分类号 TN91　　密级 公开
UDC 621.39　　编号 10299s1108028

江苏大学

硕士学位论文

社交网络影响力最大化改进算法研究与设计

The Research and Design of Improved Algorithms for Influence Maximization in Social Networks

指导教师 周莲英
作者姓名 朱锋

申请学位级别 硕士 学科（专业） 通信与信息系统
论文提交日期 2014 年 4 月 论文答辩日期 2014 年 6 月
学位授予单位和日期 江苏大学 2014 年 6 月

答辩委员会主席 宋雪桦
评阅人

独 创 性 声 明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已注明引用的内容以外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果，也不包含为获得江苏大学或其他教育机构的学位或证书而使用过的材料。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：朱辉

2014年6月9日

学位论文版权使用授权书

江苏大学、中国科学技术信息研究所、国家图书馆、中国学术期刊（光盘版）电子杂志社有权保留本人所送交学位论文的复印件和电子文档，可以采用影印、缩印或其他复制手段保存论文。本人电子文档的内容和纸质论文的内容相一致，允许论文被查阅和借阅，同时授权中国科学技术信息研究所将本论文编入《中国学位论文全文数据库》并向社会提供查询，授权中国学术期刊（光盘版）电子杂志社将本论文编入《中国优秀博硕士学位论文全文数据库》并向社会提供查询。论文的公布（包括刊登）授权江苏大学研究生处办理。

本学位论文属于不保密 [√]。

学位论文作者签名：宋辉　　指导教师签名：周建英

2014年 6月9日　　2014年 6月9日

摘要

社交网络由海量的用户及用户间复杂的关系组成。不同于传统网络，社交网络中信息的传播与扩散依赖于用户间的关系。如何使信息在网络中能被尽可能多的用户所接收，即社交网络影响力最大化问题，为当前社交网络及其应用的研究热点，可广泛应用于舆论宣传、商业广告等方面。具体到实现算法，社交网络影响力最大化问题即是在网络中如何挑选包含一定数目节点的种子节点集，去激活整个网络，使网络中最后被激活的节点数最大化。

Diffusion Degree算法为目前最流行的基于启发式的影响力最大化算法，该算法提出了节点潜在影响力的概念，即一个节点的邻居节点的影响力也可以作为当前节点的影响力的一部分。相比于其他同类算法，该算法意在优化影响范围。论文在Diffusion Degree算法概念的基础上，基于节点对邻居节点的激活概率，通过激活概率阈值进一步考虑了节点间潜在影响力的实际有效性，结合Single Discount算法实时更新节点影响力的设计思想，弥补了Diffusion Degree算法的欠缺。论文给出了Diffusion Degree算法的改进算法IDD的详细设计。

另一方面，论文通过引入节点活跃度，进一步考虑了节点本身影响力的实际有效性，扩展了启发式算法所依赖的独立级联IC模型，改进模型中节点激活规则和影响力传播过程。论文基于所提出的扩展模型AIC，设计了一种启发式算法ACH，该算法首先通过过滤网络中对种子节点选择无用的节点与边，有效减小选择范围，然后基于AIC模型的活跃度特性对候选节点进行优质筛选，最后通过模拟影响力的传播过程来实时更新节点综合影响力，从而从多方面保证所选择的种子节点集的优质性。

论文用真实的网络数据集，对所提出的 IDD 与 ACH 算法分别进行了实验仿真，仿真结果表明，基于 IC 模型的 IDD 算法在时效性上略逊于 Diffusion Degree 算法，但在影响范围上明显优于 Diffusion Degree 算法；基于 AIC 模型的 ACH 算法，在影响范围上接近爬山贪婪算法的影响范围，优于其他启发式算法，而且在时效性上也优于其他启发式算法，较适合大型社交网络。

关键字：社交网络；影响力最大化；活跃度；激活概率；启发式

Abstract

Social network consists of the mass of users and the complex relationships between the users. Different from traditional networks, the propagation and diffusion of the information in social networks depend on the relationships between users. How to make the information in the social network can be received by as many users as possible, which defined as the problem of the influence maximization in the social network, is the current research focus. It can be widely applied in the terms of propaganda and commercial advertising. The algorithm for influence maximization in the social network is how to select a seed set of a certain number of seed nodes in the network to activate the entire nodes in the network, which maximizes the number of activated nodes in the network.

Diffusion Degree algorithm, one of the most popular influence maximization algorithm based on heuristic, which proposes the concept of the potential influence of a node, considers that the influence of the neighbor nodes of a node can also be used as part of the influence of the node. Compared to other similar algorithm, it is intended to optimize the influence spread of heuristics. In this paper, based on the basis of the concept of Diffusion Degree thesis and the activation probability between neighbor nodes, we further consider about the actual effectiveness of the potential influence of neighbor nodes through the threshold of the activation probability, combined with design idea of Single Discount which updates the influence of nodes in real time, to make up for the lack of Diffusion Degree algorithm. In this paper we also give a detailed design of the improved IDD algorithm which is based on Diffusion Degree algorithm.

On the other hand, in this paper, we further consider about the actual effectiveness of the influence of the node itself by extending the Independent Cascade model to incorporate the activity aspect of the node and the influence diffusion in social networks, proposing the Activity-Independent Cascade model which improves the activation rule and the influence diffusion under IC model. We also propose a new heuristic named ACH under the AIC model to improve influence spread. Firstly the ACH algorithm reduces the range of seed nodes by pruning the nodes and the relationships which are useless to select the seed nodes. Then the algorithm filters

the candidate nodes based on the activity feature of AIC model. Finally the algorithm simulates the diffusion and propagation of the influence to update the comprehensive influence of nodes. The ACH algorithm ensures the quality of seed nodes in so many ways.

In this paper, we conduct experiments for IDD algorithm and ACH algorithm on two real-life network data sets. One of our experimental results shows that the IDD algorithm based on IC has a significantly better performance on influence spread than Diffusion Degree algorithm with a little more running time. And the other shows that the ACH algorithm based on AIC model has a better influence spread than other heuristics, matching the influence spread with the greedy algorithm, with a better performance in efficiency than other heuristics.

Key Words: Social Network; Influence Maximization; Activity; Activation Probability; Heuristics

目录

第一章 绪论

1.1 研究背景

六度分割理论认为，社会关系网中的任意两个人，通过不超过六个人就能建立起人际关系[1][2]。也就是说通过人际关系网，我们可以认识整个世界。如图 1.1 所示，社交网络是由海量的用户及用户间复杂的关系（包括亲人关系、朋友关系、同学关系以及工作关系等）组成。不同于传统网络，社交网络中信息的传播与扩散依赖于复杂的用户间的关系和用户间的相互影响。对于社交网络的研究可以追溯到 19 世纪 20 年代早期，国外学者在这个领域的研究已经取得了很多成果，并形成了相对完善的研究体系。现在这个领域的研究主要有社区发现（挖掘）、社区可视化和影响力最大化等，其中影响力最大化是目前的研究热点，它在市场营销、信息传播以及社会和谐方面都有十分重要的影响[3]。

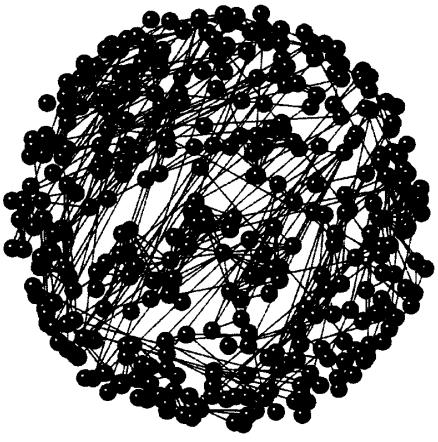

图 1.1 社交网络图

随着互联网的飞速发展，社交网络开始进入大家的视野并飞速地发展起来，在短时间内就形成了相当的规模。国外的社交网站发展较早，FaceBook、Twitter 等社交网站已经发展成熟，拥有海量的注册用户且用户互动性和活跃度很高。国内的社交网站虽然发展较晚，但是发展速度迅猛且有一种百家争鸣的趋势，新浪微博、腾讯微博、人人网等社交网站都是典型例子，还有一些特殊领域的社交网站，如针对职场的大街网、猎聘网，针对婚介相亲的百合网、有缘网。各种各样的社交网站如雨后春笋般崛起，并且开始不断影响着国内网民们的日常生活、各大企业的营销策略[4]。

社交网络对用户的消费意向与网络中信息的传播产生了巨大的影响[5]。针对社交网络的“病毒式营销”就是利用网络中用户间关系来影响用户的消费意向。由于社交网络中的用户间关系大都建立在真实的用户关系基础上，用户间都存在着互动与相互影响，当一个用户认可某个产品时会将其推荐给自己的亲戚好友，然后由此产生级联的推荐效果，亲戚好友又向自己的亲戚好友继续推荐，通过这种口碑效应使产品在网络中被更多的用户认可接受。这种营销模式不同于传统的直销或者是大众营销，不需要耗费大量的广告费用或是人力成本，只需要在社交网络中找到最具影响力的一部分用户，给他们提供免费试用，然后通过口碑效应就能使产品在网络中传播扩散。这种营销模式以最小的代价换取高额的回报，现在已经备受各大商家的青睐。尤其在这样一个电子商务高速发展的时代，各大电商纷纷开启社交营销模式，在各大社交网站上随处可见各种广告，很多电商也通过一些影响力大的用户来为自己打响口碑。现在一种新产品或者一部新电影在投入市场之前，都会通过社交网络大力宣传。一些电商还特意为注册用户经营一个社区，增强用户与电商之间的互动，以便实时了解、追踪用户需求，以此增强用户粘性。这不仅降低了电商的广告宣传费用，还优化了用户体验，提高了用户满意度。而且通过这种社区产生的口碑效应还会带来新的用户，在不丢失老用户的情况下还带来更多新的用户，形成一个良性循环，为电商带来了更多利好。

除了营销模式，社交网络对信息的传播也起到了巨大的影响[6]。在社交网络上，一个用户分享一条信息，他的所有好友都能看到，他的好友继续分享下去就形成了一种发散式的传播。无论是信息传播的速度还是最后扩散的规模都远远超越了传统的信息传播模式。这有利于公益信息的传播，让更多的人在最短的时间内能传递爱心，形成良好的社会风气。比如政府政策的推广、疾病的预防、公益爱心的传递等都能通过社交网络来达到信息快速传播的效果。但是社交网络的开放性和自由性也会导致一些恶意信息的传播，对社会带来不安定隐患。这时必须通过对社交网络上信息传播的相关研究，了解信息传播的规律和趋势，从而及时地发现并遏制恶意信息的传播，维持社会和谐。由此看出，社交网络影响力最大化的研究对于经济市场和社会和谐都有重要的研究意义。

1.2 研究现状

近些年，越来越多的学者开始关注社交网络影响力最大化算法研究，国内外学者们从不同的角度提出了很多相关算法，以下是关于这个问题的国内外的研究现状介绍。

Richardson 和 Domingos 将社交网络影响力最大化问题归纳成一个算法问题，即在社交网络中选择 k 个最具影响力的初始节点，通过这些初始节点去激活网络，从而使得影响力在整个网络中的传播最大化。同时他们给出了该问题的详细定义及评价指标，为后续研究奠定了基础[7][8]。

Kempe 和 Kleinberg 证明了社交网络影响力最大化问题是一个 NP-hard 问题，并提出了一种爬山贪婪算法[9]。这种算法虽然保证了 1-1/e 范围内接近最优解，但是由于该算法每次选择节点后都需要重新计算所有未激活节点的边际效应，其时间复杂度太高，并不适合大型的在线社交网络。他们还详细描述了两种影响力传播模型，独立级联模型（Independent Cascade Model）和线性阈值模型（Linear Threshold Model），后续相关算法的研究都是基于这两个模型或者是它们的扩展模型。

Leskove 针对爬山贪婪算法做出了改进优化，提出了 CELF（Cost-Effective Lazy Forward）算法。该算法利用独立级联模型的次模特型，大大减少了算法计算量，降低了贪婪算法的时间复杂度，但是它没有在影响范围上取得好的效果[10]。Amit Goyal 针对 CELF 算法作了进一步优化，提出了时间复杂度更低的 CELF++算法。该算法进一步避免了 CELF 算法中不必要的重复计算，使其运行速度比 CELF 算法快出 35%~55%[11]。

Wei Chen 等人在爬山贪婪算法的基础上提出了 NewGreedy 和 MixGreedy 两种优化算法。NewGreedy 算法利用独立级联模型的节点激活特性删除对影响力传播没有影响的边，构成局部小网络做影响力传播。MixGreedy 算法则是将选择过程分为两个部分，第一部分使用 NewGreedy 算法，第二部分使用 CELF 算法，结合了这两种算法使得算法整体的时间复杂度降低，但是依然不能满足大型社交网络的需求[12]。

Pablo 等人在爬山贪婪算法的基础上提出了一种改进的 SCG 算法[13]。该算法认为度数最大的节点可能互相为邻居节点，存在邻居重叠的情况，而为了使网络中影响力最大化需要尽可能避免这种情况，选择相对分散的种子节点。Pablo 改变了邻居节点集的定义，邻居节点不再是与当前节点直接连接的节点，而是与当前节点间的最短路径为小于等于 m 的节点（m 为自定义值）。每当选择一个节点作为种子节点后，SCG 算法将邻居节点排除后再继续选择。这样有效避免了种子节点影响范围重叠的情况。

由于贪婪算法的时间复杂度过高，即使是进行了时间复杂度的优化依然无法满足大型社交网络的需求，所以很多学者开始针对启发式算法进行优化，尽量提高启发式算法的影响范围。

Wei Chen 等人提出了一种启发式算法 Degree Discount。该算法基于度中心算法做了优

化，认为如果一个节点的邻居节点中有节点已经是种子节点，那就必须对该节点的出度进行“打折”，重新计算节点影响力。在文献[12]中，Wei Chen 通过实验证明 Degree Discount 算法相较于度中心算法在影响范围上得到了很大的改进。此后，Wei Chen 等人又相继提出了 MIA、PMIA 启发式算法，在影响范围上都取得了重大进展。此后 Wei Chen 等人还研究了在有限时间内的影响力最大化问题，在 IC 模型中引入了时间的概念提出 IC 模型的扩展模型 Time-Delayed IC 模型，并针对该模型提出 MIA-M 和 MIA-C 算法[14]。这两种算法也都在影响范围上取得了明显的进展。部分算法是针对特定网络的，Fa-Hsien Li 提出了一种 Maximum Coverage 算法用来解决带标记的社交网络影响力最大化问题[15]；在文献[16][17]中，WANG Y 等人考虑了在有预算限制的情况下的影响力最大化问题，并提出了相应的改进贪婪算法和启发式算法。

以上介绍的关于影响力最大化算法的研究都是针对全局网络的，还有一些学者提出了新的思路。基于社区挖掘算法的影响力最大化算法，通过社区发现或者是社区挖掘算法将社交网络分成若干个小的社区，然后分别在各个社区中选取种子节点，这一定程度上解决了种子节点堆簇的问题，使种子节点的覆盖范围最大化，但是这种算法依赖于社区挖掘算法对网络的分割效果。

T.cao 等人提出了基于社区发现算法的 OASNET 算法，该算法基于社交网络的社区性质，利用社区挖掘算法分割网络，然后为各个社区动态分配种子节点。OASNET 算法共分为三个部分，第一部分该算法利用社区发现算法把整个网络划分为若干个独立的社区；第二部分该算法根据划分出来的社区数量和种子节点集的大小，利用动态规划的方法把初始节点最佳地分配到各个社区，从而使整个网络中最终被激活的节点数（各个社区中被激活节点数的累加）最大化；第三部分该算法计算最佳分配。但是 OASNET 算法有一个缺点，它假设通过社区发现算法划分出来的社区之间是相互独立的，社区之间不存在联系，该算法则是在各个独立的社区中做影响力最大化。但是这个假设显然与实际不符，各个社区之间肯定存在或多或少的联系，而且如果社区间的联系较多，该算法提出的假设就会损失很多网络中的边[18]。

还有一种 CGA 算法　，该算法的总体思想与 OASNET 算法类似，首先使用社区发现算法划分社区，然后在各个独立的社区中做影响力最大化，最后将被激活节点的数累加起来。不同的是 OASNET 算法在各个社区中使用度中心算法来选择种子节点，而 CGA 算法则是使用爬山贪婪算法，所以 CGA 算法的时间复杂度远高于 OASNET 算法，但是由于划分出来的社区规模小于整个网络，所以该算法的时间复杂度低于爬山贪婪算法。不过该

算法也存在与 OASNET 算法一样的缺点，即假设社区间互相独立忽略了社区间的联系，损失了节点间的边[19]。

很多相关研究是针对影响力传播模型的，因为传播模型定义的科学性直接影响着影响力最大化的研究。Kempe 和 Kleinberg 提出了递减级联模型，他们认为如果一个节点被多次尝试激活后，仍然处于未激活状态，那么该节点之后被激活的概率会越来越低。在该模型中，节点间的激活概率在影响力传播过程中逐渐减小[20]。Kempe 针对 IC 模型中节点激活完全依赖于激活概率而忽略了节点度数对于节点被激活的影响这一情况，提出了加权级联模型。该模型中，节点度数越高，节点也被更多的节点影响，与之连接的边会被设置较低的激活概率。Kempe 还提出了一种非渐进模型，该模型中的节点的状态可以互相转换，不受 IC 模型中激活过程不可逆规则的约束[21]。冀进朝基于 IC 模型提出了一种完全级联模型，该模型中节点间的激活概率是随着影响力传播过程改变的，由一个可增减函数来表示[22]。

Saito 在 IC 模型和 LT 模型的基础上引入时延因素，提出了连续时间时延模型[23]。Bharathi 等人提出了基于 IC 模型的 CP(Competing Processes)模型[24]。Mayank 等人提出了一种新的信息传播模型 GADM(Genetics Algorithm Diffusion Model) [25]，通过改变模型参数，GADM 模型可以转变为 IC 模型和 CP 模型。

1.3 论文研究内容

目前传统的社交网络影响力最大化算法主要包括两大类，一类是影响范围接近最优的爬山贪婪算法，另一类是时效性优越的启发式算法。现有的很多相关研究也都是围绕对贪婪算法的时效性优化和对启发式算法的影响范围的优化展开的。本文的主要内容包括以下几个方面：

