基于多核心节点的增量式动态社区发现算法

陈晶[1]，刘志君[2,3]，杨新宇[2,3]，刘洺辛[4]，刘苗苗[5]

（1. 广东海洋大学数学与计算机学院，广东 湛江 524088；2. 燕山大学信息科学与工程学院计算机系，河北 秦皇岛 066004；
3. 河北省虚拟技术与系统集成重点实验室，河北 秦皇岛 066004；4. 广东海洋大学电子信息工程学院，广东 湛江 524088；
5. 东北石油大学计算机与信息技术学院，黑龙江 大庆 163318）

摘 要：针对动态社区发现算法通常基于社区结构平稳变化的假设，而难以应对演化过程中可能出现的大量社区消亡或涌现等突发事件的问题，提出了一种基于多核心节点的增量式动态社区发现算法 MCNIDCD。首先，将核心节点分为扩散型和内聚型，制定 4 种增量更新策略。其次，通过局部更新调整节点社区归属，并采用增量模块度方法优化社区结构。最后，实现社区合并。在人工和真实网络上对该算法的性能进行了评估，实验结果表明，在对比目前相关动态社区检测算法时，在人工网络仿真环境中，MCNIDCD 算法表现出与社区演化规律的高度契合性；在真实网络实验中，MCNIDCD 算法在模块度性能指标上平均提升了 28%，并且在稳定性方面具有良好的优势，其优势对于研究动态社区演化过程具有重要的意义。

关键词：核心节点；增量式；网络演化；社区发现

中图分类号：TP393

文献标志码：A

DOI: 10.11959/j.issn.1000−436x.2024070

Incremental dynamic community discovery algorithm based on multi-core nodes

CHEN Jing[1], LIU Zhijun[2,3], YANG Xinyu[2,3], LIU Mingxin[4], LIU Miaomiao[5]

1. College of Mathematics and Computer Science, Guangdong Ocean University, Zhanjiang 524088, China
2. College of Information Science and Engineering, Yanshan University, Qinhuangdao 066004, China
3. Key Laboratory of Virtual Technology and System Integration, Qinhuangdao 066004, China
4. College of Electronic and Information Engineering, Guangdong Ocean University, Zhanjiang 524088, China
5. School of Computer & Information Technology, Northeast Petroleum University, Daqing 163318, China

Abstract: A new incremental dynamic community discovery algorithm MCNIDCD based on multiple core nodes was proposed to address challenges in dynamic community discovery. It adapted to sudden events like the emergence or disappearance of communities during evolution. MCNIDCD categorized core nodes into diffusion and cohesion types, and devised four incremental updating strategies. It adjusted node community membership locally and optimized community structure using an incremental modularity method to facilitate community merging. Evaluation on artificial and real networks shows MCNIDCD's high conformity to community evolution patterns. In real network experiments, MCNIDCD exhibits a 28% average improvement in modularity performance and significant stability advantages. Its superiority is important for studying dynamic community evolution.

Keywords: core node, incremental, network evolution, community discovery

---

基金项目：国家自然科学基金资助项目（No.62172352, No.42306218）；中央省部共建基金资助项目（No.226Z0102G, No.226Z0305G）；河北省自然科学基金资助项目（No.2022203028, No.F2023407003）；广东海洋大学科研启动基金资助项目（No.060302102304）

Foundation Items: The National Natural Science Foundation of China (No.62172352, No.42306218), The Central Government Guides Local Science and Technology Development Fund Projects (No.226Z0102G, No.226Z0305G), The Natural Science Foundation of Hebei Province (No.2022203028, No.F2023407003), Guangdong Ocean University Scientific Research Start-up Fund Project (No.060302102304)

0　引言

社区发现的研究具有很多现实的意义，人们通过挖掘和识别社区结构可以了解网络中含有的丰富内容，理解网络社区组织结构的发展规律以及它们之间拓扑结构的相互关系等。传统的社交网络分析方法专注于静态网络[1]的研究，其研究热点体现在如何在网络中识别出有意义的社区结构。但是在许多复杂网络中，实体间的交互随时间动态变化，而网络变化必然导致其中的社区结构也随着时间发生变化。因此，仅对网络进行静态社区划分已不能准确地刻画网络的真实性。目前，针对动态社区发现的方法大多是将动态网络按时间顺序分成一系列连续的时间片网络快照，且研究方法分为独立静态式划分[2-3]、基于演化聚类划分[4-6]和增量式划分[7-9]。由于增量式划分方法相比于前 2 种方法能体现出较好的计算性能优势，为此，许多研究者从不同的角度提出了增量式动态社区发现算法。但是，其中大部分研究方法采用了如下假设：在动态社区演化过程中，多数的拓扑结构保持相对稳定，仅有小部分结构会发生变化，因此，在识别前一时刻社区结构的基础上，仅对改变的网络结构部分进行重计算，而认为其余结构保持不变，以达到提升计算效率的目的。实际上，在动态社区的演化过程中，增量节点的邻居节点也会受到影响而改变社区归属，现有的增量式社区发现算法仅对增量节点进行重新计算，往往会存在以下问题：1)计算效率的提高以降低准确性为代价；2)在对连续的时间片网络增量更新时，社区发现结果容易产生错误累加。

为了解决上述问题，本文提出了一种基于多核心节点的增量式动态社区发现算法 MCNIDCD。首先，该算法不仅考虑了增量节点，还考虑了不同情况下受到增量影响的节点；其次，基于定义的两类核心节点和最大增量模块度，判定节点的社区归属；最后，在不同规模的数据集中对本文提出的 MCNIDCD 进行了实验验证，实验结果表明，该算法在保证运算效率的同时，能准确地划定网络变化的影响范围，并有效地挖掘出动态网络的社区结构，解决了现有增量式社区发现算法准确率低的问题。本文贡献如下。

1) 定义了两类核心节点：扩散核心节点和内聚核心节点。当前的动态社区检测算法普遍依赖于特定的核心节点来驱动社区结构的演化更新，但是各类核心节点的功能差异显著，进而无法准确描述扩散性和内聚性产生的不同影响效果。因此，本文重点关注动态社区发现时扩散和内聚两类核心节点的作用机制及其对社区结构演化的影响。一类是传播效果显著的核心节点，其代表了网络中节点的信息扩散特性；另一类是破坏网络连通性的核心节点，其突显了网络内部节点间的内聚属性。

2) 基于定义的两类核心节点设计局部更新策略。将增量分为 4 种类型：增加节点、删除节点、增加边和删除边。本文将核心节点和增量类型相结合，针对不同的子场景设计了不同的策略更新方式，最大限度地避免了局部更新导致的时间错误积累。

1　相关工作

近年来，越来越多的研究工作致力于在动态时间网络中检测社区。聚类方法由于其天然的划分数据集的优势，在动态社区检测中占据了重要的地位。为此，Li 等[10]通过融合基于网络嵌入的进化非负矩阵因式分解，提出了一种用于动态社区检测的新型算法。Ren 等[11]提出了一种基于概率矩阵的新型谱聚类算法进行社区检测。Al-Sharoa 等[12]提出了一种低秩近似演化聚类方法，该方法结合低秩分解和子空间学习的技术，解决了演化聚类平滑分配问题。此外，该方法引入了一个用于跟踪动态社区结构变化的损失函数。Jia 等[13]通过结构相似度和属性相似度计算节点间的静态相似性，利用历史网络信息对当前网络信息的影响获取节点间的动态相似性，建立了加权动态社交网络，并利用 Louvain[14]实现动态社区检测。Gao 等[15]提出了一种非负矩阵分解的新型动态社区检测模型，不仅可以跟踪时间演变，还可以保持检测社区的质量。

标签传播技术源自半监督学习的技术，通过模拟标签信息在网络中的传播过程，能有效捕获社区的动态演变特征。Wang 等[16]提出了基于演化聚类框架和标签的群体智能方法，并融合标签传播和遗传算法改进的离散粒子群算法来评估社区发现的准确性。Xie 等[17-18]基于 LabelRank 提出了一个面向动态社区检测的标签传播增量聚类方法 LabelRankT。该方法结合历史社区结构信息推断当前的社区结构，研究成果适用于大规模动态网络社区发现的在线分布式算法。Sattari 等[19]提出了一种基于标签传播算法和级联信息扩散模型的方法来发现重叠社区。Berahmand 等[20]提出了一种新的标

签算法，该算法不仅能检测社区结构的内聚性和属性的同质性，并在一定程度上解决了社区发现结果不稳定和质量低的问题。

多目标优化作为一种有力工具被应用于动态社区检测，旨在平衡多个评价指标，比如模块度、稳定性以及覆盖率等。Besharatnia 等[21]基于 GWO（grey wolf optimizer）和 LP（label propagation）算法提出了一种改进的多目标和元启发式的 IGWO-LP 算法。Abbood 等[22]基于隐马尔可夫模型提出动态社区检测多目标进化算法。该算法使用多目标进化算法和 Viterbi 算法来制定目标函数，并为聚类动态网络提供随时间变化的平滑性。Jiang 等[23]为了解决不同权重参数易于影响社区检测结果进而导致检测结果在目标空间中分布不均匀的问题，提出了一种基于协同粒子群多目标优化的动态重叠社区检测算法。为了解决时间漂移准确率较低和超参数不稳定的问题，Gao 等[24]提出了一种新的多目标离散粒子群优化分解策略来平衡准确性和稳定性。

