EasyChair Preprint

№ 1980

Comparison and Analysis of Several Synchronization Clustering Models

Xinquan Chen

November 18, 2019

几种同步聚类模型的比较与分析

陈新泉 [1, 2]
1. 安徽工程大学 计算机与信息学院，芜湖 241000；
2. 重庆三峡学院 智能信息与控制重点实验室，重庆 404100
chenxqscut@126.com

摘要：聚类是一种重要的数据分析及数据预处理技术。与传统的静态聚类分析方法相比，基于同步模型的聚类算法属于一种动态演化的聚类分析技术。本文先是描述并比较了几种同步聚类模型的复杂度与聚类效果，然后分析了几种同步聚类模型的性质与特点。在人工数据集和八个 UCI 数据集的仿真实验中，将几种同步聚类模型在聚类精度、聚类速度等方面进行了适当的比较。最后对基于同步模型的聚类算法的发展进行了总结及展望。
关键词：聚类；同步模型；近邻；指数衰减

中图分类号：**TN181**　　文献标识：**A**

Comparison and Analysis of Several synchronization Clustering Models

Chen Xin-quan [1, 2]
[1]School of Computer & Information, Anhui Polytechnic University, Wuhu, 241000, China
[2]Key Laboratory of Intelligent Information Processing and Control, Chongqing Three Gorges University, Chongqing, 404100, China

Abstract：Clustering is an important data analysis and data preprocessing technology. Compared with the traditional static clustering analysis methods, the clustering algorithms based on synchronization models are a kind of dynamic evolutionary clustering analysis technique. This paper describes and compares the complexity and clustering effect of several synchronization clustering models, then analyzes their properties and characteristics. In the simulation experiments of some artificial data set and eight UCI data sets, these several synchronization clustering models were compared in clustering accuracy and clustering speed. Finally, it summarizes the development of some synchronization clustering algorithms and presents the next work.
Key words：clustering; synchronization model; near neighbour; exponential decaying

收稿日期：2019-3-28　　修订日期：　通讯作者：陈新泉 chenxqscut@126.com
基金项目：重庆市前沿与应用基础研究项目资助（cstc2016jcyjA0521），重庆三峡学院重大培育项目(grant number: 16PY08)，安徽工程大学科研启动基金计划项目（2018YQQ031）。
Foundation Item: Chongqing Cutting-edge and Applied Foundation Research Program (cstc2016jcyjA0521), Chongqing Three Gorges University of China (grant number: 16PY08), Anhui Polytechnic University (grant number: 2018YQQ031).

1 前 言

在统计学中，作为多元统计分析方法的一种，聚类试图发现数据中的结构或分布规律。在机器学习中，聚类被称为无监督学习，指的是试图发现无标记数据中的内在分布结构，让划分后的簇表现出数据的自然结构特点。聚类可以让缺乏先验信息的数据集获得其内在的分布结构（聚类的数目、大小、分布位置及空间结构等信息），进而为半监督学习、数据压缩等一些后续操作提供一定的辅助信息。通常，同一聚类中的数据应该比不同聚类的数据更为相似，不同聚类间的数据应该具有较大的差异。聚类在许多领域都有应用，如：心理学和其它社会科学、生物学、统计学、模式识别、信息检索、机器学习和数据挖掘[1]。

聚类分析属于一个交叉研究领域，它融合了多个学科的方法和技术。Qlan Wei-ning 等[2]从多个角度分析了现有的许多聚类算法。Johannes Grabmeier 等[3]从数据挖掘的角度（如相似度的定义、相关的优化标准等）分析了许多聚类算法。Arabie 和 Hubert 的论文[4]是一个关于聚类方面的很好的参考文献。Jain 等[5]对聚类分析领域作了一个较好的综述。最近，Rui Xu 等[6]和 Anil K. Jain[7]对聚类算法分别作了个较为全新的综述分析。

传统的聚类算法大致可以分为划分聚类方法，层次聚类方法，密度聚类方法，网格聚类方法，模型聚类方法，图聚类方法等。近年来，量子聚类方法，谱聚类方法，粒度聚类方法，概率图聚类方法，同步聚类方法等也流行起来。

据我们所知，Böhm 等[8]于 2010 年第一次将自然界中普通存在的同步现象引入聚类领域，在 KDD2010 上发表了第一篇同步聚类算法论文。这篇开拓性论文首先将 Kuramoto 模型进行了适当地推广，得到可应用于聚类算法中的扩展 Kuramoto 模型，提出了 SynC 聚类算法。同时该文还将最小描述长度（MDL）原理[9]应用于 SynC 算法中，提出了一种自动优化参数的方法。基于同步模型的聚类算法可以缓解聚类分析和噪声检测在传统数据上的某些难题，它具有动态性、局部性及多尺度分析等特性，可以在一定程度上解决大规模数据的聚类分析所面临的困难。自此之后，新型的基于同步模型的聚类算法形成一个研究热潮。

自 SynC 算法[8]发表后，吸引了一些研究人员的注意。近几年，邵俊明等从多个角度研究基于同步模型的数据挖掘方法，发表了多篇高水平论文[10 - 14]。例如，文献[10]提出了一种新颖的孤立点检测算法，文献[11]提出了一种新颖的有效的高效子空间聚类算法 ORSC，文献[12]提出了一种基于同步模型和最小描述长度原理的新颖的动态层次聚类算法 hSync，文献[13]提出了一种可以发现复杂图的内在模式的新颖的图聚类算法 RSGC，文献[14]提出了一种用于检测概念漂移的基于原型学习的数据流挖掘算法。2013 年，黄健斌等[15]在文献[8]的基础上，提出了一种层次型的同步聚类算法。2017 年，陈新泉找到了另外一种更为有效的同步聚类模型—Vicsek 模型的线性版本。在将这种 Vicsek 模型的线性版本应用到聚类中后，发表了基于该模型的 ESynC 算法[16]。2017 年，Hang 等[17]基于重力学中的中心力优化方法，提出了一种局部同步聚类算法(G-Sync 算法)。2018 年，陈新泉[18]在文献[8]的基础上，提出了基于三种空间索引结构的快速同步聚类算法(FSynC 算法)。FSynC 算法中的两种参数型方法在有些情况下可得到 $O(dn\log n)$的时间代价。这几篇较新的同步聚类论文[16 - 18]分别从不同的角度拓展了同步聚类算法的研究。

本文第 2 节给出了几个较为重要的概念，第 3 节给出了几种基于同步模型的聚类算法描述。第 4 节通过一个人工数据集演示了几种同步模型的聚类过程示例与相关算法比较。第 5 节通过仿真实验比较了几种同步模型的性能及聚类效果，第 6 节对本文作简要的总结并指出进一步的研究方向。

2 基本概念及性质

设数据集 $\boldsymbol{S} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)},\ldots, \boldsymbol{x}^{(n)}\}$分布在 d 维有序属性空间$(A_1 \times A_2 \times \ldots \times A_d)$的某个有限区域内。为更好地描述本文算法，先给出几个基本定义。

定义 1. 数据集 $\boldsymbol{S}$ 中的点 $\boldsymbol{x}^{(i)}$ ($i = 1, \ldots, n$)的 δ 近邻点集 $\delta(\boldsymbol{x}^{(i)})$定义为：

$$\delta(\boldsymbol{x}^{(i)}) = \{\boldsymbol{x} \mid 0 < dis(\boldsymbol{x}, \boldsymbol{x}^{(i)}) \le \delta, \boldsymbol{x} \in S, \boldsymbol{x} \ne \boldsymbol{x}^{(i)}\}, \tag{1}$$

公式(1)中，$dis(\boldsymbol{x}, \boldsymbol{x}^{(i)})$为 S 中的两个点 $\boldsymbol{x}$ 和 $\boldsymbol{x}^{(i)}$的相异性度量，δ 是一个预先设定的阈值参数。

定义 1 的实质意义是：$\delta(\boldsymbol{x}^{(i)})$是从中选取与点 $\boldsymbol{x}^{(i)}$相距不超过 δ“距离”的其它点所构成的集合。