1. 阅读调研相关资料，掌握理论基础。

2. 针对 Diffusion Degree 算法存在的问题做出改进，提出 IDD(Improved Diffusion Degree)算法，目的是改进算法的影响范围。

3. 对 IC 模型做出扩展，引入节点活跃度属性，提出 AIC (Activity Independent Cascade)模型，改变了节点激活规则和影响力传播过程，使之更加科学，贴近实际。

4. 针对 AIC 模型，提出启发式算法 ACH 来解决基于 AIC 模型的影响力最大化问题。算法根据节点活跃度重新定义了节点综合影响力，并对网络进行了精简，选择过程中实时

更新网络，最后得出优质的种子节点集。

5. 在大型网络的真实数据集上实验仿真并分析结果，证明改进的 IDD 算法与 ACH 算法的有效性。

1.4 论文组织结构

从组织结构上，本文共分为五章:

第一章为绪论部分，介绍了本文的研究背景、国内外研究现状、主要研究内容以及组织结构。

第二章主要是详细介绍了本文研究内容的相关知识，分为社交网络简介、两种经典的影响力传播模型简介和常用的影响力最大化算法简介三部分。论文首先详细描述了社交网络的概念、社交网络影响力最大化问题的定义；然后描述了两种经典的影响力传播模型的概念及特征。最后详细介绍了已有的一些常用的社交网络影响力最大化算法，包括爬山贪婪算法、度中心启发式算法以及一些其他算法。

第三章首先详细介绍了 Diffusion Degree 算法，然后指出其存在的一些问题，最后进行了改进，并提出了 IDD 算法。Diffusion Degree 算法中将所有邻居节点的影响力都加权计算进当前节点的综合影响力中，但是节点只有被激活后才拥有影响力，激活概率过小时仍然对邻居节点的影响力进行加权计算显得并不合理。因此 IDD 算法根据激活概率阈值对邻居节点进行了过滤，考虑了节点间潜在影响力的实际有效性，并在每次选择一个种子节点后更新网络和节点的综合影响力。本章最后描述了 IDD 算法的实验环境、实验数据集以及实验结果和分析。

第四章首先指出了 IC 模型中存在的问题，然后对其进行了扩展，引入了节点活跃度属性，提出了 IC 模型的扩展模型 AIC 模型，最后针对该模型提出了 ACH 启发式算法。AIC 模型中由于新增了节点活跃度属性，节点激活规则和影响力传播过程被改变，新的激活规则更加科学，贴近实际。在 ACH 算法中也根据节点活跃度重新定义了节点综合影响力，从而更科学地选择种子节点。ACH 算法首先精简了网络大小，过滤对种子节点选择无用的信息，然后根据新定义的节点综合影响力选择种子节点，并实时更新网络。本章最后描述了 ACH 算法的实验环境、实验数据集以及实验结果和分析。

第五章为总结与展望，总结了本文的主要研究内容，并提出了论文中存在的值得进一步改进和研究的地方。

第二章 社交网络相关知识简介

本章主要介绍了相关的理论知识。主要内容包括三方面，首先介绍了社交网络的概念和社交网络影响力最大化问题的定义；然后详细描述了两种经典的影响力传播模型的相关概念；最后介绍了目前常用的一些社交网络影响力最大化算法，包括爬山贪婪算法、度中心启发式算法以及其他的优化算法。

2.1 社交网络简介

2.1.1 社交网络简介

社交网络是由海量的用户及用户间复杂的关系组成，不同于传统网络，社交网络中的信息传播与扩散依赖于用户间的关系[26]。社交网络中的用户可以是个体，也可以是以集体为单位，如公司、学校、组织等各种集体性的单位。由于用户的多样性，用户间的关系就更是复杂且多种多样，包括个体之间的朋友关系、亲属关系、工作关系和集体之间的合作、互动关系等。社交网络最重要的特征就是用户间的复杂关系，网络中各种信息的传播正是依赖于这个特征[27]。

著名的六度分割理论认为社交网络中的任意两个人，通过不超过六个人就能建立起人际关系。这说明社交网络中普遍存在着一种“弱纽带”，这使网络中的用户间的距离变得“更近”。网络中的各种信息通过用户间的关系传播的速度更快范围更大[28]。

通过基于图论的图形表达法被广泛应用于描述社交网络的网络结构[29]。一般我们将社交网络抽象成一张有向（或无向）图G(V, E)；其中V是网络中节点集合，E是节点间关系集合。V中每一个节点代表网络中的一个用户，每一条边 $(u,v) \in E$（$u,v \in V$）代表用户间的关系。在有向图中(u, v)与(v, u)表示不同的边，在无向图中则是表示同一条边[30]。目前关于社交网络的主要研究都基于图结构，社交网络结构中的各种属性也是基于图的，如节点出度入度，密度，激活概率等[31]。图 2.1 是一个简单的社交网络转化后的图结构，图 2.2 是相应的邻居矩阵的存储形式，图 2.3 是相应的邻接表存储。邻居矩阵中为 1 的表示两个节点是邻居节点，为 0 的表示两个节点之间不存在直接连接。邻接表中表头存储当前节点，当前节点的邻居节点存储在该邻接表的后续链表中[32]。

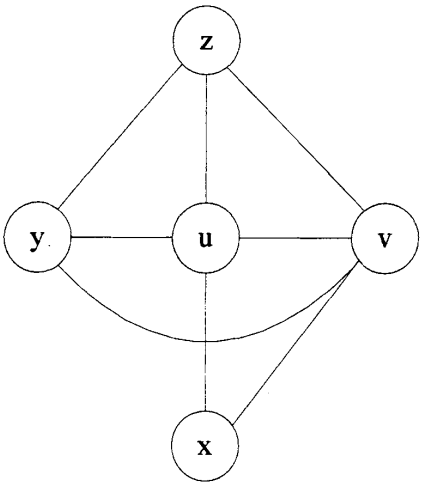

图 2.1 一个简单的社交网络结构图

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{pmatrix}$$

图 2.2 相应的邻居矩阵的存储图

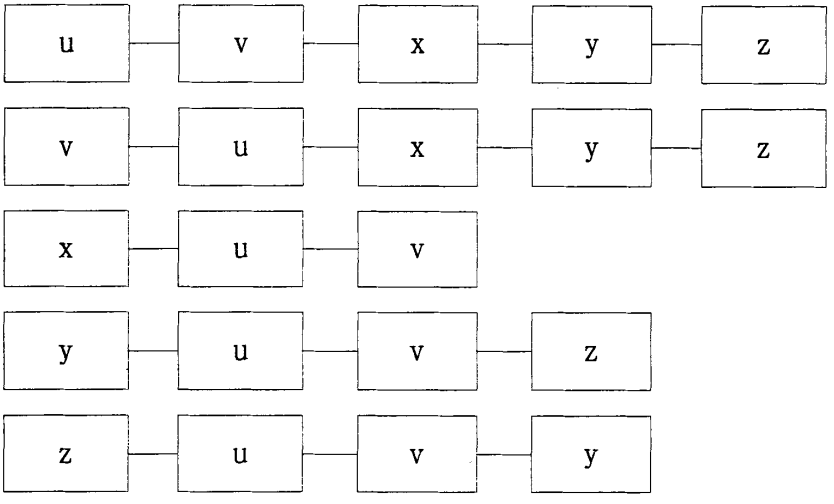

图 2.3 相应的邻接表存储图

2.1.2 社交网络影响力最大化问题

有一个网络公司开发了一个新的 WEB 应用，但是由于资金缺乏或是其他一些原因，无法进行大量的广告宣传。为了让公司产品能进入广大人民的视线，并流行起来，这时公司在社交网络中选择一些影响力大的用户，通常是一些拥有海量粉丝的公共主页或者是拥有海量好友的个人用户，并给他们提供该应用的免费试用机会，希望这些试用用户在肯定该应用的同时，会向自己的好友或粉丝推荐该应用，然后他的朋友或粉丝中肯定该应用的人又向他们的朋友推荐，通过口碑效应，如此不断迭代下去，最后这个应用会被更多的人接受，直到没人再肯定它，整个传播过程才会结束[33]。

由此可以看出最初选择哪些试用用户对整个产品的推广起到了重大的影响。这时如何选择最初的试用者能够使这个应用最终能在网络中被更多的用户肯定，得到更大的传播扩散就成了研究的热点，即社交网络影响力最大化问题[34]。Richardson 和 Domingos[7][8]将这个问题归纳成一个算法问题，即在社交网络中选择 k 个最具影响力的初始节点，由它们去激活整个网络，最终使整个网络中被激活的节点数最大化。

2.2 两种经典的影响力传播模型

目前社交网络影响力最大化算法的研究都是基于特定的影响力传播模型，两种经典的影响力传播模型是独立级联 IC 模型和线性阈值 LT 模型。在详细描述这两种模型之前，有以下几点说明：

1）网络$G(V,E)$中的每一个节点都有 2 种状态，激活状态与未激活状态。一个节点处于激活状态意味着该用户已经认可在网络中传播的信息或产品。

2）只有处于激活状态的节点对它的邻居节点中未被激活的节点才具有影响力，否则不具备该能力。

3）激活过程不可逆，即处于未激活状态的节点可以被激活而转为激活状态，反之不行，处于激活状态的节点将保持其状态不变。

4）当一个节点的邻居节点中越来越多的节点被激活时，该节点被激活的概率也就越大。

5）定义$N(v) = \{u|(u,v) \in E\}$为节点 v 的邻居节点集。

6）定义初始节点集$S \subseteq V$，且$|S| = k$ 。

2.2.1 独立级联 IC 模型

独立级联 IC 模型[9]是一个概率模型，在该模型中节点是否被激活是一个概率事件。该模型中对于种子节点集$S \subseteq V$，影响力在 IC 模型中的传播过程如下：$A_t \subseteq V$表示在步骤$t \geq 0$时，网络中被激活节点的集合，其中$A_0 = S$。在步骤$t+1$，每一个节点$v \in A_t$会去激活它的邻居节点$u \in N(v)$，激活概率为$p(v,u)$，表示节点 u 被节点 v 激活的概率。当$A_t = \emptyset$时，传播过程结束。一旦一个节点被激活，它会自发地去激活其邻居节点，如此迭代直到网络中没有节点被激活。需要注意的是，网络中任意一个处于激活状态的节点有且仅有一次机会去尝试激活它的邻居节点，并且保持自身状态不发生变化，即一直处于激活状态。而且处于激活状态的节点对当前节点进行尝试激活的顺序不会影响该节点最后是否被激活的概率，每次对当前节点的尝试激活行为都是相互独立的。独立级联模型如图 2.4 所示：

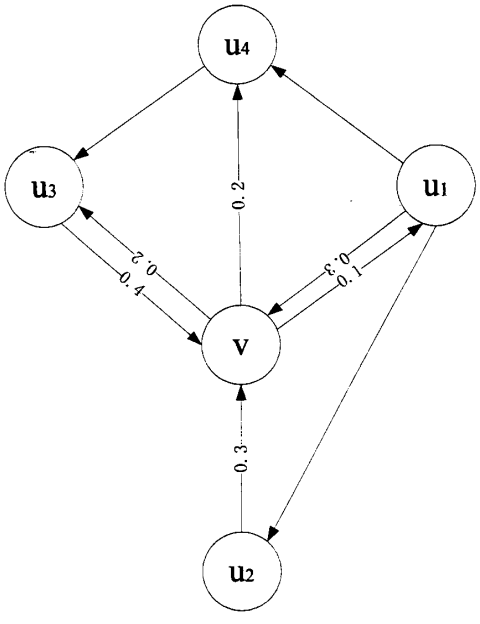

图 2.4 独立级联 IC 模型图

上图为一个有向图，其中节点 v 和其邻居节点间互相连接，且激活概率各不相同。节点 u1、u2 和 u3 如果处于激活状态则分别会去对节点 v 执行激活行为，节点 v 被节点 u1、u2 和 u3 激活的概率分别为 0.3、0.3 和 0.4，激活顺序是随机的，激活顺序的先后不会对最终结果产生影响，即各个邻居节点对节点 v 的激活过程相互独立，互不影响，且每一个邻居节点只有一次机会去尝试激活节点 v。

2.2.2 线性阈值 LT 模型

线性阈值 LT 模型[9]是一种基于节点特异性阈值的模型，也是目前被广泛研究的模型之一。LT 模型中每一个节点自身有一个阈值θ_v表示该节点被激活的难易程度。处于激活状态的节点 v 对其邻居节点$u \in N(v)$的影响定义为$b_{v,u}$，且满足$\sum_{u \in N(v)} b_{v,u} \leq 1$。节点 v 的邻居节点中已经处于激活状态的节点集被定义为$A(v) = \{u | u \in N(v), u$ 为已激活的节点$\}$。当邻居节点中的处于激活状态的节点对该节点的影响力之和大于等于该节点自身的阈值时，该节点被激活，即当满足$\sum_{u \in A(v)} b_{u,v} \geq \theta_v$条件时，节点 v 被激活。与 IC 模型不同的是 LT 模型中的这种影响是可以被累积的，并不是一次性的，因此即使此次迭代中节点没有被激活，但是影响会被累积下来，当后续其他邻居节点继续尝试激活该节点时，之前累积的影响就会同时起作用。影响力在 LT 模型中的传播过程如下：$A_t \subseteq V$表示在步骤$t \geq 0$时，网络中被激活的节点集合，其中$A_0 = S$。在步骤$t+1$，每一个节点$v \in A_t$会去激活它的邻居节点$u \in N(v)$。当节点阈值条件满足时，节点被激活，并继续传播影响力，否则影响力被累积下来。整个过程不断迭代重复，当$A_t = \emptyset$时，传播过程结束。线性阈值模型如图 2.5 所示：

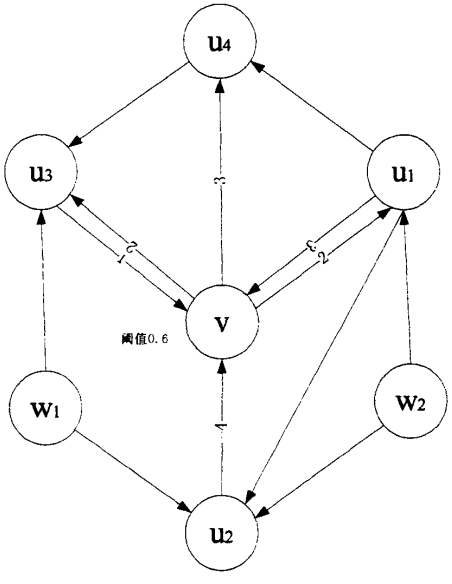

图 2.5 线性阈值 LT 模型图

上图为一个有向图，节点 v 的邻居节点中节点 u1、u2 和 u3 会对节点 v 产生影响，这三个节点中处于激活状态的节点对节点 v 的影响累计只要大于节点 v 自身的阈值 0.6，节

点 v 就会被激活。线性阈值模型中，也不考虑激活顺序对最终结果的影响，而且各个邻居节点对节点 v 的影响力是互相累计的。

2.3 常用的影响力最大化算法

社交网络影响力最大化问题就是在网络中寻找一个子集$S^* \subseteq V$且$|S^*| = k$，使得网络中最终被节点集S^*激活的节点数最大化[35]。目前已有很多学者在社交网络影响力最大化算法研究这个领域做出了贡献，提出了各自的算法。以下是对目前已有的一些常用的经典算法的介绍。

2.3.1 爬山贪婪算法

在文献[9]中，Kempe 和 Kleinberg 证明了社交网络影响力最大化问题是一个 NP-hard 问题，并基于 IC 和 LT 模型提出了一种爬山贪婪(Greedy)算法，该算法能够保证在 1-1/e 范围内接近最优解。该算法通过 k 个循环，每个循环计算所有未激活节点的边际影响范围，然后根据节点边际影响范围，挑选出当前最具影响力的节点作为种子节点，最终组成种子节点集。

为了说明算法，我们做如下说明：

1. 定义$\sigma_I(S_i)$为种子节点集S_i的影响范围，即整个影响力传播过程结束后，网络中最终被种子节点集S_i激活的所有节点数总和。
2. 定义$\sigma_I(u|S_i) = |\sigma_I(\{u\} \cup S_i)| - |\sigma_I(S_i)|$为节点 u 相对于$S_i$的边际影响范围，即网络中被节点 u 激活的所有节点数总和。

由于需要选出 k 个种子节点，所以整个选择过程分为 k 个循环，每个循环选出一个当前最具影响力的节点。初始化$S_0 = \phi$，在每一个循环中 Greedy 算法计算出每一个未激活节点的边际影响范围$\sigma_I(u|S_i)$，选出当前最具影响力最大的节点$u = \arg\max \sigma_I(u|S_{i-1})$，并将其纳入种子节点集$S_i = S_{i-1} U \{u\}$。通过 Greedy 算法能够保证种子节点集的影响范围在 1-1/e 范围内接近最优解，是目前为止能获得最好影响范围的算法。但是由于每个循环后，网络中节点状态发生变化，该算法需要重新计算所有处于未激活状态节点的边际影响范围$\sigma_I(u|S_i)$，这使该算法的时间复杂度过高，运行十分耗时。对于一个百万节点级别的在线社交网络，Greedy 算法需要消耗几天或者更多的时间才能完成计算，这显然无法满足大型社交网络的需求。

以下是 Greedy 算法的伪代码：

Algorithm Greedy: Greedy (G = (V,E),k)

1: initialize $S = \phi$

2: for i = 1 to k do

3: select $u = \text{argmax}_v\{\sigma_I(v|S)|v \in V \setminus S\}$

4: $S = S \cup \{u\}$

5: end for

6: output S

第 1 行初始化种子节点集 S，设置其为空；

第 2~5 行为整个选择过程，分为 k 个循环，每个循环计算所有处于未激活状态节点的边际影响范围，选出一个影响力最高的节点作为种子节点；

第 6 行输出结果，种子节点集 S。

2.3.2 度中心启发式算法

由于社交网络影响力最大化算法是 NP-hard 问题，而 Greedy 算法在时效性上的表现不足以满足需求，因而一般会采用启发式算法来解决。度中心(Degree)算法是一种经典的启发式算法[20]。一个节点的邻居节点的数量被称为节点度数，这是 Degree 算法的核心。一个节点的度数越高，说明该节点的邻居节点越多。一个节点的度定义为：

$$d_v = \sum_{i=1}^{n} \sigma(u_i, v) \tag{2.1}$$

当$(u_i, v) \in E$，则 $\sigma(u_i, v) = 1$；当$(u_i, v) \notin E$，则$\sigma(u_i, v) = 0$。类似于一个微博用户的粉丝数量，拥有的粉丝数量越多，该用户的影响力也越大。Degree 算法基于这样的理念，认为节点度数越大，节点影响力越大，因而在网络中选取度数最高的前 k 个节点作为种子节点。