基于增量动态社区检测的一系列研究工作致力于实时且高效地处理网络的变化。Chen 等[25]通过构建增强图引入属性中心和归属边，将属性映射到网络中，并通过模块最大化来检测社区。Zardi 等[26]引入传输消息速率的概念，获得了具有相似成员的社区。Rossetti 等[27]提出了一个在线增量聚类动态社区挖掘算法 Tiles，该算法基于模块度增量进行优化，并通过局部计算网络子结构提高了更新效率。Wang 等[28]提出了一种基于拓扑势场的动态重叠社区演化追踪方法。由于该方法在动态演化网络中可能存在同时连接在一起的子图，相关学者针对此问题提出了通过处理子图来检测社区增量的方法和相应的更新策略。郭昆等[29]利用基于密度的方法发现社区增量的变化过程，该方法不仅考虑节点直接邻居的影响，还考虑了间接邻居的影响，并通过迭代更新模块度增益进行社区合并。He 等[30]提出了一种用于时间网络中的动态社区快速检测算法，该算法利用历史社区信息构造了一个小型网络，并采用 Louvain 算法来检测新网络中的社区。

2 问题描述

为了便于对研究问题的阐述，本文引入并描述了相关的符号定义，如表 1 所示。

表 1 本文使用的符号及其描述

符号	描述
GS	演化网络的快照集合
G_t	t 时刻的快照
ΔG_{t+1}	从 t 到 t+1 时刻的增量变化
CS_t	t 时刻检测到的社区结构
C_i^t	G_t 上的第 i 个社区
V_t	G_t 的顶点集
E_t	G_t 的边集
$\mathrm{gatherL}_t$、$\mathrm{disperseL}_t$	G_t 的内聚和扩散核心节点集
$\mathrm{finishL}_t$、$\mathrm{unfinshL}_t$	G_t 的有标签和无标签节点集

动态网络可以表示为时间轴上各个时间片的网络快照的集合 $\{G_1,G_2,G_3,\cdots,G_i,\cdots,G_n\}$，其中 $G_t=(V_t,E_t)$ 表示 t 时刻的网络快照。定义 t 时刻网络快照 G_t 的一个社区划分为 $C_t=\{C_1^t,C_2^t,\cdots,C_k^t\}$，其中，$C_i^t$ 为 t 时刻网络社区集合中的第 i 个社区，k 为社区数目，则动态网络社区发现如定义 1 所示。

定义 1 动态网络社区发现。给定 $t-1$ 时刻的网络快照 G_{t-1} 和对应的网络社区划分 C_{t-1} 以及 t 时刻的网络快照 G_t，求 t 时刻的网络快照 G_t 的社区划分 C_t。

增量社区发现算法需要获得相关时刻发生变化的节点和边，定义 t 时刻增量相关的网络部分为 $\Delta G_t=(\Delta V_t,\Delta E_t)$，其中，$\Delta V_t=(V_{t-1}-V_t)\cup(V_t-V_{t-1})$ 为 t 与 $t-1$ 时刻发生变化的节点集合，$\Delta E_t=(E_{t-1}-E_t)\cup(V_t-V_{t-1})$ 为 t 与 $t-1$ 时刻发生变化的边集合，则增量社区发现如定义 2 所示。

定义 2 增量社区发现。给定 $t-1$ 时刻的网络快照 $C_t=\{C_1^t,C_2^t,\cdots,C_k^t\}$ 和对应的网络社区划分 C_{t-1}，以及 t 时刻的网络快照 G_t 和增量 $\triangle G_t$，求 t 时刻的网络快照 G_t 的一个社区划分 C_t。

3 核心节点社区发现算法

Wang 等[16]指出，在追踪动态社区检测时，采用具有代表性且结构稳定的核心节点集而非全集，可以显著提升检测效果与精确度。不同于依赖全局相似性分析的演化算法，基于核心节点的动态社区发现方法选取并关注网络拓扑结构中部分关键节点来表征各个社区结构，这些被选择的关键节点通常被称为核心节点。该类算法通过深入挖掘和利用

动态网络的内在拓扑特征信息，有效地提高了对动态社区边界划分及其演变过程的识别准确性。

然而，核心节点的不同特性可能对不同类型社区演化事件的揭示产生差异化的影响。具体而言，诸如社区生成、合并及扩张等成长型演化事件，其发生机制往往与那些在网络中展现出内聚集性特征的核心节点紧密相关；相反，社区消失、分裂以及规模缩减等收缩型演化事件，则可能更多地关联到呈现扩散性属性的核心节点。因此，单一类型的核心节点不足以全面精准地捕捉各类社区演化的复杂行为。本文在 4 种增量策略中加入了扩散和内聚 2 种不同特性的核心节点来设计相应的检测策略，其结果对细粒度级别的发现动态社区变化至关重要。

本文提出的基于多核心节点的增量式动态社区发现算法 MCNIDCD 的整体框架如图 1 所示。

首先，在 $t=0$ 的初始化阶段，利用 Louvain 算法生成相对准确的社区结构；当 $t>0$ 时，则可以将前一时刻的社区划分结果作为输入，同时利用分解 2-core 子图的方法获取内聚核心节点集 gatherL^t 和影响力最大化的方式寻找扩散核心节点集 disperseL^t。其次，在局部动态更新策略中，采用 t 时刻继承 $t-1$ 时刻社区划分结果的方式，将继承的节点放入保留历史信息的有标签集合 finishL^t 中。再次，结合 2 种类型的核心节点，即内聚核心节点集 gatherL^t 和扩散核心节点集 disperseL^t，确定其影响节点，并将增量类型分为 4 种形式。通过制定局部动态更新策略的方式，将受到影响的节点放入无标签节点集 unfinishL^t 中。最后，通过增量模块度最大化将中间社区合并成最终社区结果，在提高社区的模块度的同时平衡社区结构和时间的连贯性。这样的策略可以更好地处理动态网络的社区结构划分问题，提高社区划分的准确性和效率。

3.1　核心节点社区表示模型

在传统的动态社区发现算法中，利用局部最优的方式提高模块度值，故动态社区发现需要解决一个明显的悖论，即在不同的网络演化步骤中发现的“最优”划分并不总是形成一个连贯的动态结构，通常将网络社区结构表示为节点与社区间映射关系的集合。

这些方法求解问题的出发点是：社区通常表示为内部联系紧密的节点的集合，社区发现问题归结为如何优化当前网络的整体社区质量。但是在节点关系复杂的社区网络中，既要保证当前社区结构的质量，还要考虑动态社区的平滑性。由于不同特征的核心节点对不同事件的影响程度不同，目前的核心节点算法只使用一类核心节点来推断动态社区发现，使社区结构划分并不准确。因此，本文考虑了两类核心节点，并使核心节点和受影响的节点形成跟随关系。接下来，本文给出社区表示模型的相关定义与描述。

定义 3　节点邻域。节点 x 的邻域定义为

$$N(x)=\{x\}\bigcup \text{Neighborhood}(x) \tag{1}$$

定义 4　增量类型。增量类型指动态网络中出现的 4 种类型变化，包括增加节点、删除节点、增加边和删除边。

增加节点是指在 t 时刻存在且在 $t-1$ 时刻不存在的节点，可以用式(2)表示。

$$\text{AddNs}_t=V_t-V_{t-1} \tag{2}$$

删除节点是指在 t 时刻不存在且在 $t-1$ 时刻存在的节点，可以用式(3)表示。

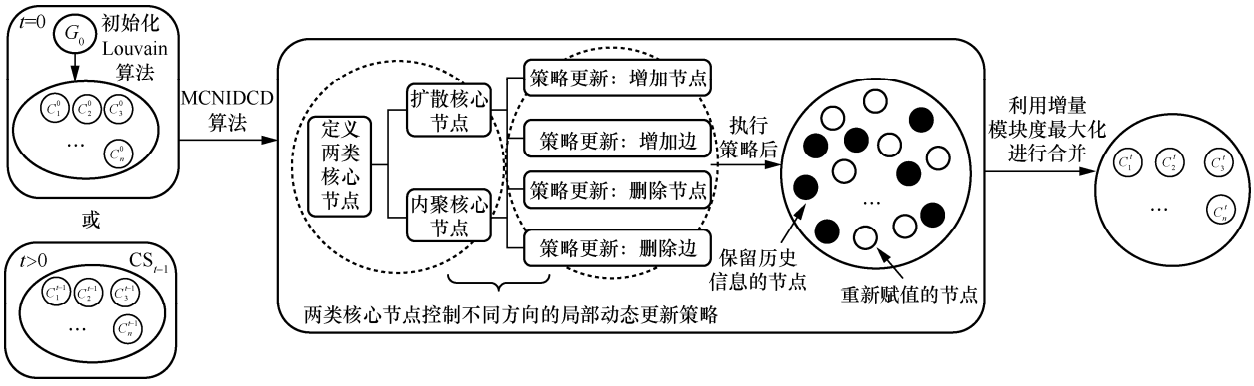


图 1　基于多核心节点的增量式动态社区发现算法的整体框架

$$\mathrm{DelNs}_t = V_{t-1} - V_t \tag{3}$$

增量边分为 2 种情况。第一种是在 t 时刻不存在、在 $t-1$ 时刻存在且边两端的节点在 $t-1$ 和 t 时刻都存在的边，称为增加边，可以用式(4)表示。

$$\mathrm{AddEs}_t = \{(u,v) \mid (u,v) \in E_t - E_{t-1}, u,v \in V_t \bigcap V_{t-1}\} \tag{4}$$