定义 2[8]**.** 应用到聚类中的扩展的 Kuramoto 模型定义为：

数据集 $\boldsymbol{S}$ 中的点 $\boldsymbol{x}^{(i)}$ ($i = 1, \ldots, n$) $= (x_1^{(i)}, \ldots, x_d^{(i)})$可看作是 d 维欧氏空间中的一个向量。根据文献[8]中的公式(1)，如果将点 $\boldsymbol{x}^{(i)}$看作是在它的 δ 近邻点集 $\delta(\boldsymbol{x}^{(i)})$里的一个相振荡器，那么点 $\boldsymbol{x}^{(i)}$在第 k 维上的分量 $x_k^{(i)}$ ($k = 1, \ldots, d$)的动力学特性可描述为：

$$x_k^{(i)}(t+1) = x_k^{(i)}(t) + \frac{1}{|\delta(x^{(i)}(t))|} \sum_{y \in \delta(x^{(i)}(t))} \sin(y_k(t) - x_k^{(i)}(t)), \tag{2}$$

公式(2)中，$x_k^{(i)}(t+1)$表示点 $\boldsymbol{x}^{(i)}$在第 t 个同步后第 k 维上的分量 $x_k^{(i)}$的更新相位值，$\boldsymbol{y} = (y_1, \ldots, y_d)$是点 $x_k^{(i)}(t)$的任一 δ 近邻点。

定义 3[16]**.** 应用到聚类中的 Vicsek 简化模型定义为：

如果将点 $\boldsymbol{x}^{(i)}$ ($i = 1, \ldots, n$) $= (x_1^{(i)}, \ldots, x_d^{(i)})$看作是在它的 δ 近邻点集 $\delta(\boldsymbol{x}^{(i)})$里的一个 agent，那么点 $\boldsymbol{x}^{(i)}$的基于 Vicsek 简化模型的动力学特性可描述为：

$$\boldsymbol{x}^{(i)}(t+1) = \boldsymbol{x}^{(i)}(t) + \frac{x^{(i)}(t) + \sum_{y \in \delta(x^{(i)}(t))} y}{\left\| x^{(i)}(t) + \sum_{y \in \delta(x^{(i)}(t))} y \right\|} \cdot v(t) \cdot \Delta t, \tag{3}$$

公式(3)中，$\boldsymbol{x}^{(i)}(t+1)$表示点 $\boldsymbol{x}^{(i)}$在第 t 个同步后的更新位置，$v(t)$表示第 t 个同步时的移动速度，$v(t) \cdot \Delta t$ 表示第 t 个同步的移动路径长度。

在一些基于 Vicsek 模型的多 agent 系统中，$v(t)$是个常量。在一些仿真实验中，当 $v(t)$是常量时，基于公式(3)的 Vicsek 模型的原始版本不能很好地应用到聚类中。所以这里我们提出 Vicsek 模型的另外一种有效版本来应用到聚类中。

定义 4[16]**.** 应用到聚类中的 Vicsek 模型的一个线性版本定义为：

如果将点 $\boldsymbol{x}^{(i)}$ ($i = 1, \ldots, n$) $= (x_1^{(i)}, \ldots, x_d^{(i)})$看作是在它的 δ 近邻点集 $\delta(\boldsymbol{x}^{(i)})$里的一个相振荡器，点 $\delta(\boldsymbol{x}^{(i)})$的基于 Vicsek 模型的一个线性版本的动力学特性可描述为：

$$\boldsymbol{x}^{(i)}(t+1) = \frac{1}{1 + \left|\delta(x^{(i)}(t))\right|} \left(x^{(i)}(t) + \sum_{y \in \delta(x^{(i)}(t))} y \right), \tag{4}$$

公式(4)中，$\boldsymbol{x}^{(i)}(t+1)$表示点 $\boldsymbol{x}^{(i)}$在第 t 个同步后的更新位置。

公式(4)与 Mean Shift 聚类算法[19 - 20]中的下一个搜索位置的计算公式相似。从中我们可以看出，点 $\boldsymbol{x}^{(i)}$的更新位置就是点 $\boldsymbol{x}^{(i)}$及它的 δ 近邻点集 $\delta(\boldsymbol{x}^{(i)})$的均值位置。

公式(4)还可以被改写为：

$$\begin{aligned} \boldsymbol{x}^{(i)}(t+1) &= \boldsymbol{x}^{(i)}(t) + \sum_{y \in \delta(x^{(i)}(t))} \left(y - x^{(i)}(t+1) \right) \\ &= \boldsymbol{x}^{(i)}(t) + \frac{1}{1 + \left|\delta(x^{(i)}(t))\right|} \sum_{y \in \delta(x^{(i)}(t))} \left(y - x^{(i)}(t) \right), \end{aligned} \tag{5}$$

公式(5)与公式(2)在外形方面有点相似，但它们有着本质的区别。我们可以看到，公式(2)所表示的更新模型是非线性的，而公式(4)与公式(5)的更新模型是线性的。

定义 5. 应用到聚类中的第一种指数衰减加权同步模型定义为：

如果将点 $\boldsymbol{x}^{(i)}$ (i = 1, …, n) = ($x_1^{(i)}$, …, $x_d^{(i)}$)看作是在 $\boldsymbol{S}$ = {$\boldsymbol{x}^{(1)}$,…, $\boldsymbol{x}^{(n)}$}范围内的一个相振荡器，根据指数衰减加权模型的启发，点 $\boldsymbol{x}^{(i)}$的一个指数衰减加权的动力学特性可描述为：

$$\boldsymbol{x}^{(i)}(t+1) = \frac{\sum_{j=1}^{n}\left(w^{(ji)}(t)\cdot x^{(j)}(t)\right)}{\sum_{l=1}^{n} w^{(li)}(t)}, \tag{6}$$

公式(6)中，$w^{(ji)}(t) = \exp(-\frac{\left\|x^{(j)}(t)-x^{(i)}(t)\right\|^2}{2})$表示在第 t 个同步时点 $\boldsymbol{x}^{(j)}$影响点 $\boldsymbol{x}^{(i)}$的指数衰减权重系数，$\boldsymbol{x}^{(i)}(t+1)$表示点 $\boldsymbol{x}^{(i)}$在第 t 个同步后的更新位置。

定义 6. 应用到聚类中的第二种指数衰减加权同步模型定义为：

如果将点 $\boldsymbol{x}^{(i)}$ (i = 1, …, n) = ($x_1^{(i)}$, …, $x_d^{(i)}$)看作是在 $\boldsymbol{S}$ = {$\boldsymbol{x}^{(1)}$, …, $\boldsymbol{x}^{(n)}$}范围内的一个相振荡器，根据指数衰减加权模型的启发，点$\boldsymbol{x}^{(i)}$在第 k 维上的分量$x_k^{(i)}$ (k = 1, …, d)的一个指数衰减加权的动力学特性可描述为：

$$x_k^{(i)}(t+1) = \frac{\sum_{j=1}^{n}\left(w_k^{(ji)}(t)\cdot x_k^{(j)}(t)\right)}{\sum_{l=1}^{n} w_k^{(li)}(t)}, \tag{6}$$

公式(6)中，$w_k^{(ji)}(t) = \exp(-\frac{\left(x_k^{(j)}(t)-x_k^{(i)}(t)\right)^2}{2})$ 表示在第 t 个同步时点 $\boldsymbol{x}^{(j)}$在第 k 维上的分量 $x_k^{(j)}$影响点 $\boldsymbol{x}^{(i)}$在第 k 维上的分量 $x_k^{(i)}$的指数衰减权重系数，$x_k^{(i)}$ (t+1)表示点 $\boldsymbol{x}^{(i)}$在第 k 维上的分量 $x_k^{(i)}$在第 t 个同步后的更新值。

定义 7. 应用到聚类中的 δ 近邻指数衰减加权同步模型定义为：

如果将点 $\boldsymbol{x}^{(i)}$ (i = 1, …, n) = ($x_1^{(i)}$, …, $x_d^{(i)}$)看作是在它的 δ 近邻点集 $\delta(\boldsymbol{x}^{(i)})$里的一个相振荡器，根据指数衰减加权模型的启发，点$\boldsymbol{x}^{(i)}$在第 k 维上的分量$x_k^{(i)}$ (k = 1, …, d)的一个指数衰减加权的动力学特性可描述为：

$$x_k^{(i)}(t+1) = \frac{\sum_{j\in NearIndex(x^{(i)}(t))\cup\{i\}}\left(w_k^{(ji)}(t)\cdot x_k^{(j)}(t)\right)}{\sum_{j\in NearIndex(x^{(i)}(t))\cup\{i\}}\left(w_k^{(ji)}(t)\right)}, \tag{7}$$