Degree 算法在时效性上表现优越，算法运行时间远少于爬山贪婪算法，因为该算法中种子节点的选取过程只是对网络中节点度数统计排序再选出前 k 个度数最高的节点，但是该算法在影响范围上效果并不好，这是因为 Degree 算法只是考虑了节点度数，而忽略了影响力传播模型中影响力的传播规则、节点间的激活概率、影响力的扩散过程等其他因素的约束。尽管如此，Degree 算法目前还是被当作以优化影响范围为目标的启发式优化算法的基础。

以下是 Degree 算法的伪代码：

Algorithm Degree: Degree(G = (V, E), k)

1: initialize $S = \phi$

2: foreach v do

3: $d_v = \sum_{i=1}^{n} \sigma(u_i, v)$

4: end for

5: for i = 1 to k do

6: select $u = \text{argmax}_v\{d_v | v \in V \setminus S\}$

7: $S = S \cup \{u\}$

8: end for

9: output S

第 1 行初始化种子节点集 S，设置其为空；

第 2~4 行通过循环逐一计算网络中的节点影响力（节点度数）；

第 6~8 行为整个选择过程，分为 k 个循环，每个循环选出一个影响力最高的节点作为种子节点（无需重新计算节点影响力）；

第 9 行输出结果，种子节点集 S。

2.3.3 其他算法

目前传统的社交网络影响力最大化算法主要包括两大类，一类是影响范围接近最优的爬山贪婪算法，另一类是时效性优越的启发式算法。现有的很多相关研究也都是围绕对贪婪算法的时效性优化和对启发式算法的影响范围优化展开的。一些研究者提出了一种新的想法，即以社区挖掘算法为基础的一类新算法[18][19][36][37]。该类算法首先通过社区发现或者社区挖掘算法将社交网络分成若干个小的社区，然后分别在各个社区中选取种子节点。

如图 2.6 所示，一个完整的社交网路被分成 3 个小社区，然后分别在这 3 个较小的网络中使用影响力最大化算法来选择种子节点，根据种子节点数量和划分出来的社区个数来分配每个社区内选择多少个种子节点，这需要考虑最优资源分配问题[38]。这一定程度上解决了种子节点堆簇的问题，使种子节点的覆盖范围最大化，但是覆盖范围最大化并不等于最终被激活的节点数量最大化，而且这种算法的效果还依赖于社区挖掘算法对网络的分割效果。所以这里主要介绍几种传统的基于全局网络的影响力最大化算法，包括贪婪算法和

启发式算法。

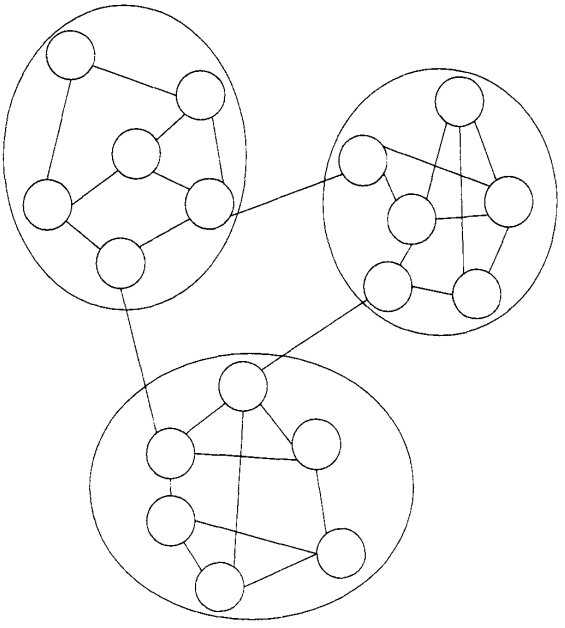

图 2.6 社交网络分割图

集合覆盖贪婪算法中认为一旦一个节点被选作种子节点，其邻居节点应该被标记为“covered”，因为该算法认为这些节点已经处于种子节点的影响力覆盖范围内，这意味着这些节点应该被排除出种子节点的候选范围，每个循环中种子节点只在标记为“uncovered”的节点中选择，整个选择过程持续 k 个循环。但是这也存在问题，在影响力传模型中，被覆盖并不意味着被激活，而且还有一种特殊情况存在，即选择过程还未完成时，网络中已经不存在“uncovered”的节点，所以该算法取得的效果并不理想[39]。

CELF 算法[10]是一种基于爬山贪婪(Greedy)算法的改进算法，该算法利用独立级联(IC)模型的次模特性在保证影响范围的同时减少了 Greedy 算法的计算量，缩短了时耗。IC 模型的次模特性即是：若种子节点集 S 的大小越小，则向 S 中添加一个种子节点 u 时，节点 u 对影响范围的增量影响就越大。次模函数为：

$$f(S\cup\{u\}) - f(S) \geq f(T\cup\{v\}) - f(T) \tag{2.2}$$

利用此特性，每次选择种子节点的时候无需计算所有节点的增量影响$\sigma_I(u|S_i)$，因为节点的增量影响在影响力传播过程中只减不增，所以在上一个循环中增量影响就小于第二高增量影响的节点在本次循环中就无需重新计算各自的增量影响，这节约了大量的计算时间。尽管在时效性上比 Greedy 算法有了很大的进步，但是依然无法满足大型社交网络对于算法时效性的需求。CELF++算法是对 CELF 算法的优化算法，该算法进一步避免了 CELF 算

法中不必要的重复计算，使其运行速度比 CELF 算法快出 35%~55%[11]。

NewGreedy 算法也是一种优化的 Greedy 算法，该算法利用 IC 模型的激活特性删除网络中对影响力传播没有影响的边，构成局部小规模网络作影响力传播，这样就减少了算法的计算量，由此降低了算法的时间复杂度。MixGreedy 算法则是将选择过程分为两个部分，第一部分使用 NewGreedy 算法来选取第一个种子节点并计算影响力传播，第二部分使用 CELF 算法来选取剩余种子节点，结合了这两种算法使得算法整体的时间复杂度降低，但是依然无法满足大型社交网络对于算法时效性的需求[12]。

相对而言，由于启发式算法能更好的适应大型社交网络，为当前研究的主要方向。

Degree Discount算法认为，考虑节点v是否被选作种子节点时，首先需要统计该节点邻居节点中的已经是种子节点的节点数量。如果该节点的邻居节点中有节点u已经是种子节点，那么节点v的出度就应该被“打折”。因此需要在每次选取一个种子节点时，更新网络中该节点的邻居节点的出度，即

$$dd_v = d_v - 2t_v - (d_v - t_v)t_v p \tag{2.3}$$
$$v \in N(u), v \in V \backslash S$$

其中dd_v为节点 v“打折”后的出度，d_v为节点 v 原来的出度，t_v为节点 v 的邻居节点中种子节点的个数，p 为节点 v 被邻居节点 u 激活的概率。实验证明，这种算法相对于度中心算法等传统启发式算法在影响范围上有了巨大的突破，而且在时效性上也表现优越[12]。

Diffusion Degree 算法提出的潜在影响力重新定义了节点影响力，在计算节点影响力的时候将该节点的邻居节点的影响力也进行加权计算。由此根据节点综合影响力来判断节点是否被选作种子节点会更合理。其影响范围优于 Degree Discount 算法。

MIA 算法是基于 Maximum Influence Arborescence(MIA)模型[40]。该算法认为网络中的节点 u 能够激活节点 v 的条件是在节点 u 到 v 的路径上所有的节点都被激活，这条路径被称为 Maximum Influence Paths(MIP)。MIA 算法通过 Dijkstra 最短路径算法计算网络中每一对节点间的 MIP，除了激活概率小于阈值θ的 MIP。对于一条路径$P = \{u = p_1, p_2, \cdots, p_m = v\}$上的节点都被激活的概率$pp(P) = \prod_{i-1}^{m-1} pp(p_i, p_i + 1)$，算法定义节点 u 到节点 v 的最具影响力的路径为

$$MIP_G = \text{argmax}\{pp(P) | P \in \Psi(G, u, v)\} \tag{2.4}$$

其中$\Psi(G, u, v)$为网络 G 中所有从节点 u 到 v 的路径。如果$\Psi(G, u, v) = \phi$，那么$MIP_G = \phi$。节点 v 对网络中其他节点的影响力为

$$MIOA(v,\theta) = \bigcup_{u \in V, pp(MIP_G(v,u)) \geq \theta} MIP_G(v,u) \tag{2.5}$$

节点 v 受到其他节点的影响力为

$$MIIA(v,\theta) = \bigcup_{u \in V, pp(MIP_G(u,v)) \geq \theta} MIP_G(u,v) \tag{2.6}$$

然后以节点 v 为根节点构建一个本地树状网络，用来近似影响力的传播过程从而更新网络。同样基于 MIA 模型的 ECE(Efficient Candidates Elimination)算法[41]，认为网络中只有部分节点能成为种子节点的候选节点，该算法提出了一种修剪策略用于修剪网络中不可能成为种子节点的节点。网络中种子节点的候选节点数量远远小于整个网络的节点总数，所以该算法的运行速度远远超过其他算法。该算法的核心是计算出节点影响力阈值，通过这个阈值来决定哪些节点成为候选节点。种子节点的选择过程在候选节点集中进行，如果候选节点集的大小正好为 k，那么该候选节点集就是种子节点集。

HPG(Hybrid Potential-Influence Greedy)算法是一种混合式的算法[42]，该算法基于 LT 模型，并利用 LT 模型的影响力可“积累”特性，将整个选择过程分为 2 个阶段。第 1 阶段是启发阶段，依次选取网络中潜在影响力 PI(Potential Influence)值最大的节点。节点 PI 值的定义如下：

$$inf(u) = \sum_{v \in N(u), v \notin A(u)} b_{uv} \tag{2.7}$$

$$PI(u) = outDegree(u) + \left(1 - e^{-inf(u)}\right) \tag{2.8}$$

N(u)为节点 u 的邻居节点集，A(u)为节点集N(u)中种子节点的集合，b_{uv}为节点 u 对节点 v 的影响力，inf(u)为节点 u 对其所有邻居节点的影响力之和，即节点 u 的影响力，outDegree(u)是节点 u 的出度。在选择过程中，如果节点出度相同，该算法会选择影响力较大的节点做种子节点。由于 PI 值的计算基本不消耗时间，所以在节约了时耗的同时在网络中积累了大量的潜在影响力；第 2 阶段是贪婪阶段，由于经历了第 1 阶段后，网络中大量节点被激活，大大减少了未激活节点的数量，贪婪算法的计算量减少了很多，同时第 1 阶段积累的潜在影响力也会被利用起来。实验证明该算法在影响范围和时间复杂度两方面都取得了明显的效果。

TBH(Threshold-Based Heuristic)算法是一种基于 HPG 算法的优化算法[43]。该算法认为 HPG 算法依然有可改进的空间，即在算法的启发阶段，计算节点影响力的时候没有考虑到每个节点激活阈值不同的情况，而只是将对所有的邻居节点的影响力做累加。TBH 算法定义节点 v 的潜在影响节点数 PIN(v)，

$$PIN(v) = \begin{cases} PIN(v) - \frac{b_{uv}}{\theta_v - r_v}, \frac{b_{uv}}{\theta_v - r_v} < 1 \\ PIN(v) - 1, \frac{b_{uv}}{\theta_v - r_v} \geq 1 \end{cases} \tag{2.9}$$

使用 PIN(v)在启发阶段快速选择种子节点，并使用一个 cover 数组来记录激活过程中覆盖到的节点用于更新节点阈值和 PIN(v)，从而使算法在启发阶段拥有更好的影响范围。实验证明，该算法在启发因子很小甚至完全不采用贪婪算法阶段的情况下就能接近 Greedy 算法的影响范围，同时在影响范围优于 HPG 算法，时效性则与其类似。

其他取得显著效果的启发式算法还有 PMIA[30]、IRIE[44]和 SIMPATH[45]等算法[46][47][48][49]，本文不再详细描述。

2.4 本章小结

本章主要介绍了相关的理论知识，第一节介绍了社交网络的概念、社交网络影响力最大化问题的定义；第二节描述了两种经典的影响力传播模型；第三节介绍了已有的一些常用的社交网络影响力最大化算法，包括爬山贪婪算法、度中心启发式算法以及一些其他算法。目前大部分研究都是围绕对贪婪算法的时效性优化和对启发式算法的影响范围优化展开的，其中后者更是现在的主流研究方向，因为启发式算法能更好地适应大型社交网络。

第三章 考虑激活概率的社交网络影响力最大化改进算法

基于IC模型的Diffusion Degree算法重新定义了节点影响力，在计算节点影响力的时候综合考虑该节点的邻居节点的影响力，并对其进行加权计算，更科学的影响力定义提高了种子节点集质量，但是该算法依然有改进的空间。本文对Diffusion Degree算法进行了详细的分析，针对其不足进行了改进，考虑了节点间潜在影响力的实际有效性提出了IDD(Improved Diffusion Degree)算法，进一步提高了算法的影响范围。

3.1 Diffusion Degree算法简介

IC模型中对于节点影响力的判断一般基于节点出度和节点对邻居节点的激活概率这两个属性，一般认为节点出度越高，节点影响力就越高，如果综合考虑节点间的激活概率，在节点出度的基础上对节点间激活概率做乘积后再做累加处理[50]。Diffusion Degree算法提出了节点潜在影响力（即节点邻居节点的影响力）的概念，重新定义了节点影响力，认为邻居节点影响力也是节点影响力的重要参考指标，因此Diffusion Degree算法在计算节点影响力时将该节点的邻居节点的影响力也进行加权计算，得出节点最终的综合影响力[51]。

这样解决了以下的一种问题，即有些节点虽然自身影响力（根据原有的节点影响力定义计算所得）不足以作为种子节点，但是该节点的邻居节点都拥有相当的影响力；而另外一些节点自身拥有相当的影响力，但是该节点的邻居节点的影响力都不高。根据原先的节点影响力定义会选择后者作为种子节点，但是Diffusion Degree算法综合考虑了邻居节点的影响力这一因素，将节点潜在影响力加权计算以后，前者的综合影响力反而超过了后者，这时原先的种子节点选择策略并不适用，而应该根据新定义的节点综合影响力选择前者作为种子节点。如图3.1所示，节点v的出度为50，其邻居节点的出度都为100；节点u的出度为100，但是其邻居节点的出度都只有30，使用Diffusion Degree算法中的种子节点选择策略综合比较，我们发现节点v的综合影响力更高，更适合作为种子节点。

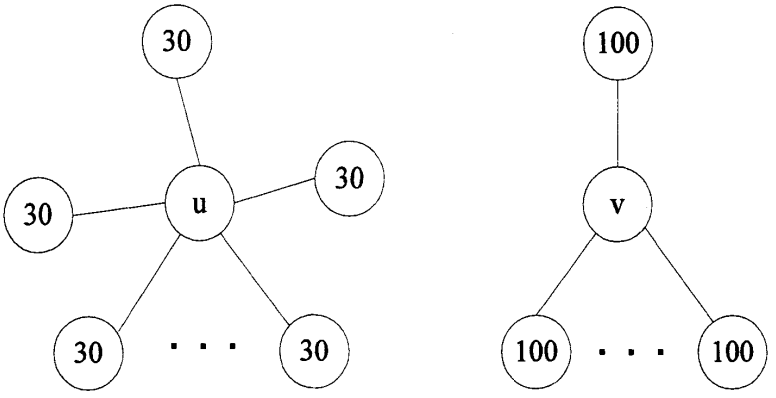

图3.1 节点u和v的综合影响力对比图

不难发现Diffusion Degree算法中提出的这种根据节点综合影响力来判断节点是否被选作种子节点的策略相对于仅仅考虑节点出度和激活概率的算法会更科学合理，其所得的种子节点集质量也更高。

度中心算法这样定义节点影响力：

$$C_D(v) = \sum_{i=1}^{n} \sigma(v, u_i) \tag{3.1}$$

当$(v, u_i) \in E$，则 $\sigma(v, u_i) = 1$；当$(v, u_i) \notin E$，则$\sigma(v, u_i) = 0$。$p(v, u_i)$是节点u_i被节点v激活的概率。在影响力传播过程中，考虑到节点间的激活概率，节点v的影响力为：

$$C_{DD}^{'}(v) = p(v,u) * C_D(v) \tag{3.2}$$

在Diffusion Degree算法中，节点影响力被重新定义，需要综合考虑节点自身影响力和节点潜在影响力（即当前节点邻居节点的影响力）两部分。N(v)为节点v的邻居节点集，节点v的邻居节点集的影响力为：

$$C_{DD}^{''}(v) = \sum_{u \in N(v)} C_{DD}^{'}(u) \tag{3.3}$$

经过加权计算，Diffusion Degree算法中的节点v的综合影响力定义为：

$$C_{DD}(v) = C_{DD}^{'}(v) + C_{DD}^{''}(v) \tag{3.4}$$

$$= p(v, u) * C_D(v) + \sum_{u \in N(v)} C_{DD}^{'}(u)$$

$$= p(v, u) * C_D(v) + \sum_{u \in N(v)} p(u, w) * C_D(u)$$

Diffusion Degree算法首先按照新的节点综合影响力定义计算网络中所有节点的综合

影响力，这个计算过程的时间复杂度为O(|V| + |E|)，|V|是网络中节点总数，|E|是网络中节点间边的数量。而且需要注意的是，在节点v影响力的定义中，我们考虑的是节点v的直接邻居节点对v的综合影响力的贡献，至于邻居节点的邻居节点则不需要继续考虑，因为节点层次越多，激活概率越小。然后进入种子节点的选择，整个选择过程分为k个循环，每个循环选出当前循环中综合影响力最高的节点作为种子节点加入种子节点集S，算法输入社交网络图结构以及种子节点集S的大小k，输出种子节点集S。

Diffusion Degree 算法的伪代码如下：

Algorithm Diffusion Degree: Diffusion Degree($G = (V, E), k$)

