朋友关系网络

摘要

随着科技的飞速发展,朋友网络的形成方式也有了很大的改变,了解朋友关系网络的形成,可能的结构,以及动态变化趋势对于某些领域和人们的生活交友意义重大. 本文就朋友关系网络形成的各个环节进行了建模分析,并通过在线采集数据对模型准确性进行检验.

本文基于 *PageRank* 算法建立了同时考虑朋友关系和兴趣爱好的链接预测 *TFPR* 模型,通过计算 *TFPR* 矩阵进而判断两个节点链接的概率,从而进行网络中的加边和删边;为了更好的刻画兴趣,本文从主题模型 *LDA* （狄利克雷分配）来挖掘个体消息中隐含的语义信息,使用吉布斯（*LDAGibbsSampling*）采样得到每个人在不同主题上的分布情况,得出对某个主题的偏好得分;同时考虑链接强度和强度比重来辅助预测新的朋友关系.

通过设计增边算法和删边算法,模拟实例形成动态的朋友关系网络;同时分析网络拓扑结构的统计特性,通过分析动态过程中统计特性的变化来查看网络的复杂性. 本文还通过提出改进的 *BA* 模型,通过对比分析提出的 *TFPR* 模型的优缺点.

关键词 朋友关系网络 PageRank 算法 TFPR 模型 主题模型 改进 BA 模型

目录

一、问题重述

朋友关系网络对社会和个人都非常重要. 科技的进步让我们朋友关系网络的形成和发展都有了很大的变化. 同时关系网络涉及很多社会现象, 如群体行为、信息传递、疾病传播等. 如何在考虑影响网络结构的个体特征(性别, 年龄, 爱好等)的情况下, 建立一个动态的网络模型来帮助人们理解朋友关系网络的形成, 都会形成一些什么样的结构, 并且随时间如何动态变化. 另外准备一份 1-2 页的海报来展示结果和数学模型, 面向社会大众发表, 使普通大众可以理解所得研究结果并产生兴趣.

二、问题分析

题目的最终目的是让我们在解决如下问题之后给出一个朋友关系网络建立的模型, 即朋友网络是如何建立和发展起来的.

首先, 朋友网络的形成, 是一个复杂网络问题. 朋友网络是根据人们之间的朋友关系所建立的网络, 以人为节点, 两人之间若有朋友关系则连接一条边. 从最初的孤立节点到通过新旧节点的链接及新边的生成形成网络, 因此节点的链接就是链接预测问题. 据此, 本文应该分析朋友关系形成的因素模型. 其次, 我们需要分析朋友网络的结构是怎样的, 现实情况下的朋友网络应是链接拓扑结构, 通过节点的连接和边的生成不断的扩大, 并且网络的拓扑性质的复杂性也决定了网络的动态生成需要考虑的网络特征.

同时题目要求我们用模型得出不同的时代对朋友网络形成和动态变化的影响. 重点是查找 1980 年代的朋友网络与现今朋友网络的特征区别以及复杂程度的差别. 提出改进 *BA* 模型,和上述模型进行比较观察优缺点.

三、模型假设

1. 个体特征在短期内相对稳定，如兴趣爱好;
2. 人人网朋友关系与现实生活中朋友关系相同;
3. 现实生活中结交主要考虑两个因素:个体兴趣和朋友关系;

四、符号说明

符号	符号说明
θ_{uj}	从个体 u 的历史消息中挖掘出的 *topic j* 所占的比重
$PR_j(u)$	个体 u 在已有朋友关系网络的基础上对 *topic j* 的偏好得分
T	最大时间步长
t	时间步长, $t \in [0,T] \cap Z$
m	第 t 个时间步长内朋友网络所添加的边数
n	第 t 个时间步长内朋友网络所删除的边数
k_i	第 i 个节点的度即节点所连边数
$\langle k \rangle$	网络的平均度

C_i	第i个节点的聚类系数,
C	网络的平均聚类系数
$d(i,j)$	第i个节点到第j个节点的最短路径的长度
$\langle d \rangle$	网络的平均路径长度

五、模型建立与求解

5.1 模型准备

首先对朋友关系网络生成机制进行思考. 两个人能否成为朋友关系可以归结为图中节点的链接预测问题. 链接预测问题定义如下：给定时刻网络的拓扑链接, 准确预测从某一时刻t_1到未来某一时刻t_2这段时间内出现的新的朋友链接. 即个体间不存在朋友链接但各方面非常相似的两个个体未来时刻可以通过链接预测成为对方的候选朋友. 总之, 链接预测的目的就是帮助个体结识新的朋友.

如何找到与个体最相似的候选朋友以及如何衡量其相似程度是本环节所重点关注的问题. 我们知道, 在现实社交网络中朋友的结交主要考虑两个因素：个体兴趣和朋友关系.

基于朋友的方法能将处在目标个体可达范围内的潜在朋友全部找到;对于不在朋友关系可达范围内的潜在朋友, 该方法无法将其推荐给目标个体. 而基于兴趣的方法, 由于过度依赖个体发布的信息, 而导致无法推荐与目标客户有着许多共同朋友的潜在朋友客户. 因此,考虑到朋友链接构成了社交网络的拓扑结构(如图 1), 个体所发的信息在该结构中流动, 而流动的信息是个体之间结交朋友, 交流互动的重要途径, 如何同时建模这两个因素对朋友推荐的影响是需要解决的问题.

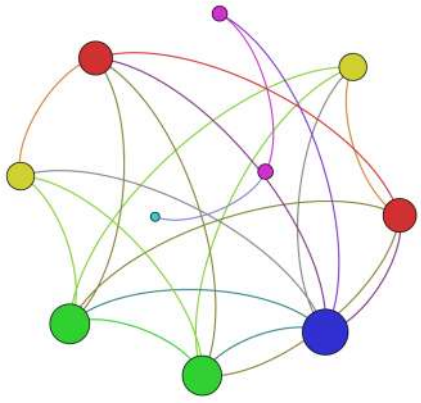


图1 一个简单的朋友网络的拓扑

主题模型（*LDA*）可以用来预测社交网络中的朋友关系. 主题模型是生成模型, 以概率描述一个文档的生成过程. 其中使用最广泛的主题生成模型是由 *Blei* 等提出的 *LDA* . 图 2 给出了 *LDA* 的图模型表示. 首先, 假设语料库 D 有 M 个文档构成, $D=\{d_1,d_2,\cdots d_m\}$；每个文档 d_i 由 N_i 个单词构成 $d_i=\{w_{i1},w_{i2},\cdots w_{iNi}\}$, 这里每个单词 $w_{ij}\in V$, V 是词典库. 那么文档的生成过程如下：

①对于每个文档 d_i :

生成它的主题分布 $\theta_i \sim Dirichlet(\alpha)$

②对于每个词 w_{ij}：

选择一个主题 $z_{ij} \sim Multinomial(z_{ij} \mid \theta_i)$

选择一个词 $w_{ij} \sim Multinomial(z_{ij})$

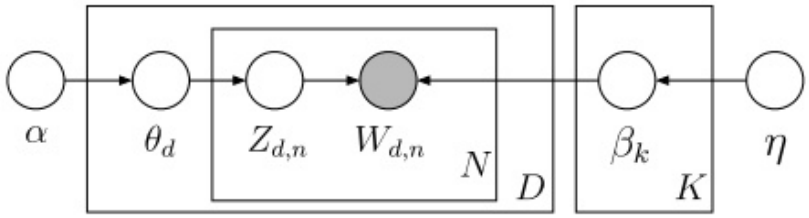


图2 *LDA* 模型

近年来, 有些工作将主题感知的 *PageRank* 算法思想引入社交网络的朋友关系预测问题中[1, 2, 3]. 此算法同时考虑了网页之间连接结构和每个网页内容的语义信息, 计算网页的不同主题下的 *PageRank* 值, 代表网页在其所处的链接结构中关于某个主题的重要性得分, 用一个向量 $\overrightarrow{PageRank}$ 表示. 算法模型如下:

$$\overrightarrow{PageRank} = dM\overrightarrow{PageRank} + (1-d)\vec{p} \tag{1}$$

这里, 用 G 表示网页链接构成的图, M 是该图 G 的平方随机矩阵, $\vec{p}$ 为 $N \times 1$ 维的个性化向量. 本文将该算法思想引入朋友网络的朋友推荐问题中, 使用 *LDA* 从个体发布的消息中挖掘个体兴趣, 构建个性化向量 $\vec{p}$ [1].