第二种是在 t 时刻存在、在 $t-1$ 时刻不存在且边两端的节点在 $t-1$ 和 t 时刻都存在的边，称为删除边，可以用式(5)表示。

$$\mathrm{DelEs}_t = \{(u,v) \mid (u,v) \in E_{t-1} - E_t, u,v \in V_t \bigcap V_{t-1}\} \tag{5}$$

采用增量类型可以更加准确地描述动态网络的变化过程，从而实现对动态网络的建模和分析。设 E_t 表示在时刻 t 存在的边集合，E_{t-1} 表示在时刻 $t-1$ 存在的边集合，V_t 表示在时刻 t 存在的节点集合，V_{t-1} 表示在时刻 $t-1$ 存在的节点集合。

定义 5 增量模块度。增量模块度描述的是新增加节点与社区的关系，定义如式(6)所示。

$$\Delta Q = \frac{1}{2m}\left[B_{ij} - \left(\frac{k_i, k_j}{2m}\right)\right]\Delta a_{ij} \tag{6}$$

其中，m 为网络中边的数量；k_i 和 k_j 分别为节点 i 和节点 j 的度数；$B_{ij} = \frac{A_{ij}}{2m}$ 为节点 i 和节点 j 之间的边在网络中所占的比例，A_{ij} 为节点 i 和节点 j 之间的边数；$\triangle a_{ij}$ 表示节点 i 和节点 j 之间增加或删除的边的数量，增加为正数，删除为负数。

$\triangle Q$ 的值可以用来评估新增加节点的模块度贡献。如果新增加的节点与现有社区结构关系更紧密，则 $\triangle Q$ 的值将更高，说明新增加节点对社交网络的稠密度和模块化程度的影响更大；反之，如果新增加的节点与社区结构关系稀疏，那么 $\triangle Q$ 的值将更低，说明新增加的节点对社交网络的模块度贡献较小。因此，增量模块度可以更好地理解动态网络的演化过程，从而实现对网络结构和功能的分析和优化。

3.2 增量核心节点影响算法

MCNIDCD 是一种基于核心节点的增量式算法，其目的是在动态网络的增量变化中最大限度地提高社区结构的模块化增益。该算法提出了一个两步方法：首先，根据增量网络变化和前一时刻网络社区结构来初始化中间社区结构；其次，利用增量模块度最大化算法将中间社区进行合并，直到模块化增益达到稳定。在初始化步骤中，MCNIDCD 将增量更改分为 4 种类型，并为每种类型的增量变化设计了对应的策略，以初始化相应的中间社区结构。这些策略旨在以增量最大化为目的，同时避免冗余和重复的计算。与原始初始化步骤的 Louvain 算法相比，MCNIDCD 的初始化步骤基于历史信息，减少了大部分不必要的计算并且增加了社区演化的平滑性，从而在动态网络中检测社区时更加高效和精准。本文提出的 4 种策略介绍如下。

3.2.1 增加边策略

在 t 时刻，节点 i 和节点 j 之间增加了一条边 e_{ij}。根据增加边两端节点在 $t-1$ 时刻是否属于同一社区，可以将这种情况分为 2 个子场景。如果节点 i 和节点 j 属于同一社区，那么增加一条边不会对当前社区结构造成任何影响，因此应该保持当前结构不变；如果节点 i 和节点 j 不属于同一社区，增加一条边会提高社区间的耦合性，因此当前社区结构可能会发生以下 3 种情况的变化，即社区结构不变、合并为同一个社区、分裂成更小的社区。为了更准确和高效地识别社区结构的变化情况，本文引入了扩散核心节点的概念。

定义 6 扩散核心节点。在 t 时刻，利用影响力最大化的方式找到扩散核心节点集合 disperseL^t。节点 i 的邻居节点存在种子节点时，由于两者的影响力存在重叠，则需要对节点 i 的度数进行度量折扣。Chen 等[31]在选择种子节点时，探索了所选种子节点对其余节点的影响，采用节点度来估计其影响，并提出了度折扣启发法来减小这种影响。假设所有边的激活概率均为 β，节点 i 的邻居中有 s_i 个激活种子时，被激活的概率为 $1-(1-\beta)^{s_i}$。因此，在考虑节点 i 是否作为种子节点时，需要综合考虑节点 i 自身的度中心性及其邻居节点的激活情况，所以节点 i 被选为种子节点时产生的期望影响力为

$$A = [1 + (d_i - s_i\beta)](1-\beta)^{s_i} \approx 1 + [d_i - 2_i - (d_i - s_i)s_i\beta]\beta \tag{7}$$

当节点 i 的邻居节点不存在种子节点时，i 作为种子节点产生的期望影响力为

$$B = 1 + d_i\beta \tag{8}$$

设 γ 是对邻居中每个种子节点的度折扣，则 $s_i\beta\gamma = B - A$，可以得到 $\gamma = 2 + (d_i - s_i)\beta$，因此，当节点 i 有 s_i 个种子邻居时，它的度折扣值定义为

$$\mathrm{dd}_i = d_i - s_i\gamma = d_i - 2s_i - (d_i - s_i)s_i\beta \quad (9)$$

依据 dd_i 将所有节点按从大到小的顺序排序，选取前 10%的节点作为扩散核心节点。

在 t 时刻，如果不同社区的节点 i 和节点 j 之间增加了一条边 e_{ij}，当节点 i 和节点 j 都是扩散核心节点时，就将节点 i 和节点 j 所属社区的所有节点放入 unfinishL^t 集合中；当节点 i 和节点 j 中有一个节点为扩散核心节点时，就将 $N(i)$ 或 $N(j)$ 放入 unfinishL^t 集合中；当 2 个节点都不是扩散核心节点时，只需要将 $N(i)$ 和 $N(j)$ 放入 unfinishL^t 集合中。增加边的伪代码如算法 1 所示。

算法 1　增加边

输入　G_t, ch, disperseL^t, finishL^t, unfinishL^t

输出　finishL^t, unfinishL^t

1) for (u,v) range Add Es$_t$:
2)　if $u.C \neq v.C$:
3)　　if u in disperseL^t and v in disperseL^t:
4)　　将 $u.C$ 和 $v.C$ 中的所有节点从 finishL^t 移到 unfinishL^t 中
5)　　else if u in disperseL^t or v in disperseL^t:
6)　　if v in disperseL^t:
7)　　将 $N(v)$ 中的所有节点从 finishL^t 移到 unfinishL^t 中
8)　　else:
9)　　将 $N(u)$ 中的所有节点从 finishL^t 移到 unfinishL^t 中
10)　　end if
11)　　else:
12)　　将 $N(v), N(u)$ 中的所有节点从 finishL^t 移到 unfinishL^t 中
13)　　end if
14)　end if
15) end for

3.2.2　删除边策略

在 t 时刻，节点 i 和节点 j 之间删除了一条边 e_{ij}。根据删除边的两端节点在 $t-1$ 时刻是否属于同一社区，将这种情况分为 2 个子场景。当节点 i 和节点 j 属于同一个社区时，当前社区的内聚性会因为边的删除而降低，可能会导致以下几种情况：社区依旧相互紧密相连，保持不变；当前社区变稀疏，变成小社区；当前社区分解并和其他社区合并。为了准确和高效地识别当前社区结构发生的变化，引入了内聚核心节点的概念。

定义 7　内聚核心节点。利用删除最少的节点拆解 2-core 子图的方法使社区不具备连通性，所删除的节点就是对应的内聚核心节点集合 gatherL^t。生成该类型核心节点的步骤描述如下。

1) 找到社区结构的 2-core 子图，得到 2-core 内每个节点的度（不考虑到外部节点的边）。

2) 识别 2-core 中度数最大的节点 i，如果存在多个，就随机选择一个；移除节点 i，更新 2-core 子图及其现有节点的度数。

3) 如果 2-core 图为空，就停止；否则，继续执行步骤 2)。根据社交网络中数据表示的图结构是稀疏图的特点，获取内聚核心节点的实例如图 2 所示。

假设最初的节点结构如图 2(a)所示，在图 2(a)中，阴影部分是一个 2-core 子图，获取到 2-core 子图后得到图 2(b)。图 2(b)中度最大的节点是节点 3，所以将节点 3 放入 gatherL^t 中，然后将节点 3 及与其相连的边都从图结构中删除得到图 2(c)。图 2(c)中得到了 2 个 2-core 子图，并且这 2 个 2-core 子图中所有节点的度数均为 2。此时随机地在每个子图中选择一个节点加入 gatherL^t 中，假设选择的是节点 6 和节点 13，同理将选择的节点及与其相连的边删除得到图 2(e)。图 2(e)中不存在 2-core 子图，所以结束循环，最终得到的内聚核心节点集合是{3,6,13}。