1: initialize $S = \phi$

2: foreach v do

3: $C_{DD}(v) = p(v,u) * C_D(v) + \sum_{u \in N(v)} p(u,w) * C_D(u)$

5: end for

6: for i = 1 to k do

7: select $u = \text{argmax}_v\{C_{DD}(v) | v \in V \setminus S\}$

8: $S = S \cup \{u\}$

9: end for

10: output S

第 1 行初始化种子节点集 S，设置其为空；

第 2~5 行通过循环逐一计算网络中的节点综合影响力；

第 6~9 行为整个选择过程，分为 k 个循环，每个循环选出一个综合影响力最高的节点作为种子节点；

第 10 行输出结果，种子节点集 S。

Diffusion Degree 算法和传统的度中心算法基于相同的影响力传播模型，不同的是采用了更科学的节点影响力定义来选择种子节点，从而达到优化种子节点集质量的效果。该算法的作者在两个真实网络的数据集(Twitter[52]和 DBLP Data Set)上进行了实验仿真，其中数据集 Twitter 规模较小，而数据集 DBLP 是大规模网络的数据集。实验中使用不同的激活概率以及不同的种子节点集的大小，对比了 Diffusion Degree、High Degree[53]和 Degree Discount 三种算法的影响范围。实验结果分析证明该算法在保证时效性的基础上，在影响范围上取得了很好的效果，其影响范围明显优于 Degree Discount 和 High Degree 算法，而

Degree Discount 算法是当时影响范围最好的启发式算法之一。

3.2 基于 Diffusion Degree 算法的改进算法 IDD

3.2.1 IDD 算法设计思想

从上述对于 Diffusion Degree 算法的描述中不难发现一个问题，即在考虑节点潜在影响力时，该算法加权计算了该节点的所有邻居节点的影响力。但这明显存在一定的问题，Diffusion Degree 算法没有考虑节点间的激活概率，忽略了节点间潜在影响力的实际有效性。首先，考虑节点间潜在影响力时必须结合节点间激活概率，即节点潜在影响力需与节点间激活概率做乘积后再加权；然后考虑以下情况，当节点 v 的一个邻居节点 u 具有相当的影响力，但是p(v,u)却非常小，这时节点 u 的影响力被加权计算进节点 v 的综合影响力就显得不合理。因为节点 u 不被激活的话，它的影响力就无法对节点 v 的综合影响力做出贡献，所以节点 u 被节点 v 激活的概率过小时，节点 u 的影响力不应该被加权计算进节点 v 的综合影响力。

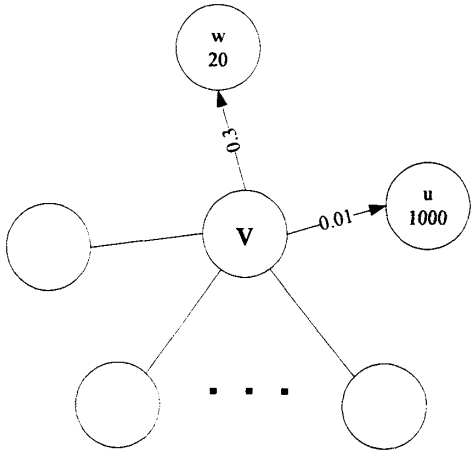

图 3.2 新定义的节点综合影响力图

如图 3.2 所示，节点 v 有若干邻居节点，假设 v 对他们的平均激活概率为 0.3，其中节点 w 被激活的概率正好为 0.3，而节点 w 自身的影响力为 20；另一个节点 u 被激活的概率却只有 0.01，而它自身影响力为 1000。这时根据 Diffusion Degree 算法中节点综合影响力的定义来看，明显节点 u 对综合影响力的贡献已经超过了 w，但是在进行加权计算的时候，节点 u 由于被激活概率远小于平均激活概率，如果不被激活，它发挥自身影响力的机会也就很小，所以节点 u 的影响力不应该被加权计算进节点 v 的综合影响力。

由此得知不是每一个邻居节点的影响力都应该被加权计算进该节点的综合影响力，为了更准确计算节点的综合影响力，应该根据激活概率对邻居节点进行过滤，剔除那些被激活概率过小的邻居节点，这样可以进一步地提高种子节点集的质量。如图 3.3 所示，节点 x 和 u 与节点 v 之间以虚线连接，这是因为节点 x 和 u 被节点 v 激活的概率过低，所以它们的影响力不被计算进节点 v 的综合影响力。

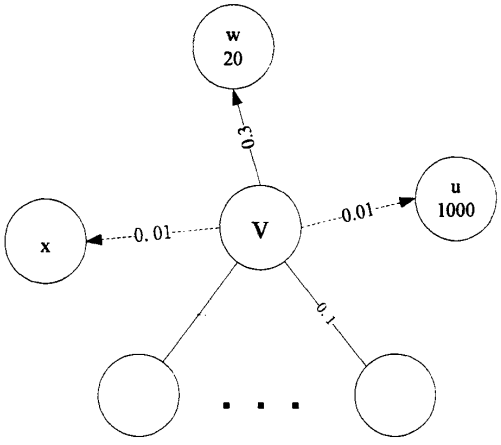

图 3.3 过滤被激活概率过小的邻居节点图

本文针对上述问题提出了Diffusion Degree算法的改进算法IDD算法，本算法定义了一个激活概率阈值θ，

$$\theta = \frac{\sum_{(v,u)\in E} p(v,u)}{|E|} \tag{3.5}$$

从而根据阈值θ对节点v的邻居节点进行进一步的过滤，只有满足条件$p(v,u) \geq \theta$时，才将邻居节点u的影响力计算进节点v的综合影响力，修改后的节点v的综合影响力定义如下：

$$C_{DD}(v) = p(v,u) * C_D(v) + p(v,u) * \sum_{u\in N(v)} p(u,w) * C_D(u), p(v,u) \geq \theta, p(u,w) \geq \theta \tag{3.6}$$

$$= p(v,u) * \left(C_D(v) + \sum_{u\in N(v)} p(u,w) * C_D(u) \right), p(v,u) \geq \theta, p(u,w) \geq \theta$$

同时，IDD算法还综合了Single Discount[54]算法中的设计思想，即考虑到IC模型中，每一个节点有且仅有一次机会去激活邻居节点，而每当一个节点被选作种子节点时，种子节点就会自发地去激活邻居节点，无论邻居节点是否被激活，该次机会都会被使用，相当于连接种子节点与其邻居节点的边已经被“废弃”，这对后续种子节点的选择将产生影响，特别是对其邻居节点的影响力大小。这里我们使用Single Discount算法中的节点出度定义来实时更新节点影响力，而非Degree Discount算法中的。后者考虑节点u在没有被邻居种子

节点激活的概率下对其进行节点出度的更新，而前者是直接减去邻居种子节点的个数，虽然Degree Discount算法似乎考虑更周全，但是本文是从IC模型中节点间有且仅有一次激活机会的角度去看待节点出度的更新问题，认为一旦激活机会被使用，无论节点是否被激活，节点间的边都应该被直接“废弃”，因此本文认为Single Discount算法的设计思想更贴合本算法的考虑角度。

如图3.4所示，当我们考虑节点v是否被选作为种子节点时，发现节点v的邻居节点中有节点u已经是种子节点，这意味着连接节点v和u边$(v,u) \in E$已经被“废弃”，此时我们必须对节点v的出度进行“打折”，将节点v的出度减去邻居节点集中已经是种子节点的节点数量。

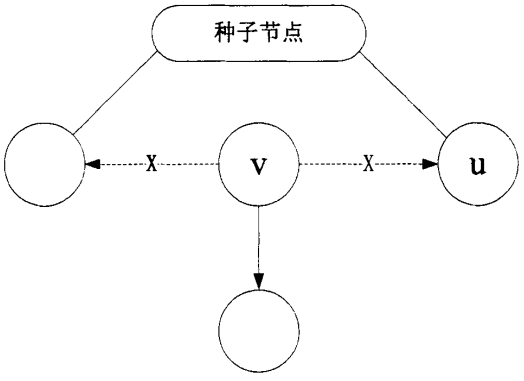

图3.4 节点出度“打折”图

因此在种子节点的选择过程中，一旦一个节点被选作种子节点，我们就更新该节点的邻居节点的出度，这样就更准确地计算了节点综合影响力。我们定义t_v为节点v的邻居节点集中种子节点的个数：

$$t_v = \sum_{u_i \in S}^{n} \sigma(v, u_i) \tag{3.7}$$

节点v“打折”后的出度为：

$$C_D^{'}(v) = C_D(v) - t_v \tag{3.8}$$

最后，节点v的综合影响力为：

$$C_{DD}(v) = p(v,u) * \left(C_D^{'}(v) + \sum_{u \in N(v)} p(u,w) * C_D^{'}(u) \right), p(v,u) \geq \theta, p(u,w) \geq \theta \tag{3.9}$$

$$= p(v,u) * \left((C_D(v) - t_v) + \sum_{u \in N(v)} p(u,w) * (C_D(u) - t_u) \right), p(v,u) \geq \theta, p(u,w) \geq \theta$$

整个种子节点的选择过程分为 k 个循环，每个循环选出一个综合影响力最高的节点作为种子节点加入种子节点集 S，一旦一个节点被选为种子节点就更新网络，重新计算并更新该节点的邻居节点中未激活节点的综合影响力，算法输入社交网络的图结构、种子节点集 S 的大小 k 以及用于过滤邻居节点的阈值θ，输出种子节点集 S。

3.2.2 IDD 算法设计与实现

Improved Diffusion Degree算法流程图3.5如下：

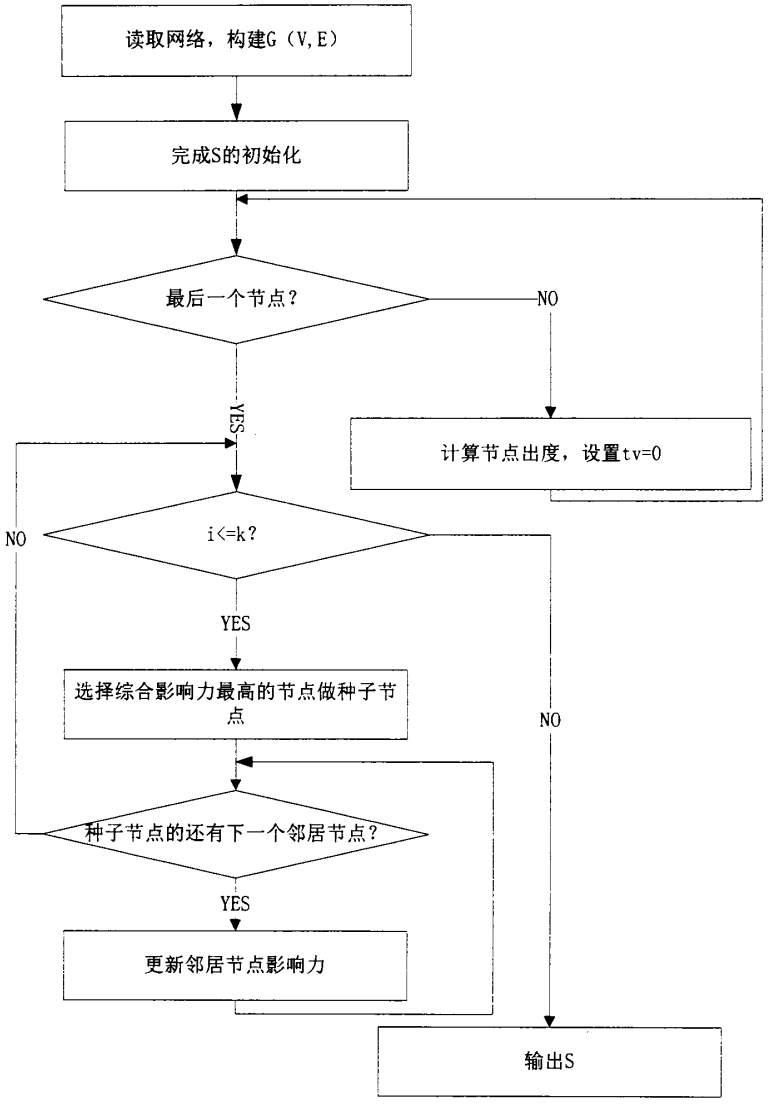

图3.5 IDD算法流程图

在算法实现部分，我们设计了Graph类[55]用来构造社交网络的数据结构，Graph类定义的核心代码如下所示：

```
struct Edge
{    int u,v,c;
     double w1,w2;
 };
class Graph
{
private:
     static bool built;
     static int n;
     static int m;
     static vector<int> degree;
     static vector<int> index;
     static vector<Edge> edges;
     static void qsort_edges(int h, int t);
public:
     static void Build();
     static int   GetN();
     static int   GetM();
     static int   GetDegree(int node);
     static int   GetNeighbor(int node);
     static Edge GetEdge(int node, int index);
     static void BuildIC();
     static void BuildWC();
     ……
     static void Build2IC();
     static void Stats();
};
```

以下是对Graph类中一些重要的数据成员变量和函数的说明：

Graph类中数据成员变量包含静态bool类型的变量built，静态int类型的变量n，静态int类型的变量m，静态数组容器degree(int类型成员)，静态数组容器index (int类型成员)，静态数组容器edges(Edge类型成员)。

bool类型的built表示网络是否被创建，为true表示网络已被创建，为false表示网络未被创建，该变量是为了防止网络被重复创建；int类型的n、m表示节点、节点间边的数量；数组容器degree中存放节点出度(数据类型为int)；数组容器edges中存放Edge类型的节点间的边；每一条边是一个结构体struct Edge，struct内部包含节点u、v、c以及节点u、v相互作用的权重。

Graph类中的成员函数中的方法qsort_edges(int h, int t)用来根据节点边的数量对节点进行排序，方法GetN()用于获取网络中节点的数量，方法GetM()用于获取网络中边的数量，方法GetDegree(int node)用于获取节点node的出度，方法GetNeighbor(int node)用于获取节点node的邻居节点，方法GetEdge(int node, int index)用于获取节点node的边，方法BuildIC()用于创建无向图中的IC模型，方法Build2IC()用于创建有向图中的IC模型，方法Stats()则是用于运行算法，得出统计结果。

Improved Diffusion Degree算法伪代码如下：

Algorithm Improved Diffusion Degree: IDD ($G = (V, E), k, \theta$)

1: initialize $S = \phi$

2: foreach v do

3: $C_D(v) = \sum_{i=1}^{n} \sigma(u_i, v)$

4: $t_v = 0$

5: end for

6: for i = 1 to k do

7: select $u = \text{argmax}_v\{C_{DD}(v) | v \in V \setminus S\}$

8: $S = S \cup \{u\}$

9: for each neighbor v of u and v and $V \setminus S$ do

10: $t_v = \sum_{i \in S}^{n} \sigma(v, u_i)$

11: $C_D'(v) = \sum_{i=1}^{n} \sigma(v, u_i) - \sum_{i \in S}^{n} \sigma(v, u_i)$

12: $C_{DD}(v) = p(v,u) * \left(C_D'(v) + \sum_{u \in N(v)} p(u,w) * C_D'(u)\right), p(v,u) \geq \theta, p(u,w) \geq \theta$

13: end for

14: end for

15: output S

第 1 行初始化种子节点集，设置其为空；

第 2~5 行通过循环逐一计算网络中所有节点的出度，初始化每个节点的邻居节点集中种子节点个数为 0，因为选择过程开始前，网络中不存在种子节点；

第 6~14 行为整个选择过程，分为 k 个循环，每个循环选出一个综合影响力最高的节点作为种子节点，其中第 9~13 行为根据新选择的种子节点更新网络中种子节点的邻居节点的出度，并重新计算节点综合影响力，为下一个循环的选择做准备；

第 15 行输出结果，种子节点集 S。

3.3 IDD 算法实验分析

3.3.1 实验环境

1、硬件平台：联想台式机一台（Intel(R) Core(TM)2 Duo CPU E7500 @ 2.93GHz，4G 内存）

2、操作系统： Windows7 旗舰版 Service Pack 1 (32 位)

3、程序开发平台： Microsoft Visual Studio 2010

Visual Studio 是目前微软公司主推的一款 Windows 平台下的程序开发环境。在 Visual Studio 2010 版本中编辑器被重新构建，这使得该平台的开发环境更加灵活、功能更加丰富。Visual Studio 2010 还提供了多显示器功能，开发者可以同时查看界面设计器、程序代码以及数据库结构。Visual Studio 2010 还提供了大量的最新功能用来支持微软的最新软件产品（如 Silverlight5），这给开发人员带来了全新的技术体验和更多的技术创新以应对云计算时代的机遇和挑战。这个版本中还新增了对于最新的 C++ 标准的支持，它的 IDE 性能也得到了增强，这使得开发人员能够真正提升自身工作效率[56]。

4、开发语言： C++

C++是当今最流行的程序开发语言之一，它基于 C 语言，秉承了 C 语言的灵活性、可移植性等精华，同时又融合了面向对象的开发理念。C++中引进了类和对象的概念，将程序设计从面向过程转变为面向对象编程，具有强大的编程功能。面向对象的编程思想使得程序设计中的建模变得更为方便，程序员能够通过类的设计轻松地构建构出程序设计中需要解决的问题，这也是 C++功能强大之处。C++编写出的程序不仅结构清晰，拥有很强的

可读性，而且代码质量高易于扩展，适用于各种系统软件和应用软件的程序设计。C++是一种静态类型语言，能够支持多种程序设计风格，包括过面向对象程序设计、程序化程序设计、泛型程序设计和数据抽象化设计等[57]。

3.3.2 实验数据集

本文选取了3个真实网络的数据集(WikiVote、Epinions和DBLP)，这三个数据集的规模如表3.1所示：

表3.1 数据集WikiVote、Epinions和DBLP的数据统计表

数据集	WikiVote	Epinions	DBLP
节点数量	7.1K	75K	655K
边的数量	101K	655K	2.0M
度平均值	26.6	13.4	6.1
度最大值	1065	3079	588

数据集 WikiVote 是一个来自 Wikipedia 的投票历史网络(投票选举管理员)的数据集，其中节点代表 Wikipedia 网络中的用户，若节点 u 投票给了节点 v（即认为节点 v 对 u 有一定的影响力)，那么存在一条从节点 u 到 v 的有向边，与此数据集对应的是个有向无权图[58]。