公式(7)中，$NearIndex(\boldsymbol{x}^{(i)}(t)) = \{j \mid \boldsymbol{x}^{(j)} \in \delta(\boldsymbol{x}^{(i)})\}$表示点 $\boldsymbol{x}^{(i)}$的近邻点集 $\delta(\boldsymbol{x}^{(i)})$中的所有点的标号构成的集合，$w_k^{(ji)}(t) = \exp(-\frac{\left(x_k^{(j)}(t)-x_k^{(i)}(t)\right)^2}{2})$ 表示在第 t 个同步时点 $\boldsymbol{x}^{(i)}$的近邻点 $\boldsymbol{x}^{(j)}$在第 k 维上的分量 $x_k^{(j)}$影响点 $\boldsymbol{x}^{(i)}$在第 k 维上的分量 $x_k^{(i)}$的指数衰减权重系数，$x_k^{(i)}$ (t+1)表示点 $\boldsymbol{x}^{(i)}$在第 k 维上的分量 $x_k^{(i)}$在第 t 个同步后的更新值。

定义 8. 数据集 $\boldsymbol{S}$ = {$\boldsymbol{x}^{(1)}$, $\boldsymbol{x}^{(2)}$,…, $\boldsymbol{x}^{(n)}$}在经历了 T 步的同步聚类后稳定下来，表示为 $\boldsymbol{S}(T)$ = {$\boldsymbol{x}^{(1)}(T)$, …, $\boldsymbol{x}^{(n)}(T)$}。设从 $\boldsymbol{S}(T)$可获得 $K_{clusters}$ 个聚类和 $K_{isolates}$ 个孤立点，$K = K_{clusters} + K_{isolates}$。$\boldsymbol{S}(T)$的聚类稳定点集记为 $\boldsymbol{C}$ = {$\boldsymbol{c}^{(1)}$, …, $\boldsymbol{c}^{(Kclusters)}$}，其中 $\boldsymbol{c}^{(j)}$是第 j 个聚类区域的稳定标记位置。如果 $\boldsymbol{x}^{(i)}(T)$满足公式(8)，则点 $\boldsymbol{x}^{(i)}$判为

属于第 j 个聚类区域。

$$dis(\boldsymbol{c}^{(j)},\ \boldsymbol{x}^{(i)}(T)) < \varepsilon \tag{8}$$

公式(8)中的参数 ε 与算法中的迭代退出阈值参数 ε 相同，是一个非常小的实数。

如果第 j 个聚类区域包含了 n_j 个数据点，则这 n_j 个数据点在原始数据集 $\boldsymbol{S} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)},\ldots, \boldsymbol{x}^{(n)}\}$ 中的均值位置 $\boldsymbol{m}^{(j)}$ 与第 j 个聚类稳定点位置 $\boldsymbol{c}^{(j)}$ 之间的距离表示为 $dis(\boldsymbol{c}^{(j)}, \boldsymbol{m}^{(j)})$，则 $K_{clusters}$ 个聚类区域的稳定位置与 $K_{clusters}$ 个聚类区域的均值位置差异值定义为

$$diffence(Steadies, Means) = \frac{1}{K_{clusters}} \sum_{j=1}^{K_{clusters}} dis(c^{(j)},\ m^{(j)}) \tag{8}$$

显然，这个指标可度量同步聚类后聚类区域的稳定位置与均值位置的平均差异。

3 几种基于同步模型的聚类算法描述

Böhm 等[8]提出的 SynC 算法，Chen[16]提出的 ESynC 算法是 3.1 节中基于近邻同步模型的聚类算法的两个典型代表，基于指数衰减加权同步模型的聚类算法将在 3.2 节中予以介绍。

3.1 基于近邻同步模型的聚类算法

算法 1: **基于近邻同步模型的聚类算法**

输入：数据点集 $\boldsymbol{S} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)},\ldots, \boldsymbol{x}^{(n)}\}$, 距离相异性度量 $dis(\cdot, \cdot)$, 近邻阈值参数 δ, 迭代退出阈值参数 ε；

输出：数据点集 $\boldsymbol{S}$ 的聚类结果，可以采用数据点的聚类归属标记数组 $FinCluResult[1 .. n]$；

1: 迭代步 t 初始化为 0，即：$t \leftarrow 0$;
2: **for** ($i = 1$; $i \le n$; i++)
3: $\quad \boldsymbol{x}^{(i)}(t) \leftarrow \boldsymbol{x}^{(i)}$;
4: **while** ($\boldsymbol{S}(t) = \{\boldsymbol{x}^{(1)}(t), \ldots, \boldsymbol{x}^{(n)}(t)\}$仍在同步移动中)
5: {
6: $\quad$ **for** ($i = 1$; $i \le n$; i++)
7: $\quad$ {
8: $\quad\quad$ 根据公式(1)为数据点 $\boldsymbol{x}^{(i)}$构造它的 δ 近邻点集 $\delta(\boldsymbol{x}^{(i)})$;
9: $\quad\quad$ 根据定义 2，或定义 3，或定义 4，或定义 7 计算 $\boldsymbol{x}^{(i)}(t)$同步后的更新值 $\boldsymbol{x}^{(i)}(t+1)$;
10: $\quad$ }
11: $\quad$ 计算 $\boldsymbol{S}(t) = \{\boldsymbol{x}^{(1)}(t), \ldots, \boldsymbol{x}^{(n)}(t)\}$与第 t 个同步后的更新值 $\boldsymbol{S}(t+1) = \{\boldsymbol{x}^{(1)}(t+1), \ldots, \boldsymbol{x}^{(n)}(t+1)\}$之间的均方差 $mse(\boldsymbol{S}(t), \boldsymbol{S}(t+1)) = \frac{1}{n}\sum_{i=1}^{n}\left\|x^{(i)}(t) - x^{(i)}(t+1)\right\|^2$;
12: $\quad$ **if** ($mse(\boldsymbol{S}(t), \boldsymbol{S}(t+1)) < \varepsilon$)
13: $\quad\quad$ 这个动态的同步聚类过程收敛了，可以从 while 循环中退出来;
14: $\quad$ **else**
15: $\quad\quad$ 迭代步 t 增加一步，即：t++，然后进入下一次 while 循环;
16: }
17: 将退出 while 循环的 t 值赋给同步总次数 T，此时可得到数据点集 $\boldsymbol{S}$ 同步后的收敛结果集 $\boldsymbol{S}(T) = \{\boldsymbol{x}^{(1)}(T), \ldots, \boldsymbol{x}^{(n)}(T)\}$;
18: 在 $\boldsymbol{S}(T)$中，那些代表一些数据点的稳定位置可看作是聚类中心，那些只代表一个或几个点的稳定位置可看作是孤立点的最终同步位置。根据 $\boldsymbol{S}(T) = \{\boldsymbol{x}^{(1)}(T), \ldots, \boldsymbol{x}^{(n)}(T)\}$很容易得到一个由若干个稳定点或孤立点构成的自然聚类簇，从而构造出 $FinCluResult[1 .. n]$。

备注：在算法 1 的第 9 步中，如果采用定义 2 来计算 $\boldsymbol{x}^{(i)}(t)$同步后的更新值 $\boldsymbol{x}^{(i)}(t+1)$，就是 Böhm 等[8]提出的

SynC 算法；如果采用定义 4 来计算 $\boldsymbol{x}^{(i)}(t)$同步后的更新值 $\boldsymbol{x}^{(i)}(t+1)$，就是 Chen[16]提出的 ESynC 算法；定义 3 的同步模型在文献[16]中已有几个对照实验；如果采用定义 7 来计算 $\boldsymbol{x}^{(i)}(t)$同步后的更新值 $\boldsymbol{x}^{(i)}(t+1)$，会得到一种同步聚类算法。

3.2 基于指数衰减加权同步模型的聚类算法

算法 2: **基于指数衰减加权同步模型的聚类算法**

输入：数据点集 $\boldsymbol{S} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n)}\}$, 距离相异性度量 $dis(\cdot, \cdot)$, 迭代退出阈值参数 ε；