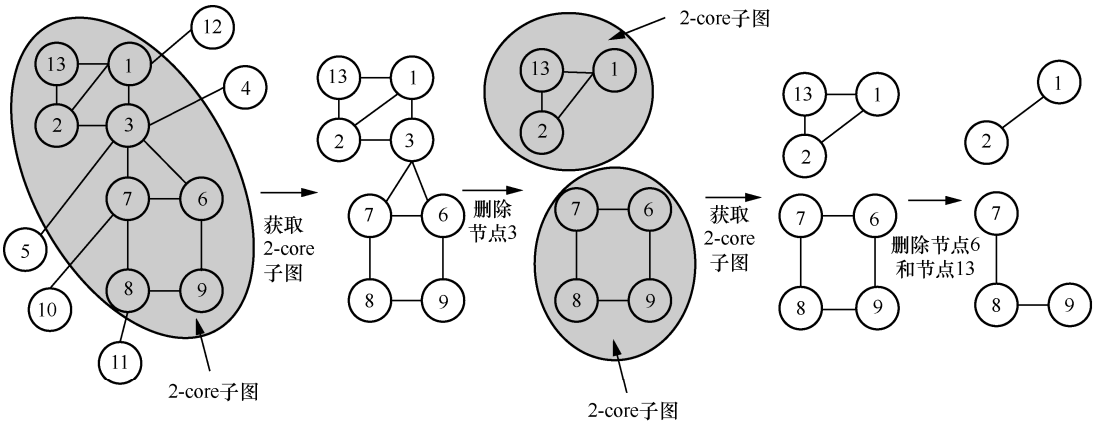
在 t 时刻，如果同一社区的节点 i 和节点 j 之间删除了一条边 e_{ij}，当节点 i 或节点 j 至少有一个节点是内聚核心节点时，将节点 i 所属社区的所有节点放入 unfinishL^t 集合中；当 2 个节点都不是内聚核心节点时，将 $N(i)$ 和 $N(j)$ 放入 unfinishL^t 集合中。当节点 i 和节点 j 属于不同社区时，2 个社区因为边的删除而提高了社区结构的耦合性，对社区结构划分是正向作用，因此保持当前社区结构不变。删除边的伪代码如算法 2 所示。

算法 2　删除边

输入　G_t, ch, gatherL^{t-1}, finishL^t, unfinishL^t

输出　finishL^t, unfinishL^t

1) for (u,v) range Del Es$_t$:
2)　if $u.C = v.C$:



(a) 最初的节点结构 (b) 获取2-core子图 (c) 删除度最大的节点及其边 (d) 重新得到2-core子图 (e) 再次删除操作

图 2 获取内聚核心节点的实例

3) 将 $N(u)$ 中的所有节点从 finishL^t 移到 unfinishL^t 中

4) if u in gatherL^{t-1} or v in gatherL^{t-1}:

5) 将 $u.C$ 中的所有节点从 finishL^t 移到 unfinishL^t 中

6) end if

7) end if

8) end for

3.2.3 增加节点策略

在 t 时刻，增加了一个新节点 i 。如果节点 i 没有其他连边，那么该节点将单独成为一个社区。当节点 i 所有的边都属于同一个社区时，将节点 i 放入所连边的社区中；否则，需要判断节点 i 是否为扩散核心节点。如果节点 i 是扩散核心节点，节点 i 会新生成一个单独社区，并且将 $N(i)$ 中所在社区的所有节点放入 unfinishL^t 中；如果节点 i 是普通节点，将 $N(i)$ 放入 unfinishL^t 集合中。添加节点的伪代码如算法 3 所示。

算法 3 添加节点

输入 G_t, ch, disperseL^t, finishL^t, unfinishL^t

输出 finishL^t, unfinishL^t;

1) for u range Add Ns$_t$:

2) if len(Neighborhood(u)) == 0:

3) 将 u 放入 unfinishL^t 中

4) else if u in disperseL^t:

5) for n in $N(u)$:

6) 将 $n.C$ 中的所有节点从 finishL^t 移到 unfinishL^t 中

7) end for

8) else:

9) 将 $N(u)$ 中的所有节点从 finishL^t 移到 unfinishL^t 中

10) end if

11) end for

3.2.4 删除节点策略

在 t 时刻，删除一个新节点 i 。如果节点 i 没有其他连边，直接删除当前节点，保持社区结构不发生变化；如果节点 i 有连边，节点 i 和其相邻社区可能会分解成更小的社区并合并到其他社区中，当前情况下，首先判断节点 i 是否为内聚核心节点。当节点 i 为内聚核心节点时，将节点 i 所属社区和 $N(i)$ 放入 unfinishL^t 集合中；当节点 i 不为内聚核心节点时，只需将 $N(i)$ 放入 unfinishL^t 集合中。删除节点的伪代码如算法 4 所示。

算法 4 删除节点

输入 G_t, ch, gatherL^{t-1}, finishL^t, unfinishL^t

输出 finishL^t, unfinishL^t

1)for u range Del Ns$_t$

2) if u in gatherL^{t-1}:

3) for n in $N(u)$:

4) 将 $n.C$ 中的所有节点从 finishL^t 移到 unfinishL^t 中

5) end for

6) else:

7) 将 $N(u)$ 中的所有节点从 finishL^t 移到 unfinishL^t 中

8) end if

9)end for

执行 4 种策略后，所有节点将被分配到 finishL^t 或 unfinishL^t 集合中。在 finishL^t 集合中，这些节点保留了历史时刻信息；在 unfinishL^t 集合中，这些节点因增量导致社区结构需要随之变动。在后续过程中，将 unfinishL^t 集合中的所有节点设为一个单独社区，得到社区中间结构。在社区中间结构上，进行以下操作。

1) 对图进行压缩，将所有在同一个社区的节点压缩成一个新节点，社区内节点之间边的权重转化为新节点的权重，社区间边的权重转化为新节点间的边权重。

2) 对每个节点 i，依次尝试把节点 i 分配到其每个邻居节点所在的社区，依据式(6)计算增量模块度的大小，并记录 $\triangle Q$ 最大的邻居节点，如果 $\triangle Q>0$，则把节点 i 分配给 $\triangle Q$ 最大的邻居节点所在的社区，否则保持不变。

3) 重复步骤 2)，直到所有节点的所属社区不再发生变化。

4) 重复步骤 1)，直到整个图的模块度不再发生变化。

本文使用了 4 种不同类型的增量更改，并相应地设计了每种类型的初始化策略，在保留部分历史信息的同时不断优化增量模块度。MCNIDCD 的伪代码描述如算法 5 所示。

算法 5　MCNIDCD

输入　$G_t, C(G_{t-1})$

输出　CS_t

1) 利用定义 5 和定义 6，获取 gatherL^{t-1} 和 disperseL^t

2) for ch range $\triangle G$

3)　$\text{finishL}^{t-1} \rightarrow \text{finishL}^t$ null $\rightarrow \text{unfinishL}^t$

4)　if ch.type equal “Remove node”:

5)　$\text{finishL}^t, \text{unfinishL}^t$ =RemoveNode(G_t, ch, gatherL^{t-1}, finishL^t, unfinishL^t)

6)　else if ch.type equal “Add node”:

7)　$\text{finishL}^t, \text{unfinishL}^t$ =AddNode(G_t, ch, disperseL^t, finishL^t, unfinishL^t)

8)　else if ch.type equal “Remove edge”: $\text{finishL}^t, \text{unfinishL}^t$ =RemoveEdge (G_t, ch, gatherL^{t-1}, finishL^t, unfinishL^t)

9)　else if ch.type equal “Add edge”: finishL^t, unfinishL^t = AddEdge (G_t, ch, disperseL^t, finishL^t, unfinishL^t)

10)　for G_t, node in finishL^t:

11)　node.C==node

12)　end for

13)　end if

14)　利用式(6)合并社区得到 CS_t

15) end for

4　实验

为了证明 MCNIDCD 算法的性能，本文在人工和真实数据集上进行了广泛的实验，将实验结果与现有的动态社区检测算法进行对比，对比算法分别是 Label-RankT[17]、DYNMOGA[32]、PDG[33]、IncNSA[34] 和 TMOGA[35]。其中，Label-RankT 以 LabelRank 为基础，是适用于大规模动态网络社区发现的在线分布式算法，具有较好的检测精度；DYNMOGA 运用了一种基于非支配排序原理的遗传优化方法，该算法旨在对函数进行高效优化，并在此过程中综合考量了电导特性、归一化的分割指标以及社区评分等多个维度的信息要素；PDG 引入了一种以个体稳定度为核心的博弈论框架，针对网络动态演化过程中的社区结构更新难题，设计并实施了诸如格局检测等的精细化优化策略；IncNSA 属于增量式检测技术，该算法关注部分节点在社区归属关系上的变更，这些节点包括新出现的节点以及上一时间快照中表现出显著活动变化的节点；TMOGA 通过从历史社区结构中提炼和保留具有稳定性的特征信息，并将这些信息集成到当前的优化进程中，从而实现对进化算法性能的有效提升与改良。本文采用模块度衡量算法的性能，以评估每个快照检测到的社区结构质量。

4.1　数据集

1) 人工动态网络

本文使用动态基准网络生成器来合成网络，每个合成网络包含了 10 个快照，每个快照有 1 000 个节点，平均度为 20，最大度为 50，混合参数 $\mu = 0.2$。合成网络演化中的 3 种演化事件类型分别为社区的生成与消亡、扩张与收缩、合并与分裂。这 3 种类型的事件介绍如下。