数据集 Epinions 是一个来自消费者评论网站(用户可以信任或者不信任其他用户，这种信任和不信任关系构成一个信任关系网络)的关系信任网络的数据集，节点代表网站的会员，一条由节点 u 到 v 的有向边意味着 u 信任 v（即 v 对 u 产生正影响)或者不信任 v（即 v 对 u 产生负影响)。与此数据集对应的是个带符号的有向图[59]。

数据集 DBLP 是一个来自 DBLP 计算机科学参考文献网络的数据集。这是一个数据规模更大的协作网络。节点代表 DBLP 网络中的作者，如果节点 u 和 v 有过合作，那么认为节点 u 与 v 是邻居节点，且互相影响。与此数据集对应的是个带符号的有向图[60]。

在这 3 个数据集上，本文分别对 IDD 算法，爬山贪婪(Greedy)算法和另外 2 种启发式算法 PMIA 和 Diffusion Degree 算法进行了评估，其中 PMIA 是目前启发式算法中表现最好的算法之一。由于 Greedy 算法能保证 1-1/e 范围内接近最优解，同时 PMIA 算法在影响范围上也非常接近 Greedy 算法，而 IDD 算法是 Diffusion Degree 算法的改进算法，因此以 Greedy 算法的影响范围作为影响范围的基准，以 PMIA 和 Diffusion Degree 算法的影响范围作为比较对象。同时以 PMIA 和 Diffusion Degree 两种启发式算法的时效性作为时效性的基准和比较对象。算法中阈值θ的取值会对算法的总体性能产生影响，它是作为一个平

衡算法影响范围和时效性的因素。本文中为了兼顾算法影响范围和时效性，θ采用的是网络中节点间激活概率的均值。

3.3.3 实验结果及分析

本文主要是在影响范围和时效性两方面对4种算法进行了对比。Greedy算法耗时太多，需要三天的时间，这与启发式算法的运行时间不在一个数量级上，所以只将其影响范围作为评估基准，选择另外2种启发式算法的运行时间作为时效性的对比基准。

（一）影响范围

影响范围（即种子节点集的质量）的评估主要是基于最终网络中被激活的节点数。IDD算法首先完成网络G = (V, E)的初始化。然后在图G = (V, E)上分别采用4种算法进行仿真，计算出种子节点集S。三个数据集中，WikiVote数据集规模最小，如图3.6所示，IDD算法在影响范围上明显优于Diffusion Degree算法，与PMIA算法的影响范围基本相当，并接近Greedy算法的影响范围，这得益于IDD算法中对节点综合影响力的科学定义以及对网络中节点综合影响力的实时更新。（以下图中X坐标为种子节点集S的大小，Y坐标为网络中被激活节点数量。）

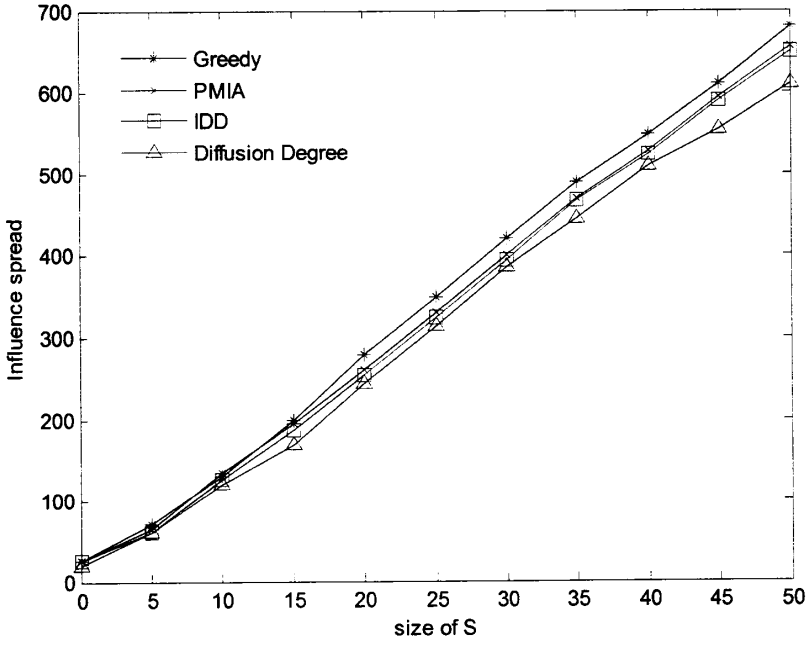

图3.6 WikiVote上的种子节点集质量对比图

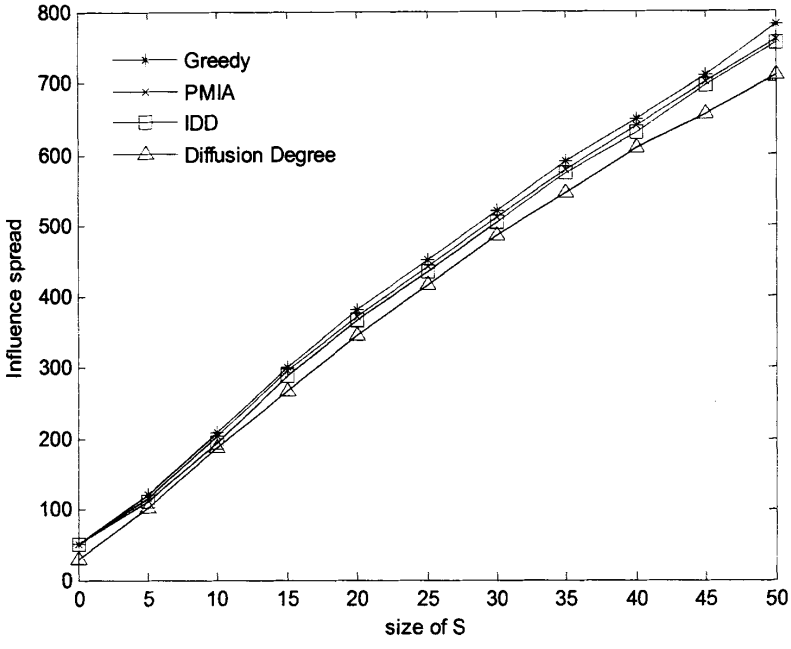

图3.7 Epinions上的种子节点集质量对比图

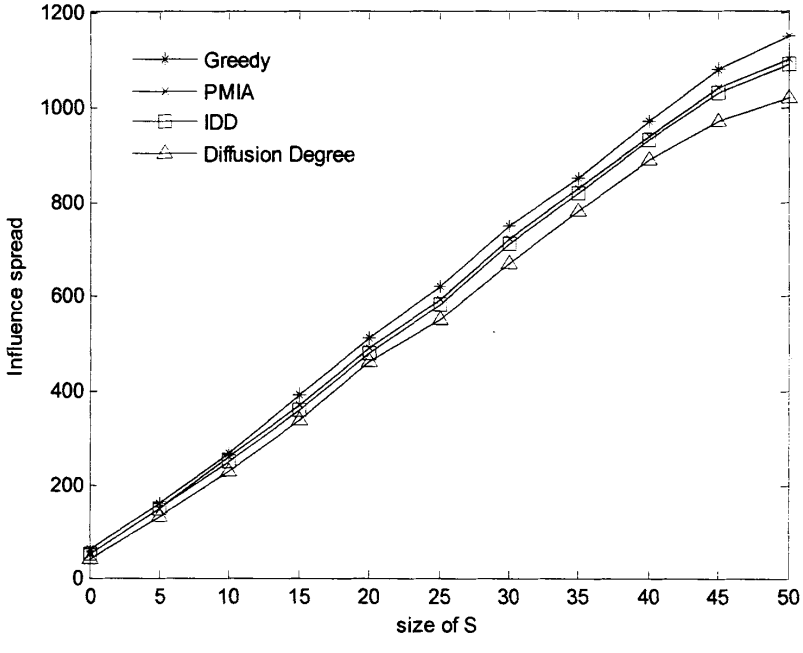

图3.8 DBLP上的种子节点集质量对比图

与数据集WikiVote相比，Epinions规模较大。如图3.7所示，在数据集Epinions上，IDD算法根据激活概率阈值对邻居节点进行了过滤，进一步考虑了节点潜在影响力的实际有效性，使节点综合影响力的计算更加科学，并且每选一个种子节点就更新网络中节点的综合影响力，所以IDD算法得到的种子节点集的质量要优于Diffusion Degree算法，其影响范围与PMIA算法基本相当，也接近Greedy算法的影响范围。

数据集DBLP在规模上大于Epinions，实验结果如图3.8所示，在数据集DBLP上，IDD算法也得到了一致的结果，其影响范围也是与PMIA算法类似，同时接近Greedy算法的影响范围，而且相较于Diffusion Degree算法有较明显的优势。在规模更大的数据集DBLP中，随着种子节点集S的大小k的增加这种优势更加明显，说明IDD算法对于大规模网络的适应能力更强。

（二）时效性

时效性即算法找出种子节点集所需运行时间。各算法的平均运行时间如表3.2所示，Greedy算法的运行时间太久，与其他启发式算法的运行时间不在一个数量级，因此不在比较范围内。数据集WikiVote规模较小，各算法在此数据集上的运行时间差距不大，且本文更关注IDD算法在大型网络数据集上的表现，因而列出了它与各算法在数据集Epinions和DBLP上的平均运行时间的对比数据。由于IDD算法在对节点综合影响力计算时，进行了进一步的过滤，还综合了Single Discount算法的更新节点影响力的操作，IDD算法的运行时间相较于Diffusion Degree算法有一点提高，但是这种程度的提高仍然在可接受范围内，而且相较于PMIA算法的运行时间依然具有相当的优势，拥有更好的时效性。在数据集Epinions上，Diffusion Degree和IDD算法的运行时间差距并不大（IDD算法的运行时间略多），但是都要优于PMIA算法；在规模更大的数据集DBLP上，PMIA算法影响范围最优，但是其运行时间大幅度增加，相对而言IDD算法的运行时间增幅较少，但是也能获得与PMIA算法十分接近的影响范围，因此综合影响范围与时效性两方面而言，IDD算法是更好的选择。

表3.2 各算法平均运行时间对比表

算法	Epinions	DBLP
Greedy	-	-
PMIA	15s	3min
Diffusion Degree	5s	15s
IDD	6s	20s

3.4 本章小结

Diffusion Degree 算法考虑并不完备，还有进步的空间，本章首先介绍了 Diffusion Degree 算法的核心思想，并给出了该算法的相关定义及伪代码；然后提出了该算法存在的问题，并针对该问题对算法进行了改进，提出了 IDD 算法。论文具体描述了 IDD 算法的设计思想、算法伪代码及具体实现，最后对实验结果进行了分析。实验结果显示 IDD 算法在影响范围上明显优于 Diffusion Degree 算法，并且已经接近 PMIA 算法的影响范围，而 PMIA 算法是目前效果最好的启发算法之一。IDD 算法的运行时间略高于 Diffusion Degree 算法，但是总体而言时效性仍然较好，在可接受范围内。因此 IDD 算法在牺牲少量时效性的基础上影响范围得到了进一步的提升。

第四章 考虑活跃度的社交网络影响力最大化算法研究与设计

IC 模型是对实际社交网络的一种抽象，模型中节点的主要属性是节点度与节点间的激活概率，节点影响力的定义也都是基于这两个属性。本文对 IC 模型进行了扩展，引入节点活跃度属性，使节点影响力的定义更科学，并提出了基于节点活跃度的社交网络影响力最大化算法。

4.1 问题提出

传统 IC 模型中对于节点影响力的判断一般基于节点出度和节点对邻居节点的激活概率这两个属性，本文考虑到现实中的两类情况：1）空有影响力却不发挥的情况，部分影响力很高的用户，但是由于自身活跃度低（即用户虽然拥有海量粉丝却很少发表状态或分享信息），没有真正发挥自身影响力来传递信息，比如阿里巴巴总裁马云在新浪微博上拥有海量粉丝，但是马云作为一名微博用户一年仅仅发布了一条状态，没有发挥自身的影响力；2）接受了朋友的推荐却没有继续向自己的朋友继续推荐的情况，例如一个用户接受了朋友推荐的想法或产品，但是并不会 100%去向自己的朋友推荐这个想法或产品，用户有可能只是自己使用该产品，或者是自己默默地收藏了该产品。

第一种情况会对种子节点的选择产生影响，我们选择的种子节点必须保证影响力能被有效地传播下去，所以活跃度低的节点不适合作为种子节点，必须选择活跃度高的节点；第二种情况会对影响力的传播过程产生影响，因为当节点被激活时如果由于活跃度低下而没有去对邻居节点执行激活行为，那么影响力就不会继续传播下去，这种情况关系到整个影响力传播过程的终止，也会对节点影响力的计算产生影响。因此考虑到以上两种情况，本文将节点活跃度属性引入 IC 模型，提出了 IC 模型的扩展模型，AIC(Activity Independent Cascade)模型。由于 AIC 模型中新增了节点活跃度这一属性，该模型中的节点激活规则和影响力传播过程都发生了相应的变化，在 4.2 节中进行详细讨论。本文中的节点活跃度指的是处于激活状态的节点对其邻居节点执行激活行为的概率，这必须与节点间的激活概率区分开来。

4.2 考虑活跃度的 IC 模型的扩展模型 AIC

传统 IC 模型中认为处于激活状态的节点就会自发地去尝试激活其邻居节点，即一旦节点 v 处于激活状态，节点 v 就以 100%的概率去对其邻居节点执行尝试激活行为，而邻居节点是否被激活则由节点间的激活概率决定。但是这明显与现实不符，以人人网用户为例，如图 4.1 所示，

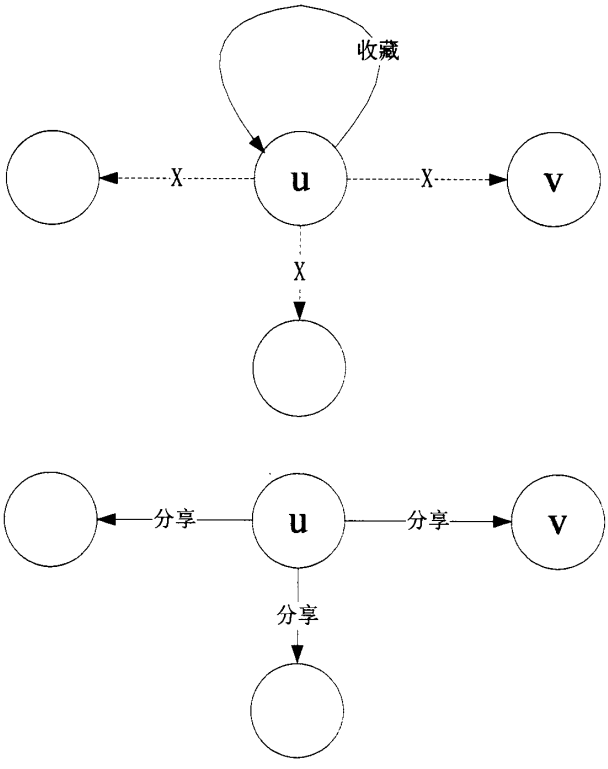

图 4.1 用户“收藏”与“分享”信息对比图

用户在看到一条新鲜事时，如果喜欢该信息可以选择分享或者收藏信息。我们将现实中的“分享”与“收藏”分别对应到影响力模型中处于激活状态的节点是否去执行激活行为的两种情况，选择“分享”信息就是自身被激活并且对邻居节点执行激活行为，选择“收藏”就是仅仅自身被激活，并不对邻居节点执行激活行为。除了人人网用户，新浪微博等其他社交网站的情况也是如此。

本文定义的活跃度就是指用户认可信息的同时选择“分享”信息的概率，活跃度越高则选择“分享”的概率越高，反之选择“收藏”的概率越高。影响力的传播则是需要用户更多地“分享”信息。对应到影响力模型中，节点活跃度就是节点被激活后同时对其邻居

节点执行激活行为的概率，节点活跃度越高，执行激活行为的概率越高。由于 AIC 模型引入了活跃度这个属性，该模型中节点的激活规则也发生了改变，即处于激活状态的节点首先根据自身活跃度，判断是否执行激活行为。若执行则以激活概率去尝试激活邻居节点，否则不执行激活行为，影响力不继续向下传播。IC 模型与 AIC 模型的激活规则对比如图 4.2 所示：在 IC 模型中，节点 v 一旦被激活就 100%地去激活节点 u；而在 AIC 模型中，假设节点 v 的活跃度为 0.8，代表节点 v 自身被激活时，由于自身活跃度较高，有 80%的概率会对其邻居节点执行激活行为，仅 20%的概率不去执行激活行为。

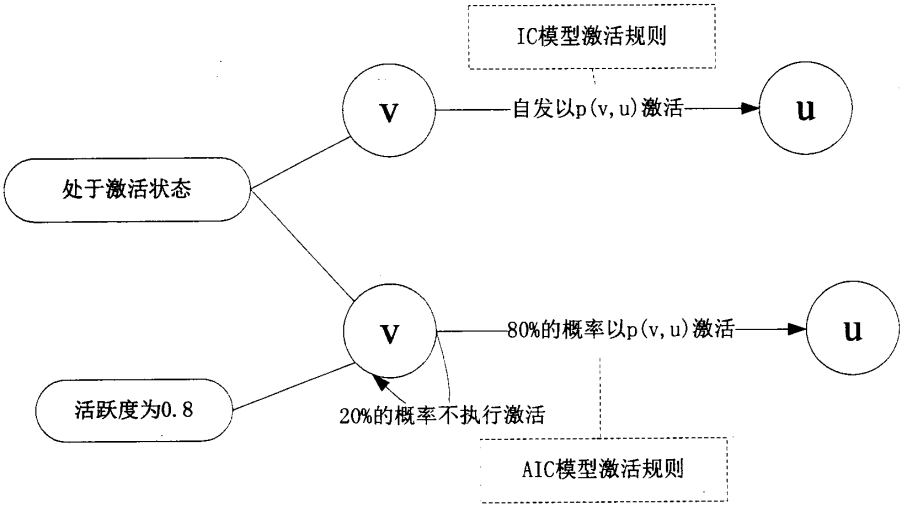

图 4.2 IC 模型与 AIC 模型的激活规则对比图

以下是对AIC模型的描述：

在AIC模型中，每一条边 $(u,v) \in E$（$u,v \in V$），$p(v,u)$代表节点u被节点v激活的概率（$p(v,u) \in [0,1]$，若$(u,v) \notin E$，则$p(v,u) = 0$），ACT_v代表节点v的活跃度属性值，$ACT_v \in [0,1]$。