输出：数据点集 $\boldsymbol{S}$ 的聚类结果，可以采用数据点的聚类归属标记数组 $FinCluResult[1 .. n]$；

1: 迭代步 t 初始化为 0，即：$t \leftarrow 0$;
2: **for** ($i = 1$; $i \le n$; i++)
3: $\quad \boldsymbol{x}^{(i)}(t) \leftarrow \boldsymbol{x}^{(i)}$;
4: **while** ($\boldsymbol{S}(t) = \{\boldsymbol{x}^{(1)}(t), \ldots, \boldsymbol{x}^{(n)}(t)\}$仍在同步移动中)
5: {
6: $\quad$ **for** ($i = 1$; $i \le n$; i++)
7: $\quad\quad$ 根据定义 5，或定义 6 计算 $\boldsymbol{x}^{(i)}(t)$同步后的更新值 $\boldsymbol{x}^{(i)}(t+1)$;
8: $\quad$ 计算 $\boldsymbol{S}(t) = \{\boldsymbol{x}^{(1)}(t), \ldots, \boldsymbol{x}^{(n)}(t)\}$与第 t 个同步后的更新值 $\boldsymbol{S}(t+1) = \{\boldsymbol{x}^{(1)}(t+1), \ldots, \boldsymbol{x}^{(n)}(t+1)\}$之间的均方差 $mse(\boldsymbol{S}(t), \boldsymbol{S}(t+1)) = \dfrac{1}{n}\sum_{i=1}^{n}\left\|x^{(i)}(t) - x^{(i)}(t+1)\right\|^2$;
9: $\quad$ **if** ($mse(\boldsymbol{S}(t), \boldsymbol{S}(t+1)) < \varepsilon$)
10: $\quad\quad$ 这个动态的同步聚类过程收敛了，可以从 while 循环中退出来;
11: $\quad$ **else**
12: $\quad\quad$ 迭代步 t 增加一步，即：t++，然后进入下一次 while 循环;
13: }
14: 将退出 while 循环的 t 值赋给同步总次数 T，此时可得到数据点集 $\boldsymbol{S}$ 同步后的收敛结果集 $\boldsymbol{S}(T) = \{\boldsymbol{x}^{(1)}(T), \ldots, \boldsymbol{x}^{(n)}(T)\}$;
15: 在 $\boldsymbol{S}(T)$中，那些代表一些数据点的稳定位置可看作是聚类中心，那些只代表一个或几个点的稳定位置可看作是孤立点的最终同步位置。根据 $\boldsymbol{S}(T) = \{\boldsymbol{x}^{(1)}(T), \ldots, \boldsymbol{x}^{(n)}(T)\}$很容易得到一个由若干个稳定点或孤立点构成的自然聚类簇，从而构造出 $FinCluResult[1 .. n]$。

备注：在算法 2 的第 9 步中，如果采用定义 5 来计算 $\boldsymbol{x}^{(i)}(t)$同步后的更新值 $\boldsymbol{x}^{(i)}(t+1)$，会得到一种同步聚类算法；如果采用定义 6 来计算 $\boldsymbol{x}^{(i)}(t)$同步后的更新值 $\boldsymbol{x}^{(i)}(t+1)$，会得到另外一种同步聚类算法。

3.3 算法的复杂度分析与改进

3.3.1 算法1的复杂度分析与改进

步 1 只需要 $O(1)$的时间代价，步 2 和步 3 需要 $O(n)$的时间代价。

步 4 需要 $O(T)$的时间代价。

在步 8 中，如果采用简单的全范围蛮力判断方法，为数据点 $\boldsymbol{x}^{(i)}$ ($i = 1, \ldots, n$)构造它的 δ 近邻点集 $\delta(\boldsymbol{x}^{(i)})$需要 Time = $O(dn)$。当数据集的维数较小时，通过为数据点集构造空间索引结构的方法，可以取得 $O(d\log n)$的时间代价。在步 9 中，计算 $\boldsymbol{x}^{(i)}(t)$ ($i = 1, \ldots, n$)同步后的更新值 $\boldsymbol{x}^{(i)}(t+1)$需要 Time = $O(d \bullet |\delta(\boldsymbol{x}^{(i)})|)$。所以步 6 到步 10 最多需要 $O(dn^2)$的时间代价。

步 11 需要 $O(dn)$的时间代价。

步 12 到步 15 只需要 $O(1)$的时间代价。

步 17 只需要 $O(1)$的时间代价。步 18 需要 $O(dnK)$的时间代价，其中 K 是聚类数和孤立点数之和，一般有 $K << n$。

根据 Böhm 等[8]和我们的分析，算法 1 未采用空间索引结构时的时间代价为 Time = $O(Tdn^2)$；算法 1 在低维数据集中，采用空间索引结构时的时间代价为 Time = $O(Tdn\log n)$。

3.3.2 算法2的复杂度分析

步 1，步 2，步 3 和步 4 的时间代价与算法 1 相同。

在步 6 和步 7 中，使用定义 5 来计算 $w^{(ji)}(t)$ (i , j = 1, …, n)需要 Time = $O(d)$，计算 $\boldsymbol{x}^{(i)}(t)$ (i = 1, …, n)同步后的更新值 $\boldsymbol{x}^{(i)}(t+1)$需要 Time = $O(dn)$；使用定义 6 来计算 $w_k^{(ji)}(t)(i,j=1,\cdots,n;k=1,\cdots,d)$ 需要 Time = $O(1)$，计算 $x_k^{(i)}(t)(i=1,\cdots,n;k=1,\cdots,d)$ 同步后的更新值 $x_k^{(i)}(t+1)$ $\boldsymbol{x}^{(i)}(t+1)$需要 Time = $O(n)$。所以步 6 到步 7 需要 $O(dn^2)$的时间代价。

步 8 需要 $O(dn)$的时间代价。

步 9 到步 12 只需要 $O(1)$的时间代价。

步 14 只需要 $O(1)$的时间代价。步 15 需要 $O(dnK)$的时间代价，其中 K 是聚类数和孤立点数之和，一般有 $K \ll n$。

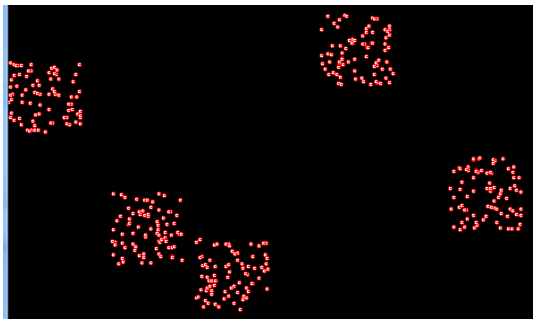
根据上面的分析，算法 2 的时间代价为 Time = $O(Tdn^2)$。

4 几种同步模型的聚类过程示例与相关算法比较

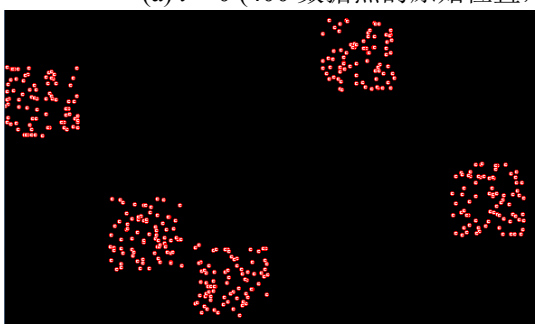
通过比较定义 2、定义 3、定义 4、定义 5、定义 6 和定义 7，可以看出，除了定义 2 是非线性的同步聚类模型，其它几个定义都是线性的同步聚类模型。

图 1 比较了一个 400 个数据点的几个同步聚类模型的同步聚类演变轨迹。参数 δ 设置为 30，参数 ε 设置为 0.00001，最大迭代次数设置为 50，定义 3 中的 $v(t)\cdot\Delta t$ 设置为 0.1。

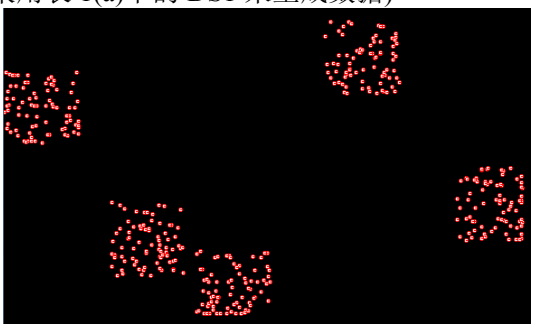
从图 1 中，我们观察到除了基于定义 4 的同步聚类模型外，其它几个同步聚类模型都不能获得良好的局部聚类效果。



