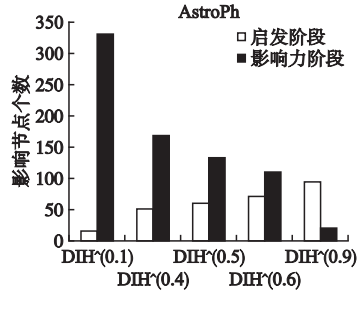

图 8 在数据集 3 中两阶段各激活的节点个数

Fig. 8 Number of activated nodes on data set 3

在下页图 9 至图 11 中,我们给出了 DIH 和 DIH$*$ 算法在影响力启发阶段随 k 值变化的影响效果. 从图中可以看出,在三个网络中,DIH 算法在影响力启发阶段的影响效果明显优于 DIH$*$ 算法. DIH 算法在进行度折启发时,处于激活状态的节点去激活未激活状态的节点,一旦失败,这时对该节点产生的影响力就会被储存起来,等到影响力启发阶段时,这些存储起来的影响力就会爆发起来,导致大量的处于激活状态的节点被激活. 而在 DIH$*$ 算法中没有储存影响力这一过程,这就造成了影响力启发阶段 DIH 算法的影响效果要优于 DIH$*$

算法,实验结果也验证了这点.

综合图 3 至图 11 的结果,表明 DIH 算法的影响力启发阶段会形成激活节点的大量爆发.

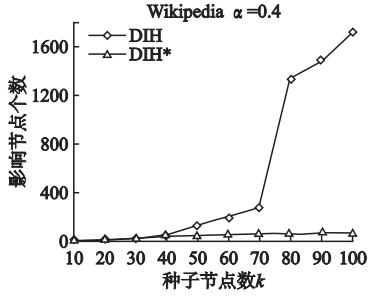

图 9　在数据集 1 中影响力启发阶段的影响效果

Fig. 9　Effect of influence-based heuristics

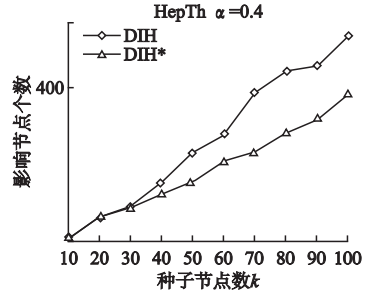

图 10　在数据集 2 中影响力启发阶段的影响效果

Fig. 10　Effect of influence-based heuristics on data set 2

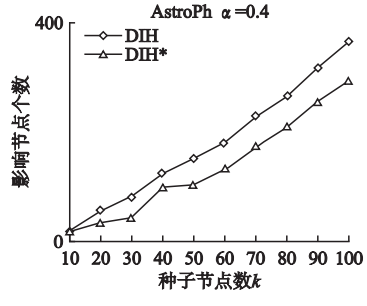

图 11　在数据集 3 中影响力启发阶段的影响效果

Fig. 11　Effect of influence-based heuristics on data set 3

5.3.2　对比算法结果分析

在图 12 至图 14 中我们给出了 DIH($\alpha = 0.4$)、DegreeDiscount[7]、SCG[12]、LDAG[14] 和 Random 算法在三个网络中的随着 k 值变化的影响效果. 从图中可以看出,在数据集 1 中,当 $k \leqslant 80$ 时,LDAG 算法的影响效果优于 DIH 算法;但此后随着 k 值的增长 LDAG 算法的影响效果渐渐趋于饱和,反之,DIH 算法的影响效果保持着持续增长的趋势,并超越了 LDAG 算法. DIH 算法的影响效果明显优于 DegreeDiscount、SCG 和 Random 算法. 在数据集 2 和 3 中,相较于其它对比算法,DIH 算法表现出了较好的影响效果.

在表 3 中我们给出了对比算法在三个网络中的运行时间. 从表中可明显看出,各对比算法的运行时间在同一数量级,除了 Random 算法外,DIH 算法的运行效率相较于 DegreeDiscount、SCG 和 LDAG 算法都略有优势.

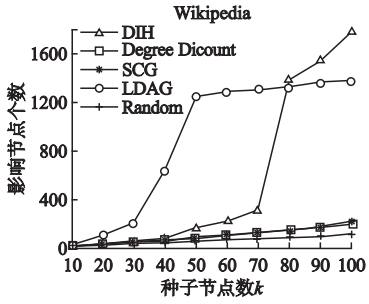

图 12　各对比算法在数据集 1 中的表现

Fig. 12　Performance of the algorithms on data set 1

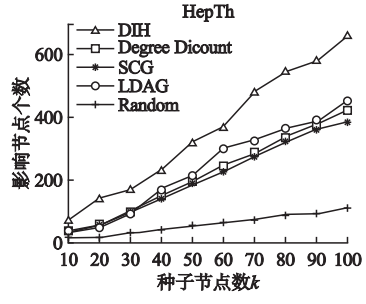

图 13　各对比算法在数据集 2 中的表现

Fig. 13　Performance of the algorithms on data set 2

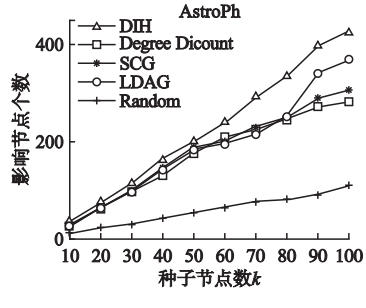

图 14　各对比算法在数据集 3 中的表现

Fig. 14　Performance of the algorithms on data set 3

表 3　不同算法的运行时间　单位(s)