① 生成和消亡：在每个快照中，从现有的社区中移除节点来创建 5 个社区，同时随机移除其他

5 个现有社区。

② 扩张和收缩：在每个快照中，随机选择 5 个社区，将其规模扩大或缩小 25%。

③ 合并和分裂：在每个快照中，选择 5 个社区进行分裂，并与另外 5 个社区进行合并。

此外，本文为每个类型事件生成 10 个具有相同参数设置的网络，并将测量指标的平均值作为事件的最终结果，以避免单次结果的偶然性。

2) 真实网络数据集

本文使用 2 个真实世界的网络 PhoneCall 和 AS-Oregon 来评估 MCNIDCD 的性能，数据集简介如表 2 所示。

表 2 数据集简介

网络	快照	$\|V\|_{\max}$	$\|V\|_{\min}$	$\|E\|_{\max}$	$\|E\|_{\min}$
PhoneCall	10	384	365	530	498
AS-Oregon	9	11 174	10 670	23 409	21 999

PhoneCall 手机通话记录涵盖了 2006 年 6 月的十天时间，并以天为单位划分出 10 张快照，得到 10 张信息通话图; AS-Oregon 是从 2001 年 3 月 31 日至 2001 年 5 月 26 日的俄勒冈州路由视图推断出的自治系统的 9 张对等信息图。

4.2 有效性分析

模块度用来评估社区划分的好坏程度。模块度越高，表示社区内部连接越紧密，社区之间连接越稀疏，社区划分的准确性越高。在一个网络中，模块度可以表示为实际边数与期望边数的差值，期望边数指的是在随机网络中同样的节点度数分布下，2 个节点之间连边的概率。设 $\boldsymbol{A}$ 表示网络的邻接矩阵，e_{ij} 表示节点 i 和节点 j 之间的边，k_i 表示节点 i 的度数，m 表示网络中边的数目，则模块度为

$$Q=\frac{1}{2m}\left(A_{ij}-\frac{k_ik_j}{2m}\right)\delta(c_i,c_j) \tag{10}$$

其中，c_i 表示节点 i 所属的社区；$\delta(c_i,c_j)$ 是一个指示函数，当节点 i 和节点 j 属于同一个社区时为 1，否则为 0。模块度的取值范围为[−1,1]。

本文考虑将 3 种不同类型的事件嵌入合成网络中，并在这些网络中评估不同社区检测算法的性能，如图 3～图 5 所示。通过比较各种算法在模块度指标上的表现可以看出，本文提出的 MCNIDCD 在合成数据集上表现良好，并在 3 种不同类型的事件中模块度值都具有优势。随着时间的变化，MCNIDCD 模块度趋于平稳，性能并不会下降，这表明 MCNIDCD 在不同时间快照中产生的社区结构是比较准确并且稳定的。

根据图 3～图 5 的实验结果可以观察到，MCNIDCD 在动态社区检测任务中表现明显优于 PDG 和 TMOGA。PDG 基于个体稳定度博弈来实现动态社区检测，在不同时间快照上模块度值的变化较大，其最大值和最小值之间存在较大差距，这表明算法的稳定性较差。值得注意的是，PDG 具有较低的时间复杂度，这是其优点之一。另一方面，TMOGA 在时间快照的初始阶段的模块度较低，但整体上呈现上升趋势。

相比之下，MCNIDCD 相对于 IncNSA、DYNMOGA 和 LabelRankT 仅有轻微的改进。IncNSA 和 DYNMOGA 是基于增量式动态社区的检测算法，在分析动态社区节点变化的时候只考虑了增量节点，时间累积可能会导致误差随着时间增长而增大，因此在整个实验过程中模块度呈现下降的趋势。而 MCNIDCD 的本质也是一种增量式算法，但相较于其他增量算法，MCNIDCD 在增量过程中通过 4 种类型策略加入了扩散和内聚 2 种核心节点来确定相关的影响节点集，并引入中间状态进行错误积累的修正，实验结果表明，MCNIDCD 在社区划分质量和划分结果稳定性方面具有良好的优势。

在扩张和收缩类型的人工网络中，MCNIDCD 相对于其他算法在性能提升上更显著。因此，实验结果表明，在面对演化变化的情况下，MCNIDCD 不会导致社区检测结果变差，并且在每个时间快照中都能稳定地获得较好的结果。

在 MCNIDCD 社区划分结果的基础上，利用式(11)表示社区之间的相似性。

$$\text{Jaccard}(A,B)=\frac{|A\cap B|}{|A\cup B|} \tag{11}$$

图 6～图 8 给出了相邻快照之间的演化关系。每个子图中，纵坐标代表当前网络快照的社区标签 c_l，横坐标代表下一个快照的社区标签 c_k，颜色深浅则表示当前快照中的社区向下一个快照的社区的转移概率大小。