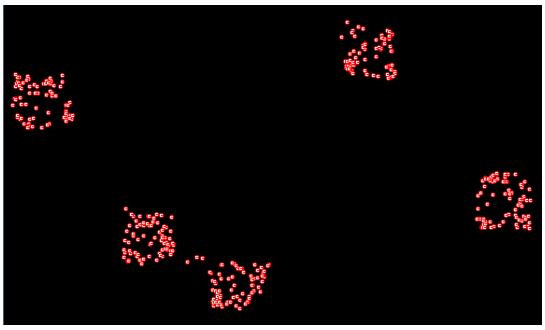
(a) t = 0 (400 数据点的原始位置，采用表 1(a)中的 DS1 来生成数据)



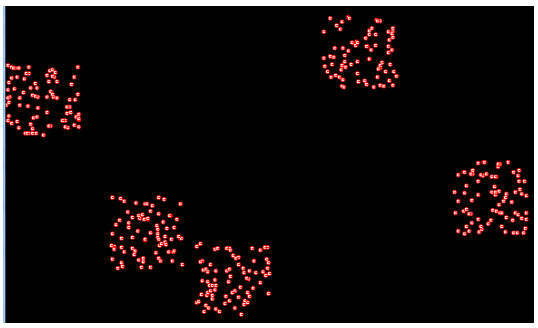
(b-1) 定义 2 同步模型, t = 1 (#*NC* = 399)



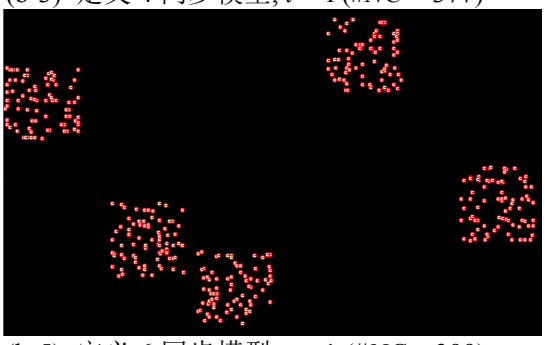
(b-2) 定义 3 同步模型, t = 1 (#*NC* = 399)



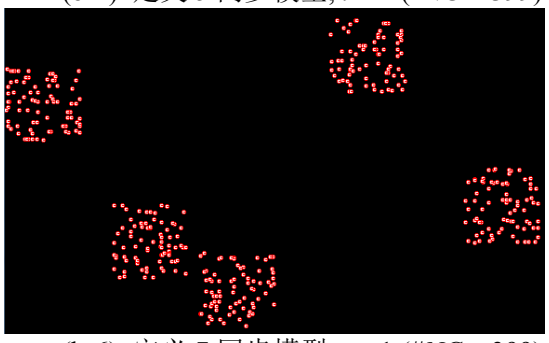
(b-3) 定义 4 同步模型, $t = 1$ ($\#NC = 377$)



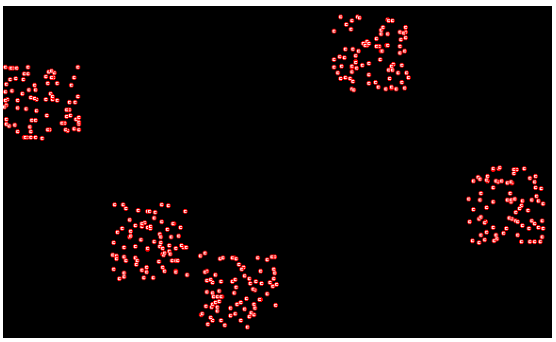
(b-4) 定义 5 同步模型, $t = 1$ ($\#NC = 399$)



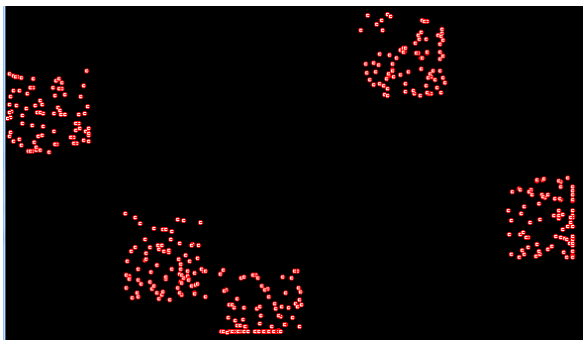
(b-5) 定义 6 同步模型, $t = 1$ ($\#NC = 399$)



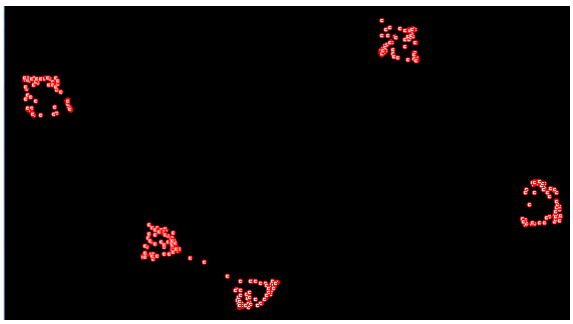
(b-6) 定义 7 同步模型, $t = 1$ ($\#NC = 399$)



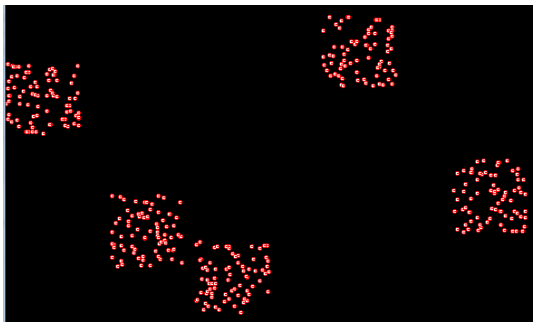
(c-1) 定义 2 同步模型, $t = 2$ ($\#NC = 399$)



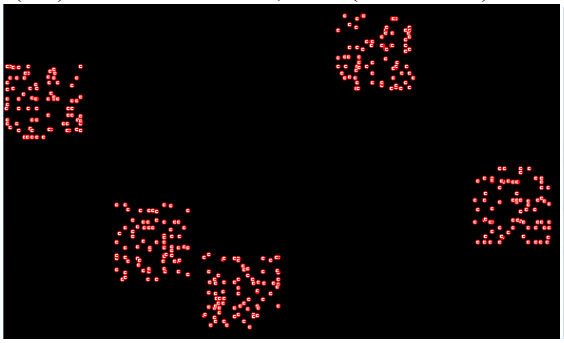
(c-2) 定义 3 同步模型, $t = 2$ ($\#NC = 399$)



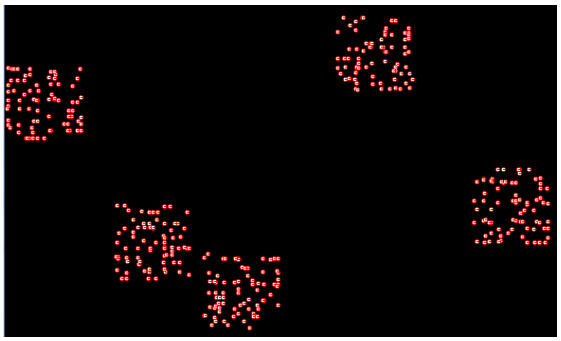
(c-3) 定义 4 同步模型, $t = 2$ ($\#NC = 325$)



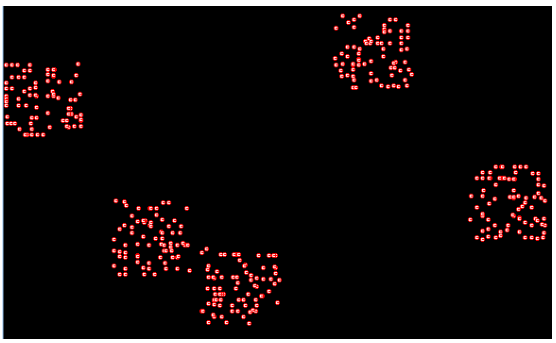
(c-4) 定义 5 同步模型, $t = 2$ ($\#NC = 399$)