Table 3　Running time of different algorithms units(s)

	数据集 1	数据集 2	数据集 3
DIH	100.2	175.5	3337.9
DegreeDiscount	146.7	224.6	4210.7
SCG	100.8	178.6	3872.6
LDAG	170.6	325.4	4386.5
Random	66.5	120.7	2808.3

综合以上分析,本文所提的 DIH 算法将整个影响最大化过程分为度折启发阶段和影响力启发阶段这一策略是可行的.

6　总　结

已有的启发式算法大都专注于单个启发因素,不能很好的适应不同的网络结构. 本文综合了考虑网络中度和影响力这两个中心性的特点,在 LT 模型的基础上提出了 DIH 算法来解决影响最大化问题. 同时,在 DIH 算法的启发阶段设计

了一种近似计算节点影响力的方法来提高算法的效率. 本文将 DIH 算法在三个真实的网络中进行了实验,实验结果表明 DIH 算法具有较好的性能及稳定性. 未来我们可以考虑将社区发现算法和并行策略引入影响最大化算法的求解中以提高算法的精度和效率.

References:

[1] Domingos P, Richardson M. Mining the network value of customers[C]. Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco: Association for Computing Machinery, 2001: 57-66.

[2] Richardson M, Domingos P. Mining knowledge-sharing sites for viral marketing[C]. Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York: Association for Computing Machinery, 2002: 61-70.

[3] Li H, Bhowmick S, Sun A, et al. Conformity-aware influence maximization in online social networks[J]. Vldb Journal — the International Journal on Very Large Data Bases, 2014, 24(1): 117-141.

[4] Cao Jiu-xin, Dong Dan, Xu Shun, et al. A k-core based algorithm for influence maximization in social network[J]. Chinese Journal of Computer, 2015, 38(2): 238-248.

[5] Kempe D, Kleinberg J, Tardos E. Maximizing the spread of influence through a social network[C]. Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York: Association for Computing Machinery, 2003: 137-146.

[6] Leskovec J, Krause A, Guestrin C, et al. Cost-effective outbreak detection in networks[C]. Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York: Association for Computing Machinery, 2007: 420-429.

[7] Chen W, Wang Y, Yang S. Efficient influence maximization in social networks [C]. Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York: Association for Computing Machinery, 2009: 199-208.

[8] Zhou S, Yue K, Fang Q, et al. An efficient algorithm for influence maximization under linear threshold model[C]. Proceedings of the 26th Chinese Control and Decision Conference, Washington: IEEE Computer Society, 2014: 5352-5357.

[9] Zhou C, Zhang P, Guo J, et al. UBLF: An upper bound based approach to discover influential nodes in social networks[C]. Proceedings-IEEE 13th International Conference on Data Mining, NJ: Institute of Electrical and Electronics Engineers, 2013: 907-916.

[10] Goyal A, Lu W, Lakshmanan L V S. CELF + +: optimizing the greedy algorithm for influence maximization in social networks [C]. Proceedings of the 20th International Conference Companion on World Wide Web, New York: Association for Computing Machinery, 2011: 47-48.

[11] Wasserman and Faust. Social network analysis: methods and applications[J]. American Ethnologist, 1997, 24(1): 219-220.

[12] Estevez P A, Vera P, Saito K. Selecting the most influential nodes in social networks[C]. The 2007 International Joint Conference on Neural Networks, NJ: Institute of Electrical and Electronics Engineers, 2007: 2397-2402.

[13] Jiang Q, Song G, Cong G, et al. Simulated annealing based influence maximization in social networks[C]. Proceedings of the 25th AAAI Conference on Artificial Intelligence and the 23rd Innovative Applications of Artificial Intelligence Conference. CA: AI Access Foundation, 2011: 127-132.

[14] Chen W, Yuan Y, Zhang L. Scalable influence maximization in social networks under the linear threshold model[C]. Proceedings of the 10th IEEE International Conference on Data Mining, NJ: Institute of Electrical and Electronics Engineers, 2010: 88-97.

附中文参考文献:

[4] 曹玖新,董 丹,徐 顺,等. 一种基于 k-核的社会网络影响最大化算法[J]. 计算机学报, 2015, 38(2): 238-248.