5.2 基于 PageRank 算法的 TFPR 模型[1]

5.2.1 抽取个人的主题分布

采用主题模型 *LDA*（狄利克雷分配）来挖掘个体消息中隐含的语义信息, 在图2中使用吉布斯（*LDAGibbsSampling*）[2]采样得到每个人 u 在不同主题上的分布情况, 并将其记为 θ_u. 则由主题感知的 *PageRank* 算法得到个体 u 在已有朋友网络的基础上对某个主题的偏好得分：

$$PR_j(u) = (1-d)\frac{\theta_{uj}}{\sum_{u^* \in U}\theta_{u^* j}} + d\sum_{v \in Friend(u)}\frac{PR_j(v)}{L(v)} \tag{2}$$

其中, θ_{uj} 表示从个体 u 的历史消息中挖掘出的 *topic j* 所占的比重. $Friend(u)$ 表示个体 u 的朋友集合, 而 U 表示所有个体的集合, 任意个体 $u^* \in U$.

5.2.2 计算朋友的链接强度

朋友之间亲密程度不同产生不同的链接强度. 本文中将这种强弱差别加入 *PageRank* 算法来辅助预测新的朋友关系. 对于人人网等由无向链接构成的社交网络, 如果个体 u 在 v 的朋友列表中, 则个体 v 必在 u 的朋友列表中. 其中, $Friend(u) = \{f_{u1}, f_{u2}, \cdots f_{um}\}$, $Friend(v) = \{f_{v1}, f_{v2}, \cdots f_{vm}\}$. 链接 (u,v) 的强弱记为 $stren(u,v)$, 其计算过程如下:

$$Stren(u,v) = \frac{|Friend(u) \cap Friend(v)|}{|Friend(u)| + |Friend(v)|}$$

其中, $|Friend(u) \cap Friend(v)|$ 代表个体 u 和个体 v 共同朋友的数目, $|Friend(u)| + |Friend(v)|$ 表示两个个体总的朋友数目. 对于个体 u 的任意一个朋友 $v^* \in Friend(u)$, 链接 (u,v^*) 的强度为 $stren(u,v^*)$, 链接 (u,v) 与其他链接 (u,v^*)

相比, 其强度比重为

$$\pi(u,v)=\frac{Stren(u,v)}{\sum_{v^*\in Friend(u)}Stren(u,v^*)} \tag{3}$$

这里, $\sum_{v^*\in Friend(u)}\pi(u,v^*)=1$.

5.2.3 TFPR 模型的建立

将上述两部分融合在一起, 提出 *Tpic _ Friend _ PageRank* 模型来预测朋友关系（如图3）.

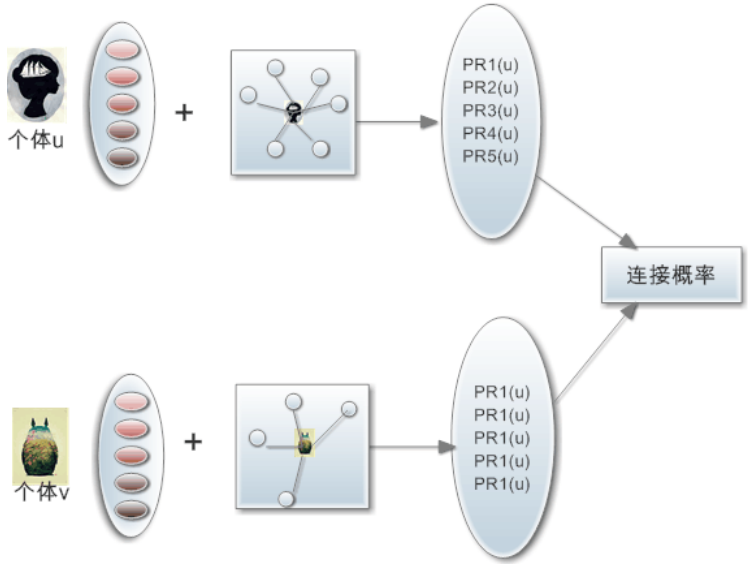


图3 *TFPR*模型

朋友链接生成的因素主要是兴趣和朋友关系. 如果两个个体有许多相似的爱好, 那么他们成为朋友的可能性很大; 同时, 两个个体有许多相同的朋友, 那么他们成为朋友的可能性同样很大. 在该模型中, 从每个个体发布的消息集合中抽取该个体的主题分布, 同时计算好强度比重, 由(2)式和(3)式计算相应加入链接信息的 *PageRank* 值如下：

$$PR_j(u)=(1-d)\frac{\theta_{uj}}{\sum_{u^*\in U}\theta_{u^*j}}+d\sum_{v\in Friend(u)}\pi(u,v)PR_j(v) \tag{4}$$

其中, u 以概率$(1-d)$按照某一主题 j 找到新的朋友，以概率 d 依循已有的朋友关系找到新的朋友. 按照这个思想, 针对每个主题得到的 *PageRank* 值可以将个体表示为一个向量 $\overrightarrow{PageRank}$. 按照$(4)$式计算所以个体$U(u_1,u_2,\cdots,u_n)$的向量矩阵如下

$$TFP=\begin{bmatrix}\overrightarrow{PR(u_1)}\\ \overrightarrow{PR(u_2)}\\ \vdots\\ \overrightarrow{PR(u_n)}\end{bmatrix}=\begin{bmatrix}PR_1(u_1) & PR_2(u_1) & \cdots & PR_K(u_1)\\ PR_1(u_2) & PR_2(u_2) & \cdots & PR_K(u_2)\\ \vdots & \vdots & \vdots & \vdots\\ PR_1(u_n) & PR_2(u_n) & \cdots & PR_K(u_n)\end{bmatrix} \tag{5}$$

任意两个个体之间存在链接的概率可以计算对应向量的距离来描述. 距离最短的个体作为可能的新链接, 距离最长的个体作为可能删减的节点. 因此, 我们根据所有的 $TFP(5)$ 矩阵行向量的欧氏距离作为判断加边和减边的原则, 从图 G 的

$TFP(5)$ 中求得这些欧氏距离的排序, 从距离最小的开始加边, 从距离最大的开始减边, 规定每个时间步长可以加减边的条数, 即可得到具体算法.

文献[1]证明向量 $TFP(5)$ 是一个矩阵的特征向量, 即 $TFP(5)$ 是下式的一个解.

$$TFP = d\begin{bmatrix} \pi(u_1,u_1),\pi(u_1,u_2),\cdots,\pi(u_1,u_n) \\ \pi(u_2,u_1),\pi(u_2,u_2),\cdots,\pi(u_2,u_n) \\ \cdots \\ \pi(u_n,u_1),\pi(u_n,u_2),\cdots,\pi(u_n,u_n) \end{bmatrix} TFP + (1-d)\begin{bmatrix} r(u_{1,1}),r(u_{1,2}),\cdots,r(u_{1,K}) \\ r(u_{2,1}),r(u_{2,1}),\cdots,r(u_{2,K}) \\ \cdots \\ r(u_{n,1}),r(u_{n,2}),\cdots,r(u_{n,K}) \end{bmatrix} \quad (6)$$

这里, *topic j* 下,对于任意的个体 u , $r\left(u_{i,\mathrm{j}}\right)=\dfrac{\theta_{uj}}{\sum_{u^*\in U}\theta_{u^*j}}$.同时,文献[1]指出 d 适合0.5时, 本文模型的准确率会随着数据集规模增加而提高.

关于 TFP 的计算可以通过循环迭代进行, 迭代公式即为 (6) 式, 预先给定一个规定的迭代误差 ε, 当迭代过程进行到小于 ε 时停止, 即可得到矩阵 TFP .

5.2.4 TFPR 模型算法

一个朋友关系网络图在程序中对应一个邻接矩阵 A ,矩阵 A 的变化就是网络的变化;特别的我们考虑的朋友关系网络是个无向图,即矩阵 A 是个对称矩阵.