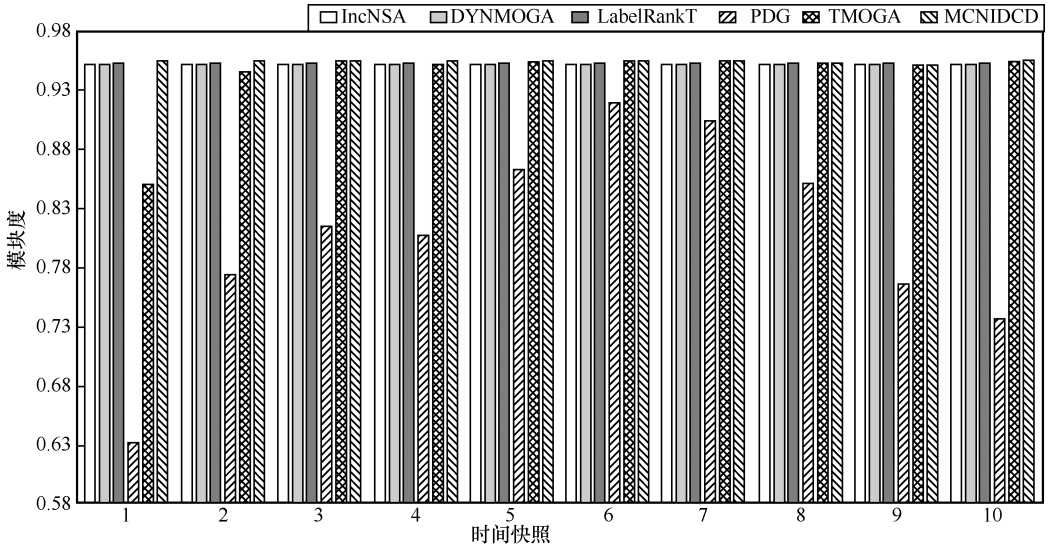


图 3　各算法在生成和消亡人工网络数据集上的模块度

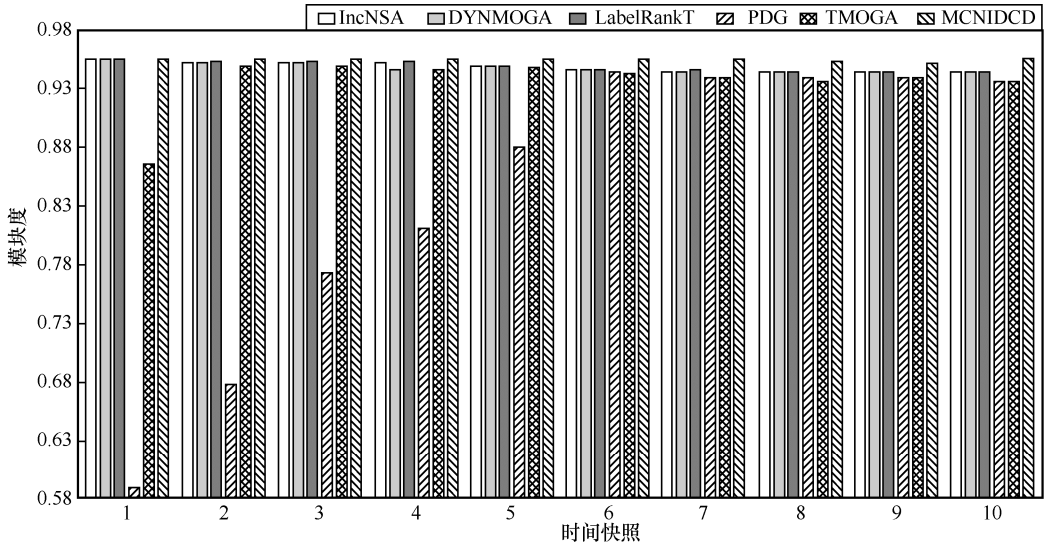


图 4　各算法在扩张和收缩人工网络数据集上的模块度

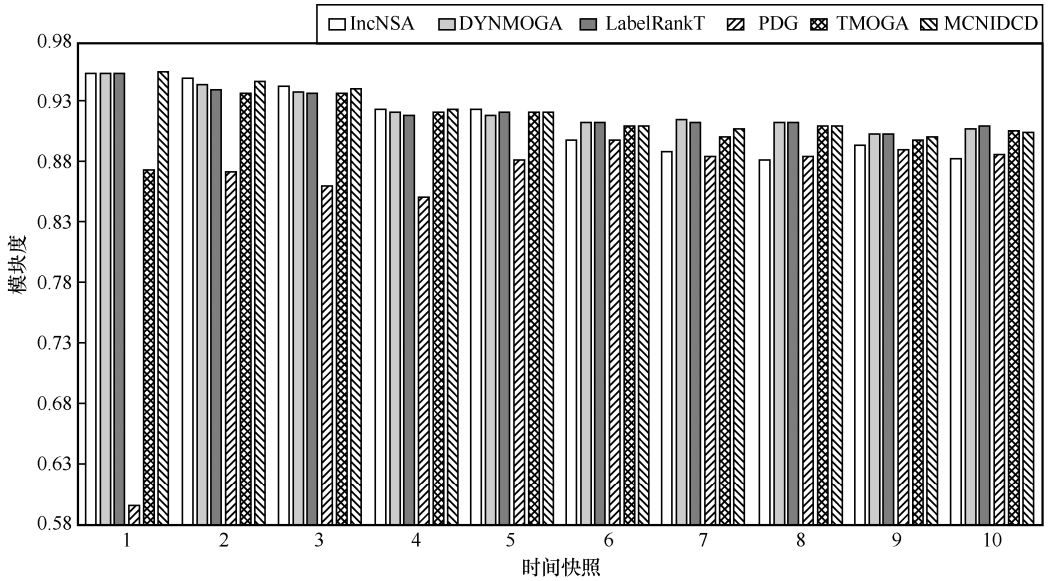
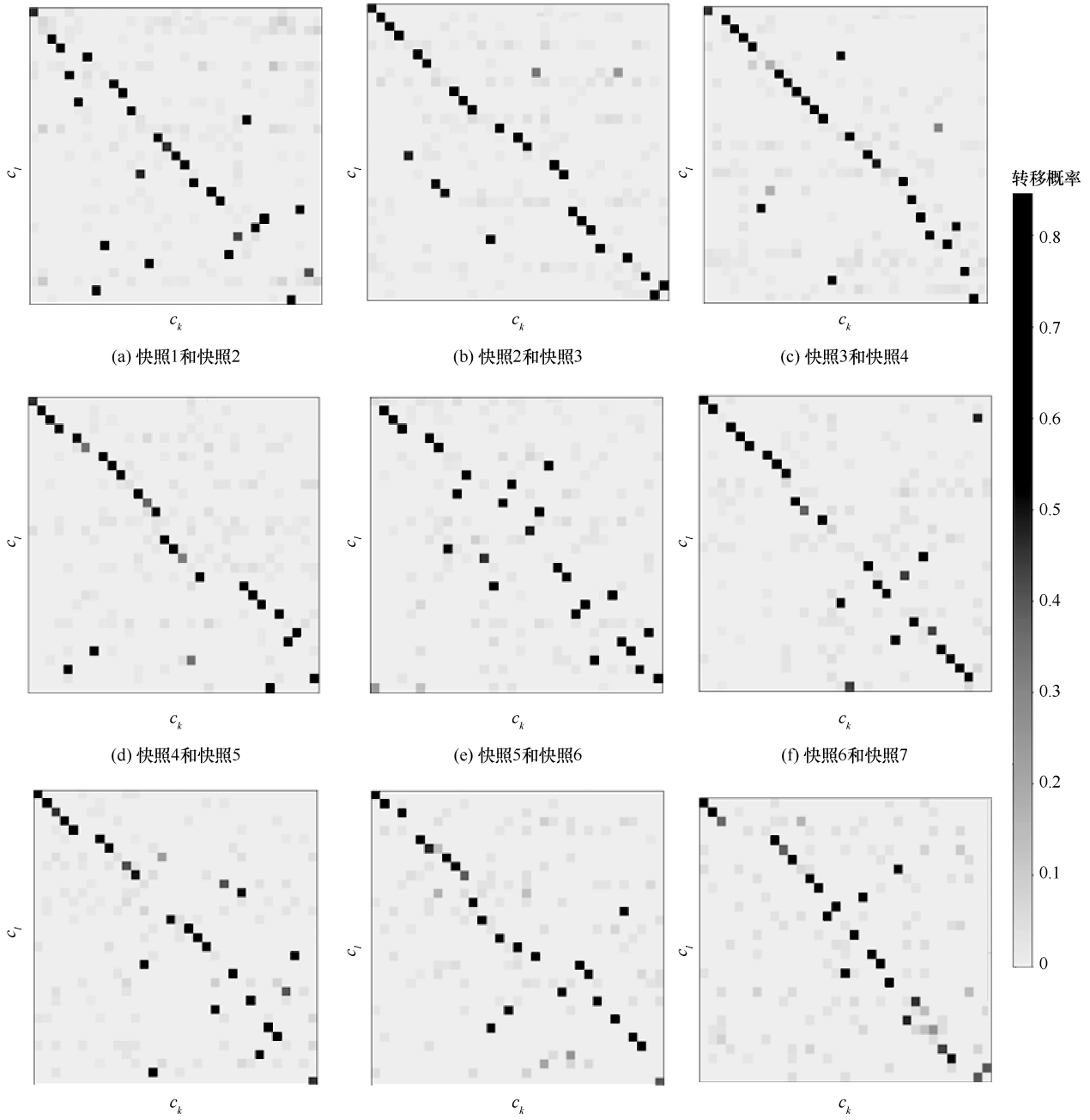


图 5　各算法在合并和分裂人工网络数据集上的模块度



(a) 快照1和快照2　(b) 快照2和快照3　(c) 快照3和快照4

(d) 快照4和快照5　(e) 快照5和快照6　(f) 快照6和快照7

(g) 快照7和快照8　(h) 快照8和快照9　(i) 快照9和快照10

图 6　MCNIDCD 在生成和消亡人工网络数据集上的相关性

从图 6～图 8 可观察到，大多数子图的对角线呈现深色，表明大多数社区内部的节点有较大的概率保留在当前社区中。这反映了动态社区演化的缓慢性，展示了 MCNIDCD 的平滑特性。在图 6 中，大部分事件是生成和消亡事件，导致 t 时刻社区和 $t+1$ 时刻社区相似度较低，因此对角线部分区域白色较多。在图 7 中，主要事件是扩张和收缩，$t+1$ 时刻社区保留了 t 时刻社区中大多数节点，因此对角线区域呈现黑色。在图 8 中，主要事件是合并和分裂，t 时刻的某个社区与 $t+1$ 时刻的几个社区有一定的相似度，因此对角线区域集中在灰色。图 6～图 8 所示的演化情况证明了 MCNIDCD 社区划分结果的合理性。

为了说明每种算法的总体性能，表 3 呈现了 2 个真实网络的动态社区发现算法实验结果所有快照的平均模块度。由表 3 可以看出，MCNIDCD 表现最佳。由于 AS-Oregon 数据集相对比较大，而