AIC模型中的影响力传播过程也类似于IC模型，$A_t \subseteq V$表示在步骤$t \geq 0$时，网络中被激活的节点集合，其中$A_0 = S$。在步骤$t+1$，每一个节点$v \in A_t$拥有唯一的一次机会去激活它的邻居节点$u \in V$，激活概率为$p(v,u)$。当$A_t = \emptyset$时，传播过程结束。与IC模型不同的地方是，在AIC模型中根据节点间活跃度ACT（$ACT \in [0,1]$）值的不同，处于激活状态的节点是否对邻居节点执行激活行为的概率不同，活跃度越高，对邻居节点执行激活行为的概率越高，反之则越低（若$ACT = 1$则100%去激活邻居节点，$ACT = 0$则不会去激活邻居节点）。因此，在AIC模型中影响力在传播过程中，除了$A_t = \emptyset$这种情况，当被激活节点的$ACT = 0$，或者节点根据自身活跃度值判断并不去对邻居节点执行激活行为时，虽然该节点被激活，但是影响力也不会继续向下传播，影响力在该个分支的传播过程就到此结束，其他分支的

传播过程也是类似，都受节点活跃度的影响。

需要注意的是IC模型是AIC模型的一个特例，即所有节点ACT = 1的情况。两个模型中，处于未激活状态的节点被激活的过程是一致的，都是由节点间的激活概率决定。不同的是，处于激活状态的节点对邻居节点的激活过程。在AIC模型中，必须首先考虑节点ACT属性，根据ACT属性的值判断是否执行激活行为，若判断结果为执行激活行为，那么再由节点间激活概率决定邻居节点是否被激活，否则即使该节点的影响力再高也无法发挥其作用。

4.3 基于 AIC 模型的影响力最大化算法 ACH

4.3.1 ACH 算法设计思想

基于AIC模型，本文针对节点ACT属性提出了一种启发式算法ACH，该算法主要分为三部分，第一部分的主要思想是精简网络，因为传统算法（包括贪婪算法和启发式算法）都是默认遍历网络中的每一个节点，经过影响力计算后进行判断该节点是否成为种子节点，但是事实上社交网络中的海量节点中只有一部分节点有机会成为种子节点，而且很多节点间的关系（边）对于种子节点的挑选也是没有影响的（当激活概率过低时，可认为当前节点间的关系是无效的），因此在构建网络时就可以先对网络中的无效信息（包括无效的节点与节点间的边）进行过滤，这大大降低了网络的大小，从而可以降低计算的时耗。第二部分主要是定义针对AIC模型的节点综合影响力，在精简之后的网络G'中，首先需要考虑节点ACT属性，也就是首先选取ACT值高的节点进入ACT节点集H，然后对于ACT节点集中的每一个节点做遍历，综合考虑节点的ACT值与影响力大小，最后选出综合值aps(v)最高的节点作为种子节点，节点综合影响力定义如下：

$$aps(v) = ACT_v * \sum_{u \in N(v)} d_v * p(v,u), v \in H \tag{4.1}$$

其中ACT_v为节点v的活跃度属性值，d_v为节点v的出度，N(v)为节点v的邻居节点集，节点$u \in N(v)$为节点v的邻居节点，$p(v,u)$为节点v的邻居节点被节点v激活的概率，计算aps(v)时，也需要考虑节点v的邻居节点u的ACT值。第三部分主要是实时地更新网络中节点综合影响力，当一个节点被选择作为种子节点时，我们模拟这个节点的影响力传播过程，这个过程的模拟可以看作是在一个小规模的网络上做节点广度优先遍历，遍历过程如图4.3所示：

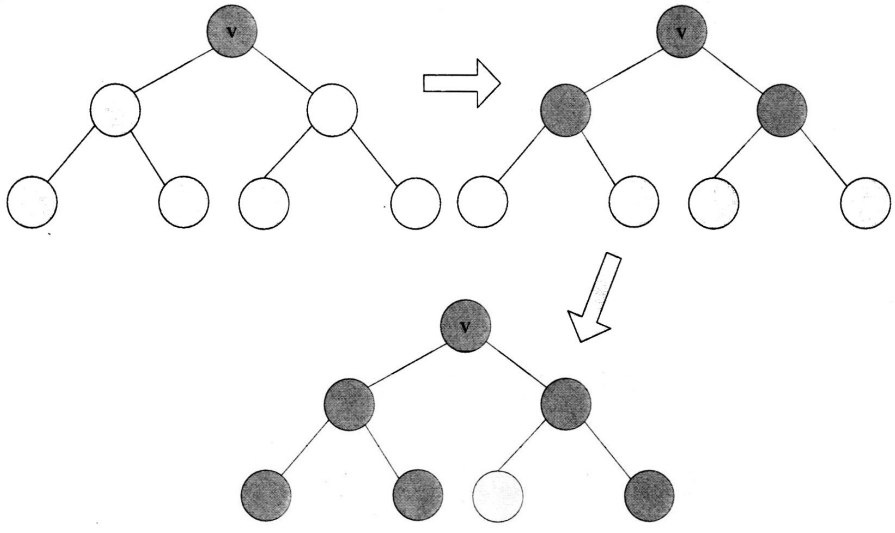

图4.3 影响力传播过程模拟图

图中节点v为刚刚选择的种子节点，节点v根据AIC模型中的激活规则去激活邻居节点（深色节点即为被激活的节点），被激活的节点又会继续去激活自身的邻居节点，如此不断迭代下去直到没有节点被激活或者激活节点不再继续执行激活行为，由此模拟影响力传播过程。在这个过程中标记传播路径上的节点间的边，因为AIC模型中节点间的激活也是有且仅有一次机会，因此，经过一次影响力的模拟传播之后我们将这些被“废弃”的边从网络中删去，如图4. 4所示，虚线表示的边就是需要被删去的边。

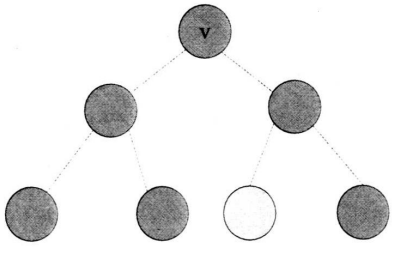

图4.4 标记删除传播过程中的边后的网络图

同时更新网络中影响力传播过程中涉及的节点及它们的邻居节点的出度，重新计算节点综合影响力，这样就进一步提高了选择精度，缩小了选择范围。整个算法的具体步骤如下：

1）构建精简的网络，主要是对节点和节点间边的过滤：

$$G' = (V', E') \tag{4.2}$$

$$V' = \{v | v \in V, d_v \geq \theta\}$$

$$E' = \{(u, v) \in E | p(v, u) \geq \beta\}$$

2）构建ACT节点集H，主要是对节点活跃度的过滤：

$$H = \{v \in V' | ACT_v \geq \gamma\} \tag{4.3}$$

3）计算ACT节点集H中节点的综合影响力aps(v)：

$$aps(v) = ACT_v * \sum_{u \in N(v)} d_v * p(v,u), v \in H \tag{4.4}$$

4）挑选aps(v)值最大的节点作为种子节点：

$$S = S \cup \{u\} \tag{4.5}$$

$$u = argmax_v\{aps(v) | v \in H \setminus S\} \tag{4.6}$$

5）模拟影响力传播过程并对传播过程中的边作标记：

$$flag(u,v) = true, (u,v) \in E' \tag{4.7}$$

6）更新网络节点与节点间的边：

$$G' = (V'', E'') \tag{4.8}$$

$$V'' = \{u \in V' | u \notin A\}$$

$$E'' = \{(u,v) \in E' | flag(u,v) \neq true\}$$

7）更新节点出度：

$$t_v = \sum_{flag(u,v)=true, u \in N(v)} 1 \tag{4.9}$$

$$d_v' = d_v - t_v \tag{4.10}$$

8）更新节点综合值aps(v)：

$$aps(v) = ACT_v * \sum_{u \in N(v)} d_v' * p(v,u) \tag{4.11}$$

9）重复4）至 8）直到$|S| = k$，整个选择过程结束。

4.3.2 ACH 算法设计

ACH算法流程图如图4.5所示：

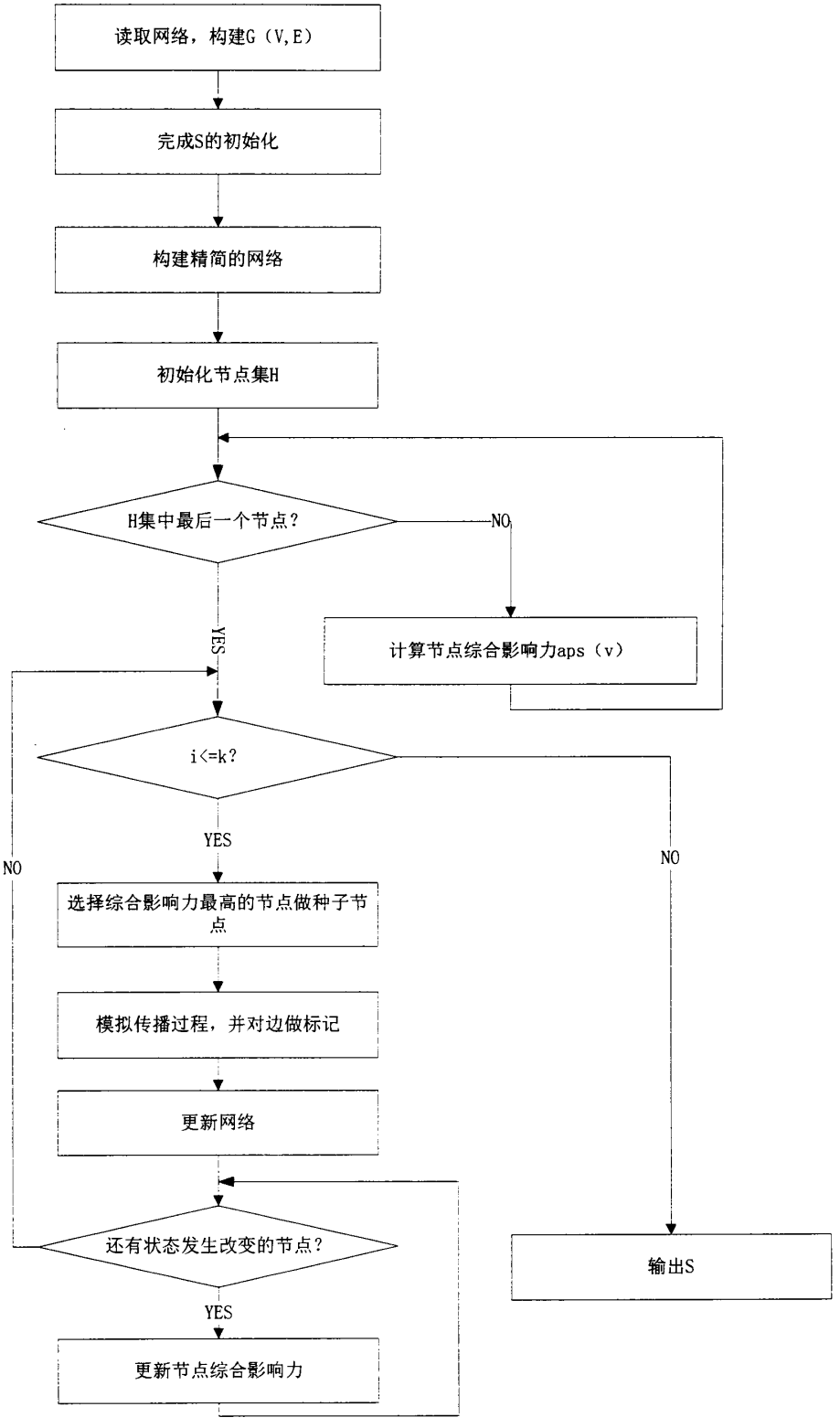

图4.5 ACH算法流程图

流程图中第一步构建网络G时，除了读取节点、节点之间的边和节点间激活概率以外，还必须对每一个节点加入人工的数据来模拟节点活跃度，这里节点活跃度值是根据一定的规则生成的一个0~1之间的随机数，在实验部分4.4.2会详细介绍节点活跃度的生成规则。构建网络的同时还会统计节点出度、节点间激活概率和节点活跃度值，从而计算出阈值θ,β和γ。节点出度阈值θ和节点间激活概率阈值β用于过滤节点和节点间的边，从而精简网络，节点活跃度阈值γ用于初始化节点集H，选择活跃度高的节点。流程图中种子节点的选择过程都是基于节点集H的。

4.3.3 ACH 算法实现

在算法实现部分，因为AIC模型对IC模型作了扩展引入了节点活跃度属性，为了适应新的AIC模型，尤其是模型扩展后发生变化的节点激活规则和影响力传播过程，我们改造了Graph类用来构造社交网络的数据结构，改造后的Graph类定义的核心代码如下所示：

```
struct Edge
{
    int u,v,c;
    double w1,w2;
    double ACT1,ACT2;
};
class Graph
{
private:
    static bool built;
    static int n;
    static int m;
    static vector<int> degree;
    static vector<int> index;
    static vector<Edge> edges;
    static void qsort_act(int h, int t);
    static void qsort_edges(int h, int t);
```

```
public:
    static void Build();
    static int GetN();
    static int  GetM();
    static int  GetDegree(int node);
    static int  GetNeighbor(int node);
    static Edge GetEdge(int node, int index);
    static double GetACT(int node);
    static void BuildIC();
    static void BuildAIC();
    static void BuildWC();
    ..........
    static void Build2IC();
    static void Build2AIC();
    static void Stats();
};
```

由于AIC模型是对IC模型的扩展，所以在第3章描述的Gragh类的基础上加入了与节点活跃度相关的数据成员变量和对应函数。在结构体struct Edge中加入了与节点u、v相对应的double类型的活跃度ACT1、ACT2，在Graph类中加入了方法GetACT(int node)用于获取节点node的活跃度值，加入了方法BuildAIC()与Build2AIC()用于创建无向与有向图中的AIC模型，加入了方法qsort_act(int h, int t)用于根据节点活跃度对节点进行排序，同时重写了方法Stats()的具体实现，使之适应AIC模型，主要是根据AIC模型中的节点激活规则和影响力传播过程对原有的基于IC模型的代码作了修改。

ACH算法的伪代码如下所示：

Algorithm ACH: ACH $(G = (V, E), k, ACT, \theta, \beta, \gamma)$

1: initialize $S = \phi$

2: initialize $G' = (V', E')$, $V' = \{v | v \in V, d_v \geq \theta\}, E' = \{(u, v) \in E | p(v, u) \geq \beta\}$

3: initialize $H = \{v \in V' | ACT_v \geq \gamma\}$

4: foreach $v \in H$ do

5: initialize $aps(v) = ACT_v * \sum_{u \in N(v)} d_v * p(v,u), v \in H, ACT_u \geq \gamma$

6: for i = 1 to k do

7: select $u = argmax_v\{aps(v) | v \in H \setminus S\}$

8: $S = S \cup \{u\}$

9: foreach $(u,v) \in E'$ do

10: $flag(u,v) = true$

11: $G' = (V', E'')$

12: $V'' = \{u \in V' | u \notin A\}$

13: $E'' = \{(u,v) \in E' | flag(u,v) \neq true\}$

14: for each neighbor v of u and v and $V \setminus S$ do

15: $t_v = \sum_{flag(u,v)=true, u \in N(v)} 1$

16: $d'_v = d_v - t_v$

17: $aps(v) = ACT_v * \sum_{u \in N(v)} d'_v * p(v,u)$

18: end for

19: end for

20: output S

第 1~2 行是初始化种子节点集 S 为空，并根据节点出度和节点间激活概率阈值构建精简的网络；

第 3~5 行是初始化 H 集，根据节点活跃度阈值选出节点活跃度较高的节点，并逐一计算 H 集中所有节点的综合影响力；

第 6~19 行为整个选择过程，分为 k 个循环，每次循环选择当前网络中综合影响力最高的节点，其中第 9~18 行为根据新选择的种子节点模拟影响力在 AIC 模型上的传播过程，并标记影响力传播过程中节点间的边，更新网络状态，更新节点出度，重新计算节点综合影响力；

第 20 行输出结果，种子节点集 S。

4.4 ACH 算法实验分析

4.4.1 实验环境

本部分实验环境与 3.3.1 节的实验环境类似，具体的硬件平台、操作系统、程序开发平台和开发语言详情参见 3.3.1 节介绍。

4.4.2 实验数据集

本文针对的是大型社交网络，因此选取了2个大型真实网络的数据集(Epinions和DBLP)，这两个数据集的规模如表4.1所示：

表4.1 数据集Epinions和DBLP的数据统计表

数据集	Epinions	DBLP
节点数量	75K	655K
边的数量	655K	2.0M
度平均值	13.4	6.1
度最大值	3079	588
节点活跃度	-	-

在本文 3.3.2 节中介绍了三个真实网络的数据集，其中数据集 Epinions 和 DBLP 属于两个大规模网络的数据集。而本文提出的 ACH 算法主要针对的是大型社交网络，所以本部分的实验数据选取数据集 Epinions 和 DBLP，它们的详细介绍见 3.3.2 节。

在这 2 个数据集的基础上，本文分别对 ACH 算法、爬山贪婪算法(Greedy)和另外 2 种启发式算法 Degree 和 MIA 算法在 AIC 模型上进行了评估。由于 Greedy 算法能保证 1-1/e 范围内接近最优解，因此以 Greedy 算法的影响范围作为影响范围的基准，以 Degree 和 MIA 算法的影响范围作为比较对象，以另 2 种启发式算法的时效性作为时效性的基准和比较对象。

AIC 模型中的 ACT 属性值的生成是采用根据节点出度来产生一个 0~1 之间的随机数的方式，在ε的情况下是 ACT 值与节点出度成正比，$1-\varepsilon$情况下 ACT 值与节点出度成反比，$\varepsilon \in (0.5,1]$，即大部分情况下可认为节点影响力越大，节点活跃度也越高，但是也有部分情况是影响力较大的节点活跃度却很低。由此可以模拟出现实情况中影响力不发挥的情况。算法中阈值θ、β和γ的取值会对算法的总体性能产生影响，它们是作为平衡算法影