为了考虑随时间的动态变化，令 T 为动态网络结束的时间,在单位时间步长内朋友网络增加边数是 m ,减少的边数是 n .计算模型矩阵 TFP ,通过计算 TFP 各个节点之间欧氏距离,判断是否增边和减边.(如图4)

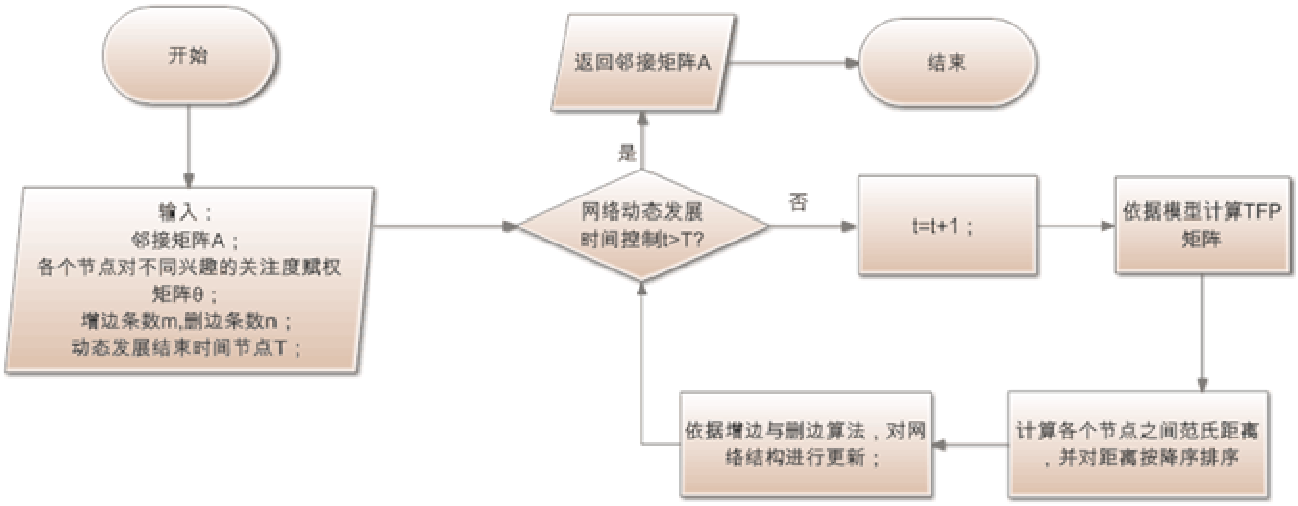


图4 主算法

对于增边和减边这两个过程思路基本一致,只给出增边更新邻接矩阵的算法(图5).

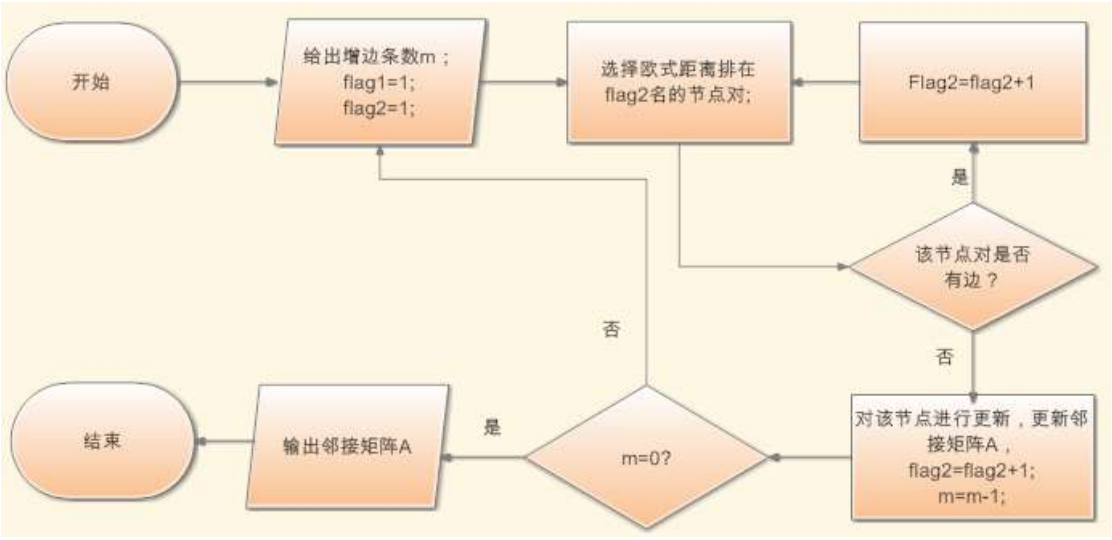
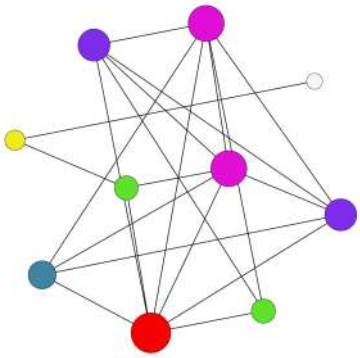


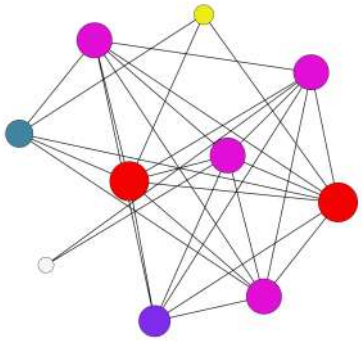
图5 增边算法

5.3 朋友关系网络动态变化实例

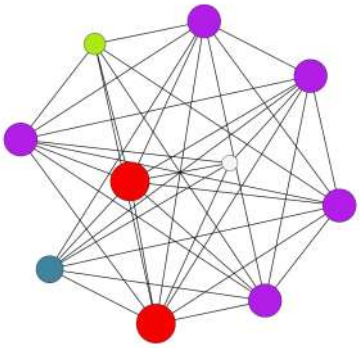
实例 对一个固定总人数 10 人的封闭交友系统进行研究，采用 *XNanalysis* [3] 工具获取这 10 人的人人网好友数据，并作为初始邻接矩阵；采用 *LDAGibbsSampling* 获取兴趣关注度矩阵；这里根据现代大学生交往能力的调查，令每一个时间步长内 $m=3, n=1$；当选取 3 个兴趣主题时，根据 *TFPR* 模型算法此 10 人朋友关系网络动态变化过程如下图6（本文采用 *Gephi* [4] 工具进行生成朋友关系网络，并通过 *Gephi* 进行网络拓扑性质的若干统计指标计算）：



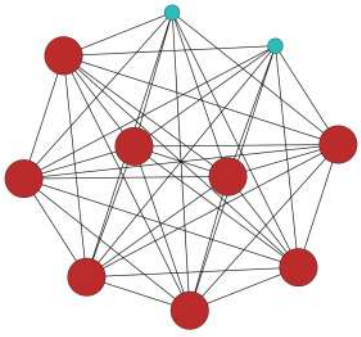
(1) 初始网络（时间步长 $i=0$ 时）



图(2) 时间步长 $i=6$ 时的网络



(3) 时间步长 $i=11$ 时的网络



(4) 稳定网络（时间步长 $i=17$）

图6 个体总数为 10 的好友网络形成过程

从上图变化可以看出, 随着时间步长的增加, 朋友关系网络由开始时的稀疏变得稠密, 而且趋势是有扎堆抱团的现象, 在许多文献中称此现象为富人俱乐部效应. 而我们想刻画网络的拓扑结构的复杂性, 考虑如下几个网络统计特征.

5.3.1 无向网络的度与平均度

度(deg *ree*)是复杂网络的一个基本且重要的统计特征.第 i 个节点的度 k_i 指的是与该节点相连接的其他节点的数目,其数值可以在一定程度上反应该节点在整个网络中的重要程度.全部节点的度的算术平均值称之为网络的平均度,用 $\langle k \rangle$ 来表示,即

$$\langle k \rangle = \frac{\sum_i k_i}{N} \tag{7}$$

其中, N 表示网络中的节点数目;在朋友网络中, k_i 表示一个人的朋友数, $\langle k \rangle$ 则表示网络中平均每人有多少朋友.经过计算, 3,4,5,6 个兴趣主题的朋友网络在不同时间步长上的平均度得图 7 .