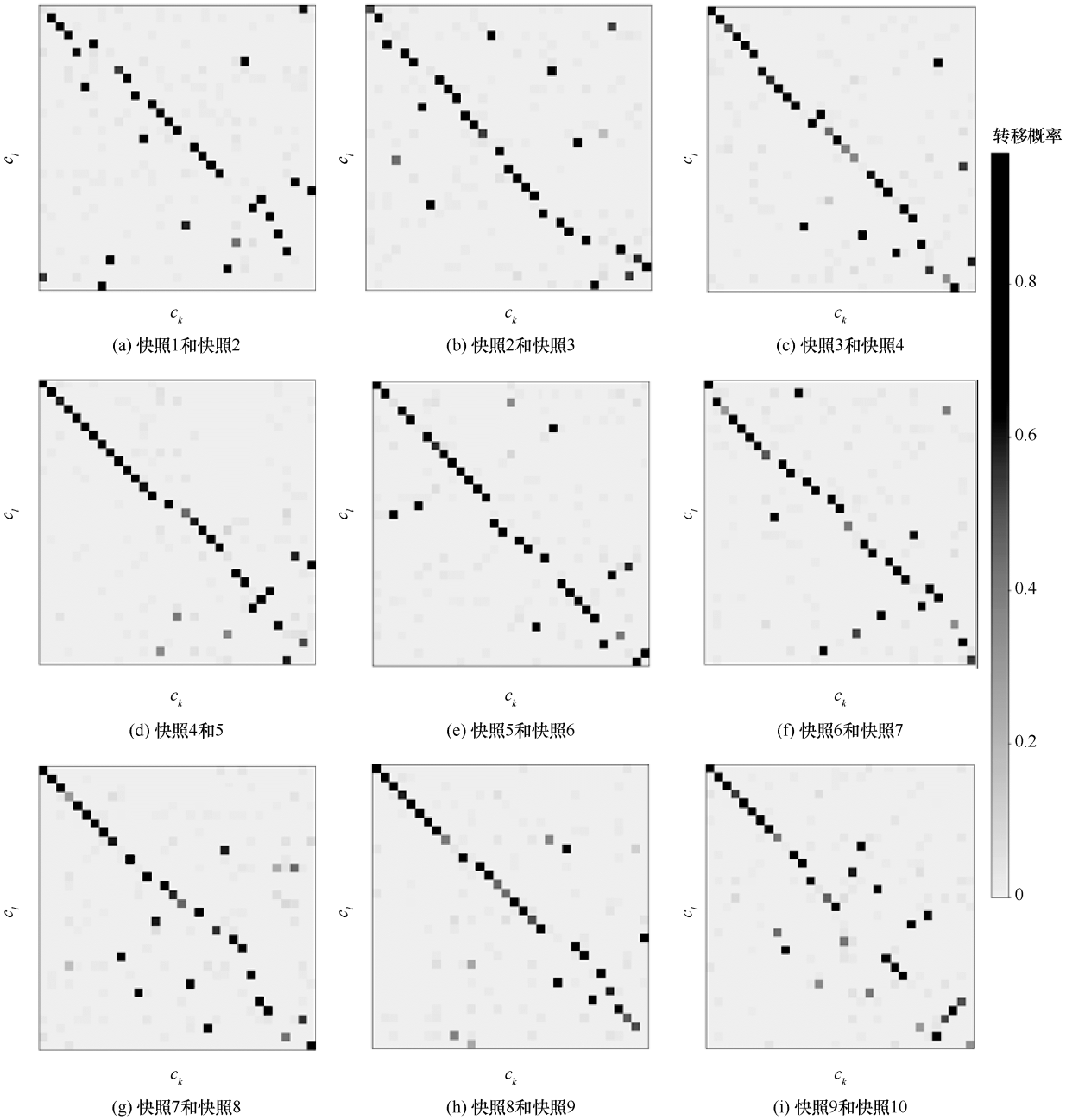
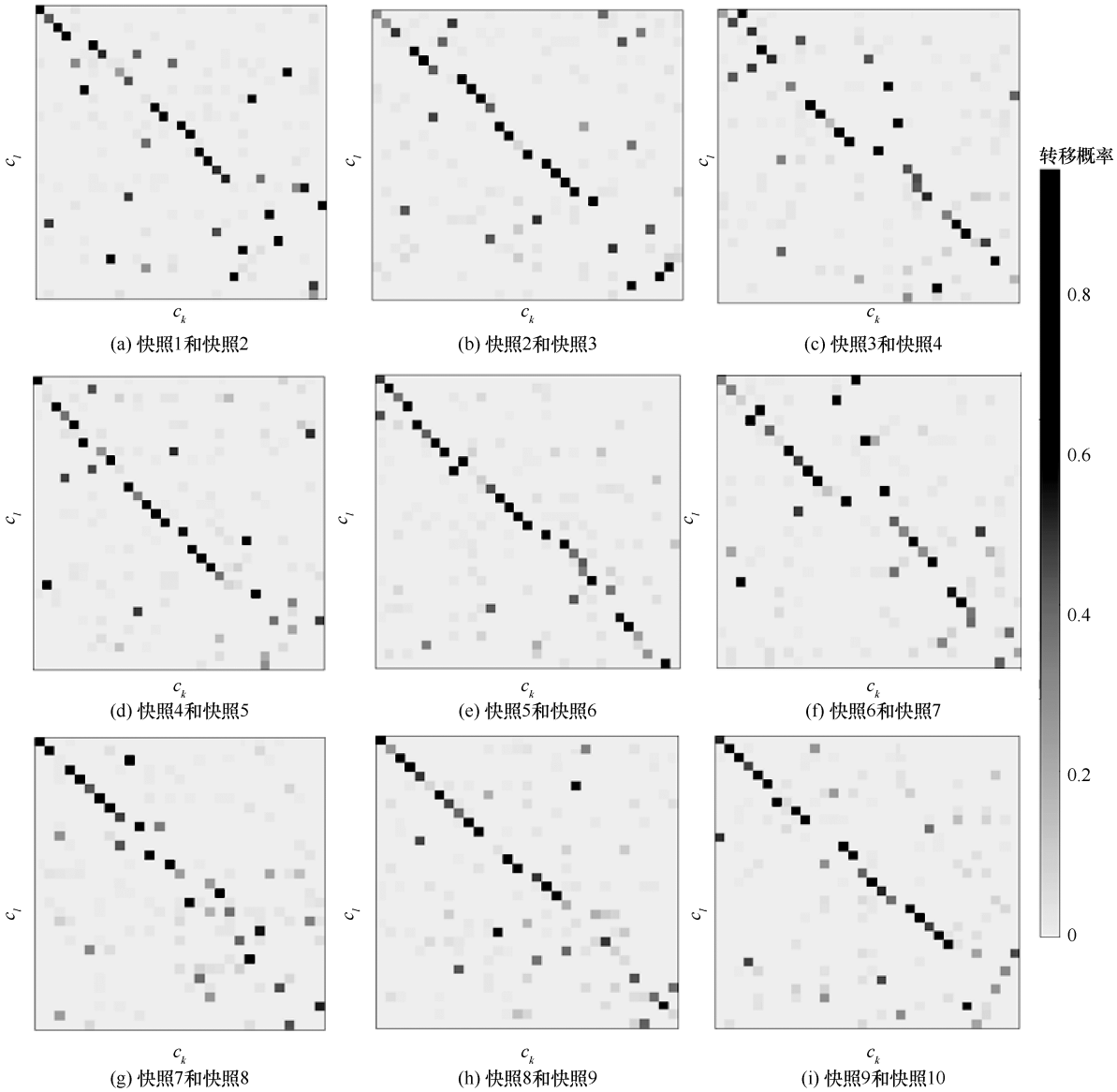


图 7　MCNIDCD 在扩张和收缩人工网络数据集上的相关性

TMOGA 时间复杂度非常高，TMOGA 运行时长是其他算法的几十倍，因此在 AS-Oregon 数据集中不和 TMOGA 进行对比。

在真实数据集中，MCNIDCD 在 2 个数据集中都展现出了较好的模块度结果，如图 9 和图 10 所示。相较于人工网络，真实网络中的社区演化类型更加复杂，使 MCNIDCD 相对于其他算法在性能提升方面相比人工网络数据集更显著。MCNIDCD 在不同时间快照下的模块度值波动较小，而其他算法的模块度值波动较大，这一现象在图 9 中特别明显。在图 9 中，MCNIDCD 在每个时间快照中都获得了最大的模块度值。尽管在图 10 中，MCNIDCD 在某些时间快照中的模块度相对于 IncNSA 较低，但总体平均值更高，并且 MCNIDCD 呈现出更好的稳定性。综上所述，MCNIDCD 相对于其他增量算法来说更稳定，能够得到更合理的社区划分结果，并且其模块度随着时间推移波动平稳。



(a) 快照1和快照2　(b) 快照2和快照3　(c) 快照3和快照4

(d) 快照4和快照5　(e) 快照5和快照6　(f) 快照6和快照7

(g) 快照7和快照8　(h) 快照8和快照9　(i) 快照9和快照10

图 8　MCNIDCD 在合并和分裂人工网络数据集上的相关性

表 3　　2 个真实网络的动态社区发现算法实验结果所有快照的平均模块度

网络	IncNSA	DYNMOGA	LabelRankT	PDG	TMOGA	MCNIDCD
PhoneCall	0.670 8	0.652 3	0.536 1	0.549 0	0.581 4	**0.683 3**
AS-Oregon	0.617 1	0.528 1	0.508 7	0.354 5	—	**0.620 4**

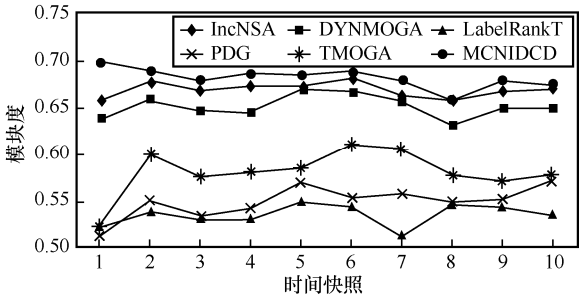


图 9　各算法在 PhoneCall 数据集上的模块度

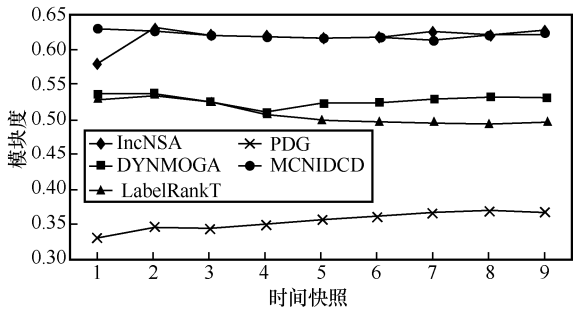


图 10　各算法在 AS-Oregon 数据集上的模块度

根据人工和真实网络数据集的实验结果可知，MCNIDCD 通过引入内聚和扩散两类核心节点，针对 4 种增量类型设计了多核心节点局部更新策略。将节点划分为核心节点和其他节点的目的是准确地识别在增量过程中受到影响的节点。同时，将核心节点进一步划分为 2 种类型，可以控制社区演化中的扩散和内聚 2 种方向。这样的设计使 MCNIDCD 能够在保留历史信息的同时，准确且稳定地获取社区结构。这种稳定性的特征在表 3 中得到了体现。

5　结束语

本文提出了一种新的动态社区检测算法 MCNIDCD，并使用增量式方法来检测社区随时间的变化趋势。该算法克服了传统算法中忽略历史信息的缺陷，并通过解决错误积累引起的时间漂移误差来提高准确性。首先，初始状态利用静态社区检测算法；其次，通过使用两类核心节点控制内聚和扩散 2 个方向的演化过程，该算法制定不同的更新策略以修改社区结果，从而形成中间社区；最后，基于定义的增量模块度最大化方法将小社区合并，并对合并后的社区划分结果进行性能评价。通过在 2 个真实网络和 3 个人工网络上进行实验可知，MCNIDCD 能够从各种时间演化的网络中提取出有效的社区结构，且提取出的社区质量具有较好的优势，总体上优于其他算法。

本文通过将核心节点划分为扩散和内聚 2 种类型，使 MCNIDCD 具有以下优点：首先，它可以在不丢失历史信息的情况下对社区进行动态检测；其次，它通过使用两类核心节点控制内聚和扩散 2 个方向的演化过程，能够生成许多小社区，从而提高了社区检测的准确性；最后，它使用增量模块度最大化的方法将小社区合并，从而提高了社区合并的质量。未来可以探索如何将该算法应用于更大规模的网络中，并进一步优化算法的性能。

参考文献：

[1] SU X, XUE S, LIU F Z, et al. A comprehensive survey on community detection with deep learning[J]. IEEE Transactions on Neural Networks and Learning Systems, 2022, PP(99): 1-21.