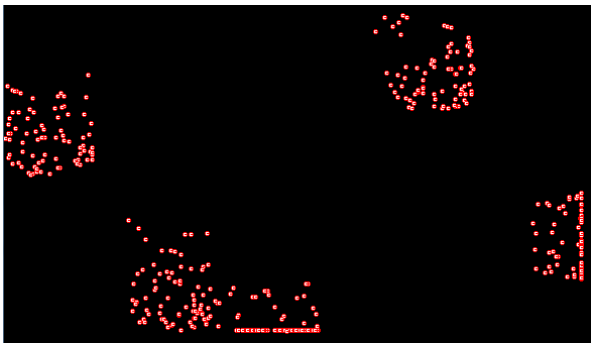
(c-5) 定义 6 同步模型, $t = 2$ ($\#NC = 399$)



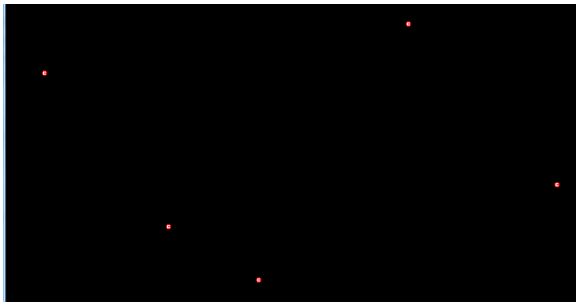
(c-6) 定义 7 同步模型, $t = 2$ ($\#NC = 397$)



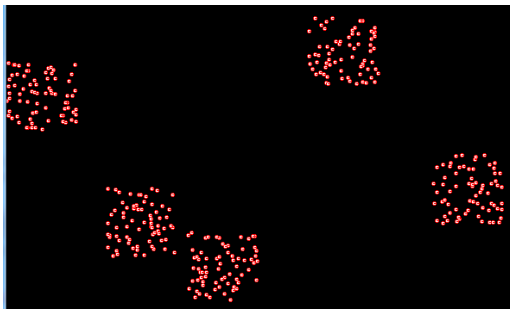
(d-1) 定义 2 同步模型, $t = 5$ ($\#NC = 399$)



(d-2) 定义 3 同步模型, $t = 5$ ($\#NC = 398$)

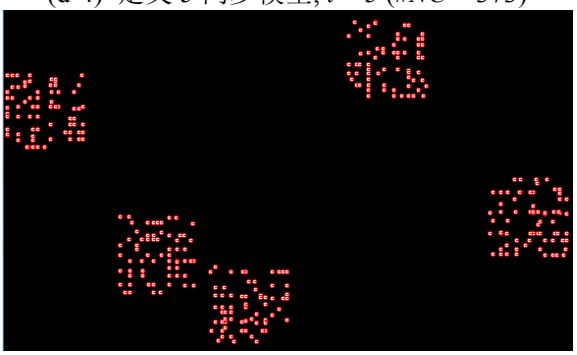


(d-3) 定义 4 同步模型, $t = 5$ ($\#NC = 5$)

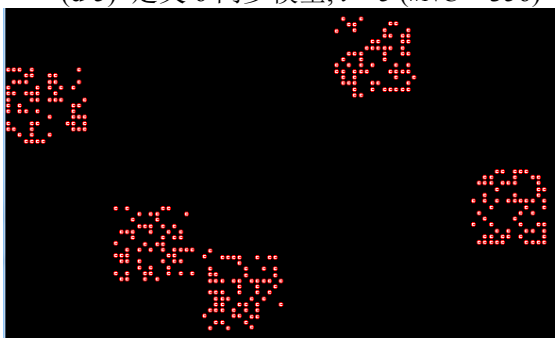


(d-4) 定义 5 同步模型, $t = 5$ ($\#NC = 373$)

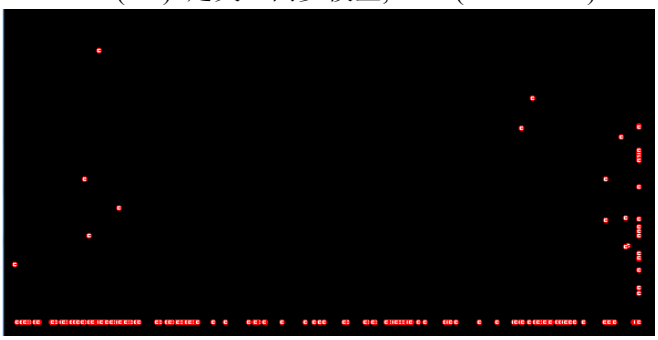
(d-5) 定义 6 同步模型, $t = 5$ ($\#NC = 356$)



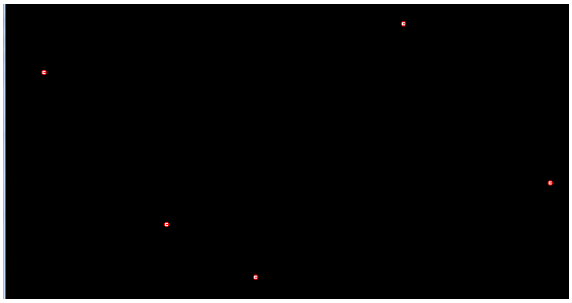
(d-6) 定义 7 同步模型, $t = 5$ ($\#NC = 351$)



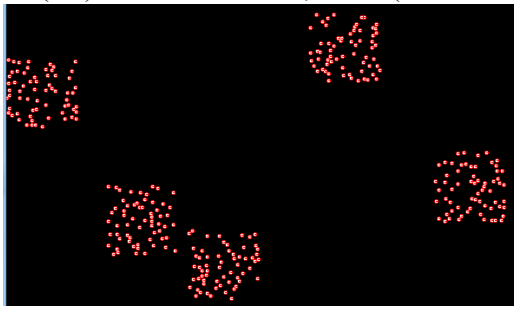
(e-1) 定义 2 同步模型, $t = 50$ ($\#NC = 324$)



(e-2) 定义 3 同步模型, $t = 50$ ($\#NC = 398$)

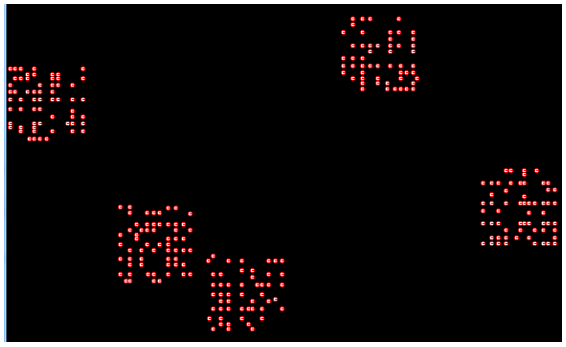


(e-3) 定义 4 同步模型, $t = 50$ ($\#NC =$)



(e-4) 定义 5 同步模型, $t = 50$ ($\#NC = 317$)

(e-5) 定义 6 同步模型, $t = 50$ ($\#NC = 286$)



(e-6) 定义 7 同步模型, $t = 50$ ($\#NC = 283$)

图 1. 基于定义 2、定义 3、定义 4、定义 5、定义 6 和定义 7 的同步聚类模型的动态聚类过程比较

5 仿真实验

5.1 仿真实验设计

相关实验均在 Intel(R) Celeron(R) CPU 3855U 1.6GHz 上进行，配备 8G 内存，在 Windows 7 下的 Visual C++6.0 开发环境下采用 C 语言和 C++语言混合编程进行仿真实验验证。

为证实本文算法的聚类效果和时间效率上的改进，我们采用表 1 中描述的多个人工数据集(由 2 个函数根据设定的参数生成数据)，八个 UCI 数据集(Frank et al., 2010)和三张图片数据集做一些仿真实验。

实验数据集采用多个人工数据集和 3 张大小分别为 100*100 和 200*200 像素的图片。表 1(a)给出 12 类人工数据集的基本描述信息，它们是由两个程序在区域[0, 600] × [0, 600]内生成的多维数据，数据点数目有 1000, 5000, 10000 三种。表 1(b)是八个 UCI 数据集(Frank et al., 2010)的简单描述。表 1(c)是三张图片数据集的简单描述。

表 1. 仿真实验数据集的描述

Tab. 1 Twelve kinds artificial data sets

(a) 一些 2 维实值型人工数据集的描述

人工数据集	预设 (实际)聚类数目	有无噪声	聚类边界最大半径	维数(d)
DS1	5 (5)	no	40	2
DS2	5 (4)	yes	50	2
DS3	7 (6)	no	30	2
DS4	7 (5)	yes	40	2