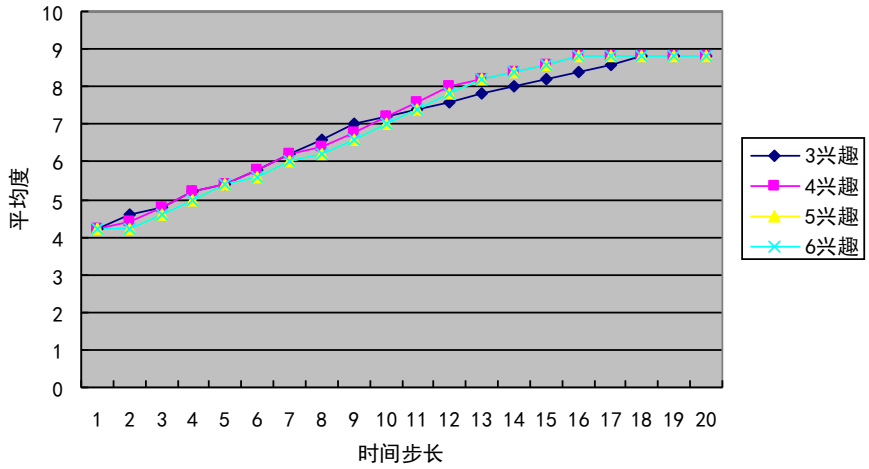


图 7 朋友网络平均度随时间步长变化

可以看出, 随着步长的增加, 即网络的发展, 基于 *TFPR* 模型生成的网络的平均度逐步增加. 并且在一定步长后, 平均度呈常数, 即网络各点的度均不发生变化, 或称之为网络稳定.

同时可以看出, 兴趣主题提前个数不大影响图像平均度的改变, 即兴趣的差异性几乎不影响网络平均度.

5.3.2 无向网络的聚类系数

聚类系数 $(clustering\ coneffficient)$ 也称为群聚系数、集群系数或集聚系数, 用来描述节点的邻点之间也互为邻点的比例. 节点 i 的聚类系数 C_i 等于所有与它相连的节点相互之间所连的边的数目除以这些节点之间可以连出的边数的最大值, 即

$$C_i = \frac{2e_i}{k_i(k_i - 1)} \tag{8}$$

e_i 表示节点 i 的所有邻接点之间实际存在的边数. 显然 $C_i \in [0,1]$. C_i 越接近 1, 表示这个节点附近点有“抱团”的趋势. 整个网络的聚类系数 C 为所有聚类系数的算术平均值, 即

$$C=\frac{1}{N}\sum_{i=1}^{N}C_i \tag{9}$$

经过计算, 3,4,5,6 个兴趣主题的朋友网络在不同时间步长上的平均聚类系数得图8.

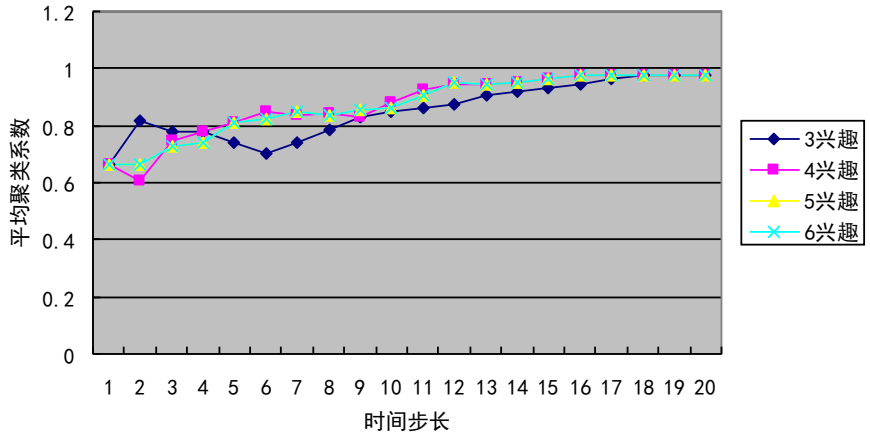


图8 朋友网络平均聚类系数随时间步长变化

可以看出随着网络的发展, 网络的平均聚类系数逐步增加. 并且在一定步长后, 平均聚类系数又归为常数. 同时, 兴趣的差异在开始的几段步长中影响不尽相同, 但随着时间的增长, 该系数都归于一稳定值.

5.3.3 无向网络的平均路径长度

最短路径长度指的是从节点 i 到节点 j 要经历的边的最小值, 记作 $d(i,j)$. 平均路径长度 $\langle d\rangle$ 指的是所有节点对之间最短路径长度的算术平均值, 即

$$\langle d\rangle=\frac{1}{N(N-1)}\sum_{i\neq j}d(i,j) \tag{10}$$

经计算, 3,4,5,6 个兴趣主题的朋友网络在不同时间步长上的平均路径长度得图9.

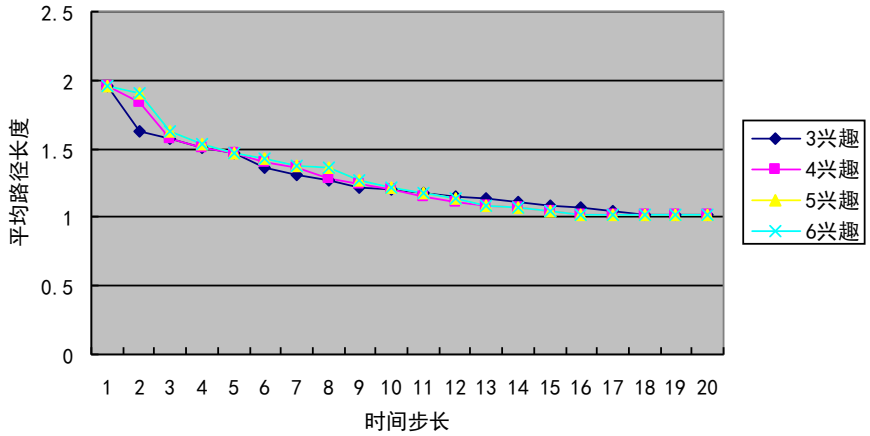
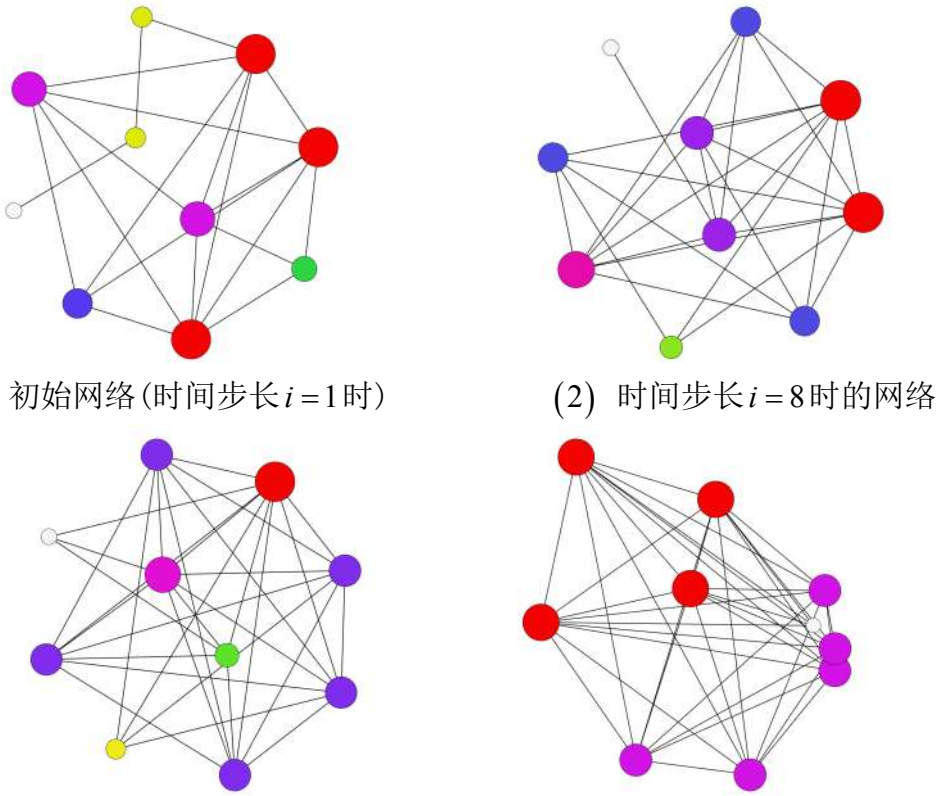


图9 朋友网络平均路径长度随时间步长变化

由上图可知, 朋友关系网络的平均路径长度是趋于 1 的, 就好像中国人那一句话“多一个朋友多一条道路”, 正好能从此看出. 兴趣的差异性显然对平均路径长度影响较弱.