响范围和时效性的因素。本文中为了兼顾算法影响范围和时效性，阈值θ采用的是网络中节点出度均值，β采用的是节点间激活概率均值，γ采用的是节点 ACT 值均值。

4.4.3 实验结果及分析

本文主要是在影响范围和时效性两方面对Greedy、Degree、MIA和ACH四种算法进行了对比。Greedy算法的耗时太多，需要三天的时间，无法在两个数据集上实时地完成计算，所以只将其影响范围作为各算法影响范围的评估基准，以另外2种启发式算法的运行时间作为时效性的对比基准。

（一）影响范围

影响范围即种子节点集的质量的评估主要是基于网络中最终被激活的节点数量。本文在两个数据集的基础上根据4.4.2节所述规则产生随机数作为节点的ACT属性值，由此完成网络$G = (V, E)$的初始化。然后在$G = (V, E)$上分别对4种算法进行仿真，计算出种子节点集S。

本文分别对$\varepsilon = \{0.6, 0.7, 0.8, 1\}$的4种情况在Epinions和DBLP两个数据集上作了实验仿真和讨论。图4.6和4.7为$\varepsilon = 0.8$时，Greedy、Degree、MIA和ACH四种算法在数据集Epinions和DBLP上的种子节点集质量对比图；图4.8和4.9为$\varepsilon = 0.7$时的四种算法在两个数据集上的种子节点集质量对比图；图4.10和4.11为$\varepsilon = 0.6$时的四种算法在两个数据集上的种子节点集质量对比图。

如图4.6、4.8和4.10所示，在数据集Epinions上，由于ACH算法针对AIC模型的ACT属性进行了针对性的处理，对于节点综合影响力的计算更加科学，同时实时更新网络中节点综合影响力，而Degree和MIA算法都没有考虑节点的活跃度属性，所以ACH算法得到的种子节点集的质量都要优于Degree和MIA算法，其影响范围基本与Greedy算法的影响范围一致。其中Degree算法的种子节点集质量较差，这是因为该算法只考虑节点出度这一因素。同时对比这3张图中，当ε值不同时，ACH算法在Epinions上的表现，可以发现ε值越小，ACH算法相对于MIA算法在影响范围上的优势越明显，同时其影响范围也越接近Greedy算法，这是因为ε值越小，节点活跃度属性越明显，AIC模型与IC模型的差异越大，节点活跃度属性对于节点激活规则、影响力传播及种子节点选择的影响也显得越明显。MIA算法没有考虑节点活跃度，因此算法的性能随着ε值的减小而不断下降，而ACH算法没有这个问题。

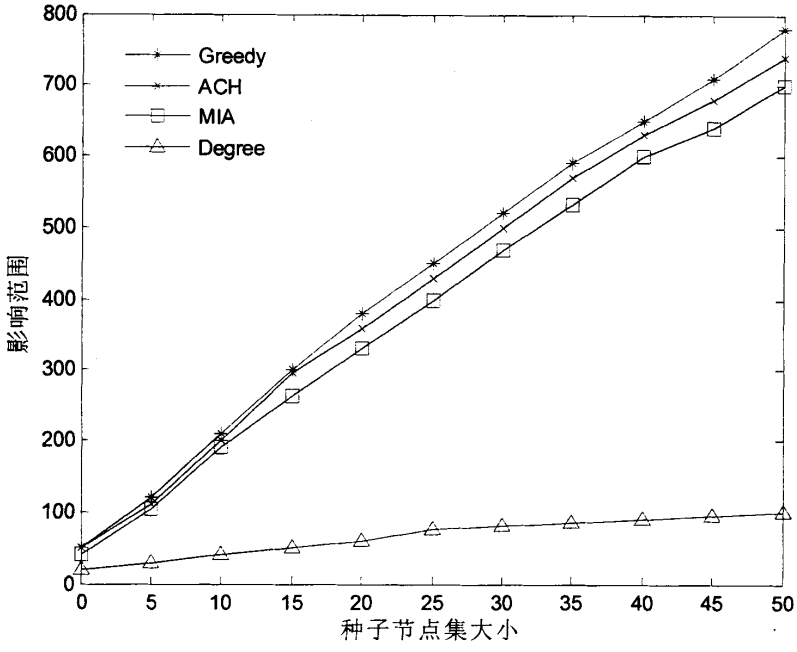

图4.6 ε = 0.8时Epinions上的种子节点集质量对比图

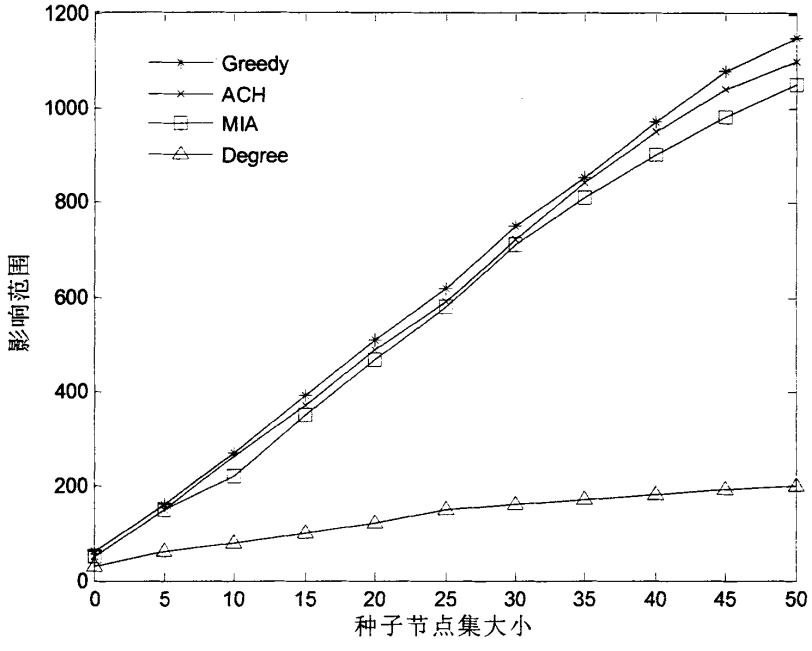

图4.7 ε = 0.8时DBLP上的种子节点集质量对比图

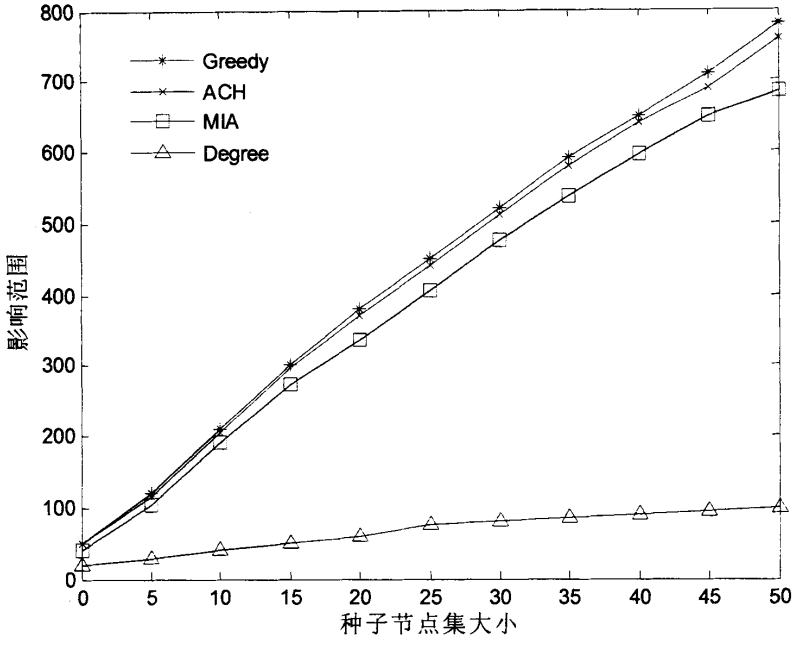

图4.8 ε = 0.7时Epinions上的种子节点集质量对比图

如图4.7、4.9和4.11所示，在规模更大的数据集DBLP也得到了一致的结果，ACH算法得到的种子节点集的质量要优于Degree和MIA算法，其影响范围基本与Greedy算法的影响范围一致。而且比较这三张图，发现ε值越小，ACH算法相对于MIA算在影响范围上的优势越明显，同时ACH算法的影响范围也越接近Greedy算法。这是因为ε值越小，模型中节点活跃度属性越明显，ACH算法对节点活跃度做了针对性的处理，而MIA算法没有考虑节点活跃度，所以ACH算法的性能优势也就越明显。

同时，观察图4.6-4.11可以发现，随着种子节点集的增大，在两个数据集上ACH算法相对于MIA算法的优势也越来越明显，这是因为选择少量种子节点时并不能全面的体现算法的性能，当种子节点数量增大后，算法的性能得到更好的表现。

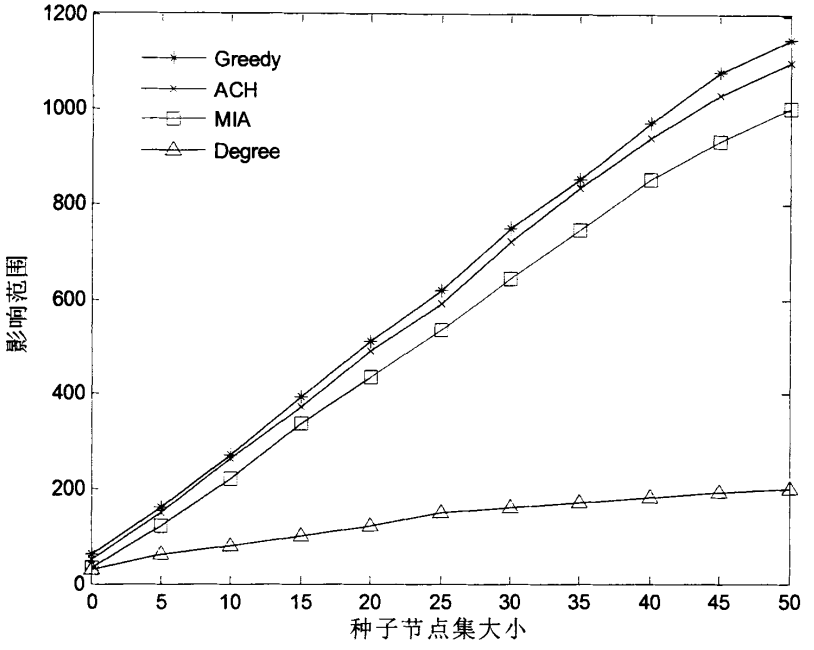

图4.9 ε = 0.7时DBLP上的种子节点集质量对比图

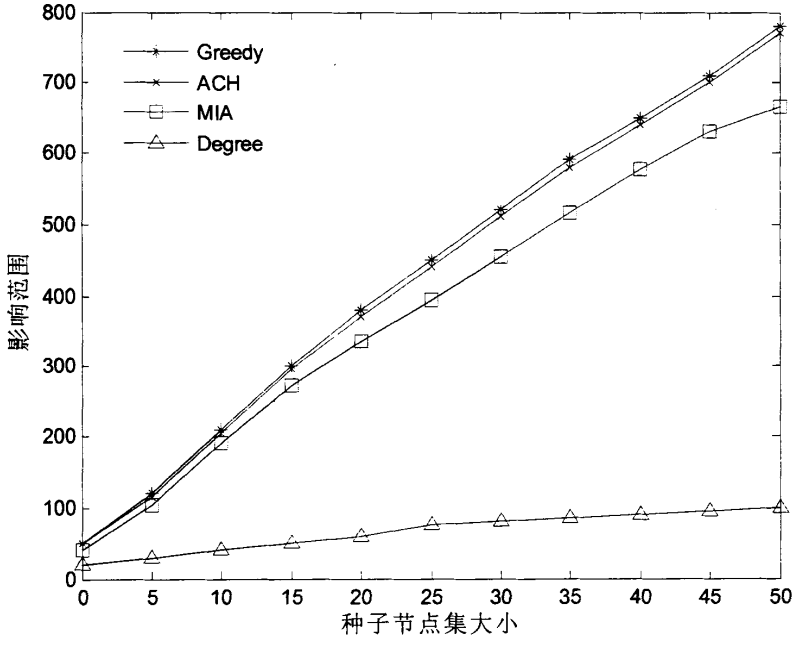

图4.10 ε = 0.6时Epinions上的种子节点集质量对比图

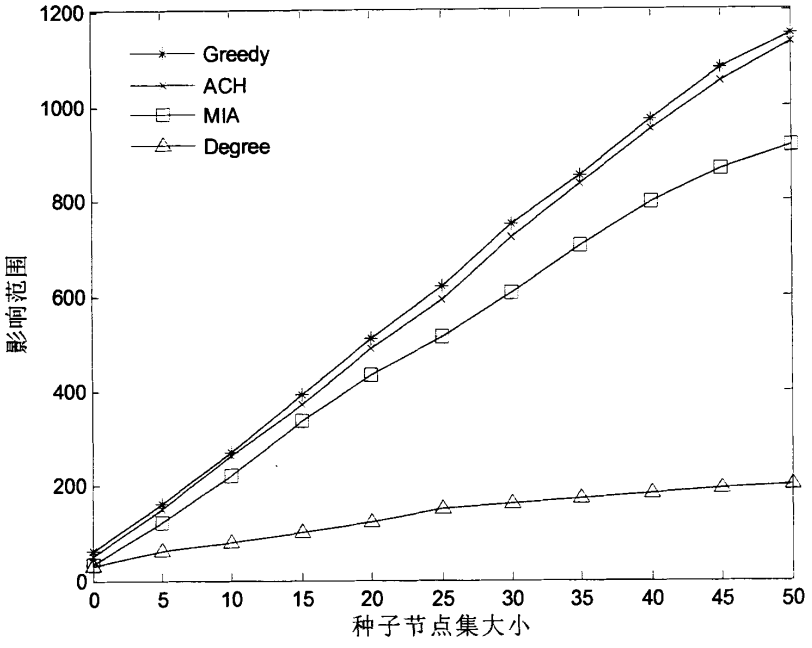

图4.11 ε = 0.6时DBLP上的种子节点集质量对比图

如图4.6和4.7、4.8和4.9、以及4.10和4.11所示，对比ε值相等时，各算法在两个数据集上的表现可得，在更大规模的数据集DBLP上，ACH算法对于MIA算的优势也更明显，说明ACH算法对更大规模的网络适应能力更强，这是因为ACH算法每选择一个种子节点就会实时更新网络状态，更新状态发生改变的节点的综合影响力，这提高了种子节点的质量。而且由于ACH算法在预处理阶段会精简网络，对于精简的网络还会根据节点活跃度进行优质筛选，种子节点的选择都是在精简后的较小规模的候选节点集中进行，这使ACH算法的性能受网络规模变化的影响较小。即使网络规模变大很多，算法性能依然稳定。

图4.12和4.13为ε = 1时，四种算法在数据集Epinions和DBLP上的种子节点质量对比图。当ε = 1时，节点活跃度值100%与节点出度成正比，即节点影响力越大，节点活跃度也越高，此时不再有影响力高的节点反而活跃度很低的情况。节点活跃度属性对节点激活规则、种子节点选择的影响变弱，AIC模型十分接近IC模型，所以设置所有节点的ACT = 1，不采取随机数赋值的方式。如图4.12和4.13所示，在数据集Epinions和DBLP上，ACH算法在影响范围上依然维持着良好的表现，ACH算法所得种子节点集的质量优于MIA算法，且接近Greedy算法。由此得出，ACH算法拥有良好的稳定性。

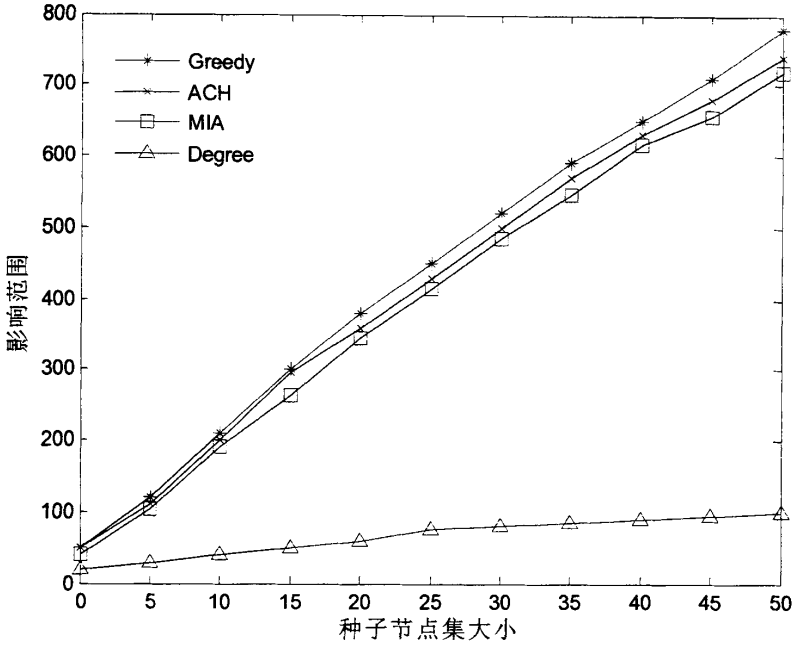

图4.12 ε = 1&&ACT = 1时Epinions上的种子节点集质量对比图

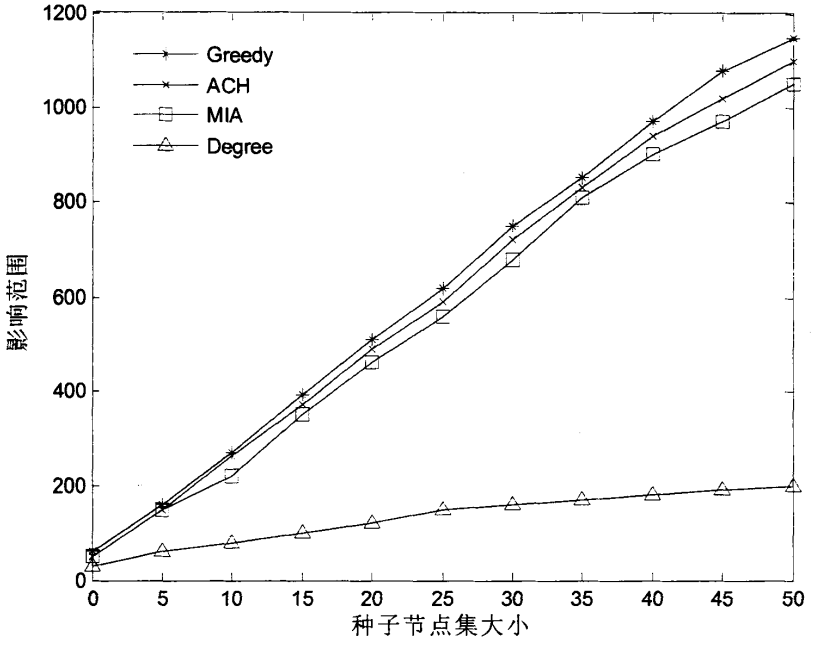

图4.13 ε = 1&&ACT = 1时DBLP上的种子节点集质量对比图

（二）时效性

时效性即算法找出种子节点集所需的运行时间。由于节点活跃度值是在构建网络G的时候与节点完成绑定，所以ε的人工取值并不影响各算法的运行时间，只是对算法影响范围产生影响。各算法的平均运行时间如表4.2所示，Greedy算法的运行时间太久，而Degree算法由于其简易性运行时间极短，因此不在比较范围内。由于ACH算法对整个网络进行了精简，过滤了大量对种子节点集的选择不会产生影响的节点及边，同时ACH算法还针对节点活跃度对候选节点集做了进一步的过滤，大大缩减了选择范围。从表中数据得知，ACH算法的运行时间相较于MIA算法缩短了近一半，拥有更好的时效性。