(b) 八个 UCI 数据集(Frank et al., 2010)

(b) The description of eight UCI data sets (Frank and Asuncion, 2010)

UCI 数据集	点数目 (n)	维数(d)	类别分布	类别数目
Iris	150	4	{Setosa: 50, Versicolor: 50, Virginica: 50}	3
Wine	178	13	{1: 59, 2: 71, 3: 48}	3
Wdbc	569	30	{B: 357, M: 212}	2
Glass	214	9	{Window: {FB: 70, FV: 17, NFB: 76}, Non-window: {C: 13, T: 9, H: 29}}	6
Ionosphere	351	34	{Good: 225, Bad: 126}	2
Letter-recognition	20000	16	{A: 443, B: 460, C: 449, …, Z: 408}	26
Segmentation	210	19	{Brickface: 30, Sky: 30, Foliage: 30, Cement: 30, Window: 30, Path: 30, Grass: 30}	7
Cloud	2048	10	{1: 2014, 2: 2014}	2

(c) 三张图片数据集(来自于 Internet)

(c) Three picture pixel data sets

图片数据集	象素点数目(*n*)	维数(*d*)
Picture1	100*100	3
Picture2	100*100	3
Picture3	200*200	3

这里采用多个不同类型的数据集对几种基于同步模型的聚类算法(分别基于定义 2、定义 3、定义 4、定义 5 和定义 6 构造的同步聚类算法)在时间代价(单位为秒)、同步迭代次数、聚类质量或聚类纯度(采用三个基于信息论的度量指标 AMI / AVI / NMI)方面进行适当的比较，从而比较验证几种同步聚类算法的聚类有效性差异。

5.2 仿真实验结果

5.2.1 人工数据集的实验结果

表 2 是六种基于同步模型的聚类算法在人工数据集上的实验结果比较。在表 2 中，数据点数目 n 及算法所需设置的参数 δ 的具体值附在数据集后的括号内。三个指标 AMI /AVI / NMI 的取值基本上可以反映出聚类质量（采用人工观察到的聚类区域或孤立点来设置基准聚类标号）的好坏。差异值反映出同步聚类后聚类区域的稳定点集与平均点集之间的平均差异。时间和迭代次数反映出聚类速度。

表 2. 人工数据集的实验结果

Tab. 2 The expermental results of artifical data sets

数据集	同步聚类模型	聚类结果度量指标				
		AMI /AVI / NMI	聚类数目	差异值	时间(Sec)	迭代次数
DS1 (n = 1000, δ = 30)	定义 2	0.0948 / 0.1731 / 0.5067	598	2.162558	94	613
	定义 3	-	-	-	-	- (不收敛)
	定义 4	**1 / 1 / 1**	**5**	1.011055	**1**	**6**
	定义 5	0.1258 / 0.2234 / 0.5158	492	0.088730	725	1185
	定义 6	0.1377 / 0.2421 / 0.5195	456	1.962208	2206	1820
	定义 7	0.1394 / 0.2447 / 0.5201	451	1.923661	219	1153
DS2 (n = 1000, δ = 70)	定义 2	0.0550 / 0.1043 / 0.4623	720	2.381432	1663	9819
	定义 3	-	-	-	-	- (不收敛)
	定义 4	**0.9765 / 0.9881 / 0.9883**	7 (**4** clusters + 3 noises)	0.118941	**1**	**6**
	定义 5	0.0852 / 0.1570 / 0.4714	590	0.063084	1124	1941
	定义 6	0.0902 / 0.1655 / 0.4729	568	2.161002	1041	840
	定义 7	0.0934 / 0.1708 / 0.4741	558	2.092545	231	749
DS3 (n = 1000, δ = 28)	定义 2	0.1235 / 0.2198 / 0.5348	518	2.054773	70	405
	定义 3	-	-	-	-	- (不收敛)
	定义 4	**1 / 1 / 1**	**6**	0.656240	**1**	**5**
	定义 5	0.1710 / 0.2920 / 0.5496	385	0.142658	928	1192
	定义 6	0.1889 / 0.3178 / 0.5557	345	1.780673	1796	1402
	定义 7	0.1917 / 0.3217 / 0.5569	340	1.713524	75	388
DS4 (n = 1000, δ = 35)	定义 2	-	-	-	-(时间长)	-
	定义 3	-	-	-	-	- (不收敛)
	定义 4	**1 / 1 / 1**	20 (**5** clusters + 15 noises)	0.247009	**2**	**12**
	定义 5	0.1076 / 0.1943 / 0.5174	550	0.063754	703	1102
	定义 6	0.1225 / 0.2182 / 0.5223	503	2.123284	845	611
	定义 7	0.1301 / 0.2302 / 0.5245	476	2.062374	262	1318

5.2.1 UCI 数据集的实验结果

表 3 是几种基于同步模型的聚类算法在八个 UCI 数据集上的实验结果比较。在表 3 中，算法所需设置的参数δ的具体值附在数据集后的括号内。三个指标 AMI /AVI / NMI 的取值基本上可以反映出聚类纯度（采用数据集本身的类别号来设置基准聚类标号）的好坏。差异值反映出同步聚类后聚类区域的稳定点集与平均点集之间的平均差异。时间和迭代次数反映出聚类速度。

表 3. 八个 UCI 数据集的实验结果

Tab. 3 The expermental results of eight UCI data sets

(a)

数据集	同步聚类模型	聚类结果度量指标				
		AMI /AVI / NMI	聚类数目	差异值	时间(Sec)	迭代次数
Iris (δ = 120)	定义 2	-	-	-	-(时间长)	-
	定义 3	-	-	-	-(时间长)	-
	定义 4	**0.7143 / 0.7190 / 0.7265**	5 (**3** clusters + 2 isolates)	24.152181	**0**	9
	定义 5	0.0050 / 0.0100 / 0.4697	147 (2 clusters + 145 noises)	**0**	**0**	**1**
	定义 6	0.0050 / 0.0100 / 0.4697	147 (2 clusters + 145 noises)	**0**	**0**	**1**
	定义 7	0.0050 / 0.0100 / 0.4697	147 (2 clusters + 145 noises)	**0**	**0**	**1**
Wine (δ = 305)	定义 2	-	-	-	-(时间长)	-
	定义 3	-	-	-	-(时间长)	-
	定义 4	**0.6057 / 0.7259 / 0.7615**	19 (**3** clusters + 16 isolates)	2.985147	**0**	**6**
	定义 5	3.2528e-16 / 6.5056e-16 / 0.4578	178 noises	**0**	**0**	**1**
	定义 6	3.2528e-16 / 6.5056e-16 / 0.4578	178 noises	6.269711	3245	12563
	定义 7	3.2528e-16 / 6.5056e-16 / 0.4578	178 noises	3.178303	77	2848
Wdbc (δ = 325)	定义 2	-	-	-	-(时间长)	-
	定义 3	-	-	-	-(时间长)	-
	定义 4	**0.3277 / 0.4205 / 0.4717**	47 (3 clusters + 44 isolates)	3.571035	**1**	**8**
	定义 5	6.8369e-16 / 1.3674e-15 / 0.3226	569 noises	**0**	**0**	**1**
	定义 6	-	-	-	-(时间长)	-
	定义 7	-	-	-	-(时间长)	-
Glass (δ = 148)	定义 2	-	-	-	-(时间长)	-
	定义 3	-	-	-	-(时间长)	-
	定义 4	**0.2872** / **0.3432** / 0.4540	35 (**6** clusters + 29 isolates)	5.444073	**0**	**6**
	定义 5	0.0012 / 0.0025 / **0.5306**	213 noises	**0**	**0**	**1**
	定义 6	0.0012 / 0.0025 / **0.5306**	213 noises	5.421137	2482	11395
	定义 7	0.0012 / 0.0025 / **0.5306**	213 noises	4.169292	282	6457

(b)