5.4 时代因素对朋友关系网络的影响

题目中提到 1980 年的朋友关系网络的形成和动态变化和 2015 年有什么差别，我们同样是看到网络的拓扑性质的复杂性，使用统计指标进行对比. 首先，1980 年的人们交往交流并不能想如今这样方便快捷，如是假设 1980 年相比如今交友数量的相对较少关系，模拟 1980 年每个时间步长新增边条数变为 $m=2$；同时改革开放前，人们的精神生活并不是十分丰富，很少有人能兼顾多个兴趣主题，如此假设最简单的情况，提前主题分布时只选取一种兴趣主题. 同样将实例中的 10 个人作为初始网络，生成网络的形成和发展图 10.



(1) 初始网络(时间步长 $i=1$ 时)　(2) 时间步长 $i=8$ 时的网络

(3) 时间步长 $i=15$ 时的网络　(4) 稳定网络(时间步长 $i=23$)

图10 个体总数为 10 的好友网络形成过程

显然，1980 年的朋友关系网络到达稳定的时间花费更多得步长，这主要是由于交际范围有限导致的；由于交际范围的限制，人们要形成自己稳定的交际网络，需要长时间的通信和联系的现象也是当时改革前的实际类似情况.

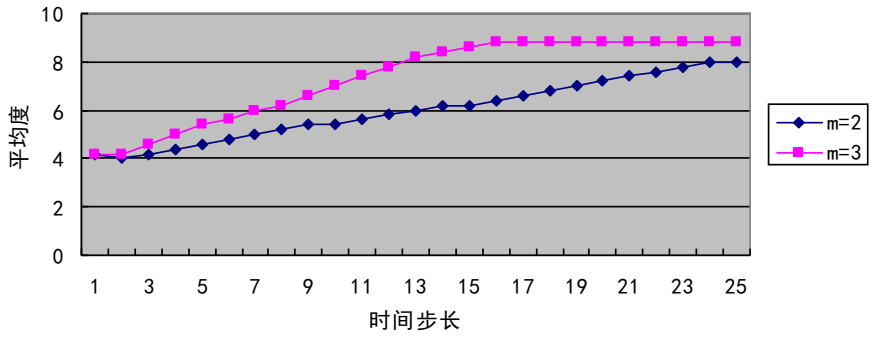


图11 1980 年和 2015 年朋友关系网络的平均度变化

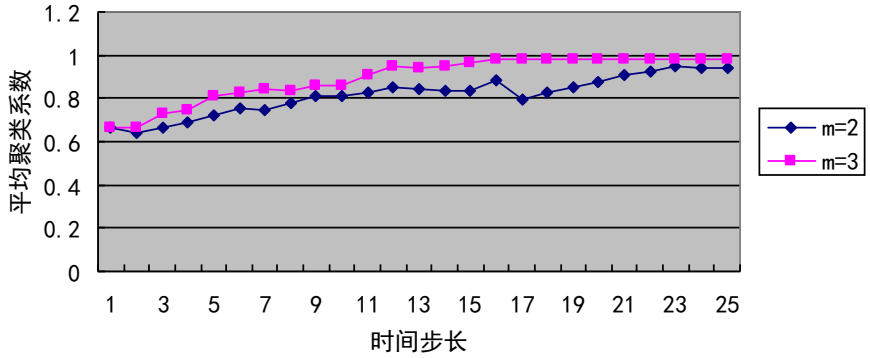


图12 1980 年和 2015 年朋友关系网络的平均聚类系数变化

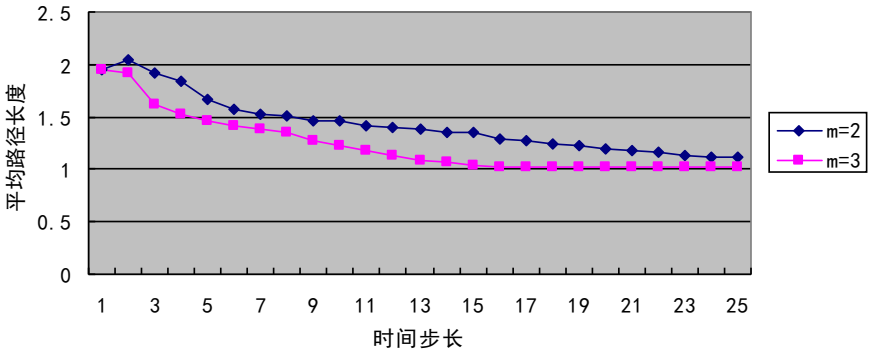


图12 1980 年和 2015 年朋友关系网络的平均路径长度变化

从以上三个图中可以看到, 1980 年朋友关系网络较 2015 年朋友关系网络稳定下来实际较长, 2015 年的朋友关系网仅需 17 个时间步长, 而 1980 年需要 23 个步长, 并且 1980 年较 2015 年平均度和平均聚类系数比较小, 平均路径长度比较大, 反应了在同一时间步长内其朋友网络的复杂程度较低.

5.5 改进的 BA 无标度模型

5.5.1 BA 模型的简介

BA 模型主要是用来说明：假设初始网络中有 m_0 个节点，在每个时间步长上，对新节点连 m 个边($m \le m_0$)节点到已经存在的节点上.择优的体现:假定一个这样的概率 Π ,即新的节点能够连接到节点 i 依赖于节点 i 的度 k_i 的择优概率, $\Pi(k_i) = \dfrac{k_i}{\sum_j k_j}$.

5.5.2 改进的 BA 模型

考虑到了个体特征(兴趣因素)对网络动态生长的影响.考虑到 *BA* 无标度模型没有考虑到删边的情况,而在实际的朋友关系网络中,删边的出现很常见，基于对个体特征因素,本模型算法考虑到了删边的情况,具体如下:

①.基于原 *BA* 模型的流通性 k_i 与本问题的个性特征,引入参数 k_i' ,用它来代替原 *BA* 模型中的 k_i ,计算择优概率,公式如下：

$$k_i' = d_2 \sum_{j=1, j\neq i}^{N} x_{ij} + (1-\mathrm{d}_2)k_i, \qquad \Pi(k_i') = \frac{k_i'}{\sum_j k_j'}. \tag{10}$$

其中, $d_2 = 0.5$, 它是个常数.

x_{ij} 表示节点 i 与节点 j 之间由于兴趣一致性产生的吸引程度.首先，得到个体关于不同主题的关注度矩阵 θ,通过矩阵 θ,为了能刻画个体的兴趣点一致性,计算出个体之间关于主题的欧氏距离矩阵 D.

$$\forall i, j \quad d_{ij} = norm(\theta_i - \theta_j), \quad D = \left\{ d_{ij} \middle| (i, j) \right\}$$

考虑到欧氏距离与度 k_i 数量级上存在巨大差异,我们将欧氏距离矩阵 D 元素的数值域 $[d_{\min}, d_{\max}]$ 映射到 $[0, \max(k)]$ (其中 $\max(k)$ 表示所有节点度的最大值),公式为:

$$F(\mathrm{D}) = \frac{\max(k)}{d_{\max} - d_{\min}} (d - d_{\min}).$$

又由于兴趣越一致的个体其欧氏距离越小,在实际朋友网络中,类比流通性 k_i (即节点的度)的含义,应该是兴趣越一致的个体之间,其相互吸引程度越高,即 x_{ij} 越大.它们呈负相关.故若用欧氏距离矩阵 D 刻画 x_{ij} ,应该有必要的数学处理,处理过程如下:

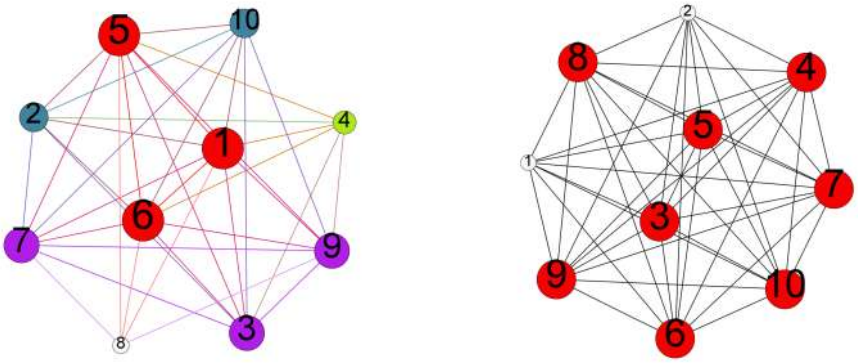
$$x_{ij} = \max(k) - F\left(D\right)_{ij}$$

②.关于删边算法[5]:

在原网络中删除 n 条旧连边: 删除的连线的两个端点均以反择优概率被选取。

5.5.3 模型的解释

改进的 *BA* 模型对实例中 4 个兴趣主题进行模拟计算, 并和 *TFPR* 模型进行比较.