[2] DHOUIOUI Z, AKAICHI J. Tracking dynamic community evolution in social networks[C]//Proceedings of the 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. New York: ACM Press, 2014: 764-770.

[3] İLHAN N, ÖĞÜDÜCÜ Ş G. Predicting community evolution based on time series modeling[C]//Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. New York: ACM Press, 2015: 1509-1516.

[4] YIN Y, ZHAO Y H, LI H, et al. Multi-objective evolutionary clustering for large-scale dynamic community detection[J]. Information Sciences, 2021, 549: 269-287.

[5] ZENG X X, WANG W, CHEN C, et al. A consensus community-based particle swarm optimization for dynamic community detection[J]. IEEE Transactions on Cybernetics, 2020, 50(6): 2502-2513.

[6] LI W M, ZHOU X K, YANG C, et al. Multi-objective optimization algorithm based on characteristics fusion of dynamic social networks for community discovery[J]. Information Fusion, 2022, 79: 110-123.

[7] ZHAO Z Y, LI C, ZHANG X J, et al. An incremental method to detect communities in dynamic evolving social networks[J]. Knowledge-Based Systems, 2019, 163: 404-415.

[8] SEIFIKAR M, FARZI S, BARATI M. C-blondel: an efficient louvain-based dynamic community detection algorithm[J]. IEEE Transactions on Computational Social Systems, 2020, 7(2): 308-318.

[9] ZARAYENEH N, KALYANARAMAN A. Delta-screening: a fast and efficient technique to update communities in dynamic graphs[J]. IEEE Transactions on Network Science and Engineering, 2021, 8(2): 1614-1629.

[10] LI D Y, ZHONG X X, DOU Z F, et al. Detecting dynamic community by fusing network embedding and nonnegative matrix factorization[J]. Knowledge-Based Systems, 2021, 221: 106961.

[11] REN S X, ZHANG S B, WU T. An improved spectral clustering community detection algorithm based on probability matrix[J]. Discrete Dynamics in Nature and Society, 2020, 2020: 1-6.

[12] AL-SHAROA E, AL-KHASSAWENEH M, AVIYENTE S. Low-rank estimation based evolutionary clustering for community detection in temporal networks[C]//Proceedings of the 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Piscataway: IEEE Press, 2019: 5381-5385.

[13] JIA J J, LI L F. Dynamic community detection based on similarity of social network nodes[C]//Proceedings of the 2022 4th International Academic Exchange Conference on Science and Technology Innovation (IAECST). Piscataway: IEEE Press, 2022: 1147-1151.

[14] BLONDEL V D, GUILLAUME J L, LAMBIOTTE R, et al. Fast unfolding of communities in large networks[J]. Journal of Statistical Mechanics: Theory and Experiment, 2008, 2008(10): 10008.

[15] GAO F, YUAN L, WANG W J, et al. Dynamic community detection using nonnegative matrix factorization[C]//Proceedings of the 2017 International Conference on Computing Intelligence and Information System (CIIS). Piscataway: IEEE Press, 2017: 39-45.

[16] WANG C Y, DENG Y, LI X H, et al. Dynamic community detection based on a label-based swarm intelligence[J]. IEEE Access, 2019, 7: 161641-161653.

[17] XIE J R, CHEN M M, SZYMANSKI B K. LabelRankT: incremental community detection in dynamic networks via label propagation[C]//Proceedings of the Workshop on Dynamic Networks Management and Mining. New York: ACM Press, 2013: 25-32.

[18] XIE J R, SZYMANSKI B K. LabelRank: a stabilized label propagation algorithm for community detection in networks[C]//Proceedings of the 2013 IEEE 2nd Network Science Workshop (NSW). Piscataway: IEEE

Press, 2013: 138-143.
[19] SATTARI M, ZAMANIFAR K. A cascade information diffusion based label propagation algorithm for community detection in dynamic social networks[J]. Journal of Computational Science, 2018, 25: 122-133.
[20] BERAHMAND K, HAGHANI S, ROSTAMI M, et al. A new attributed graph clustering by using label propagation in complex networks[J]. Journal of King Saud University-Computer and Information Sciences, 2022, 34(5): 1869-1883.
[21] BESHARATNIA F, TALEBPOUR A, ALIAKBARY S. An improved grey wolves optimization algorithm for dynamic community detection and data clustering[J]. Applied Artificial Intelligence, 2021, 36(4): 1-22.
[22] ABBOOD A D, ATTEA B A, HASAN A A, et al. Community detection model for dynamic networks based on hidden Markov model and evolutionary algorithm[J]. Artificial Intelligence Review, 2023, 56(9): 9665-9697.
[23] JIANG W C, PAN S C, LU C H, et al. Label entropy‐based cooperative particle swarm optimization algorithm for dynamic overlapping community detection in complex networks[J]. International Journal of Intelligent Systems, 2022, 37(2): 1371-1407.
[24] GAO C, CHEN Z P, LI X H, et al. Multiobjective discrete particle swarm optimization for community detection in dynamic networks[J]. Europhysics Letters, 2018, 122(2): 28001.
[25] CHEN Z, SUN A X, XIAO X K. Incremental community detection on large complex attributed network[J]. ACM Transactions on Knowledge Discovery from Data, 2021, 15(6): 1-20.
[26] ZARDI H, ALHARBI B, KARAMTI W, et al. Detection of community structures in dynamic social networks based on message distribution and structural/attribute similarities[J]. IEEE Access, 2021, 9: 67028-67041.
[27] ROSSETTI G, PAPPALARDO L, PEDRESCHI D, et al. Tiles: an online algorithm for community discovery in dynamic social networks[J]. Machine Learning, 2017, 106(8): 1213-1241.
[28] WANG Z X, LI Z C, YUAN G, et al. Tracking the evolution of overlapping communities in dynamic social networks[J]. Knowledge-Based Systems, 2018, 157: 81-97.
[29] 郭昆, 彭胜波, 陈羽中, 等. 基于密度聚类的增量动态社区发现算法[J]. 模式识别与人工智能, 2018, 31(11): 965-978.
GUO K, PENG S B, CHEN Y Z, et al. Incremental dynamic community detection algorithm based on density clustering[J]. Pattern Recognition and Artificial Intelligence, 2018, 31(11): 965-978.
[30] HE J L, CHEN D B. A fast algorithm for community detection in temporal network[J]. Physica A: Statistical Mechanics and Its Applications, 2015, 429: 87-94.
[31] CHEN W, WANG Y J, YANG S Y. Efficient influence maximization in social networks[C]//Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM Press, 2009: 199-208.
[32] FOLINO F, PIZZUTI C. An evolutionary multiobjective approach for community discovery in dynamic networks[J]. IEEE Transactions on Knowledge and Data Engineering, 2013, 26(8): 1838-1852.
[33] 许宇光, 蒋飞, 朱恩强, 等. 基于个体稳定度博弈的动态社区发现算法研究[J]. 电子与信息学报, 2017, 39(4): 763-769.
XU Y G, JIANG F, ZHU E Q, et al. Research on dynamic community discovery algorithm based on individual stability game[J]. Journal of Electronics & Information Technology, 2017, 39(4): 763-769.
[34] SU X, CHENG J J, YANG H J, et al. IncNSA: detecting communities incrementally from time-evolving networks based on node similarity[J]. International Journal of Modern Physics C, 2020, 31(7): 2050094-921.
[35] ZOU J G, LIN F, GAO S Y, et al. Transfer learning based multi-objective genetic algorithm for dynamic community detection[J]. arXiv Preprint, arXiv: 2109.15136, 2021.

[作者简介]

陈晶（1976− ），女，黑龙江哈尔滨人，博士，广东海洋大学教授、硕士生导师，主要研究方向为 Web 服务、社交网络分析、数据挖掘。

刘志君（1998− ），男，江西吉安人，燕山大学硕士生，主要研究方向为动态社区发现与社区演化分析。

杨新宇（1998− ），男，山东东营人，燕山大学硕士生，主要研究方向为社交网络分析与链接预测。

刘洺辛（1976− ），男，黑龙江绥化人，博士，广东海洋大学教授、博士生导师，主要研究方向为物联网、无线传感器网络。

刘苗苗（1982− ），女，河南洛阳人，博士，东北石油大学教授、硕士生导师，主要研究方向为社会网络分析及预测、基于深度学习的大数据分类和数值模拟分析。