表4.2 平均运行时间对比表

算法	Epinions	DBLP
Greedy	-	-
Degree	-	-
MIA	15s	41s
ACH	7s	23s

4.5 本章小结

IC 模型中的节点激活规则没有很好地结合现实。本文针对这个问题在 IC 模型中引入节点活跃度属性，提出了 IC 模型的扩展模型 AIC，并针对该模型提出了 ACH 启发式算法。本章首先详细描述了 AIC 模型的特性及其影响力传播过程，然后给出了 ACH 算法的设计思想、算法具体实现及伪代码描述。ACH 算法是针对大型社交网络，因此实验部分选取了 2 个大型真实网络的数据集。实验结果显示在 AIC 模型中，ACH 算法在影响范围和时效性上都拥有一定的优势。

第五章 总结与展望

5.1 论文总结

随着互联网社交的流行，越来越多的学者开始关注社交网络领域的研究，其中影响力最大化问题是当前的研究热点。这个问题的研究在市场营销、舆论预警及社会和谐安定等方面都有着非常重要的作用和研究价值。庞大的网络规模，海量的用户及复杂的用户间关系都给这个问题的研究带来了巨大的挑战。

本文首先整理并介绍了社交网络影响力最大化问题的相关概念，包括社交网络的概念介绍、社交网络影响力最大化问题的定义、两种经典的影响力传播模型的详细介绍以及现有的一些常用的经典算法的描述。然后基于这些理论知识和相关工作，提出了本文的主要研究内容，分为两方面：

第一是针对 Diffusion Degree 算法存在的一些问题进行了改进，并提出了 IDD 算法。Diffusion Degree 算法中将所有邻居节点的影响力都加权计算进当前节点的综合影响力中，但是考虑到节点只有被激活后才拥有影响力，激活概率过小时仍然对邻居节点的影响力进行加权计算显得并不合理。因此 IDD 算法根据节点间激活概率阈值对邻居节点进行了过滤，考虑了节点间潜在影响力的实际有效性，并在每次选择一个种子节点后更新网络。实验结果证明，IDD 算法通过牺牲少量的时间复杂度的方式使算法在影响范围上取得了明显的进步。IDD 算法虽然在运行时间上略大于 Diffusion Degree 算法，但是在影响范围方面已经接近目前最好的启发式算法 PMIA，明显优于 Diffusion Degree 算法。

第二是针对 IC 模型进行了扩展，引入节点活跃度属性，提出了 AIC 模型，并针对该模型提出了 ACH 启发式算法。AIC 模型中由于新增了节点活跃度属性，节点激活规则和影响力传播过程被改变，新的激活规则更加科学，贴近实际。在 ACH 算法中也根据节点活跃度重新定义了节点综合影响力，从而更科学地选择种子节点。ACH 算法首先精简了网络大小，过滤对种子节点选择无用的信息，然后根据新定义的节点综合影响力选择种子节点，并实时更新网络。实验结果证明，针对 AIC 模型，ACH 算法相对于其他启发式算法在时效性和影响范围两方面都具有一定的优势。

5.2 论文展望

虽然本文提出的IDD算法以及针对AIC模型的ACH算法都取得了较满意的结果。IDD算法牺牲少量运行时间换取影响范围的明显优化，ACH算法则在时效性和影响范围两方面都取得了较满意的效果。但是本文依然还有值得改进与进一步研究的地方。现将其总结如下：

1. AIC模型中节点的活跃度属性的数据是人造数据，根据节点出度按一定规则设定为0~1之间的随机数。虽然本文模拟的数据尽可能地考虑了实际情况，但是无法完全替代真实数据集，今后应该搜集相应的数据来评估和设定节点的活跃度数值。
2. IDD算法是基于IC模型的，ACH算法是基于AIC模型（IC模型的扩展模型）的。今后应考虑拓展算法的应用范围，将其应用于线性阈值模型或其扩展模型。
3. 节点影响力更新对种子节点选择起着重要的影响，如何更加高效准确地模拟影响力传播过程，从而动态的实现网络更新需要投入更多的关注和研究。
4. 传播模型中可以更加实际地考虑传播时间这一因素。因为现实中任何产品的推广，信息的传播都是有时间限制的。

参考文献

[1] M. Kochen, I. de Sola Pool, S. Milgram, and T. Newcomb, The small world [M]. Norwood, NJ: Ablex, 1989.

[2] Kleinberg J. The small-world phenomenon: An algorithmic perspective[C]. Proceedings of the thirty-second annual ACM symposium on Theory of computing. ACM, 2000: 163-170.

[3]胡海波, 徐玲, 王科等. 大型在线社会网络结构分析[J]. 上海交通大学学报, 2009 (4): 587-591.

[4]景天魁, 林聚任. 社会网络分析: 理论, 方法与应用[J]. 北京:北京师范大学出版社, 2009: 83-86,107-116

[5] Wang C, Chen W, Wang Y. Scalable influence maximization for independent cascade model in large-scale social networks[J]. Data Mining and Knowledge Discovery, 2012, 25(3): 545-576.

[6]胡海波, 王科, 徐玲, 等. 基于复杂网络理论的在线社会网络分析[J]. 复杂系统与复杂性科学, 2008, 5(2): 1-14.

[7] Domingos P, Richardson M. Mining the network value of customers[C].Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2001: 57-66.

[8] Richardson M, Domingos P. Mining knowledge-sharing sites for viral marketing[C].Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2002: 61-70.

[9] Kempe D, Kleinberg J, Tardos É. Maximizing the spread of influence through a social network[C].Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2003: 137-146.

[10] 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. ACM, 2007: 420-429.

[11] 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. ACM, 2011: 47-48.

[12] 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. ACM, 2009: 199-208.

[13] Estevez P A, Vera P, Saito K. Selecting the most influential nodes in social networks[C].Neural Networks, 2007. IJCNN 2007. International Joint Conference on. IEEE, 2007: 2397-2402.

[14] Chen W, Lu W, Zhang N. Time-Critical Influence Maximization in Social Networks with Time-Delayed Diffusion Process[C].AAAI. 2012.

[15] Li F H, Li C T, Shan M K. Labeled Influence Maximization in Social Networks for Target Marketing[C].Privacy, security, risk and trust (passat), 2011 ieee third international conference on and 2011 ieee third international conference on social computing (socialcom). IEEE, 2011: 560-563.

[16] Nguyen H, Zheng R. On Budgeted Influence Maximization in Social Networks[J]. Selected Areas in Communications, IEEE Journal on, 2013, 31(6): 1084-1094.

[17] Wang Y, Huang W J, Zong L, et al. Influence maximization with limit cost in social network[J]. Science China Information Sciences, 2013, 56(7): 1-14.

[18] Cao T, Wu X, Wang S, et al. OASNET: an optimal allocation approach to influence maximization in modular social networks[C].Proceedings of the 2010 ACM Symposium on Applied Computing. ACM, 2010: 1088-1094.

[19] Wang Y, Cong G, Song G, et al. Community-based greedy algorithm for mining top-k influential nodes in mobile social networks[C].Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2010: 1039-1048.

[20] Kempe D, Kleinberg J, Tardos É. Influential nodes in a diffusion model for social networks[M].Automata, languages and programming. Springer Berlin Heidelberg, 2005: 1127-1138.

[21] Even-Dar E, Shapira A. A note on maximizing the spread of influence in social networks[M].Internet and Network Economics. Springer Berlin Heidelberg, 2007: 281-286.

[22]冀进朝, 韩笑, 王喆. 基于完全级联传播模型的社区影响最大化[J]. 吉林大学学报: 理学版, 2009, 47(5): 1032-1034.

[23] Kimura M, Saito K. Tractable models for information diffusion in social networks[M].Knowledge Discovery in Databases: PKDD 2006. Springer Berlin Heidelberg, 2006: 259-271.

[24] Bharathi S, Kempe D, Salek M. Competitive influence maximization in social networks[M].Internet and Network Economics. Springer Berlin Heidelberg, 2007: 306-311.

[25] M. Lahiri and M. Cebrin. The Genetic Algorithm as a General Diffusion Model for Social Networks[C], Proc. of the 24th AAAI Conference on Artificial Intelligence, 2010.

[26]冀进朝, 黄岚, 王喆等. 一种新的基于社区结构的影响最大化方法[J]. 吉林大学学报: 理学版, 2011, 49(001): 93-97.

[27] Li C T, Shan M K, Lin S D. Dynamic selection of activation targets to boost the influence spread in social networks[C]. Proceedings of the 21st international conference companion on World Wide Web. ACM, 2012: 561-562.

[28]才华, 周春光, 王喆等. 动态社会网络中的社区挖掘算法研究[J]. 吉林大学学报: 信息科学版, 2008, 26(4): 380-385.

[29]曲鹏程. 动态社会网络挖掘算法研究[D]. 吉林大学, 2008.

[30] Liu B, Cong G, Xu D, et al. Time Constrained Influence Maximization in Social Networks[C]. ICDM. 2012: 439-448.

[31] Mossel E, Roch S. On the submodularity of influence in social networks[C]. Proceedings of the thirty-ninth annual ACM symposium on Theory of computing. ACM, 2007: 128-134.

[32] Yang Y, Li V O K, Xu K. Influence maximization in noncooperative social networks[C].Global Communications Conference (GLOBECOM), 2012 IEEE. IEEE, 2012: 2834-2839.

[33] Nguyen H, Zheng R. Influence spread in large-scale social networks–A belief propagation approach[M].Machine Learning and Knowledge Discovery in Databases. Springer Berlin Heidelberg, 2012: 515-530.

[34]周春光, 曲鹏程, 王曦等. DSNE: 一个新的动态社会网络分析算法[J]. 吉林大学学报: 工学版, 2008, 38(2): 408-411.

[35] Liu B, Cong G, Xu D, et al. Time Constrained Influence Maximization in Social Networks[C].ICDM. 2012: 439-448.

[36] Li J, Yu Y. Scalable Influence Maximization in Social Networks Using the Community Discovery Algorithm[C].Genetic and Evolutionary Computing (ICGEC), 2012 Sixth International Conference on. IEEE, 2012: 284-287.

[37] Chen Y C, Chang S H, Chou C L, et al. Exploring Community Structures for Influence Maximization in Social Networks[C].The 6th SNA-KDD Workshop on Social Network Mining and Analysis Held in conjunction with KDD. 2012, 12: 1-6.

[38] Fan X, Li V O K. The probabilistic maximum coverage problem in social networks[C].Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE. IEEE, 2011: 1-5.

[39] Li H, Bhowmick S S, Sun A. CINEMA: conformity-aware greedy algorithm for influence maximization in online social networks[C].Proceedings of the 16th International Conference on Extending Database Technology. ACM, 2013: 323-334.

[40] Chen W, Wang C, Wang Y. Scalable influence maximization for prevalent viral marketing in large-scale social networks[C].Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2010: 1029-1038.

[41] Yeh T A, Wang E T, Chen A L P. Finding Leaders with Maximum Spread of Influence through Social Networks[M].Advances in Intelligent Systems and Applications-Volume 1. Springer Berlin Heidelberg, 2013: 269-279.

[42]田家堂, 王轶彤, 冯小军. 一种新型的社会网络影响最大化算法[J]. 计算机学报, 2011, 34(10): 1956-1965.

[43]陈浩, 王轶彤. 基于阈值的社交网络影响力最大化算法[J]. 计算机研究与发展, 2012, 49(10): 2181-2188.

[44] Jung K, Heo W, Chen W. IRIE: Scalable and robust influence maximization in social networks[J]. arXiv preprint arXiv:1111.4795, 2011.

[45] Goyal A, Lu W, Lakshmanan L V S. Simpath: An efficient algorithm for influence maximization under the linear threshold model[C].Data Mining (ICDM), 2011 IEEE 11th International Conference on. IEEE, 2011: 211-220.

[46] Chen W, Collins A, Cummings R, et al. Influence Maximization in Social Networks When Negative Opinions May Emerge and Propagate[C].SDM. 2011: 379-390.

[47] Chen W, Yuan Y, Zhang L. Scalable influence maximization in social networks under the linear threshold model[C].Data Mining (ICDM), 2010 IEEE 10th International Conference on. IEEE, 2010: 88-97.

[48] Goyal A, Bonchi F, Lakshmanan L V S. Learning influence probabilities in social networks[C].Proceedings of the third ACM international conference on Web search and data mining. ACM, 2010: 241-250.

[49] Narayanam R, Narahari Y. A shapley value-based approach to discover influential nodes in social networks[J]. IEEE Transactions on Automation Science and Engineering, 2010 (99): 1-18.

[50] Kim J, Kim S K, Yu H. Scalable and parallelizable processing of influence maximization for large-scale social networks[C]. Data Engineering (ICDE), 2013 IEEE 29th International Conference on. IEEE, 2013:

266-277.

[51] Kundu S, Murthy C A, Pal S K. A new centrality measure for influence maximization in social networks[M]. Pattern Recognition and Machine Intelligence. Springer Berlin Heidelberg, 2011: 242-247.

[52] Yang H, Wang C, Xie J. Maximizing influence spread in a new propagation model[M].Rough Sets and Knowledge Technology. Springer Berlin Heidelberg, 2012: 292-301.

[53] Borgs C, Brautbar M, Chayes J T, et al. Influence maximization in social networks: Towards an optimal algorithmic solution[J]. arXiv preprint, 2012.

[54] Zhu Y, Wu W, Bi Y, et al. Better approximation algorithms for influence maximization in online social networks[J]. Journal of Combinatorial Optimization, 2013: 1-12.

[55] Suri N R, Narahari Y. Determining the top-k nodes in social networks using the shapley value[C].Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 3. International Foundation for Autonomous Agents and Multiagent Systems, 2008: 1509-1512.

[56] Wang Y, Feng X. A potential-based node selection strategy for influence maximization in a social network[M].Advanced Data Mining and Applications. Springer Berlin Heidelberg, 2009: 350-361.

[57] Hao F, Chen M, Zhu C, et al. Discovering influential users in micro-blog marketing with influence maximization mechanism[C].Global Communications Conference (GLOBECOM), 2012 IEEE. IEEE, 2012: 470-474.

[58]Li Y, Chen W, Wang Y, et al. Influence diffusion dynamics and influence maximization in social networks with friend and foe relationships[C].Proceedings of the sixth ACM international conference on Web search and data mining. ACM, 2013: 657-666.

[59] Leskovec J, Huttenlocher D, Kleinberg J. Signed networks in social media[C].Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, 2010: 1361-1370.

[60] Goyal A, Bonchi F, Lakshmanan L V S. A data-based approach to social influence maximization[J]. Proceedings of the VLDB Endowment, 2011, 5(1): 73-84.

致谢

时光飞逝，转眼间硕士生涯行将结束。回首往事，心中感慨万千。江苏大学博学、求是、明德的校训，严谨务实的学风，纯正浓厚的科学氛围都给我留下了深刻的印象，也深深影响了我，这些都让我获益匪浅，受用终身！而能够进入江苏大学，向众多杰出的老师和优秀的同学学习是我莫大的荣幸，我终生难忘！感谢母校，感谢周老师！

首先，非常感谢我的导师周莲英教授！三年来，科研上周老师循循善诱，精心指导，严格要求，生活上也是是无微不至，学生的每一点成绩，每一点进步无不包含着老师的栽培和心血！周老师的严谨、勤奋以及精益求精事必躬亲的工作态度，对科学真理的孜孜追求让我深受感动，也备受激励！深厚的学术功底更让学生佩服不已！周老师不但教会了我科研方法，更是用自己的身体力行教会了我为人做事的道理！这些东西学生将终身获益！在此谨向我们敬爱的周老师以崇高的敬意和最诚挚的感谢和祝福！谢谢周老师！

感谢同项目组的好搭档蒋大飞、吴淑跃、王斌同学，我们并肩作战，相互帮助，相互鼓励，共同进步，留下了多少快乐而又难忘的时光！回想起来，往事历历在目！课题的进程中大家都给了我很大的帮助，谢谢大家！衷心地祝大家在以后的人生旅途中工作顺利，身体健康，飞黄腾达！

感谢实验室一起学习的师兄、师姐、师弟、师妹！感谢你们对我的支持！与你们探讨问题常让我感觉很有启发！

最后向所有帮助过我的人表示最真诚的感谢！祝你们永远健康幸福！

攻读硕士学位期间发表的论文及科研情况

发表的论文：

[1]周莲英, 朱锋. 一种改进的社交网络影响力最大化算法[J]. 信息技术, 2014.
[2]周莲英,朱锋.一种基于活跃度的社交网络影响力最大化方法.代理中.已获申请号【201410243677.8】(第二作者，导师第一作者)

参加的科研项目：

[1]江苏大校关工委信息服务平台（江苏大学校内项目）。

[2]面向留学生的信息服务平台。

[3]江苏大学中层干部网上测评后台管理系统（江苏大学校内项目）。

[4]江苏大学食品学院数据库管理系统（江苏大学校内项目）。