(1) 改进 *BA* 模型稳定状态　　(2) *TFPR* 模型稳定状态

图11 稳定状态时朋友关系网络

可以明显看出改进 *BA* 模型形成的网络边数较 *TFPR* 模型少,通过改进的 *BA* 模型择优概率公式可以看出随着时间步长的增长,兴趣因素所占比重会越来越小,故综合来看在网络结构的发展上看,改进 *BA* 模型较 *TFPR* 模型保守.

六、模型的评价及推广

6.1 TFPR 模型的评价

6.1.1 模型的优点

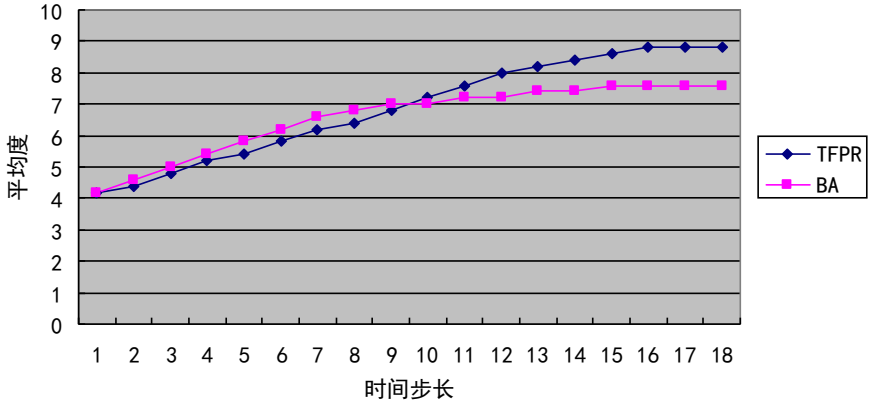


图12 TFPR 和改进 BA 模型平均度变化

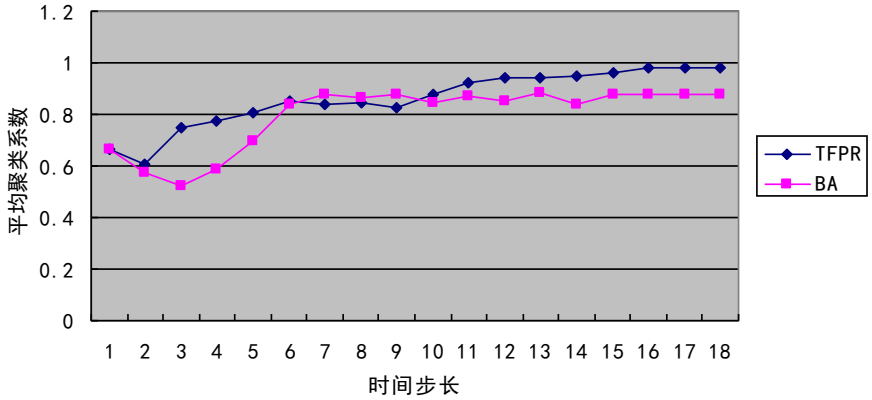


图13 TFPR 和改进 BA 模型平均聚类系数变化

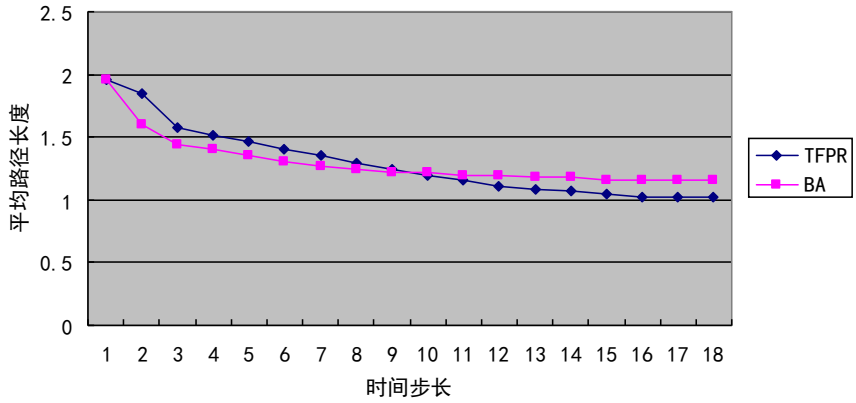


图14 TFPR 和改进 BA 模型平均聚类系数变化

从上面的几个图可以看出无论是*TFPR*模型还是改进*BA*模型随都显示出平均度的增加、平均聚类系数的递增趋于 1,平均路径长度递减趋于 1.显示这两个模型刻画朋友关系网络形成的基本特点,即小世界模型.

6.1.1 模型的不足

模型没有考虑兴趣变化的情况,没有对开放系统进行测试,和现实生活中有一定的差距.同时仅对小规模朋友进行了实例验证,缺少大规模的实验.

6.2 模型的推广

朋友网络是复杂网络,输入社科类问题.该模型的建立对朋友推荐和社会舆论控制都有一定的借鉴作用.能够看出舆论主题队朋友关系网络的影响.

七、参考文献

[1] 尚燕敏,张鹏,曹亚男, 融合链接拓扑结构和用户兴趣的朋友推荐方法,通信学报, Vol.36,No.2,2015/2.

[2]yangliuy,LDAGibbsSampling,https://github.com/yangliuy/LDAGibbsSampling,2015/5/17.

[3]XNanalysis,http://donghaoren.org/projects/xnanalysis,2015/5/17.

[4]Gephi,http://gephi.github.io/,2015/5/16.

[5] 贾秀丽,蔡绍洪.张芙蓉.一种动态的无标度网络模型[J].四川师范大学学报:自然科学版,2009(6):839-842.

[6] Barabási A L, Albert R. Emergence of scaling in random networks[J]. science,1999,286(5439):509-512.

八、附录

使用的软件: Matlab

8.1 TFPR 模型代码:

(1).初始数据:

```
A=[  0     1     1     1     1     0     0     0     0     0
     1     0     1     1     1     1     0     0     1     1
     1     1     0     1     1     1     0     0     0     1
     1     1     1     0     1     1     0     0     0     0
     1     1     1     1     0     1     0     0     0     1
     0     1     1     1     1     0     0     0     0     1
     0     0     0     0     0     0     0     1     0     0
     0     0     0     0     0     0     1     0     1     0
     0     1     0     0     0     0     0     1     0     0
     0     1     1     0     1     1     0     0     0     0];
cita=[    0.0493    0.0798    0.0341    0.2601
         0         0    0.0048         0
         0    0.0050    0.0041    0.0041
    0.0401    0.0115    0.0062    0.0080
    0.0138         0    0.0080    0.0060
         0    0.0049    0.0193         0
    0.0216    0.0368    0.0073    0.0057
         0    0.0039    0.0041         0
    0.0068    0.0074    0.0048         0
         0    0.0158    0.0062    0.0108];
```

(2)主程序:

```
%main_mathorcup:  pagerank 思路;
clc,clear
%load('qqq.mat');
load('main_mathorcup_data1.mat');
A=evalin('base','A');
cita=evalin('base','cita');
out_data=cell(21,1);

    for i0=1:21
        TFP=fun_TFP(A,cita),

        %1.distance's calculate
        dist=dist_cal(TFP),

        %2.排序; rsp==reshape; 升序;
        [dist_tril]=paixu(dist),

        %3.增边子程序;
        m=3,
        [A1]=add_path(m,A,dist,dist_tril),

        A=A1,  %对 A 更新;
        %4.减边子程序

        %减边子程序;
        n=1;
        [A2]=delete_path(n,A,dist,dist_tril);
        A=A2,  %对 A 更新,

        display('当前网络已经发展如下次数'),
        i0,
        [index]=A_analysis(A);
        %Index=[Index,index],
       out_data{i0,1}=index;
    end
 [y]=myoutput(out_data),
```