数据集	同步聚类模型	聚类结果度量指标				
		AMI /AVI / NMI	聚类数目	差异值	时间(Sec)	迭代次数
Ionosphere (δ = 615)	定义 2	-	-	-	-(时间长)	-
	定义 3	-	-	-	-(时间长)	-
	定义 4	**0.1073 / 0.1701 / 0.3106**	85 (**2** clusters + 83 isolates)	1.622768	**0**	**9**
	定义 5	3.5016e-04 / 7.0007e-04 / 0.3339	350 isolates	**0**	**0**	**1**
	定义 6	-	-	-	-(时间长)	-
	定义 7	-	-	-	-(时间长)	-
Letter-recognition (δ = 210)	定义 2	-	-	-	-(时间长)	-
	定义 3	-	-	-	-(时间长)	-
	定义 4	**0.3986 / 14.8254 / 0.3986**	34(**26** clusters + 8 isolates)	46.197392	**4006**	**23**
	定义 5	0.0161 / 0.0317 / 0.5768	18668	**0**	**546**	**1**
	定义 6	-	-(空间不够)	0	-	-
	定义 7	0.0161 / 0.0317 / 0.5768	18668	-	206	1
Segmentation (δ = 205)	定义 2	-	-	-	-(时间长)	-
	定义 3	-	-	-	-(时间长)	-
	定义 4	**0.4212 / 0.5093 / 0.6086**	38 (**7** clusters + 31 isolates)	7.615040	**0**	**7**
	定义 5	-1.6974e-15 / -3.3948e-15 / 0.6033	210 isolates	0	0	1
	定义 6	-1.6974e-15 / -3.3948e-15 / 0.6033	210 isolates	8.502851	6674	16898
	定义 7	-1.6974e-15 / -3.3948e-15 / 0.6033	210 isolates	4.997892	378	13173
Cloud (δ = 380)	定义 2	-	-	-	-(时间长)	-
	定义 3	-	-	-	-(时间长)	-
	定义 4	**1** / **1** / **1**	2 clusters	31.107646	**8**	**6**
	定义 5	1.98689e-04 / 3.9729e-04 / 0.3016	2044 isolates	0.002257	123	40
	定义 6	-	-	-	-(时间长)	-
	定义 7	-	-	-	-(时间长)	-

5.3 实验结果分析及结论

从表 2 的实验结果可以看出，基于定义 4 的同步聚类算法的聚类质量总体上最优，聚类速度最快。从表 3 的实验结果可以看出，基于定义 4 的同步聚类算法的聚类纯度总体上最优。

从表 2 和表 3 还可以看出，有的 UCI 数据集采用几个同步聚类模型时，要不运行时间过长，难以在有效时间范围内达到迭代的终止条件，要不就是只经历一次迭代就满足了迭代终止条件。基于定义 4 的同步聚类算法中迭代次数、聚类效果上面总体上是最为有效的。这种基于近邻范围内的均值位置更新式（即基于定义 4）的迭代同步模型在聚类分析领域非常有效，其效果超过基于定义 2 的同步聚类模型（即文献[8]首次提出的扩展 Kuramoto 同步聚类模型）、基于定义 3 的同步聚类模型和本文提出的另外三种同步聚类模型（分别基于定义 5、定义 6 和定义 7）。

另外，我们还观察到，AMI 和 AVI 指标能较好的度量聚类质量，而 NMI 指标有时偏好于聚类结果中簇数目较多的情况。具体来说就是，当聚类结果与基准聚类不一致时，簇数目较多的聚类结果的 NMI 值远远大于簇数目较少的聚类结果。例如，Glass 数据集有 210 个数据点，当把每个点当作一个单聚类（即孤立点）时，其 NMI 值为 0.5306，大于 **6** 聚类和 29 孤立点时的 NMI 值 0.4540。

6 结束语

在聚类领域，常用的同步模型有扩展的 Kuramoto 模型[8]，基于 Vicsek 模型的线性版本[16]，基于第二个线性版本的 Vicsek 模型[21]，万有引力同步模型[22]等。扩展的 Kuramoto 模型采用一种非线性的动态同步迭代公式来计算每一个迭代步骤的新位置，而基于 Vicsek 模型的线性版本采用一种线性的动态同步迭代公式来计算每一个迭代步骤的新位置，基于第二个线性版本的 Vicsek 模型采用一种加权的线性同步迭代公式来计算每一个迭代步骤的新位置。从本文及文献[16]和文献[21]的一些仿真实验可以看出，后两种同步模型更适合应用于聚类分析。

下一步，我们将试图从理论上证明这些同步聚类模型的迭代收敛性，并将 ESynC 算法应用到文本聚类和 Web 日志分析中。

参考文献

[1] Tan, P. N., Steinbach, M., Kumar, V. Introduction to Data Mining [M]. Pearson Education Publishers, Inc, 2006.

[2] Qlan W., Zhou A. Analyzing Popular Clustering Algorithms from Different Viewpoints [J]. Journal of Software (China). 2002, 13(8): 1382-1394.

[3] Johannes, G. & Andreas, R. Techniques of Cluster Algorithms in Data Mining [J]. Data Mining and Knowledge Discovery, 2002, 6(4): 303-360.

[4] Arabie P. & Hubert L. J. An overview of combinatorial data analysis [M]. Clustering and Classifications, World Scientific Publishing Co. 1996, 188-217.

[5] Jain A. K., Murty M. N., & Flynn P. J. Data Clustering: A review [J]. ACM Computing Surveys, 1999, 31(3): 264-323.

[6] Xu, R. Wunsch, D. Survey of clustering algorithms [J]. IEEE Transactions on Neural Networks, 2005, 16(3): 645-678.

[7] Jain A. K. Data clustering: 50 years beyond K-means [J]. Pattern Recognition Letters, 2010, 31(8): 651-666.

[8] Böhm, C., Plant, C., Shao, J., et al. Clustering by synchronization [C]. In SIGKDD, 2010, 583-592.

[9] GrÄunwald, P. A tutorial introduction to the minimum description length principle [M]. In: Grunwald P, Myung I J, PittM, eds. Advances in Minimum Description Length: Theory and Applications. Cambridge: MIT Press, 2005. 1-80.

[10] Shao, J., Hahn, K., Yang, Q., et al. (2010). Hierarchical density-based clustering of white matter tracts in the human brain [J]. International Journal of Knowledge Discovery in Bioinformatics, 1(4):1–25.

[11] Shao, J., Yang, Q., Böhm, C. et al. Detection of arbitrarily oriented synchronized clusters in high-dimensional data [C]. In ICDM, 2011, 607-616.

[12] Shao, J., He, X., C. Böhm, C. et al. Synchronization inspired partitioning and hierarchical clustering [J]. IEEE Transaction on Knwoledge and Data Engineering, 2013, 25(4): 893-905.

[13] Shao, J., He, X., Plant, C. et al. Robust synchronization-based graph clustering [C]. In PAKDD, 2013, 249-260.

[14] Shao, J., Ahmadi, Z., & Kramer, S. Prototype-based Learning on Concept-drifting Data Streams [C]. In KDD, 2014, 412-421.

[15] Huang, J., Kang, J., Qi, J., & Sun, H. A hierarchical clustering method based on a dynamic synchronization model [J]. Science in China Series F: Information Sciences, 2013, 43: 599-610.

[16] Chen, X. (2017). An effective synchronization clustering algorithm [J]. Applied Intelligence, 46(1): 135 – 157.

[17] Hang, W., Choi, K., & Wang, S. Synchronization clustering based on central force optimization and its extension for large-scale datasets [J]. Knowledge-Based Systems, 2017, 118, 31–44.

[18] Chen, X. Fast synchronization clustering algorithms based on spatial index structures [J]. Expert Systems with Applications, 2018, 94: 276 - 290.

[19] Fukunaga, K. & Hostetler, L. The estimation of the gradient of a density function, with applications in pattern recognition [J]. IEEE Transactions on Information Theory, 1975, 21(1): 32–40.

[20] Comaniciu, D. & Meer, P. Mean shift: A robust approach toward feature space analysis [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002, 24(5): 603–619.

[21] Xinquan Chen. Synchronization clustering based on a linearized version of Vicsek model [cs.DB]. arXiv: 1411.0189. http://arxiv.org/abs/1411.0189, 2014.

[22] 乔颖，王士同，杭文龙.大规模数据集引力同步聚类[J]. 控制与决策, 2017, 32(6):1075-1083

作者通讯地址、邮编、电话和 Email 地址：
安徽省芜湖市北京中路安徽工程大学计算机与信息学院
陈新泉 （收）
邮编 241000
15123428097
chenxqscut@126.com