(3)各部分子程序

```
function [index]=A_analysis(A)
%find the element who is 1;
temp=tril(A,-1);
temp=temp+temp';
[m1,n1]=find(temp==1);
```

```
index=[m1,n1];
end

function [A1]=add_path(m,A,dist,dist_tril);
%增边，更新 A

flag2=1,   %设置目标：对增多边，查找第二条边的起点更新，更新后的值，靠它标记;
flag1=1;   %help flag1 update!
for j0=1:m
    %注：暂时不考虑存在距离相等的情况
    if flag2==length(dist_tril)
        A1=A;
        break; %跳出最外层的 for 循环;
    else
        for i0=flag2:length(dist_tril)
            [m1,n1]=find(dist_tril(i0,1)==dist), %返回坐标
            if A(m1,n1)==1
                flag1=flag1+1,
                flag2=flag1, %更新 flag2;

                %对一个特殊情况的考虑;
                if i0==length(dist_tril)  %考虑该组合下最后一对节点也已经有边的情况
                    A1=A,                      %说明该组距离组合下，不能再加边了，无循环的必要了;
                    flag2=i0,              %即不再查找第二条边;
                    break;
                else
                    ;
                end
                continue;       %考虑下一对节点情况;
            else
               A(m1,n1)=1,
               A(n1,m1)=1,
               A1=A,
               flag1=flag1+1,
               flag2=flag1,           %对下次应该的循环“对序号”更新;
               break;   %跳出内层 for 循环
            end
        end
    end
end
end
```

```
function [A2]=delete_path(n,A,dist,dist_tril);
%增边，更新 A

dist_tril=sort(dist_tril,'descend'),    %对 dist_rsp 改为降序

flag2=1,    %设置目标：对增多边，查找第二条边的起点更新，更新后的值，靠它标记;
flag1=1;    %help flag1 update!
for j0=1:n
    %注：暂时不考虑存在距离相等的情况
    if flag2==length(dist_tril)
        A2=A;
        break; %跳出最外层的 for 循环;
    else
        for i0=flag2:length(dist_tril)
            [m1,n1]=find(dist_tril(i0,1)==dist), %返回坐标
            if A(m1,n1)==0          %无边，找有边的删;
                flag1=flag1+1,
                flag2=flag1,         %更新 flag2;

                %对一个特殊情况的考虑;
                if i0==length(dist_tril)  %考虑该组合下最后一对节点也已经有边的情况
                    A2=A,                       %说明该组距离组合下，不能再加边了，无循环的必要了;
                    flag2=i0,                %即不再查找第二条边;
                    break;
                else
                    ;
                end
                continue;        %考虑下一对节点情况;
            else                 %说明有边，下面在删之
               A(m1,n1)=0,
               A(n1,m1)=0,
               A2=A,
               flag1=flag1+1,
               flag2=flag1,            %对下次应该的循环“对序号”更新;
               break;    %跳出内层 for 循环
            end
        end
    end
end
```

```
end



function [ dist ] = dist_cal( TFP )
[m,n]=size(TFP);
for i=1:m
    for j=1:m
        if i==j
            dist(i,j)=1000;
        else
            dist(i,j)=norm(TFP(i,:)-TFP(j,:));
        end

    end
end
end


function [PI]=fun_PI(stren)
N=length(stren);
PI=zeros(N,N);
for i=1:N
    for j=1:N
        if i==j
            PI(i,j)=0;
        else
        PI(i,j)=stren(i,j)/sum(stren(i,:));
    end
end
end


function [ R ] = fun_R( cita )
[m,n]=size(cita);
for i0=1:m
    for j0=1:n
        R(i0,j0)=cita(i0,j0)/sum(cita(:,j0));
    end
end
end
```

```
function [ stren ] = fun_stren( A )
%UNTITLED8 Summary of this function goes here
%   Detailed explanation goes here
N=length(A);
stren=zeros(N,N);
for i=1:N
    for j=1:N
        if i==j
            stren(i,j)=0;
        else

stren(i,j)=length(intersect(find(A(i,:)==1),find(A(j,:))))/(sum(A(i,:
))+sum(A(j,:)));
    end
    end
end


function [ TFP ] = fun_TFP( A,cita )
%1.A-->stren
stren=fun_stren(A);

%stren-->PI;
PI=fun_PI(stren);

%cita-->R
R=fun_R(cita);

%迭代计算;
TFP=repeat_cal(PI,R);
end


function [ ] = mat2excel( i,index )
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here
i=num2str(i),
str1=strcat('Sheet',i ),
xlswrite('data1',index,str1)
end


function [ ] = mat2excel_1( i,index )
```

```
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here
i=num2str(i),
str1=strcat('data',i ),
xlswrite(str1,index),

end

function [ y ] = myoutput( x )
%把输出数据 2——>mat;
m=length(x);
for i0=1:m
    display('当前的代数：');
    display(i0);
    index=cell2mat(x(i0,1)),
    %把每下标导入到 Excel 中的子程序;
    %mat2excel(i0,index);
    mat2excel_1(i0,index);


    n(i0,1)=length(index);

    if i0>=3
        if   n(i0,1)   ==   n(i0-1,1)   &   n(i0-1,1)==n(i0-2,1)   &
n(i0-1,1)==n(i0-2,1)
            display('网络发展已经稳定');
            display('边数总数：')
            display(n(i0,1)/2);
        else
            ;
        end
    end

end
y=1;
end



function [ TFP ] = fun_TFP( A,cita )
%1.A-->stren
stren=fun_stren(A);

%stren-->PI;
```

```
PI=fun_PI(stren);

%cita-->R
R=fun_R(cita);

%迭代计算;
TFP=repeat_cal(PI,R);
end

PI=
end
function [ dist_tril ] = paixu( dist )
%goal:排序，只要下三角矩阵元素;
%输出：升序的距离元素顺序，列向量;
m=length(dist);
temp=1;
for i0=2:m
    for j0=1:i0-1
        dist1(temp,1)=dist(i0,j0);
        temp=temp+1;
    end
end
dist_tril=sort(dist1);
end


function [ TFP ] = repeat_cal( PI,R )

%数据部分初始化;
[m,n]=size(R);
TFP=3*ones(m,n);
X=ones(m,n);
TFP=0.5.*PI*X+0.5*R;
flag=2;
while(flag>=1)
    if norm(TFP-X,1)<=0.1
        flag=0;
    end
    X=TFP;
    TFP=0.5.*PI*X+0.5*R;
    %norm(TFP-X),
end
end
```

8.2 BA 模型代码:

(1)主程序：

```
%main:BA 模型
clc,clear;
rand('state',sum(clock));

%initial value
load('main_mathorcup_data1.mat');
A=evalin('base','A');
cita=evalin('base','cita');
k=sum(A');
out_data=cell(25,1);


%k_pie 计算子程序;
%只需要执行一次;
[k_pie]=k_pie_cal(cita,k);

%main 主体部分;
for  i0=1:25
    %择优子程序;
    m=3;
    [A]=select_good(m,A,k_pie);

    %删边子程序;
    n=1;
    [A]=delete_path(n,A,k_pie);

    %局部迭代计算步长显示;
    display('当前网络已经迭代次数：'),
    display(i0);

    %网络稳定性分析子程序;
    [index]=A_analysis(A);
        %Index=[Index,index],
     out_data{i0,1}=index;
end
```

（2）各部分子程序

```
[y]=myoutput(out_data),
```

```
function [k_pie]=k_pie_cal(cita,k);
[m,n]=size(cita);
for i=1:m
    for j=1:m
        dist(i,j)=norm(cita(i,:)-cita(j,:));
    end
end
dist,
%计算 norm_max;
norm_max=max(max(dist)),
%计算 norm_min;
temp=dist,
N=length(temp),
temp=temp+diag(100000*ones(N,1)),
norm_min=min(min(temp)),
%k_max 计算;
%注：此值只计算一次，之后不再更新;
k_max=max(k);

%every dist(i,j)-->G 域：[0,max(k)]
g_dist=k_max*(dist-norm_min)/(norm_max-norm_min),
fact_grade=k_max-g_dist,
%去掉对角线的影响;
fact_grade=fact_grade+diag(-diag(fact_grade)),
k_pie=0.5*sum(fact_grade')+0.5*k,
end
function [ A ]=select_good(m,A,k_pie)

rand('state',sum(clock));

%随机选择主连接节点集 NS1;0
%core:防节点重复;
while(1)
    N=length(A),
    NS1=randi(N,m,1),
    y=is_mat_repeat(NS1);
    if y==1
        continue;
    else
       break;
    end
end
```

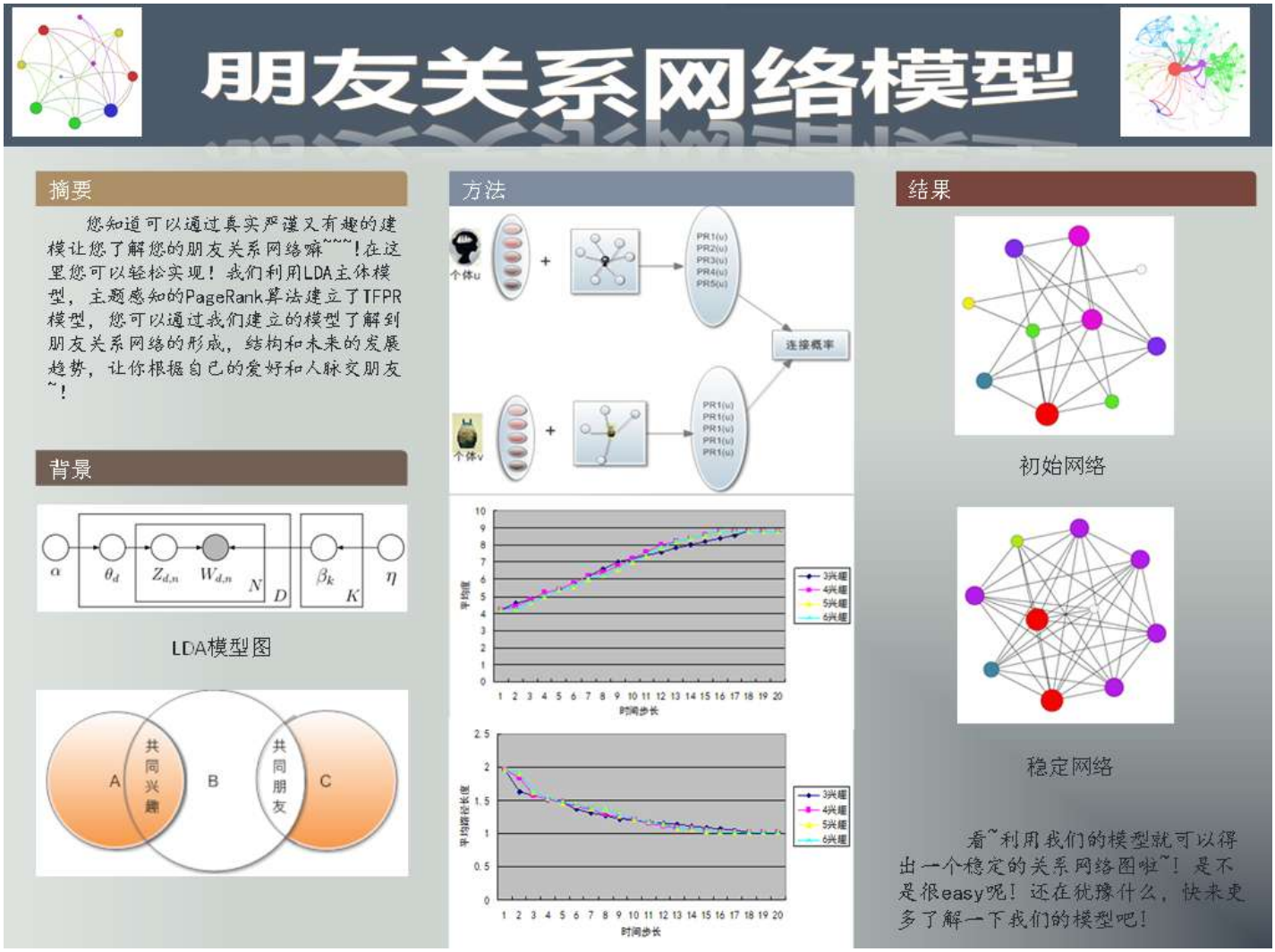
```
%随机选择从连接节点;
for j0=1:m
    flag1=0;  %flag1 的初始化;
    i0=NS1(j0,1),

    y=1;
    NS2=(1:N)',    %建立第个节点的不能边连边的节点集合，该过程的初始节点集;

    while(y==1 & flag1==0)
        select_node=randi(N,1),
        %是否重选节点的判断子程序;
        i0,
        [y]=is_repeat_select_sg(i0,select_node,A),
        if y==1
            [ NS2 ]=update_NS2(select_node,NS2),  %更新 NS2 节点集;
            if length(NS2)==0
                break;    % consider next node;
            else
                continue;    %没有执行必要，end this time loop;
            end
        else
            ;
        end

        p1=k_pie(select_node)/sum(k_pie),

        if randi(100,1)/100<p1
           m=m-1,
           A(i0,select_node)=1,
           A(select_node,i0)=1,
           k=update_k(A),
        else
            y=1,    %说明需要重选节点，计算;
        end
    end
end
end
function [ A ] = delete_path( n,A,k_pie )
rand('state',sum(clock));
%随机选择主删节点集 NS1;
%core:防节点重复;
while(1)
    N=length(A),
```

```
    NS1=randi(N,n,1),
    y=is_mat_repeat(NS1);
    if y==1
        continue;
    else
       break;
    end
end

%随机选择从删接节点;
for j0=1:n
    flag1=0;  %flag1 的初始化;
    i0=NS1(j0,1),

    y=1;
    NS2=(1:N)',   %建立第个节点的不能边删边的节点集合，该过程的初始节
点集;

    while(y==1 & flag1==0)
        select_node=randi(N,1),
        %是否重选节点的判断子程序;
        i0,
        [y]=is_repeat_select_dp(i0,select_node,A),
        if y==1
            [ NS2 ]=update_NS2(select_node,NS2),  %更新 NS2 节点集;
            if length(NS2)==0
                break;    % consider next node;
            else
                continue;    %没有执行必要，end this time loop;
            end
        else
            ;
        end

        p1=k_pie(select_node)/sum(k_pie),

        if randi(100,1)/100>p1
           n=n-1,
           A(i0,select_node)=0,
           A(select_node,i0)=0,
           k=update_k(A),
        else
            y=1,   %说明需要重选节点，计算;
        end
```

```
    end
end
end
function [index]=A_analysis(A)
%find the element who is 1;
temp=tril(A,-1);
temp=temp+temp';
[m1,n1]=find(temp==1);
index=[m1,n1];
end
function  [k_pie]=k_pie_change(x,k);
b0=0.5;
b1=0.5;
[m,n]=size(x);
for i0=1:m
    k_pie(i0,1)=b0* sum(x(i0,:))+b1*k(i0,1);
end
end

function [ y ] = myoutput( x )
%把输出数据 2——>mat;
m=length(x);
for i0=1:m
    display('当前的代数：');
    display(i0);
    index=cell2mat(x(i0,1)),
    %把每下标导入到 Excel 中的子程序;
    %mat2excel(i0,index);
    mat2excel_1(i0,index);

    n(i0,1)=length(index);

    if i0>=3
        if   n(i0,1)   ==   n(i0-1,1)   &   n(i0-1,1)==n(i0-2,1)   &
n(i0-1,1)==n(i0-2,1)
            display('网络发展已经稳定');
            display('边数总数：')
            display(n(i0,1)/2);
        else
            ;
        end
    end
end
y=1;
```

end

九、海报


朋友关系网络模型
摘要
您知道可以通过真实严谨又有趣的建模让您了解您的朋友关系网络嘛~~~~!在这里您可以轻松实现！我们利用LDA主体模型，主题感知的PageRank算法建立了TFPR模型，您可以通过我们建立的模型了解到朋友关系网络的形成，结构和未来的发展趋势，让你根据自己的爱好和人脉交朋友~!
背景
LDA模型图
共同兴趣
共同朋友
A
B
C
方法
个体u
个体v
连接概率
PR1(u)
PR2(u)
PR3(u)
PR4(u)
PR5(u)
平均聚集度
时间步长
3兴趣
4兴趣
5兴趣
6兴趣
平均路径长度
结果
初始网络
稳定网络
看~利用我们的模型就可以得出一个稳定的关系网络图啦~！是不是很easy呢！还在犹豫什么，快来更多了解一下我们的模型吧！