具有虚拟领导的 Flocking 聚类算法

李 强 何 衍 蒋静坪
(浙江大学电气工程学院 杭州 310027)

摘 要：该文提出一种改进的带虚拟领导的 Flocking 模型，并基于此模型开发了一种数据聚类算法。在此算法中，数据集中的数据点被考虑为可以在空间中移动的 Agent，并且根据改进的模型，生成有权无向图。然后从数据集中选定一组虚拟领导，每个数据点与其中 γ 个虚拟领导建立连接。所有与这个数据点有连接的邻居，都通过一个势函数产生场，对这个数据点进行作用，此数据点将沿着所有场矢量叠加的方向移动一段距离。算法中，虚拟领导的加入有效减少了数据点，特别是邻居较少的数据点向某个中心收敛的时间。在所有数据点不断受到作用而移动的过程中，同类的数据点就会逐渐地聚集到一起，而不同类的数据点则相互远离，最后自动形成聚类。此算法的实验结果表明，数据点能合理有效地被聚类，并且算法具有较快的收敛速度，同时，与其他算法对比也验证了此算法的有效性。

关键词：数据聚类；无监督学习；Flocking 模型；虚拟领导
中图分类号：TP391 **文献标识码**：A **文章编号**：1009-5896(2009)08-1846-06

A New Clustering Algorithm Based on Flocking with Virtual Leaders

Li Qiang He Yan Jiang Jing-ping
(*College of Electrical Engineering*, *Zhejiang University*, *Hangzhou* 310027, *China*)

Abstract: A modified flocking model with virtual leaders is proposed, and then a clustering algorithm based on it is developed. In the algorithm, firstly a weighted and undirected graph is established by all data points in a dataset according to the model, where each data point is regarded as an agent who can move in space. Then a set of virtual leaders are identified, from which γ virtual leaders are chosen and linked by each data point. Furthermore, each data point is acted by all linked data points through fields established by them in space, so it will take a step along the direction of the vector sum of all fields. Thanks to those virtual leaders, they accelerate the rate of convergence of the algorithm. As all data points move in space according to the proposed model, data points that belong to the same class are located at a same position, whereas those that belong to other classes are away from one another. Consequently, the experimental results demonstrate that data points in datasets are clustered reasonably and efficiently, and in several iterations the convergence of the algorithm will be reached. Moreover, the comparison with other algorithms also provides an indication of the effectiveness of proposed approach.

Key words: Data clustering; Unsupervised learning; Flocking model; Virtual leaders

1 引言

数据聚类是模式识别领域中一个重要的研究课题。在过去几十年中，许多优秀的聚类算法被相继提出，如 K-means[1]，支持向量聚类(SVC)[2]等，这些算法主要将重点放在找到聚类中心和聚类边界上。但是要聚类或分类的数据点都是固定不动的，研究者提出各种算法，设计各种函数去找到复杂的聚类或分类平面。然而，近年来，一些研究者提出，为什么数据点不能像 Agent 一样，可以自己根据一些规则，在空间中自由移动，而自动聚集到一起呢？因此，这些研究者根据他们的新思想，开发出一些令人兴奋的新的聚类算法，如基于粒子群(PSO)[3]，基于蚂蚁群[4]，基于 Flocking[5]的聚类算法。这些新的聚类算法都有一个显著的特征，那就是数据点本身可以根据一些规则在空间中移动，从而自动地实现数据聚类。

Flocking 是群居动物的一种合作行为，它通过种群中个体间的相互作用而实现某种整体的目标[6]，这种行为在鸟群、蜂群、鱼群等动物中大量存在。在过去十年中，这一领域吸引了来自生物学、物理学、社会学、计算机科学等不同背景研究者的浓厚兴趣，共同探索通过局域相互作用而导致群体行为

2008-07-28收到，2009-03-02改回
国家自然科学基金(60405012, 60675055)和浙江省自然科学基金(Y1080776)资助课题

自然涌现的机理[7]。同时，Flocking 模型也被广泛应用于工程之中。

但是，Flocking 模型在数据聚类方面应用得并不多。例如 Cui 等[5]提出一种基于 Flocking 的文件聚类算法，在他们的算法中，首先将高维数据点映射为一个二维虚拟空间的 boids；然后，利用 Reynolds 提出的 3 条规则[8]：聚集(Cohesion)、分离(Separation)、对齐(Alignment)，并且另外设计 1 条“特征相似与不相似”规则；最后，通过这 4 条规则和一组预设的权值来得到一个速度向量，控制每个 boid 的移动，从而实现文件聚类。与他们的算法不同，在本文的算法中，并不将高维数据点进行映射，而是将每个数据点看作是可以在空间中移动的 Agent，直接在高维空间中进行聚类。另外，通过引入一个势函数 $\phi(\bullet)$，只需要 Reynolds 的两条规则：聚集和分离，就能实现数据聚类。并且，在要聚类的数据点中，选定少量的虚拟领导，从而解决边界点由于缺乏局域作用而移动缓慢的问题。

2　改进模型

假设一个有 N 个可移动 Agent 的集合 $\boldsymbol{X}=\{\boldsymbol{X}_1,\boldsymbol{X}_2,\cdots,\boldsymbol{X}_N\}$，分布在一个 m 维距离空间中，其中每个 $\boldsymbol{X}_i$ 表示一个 Agent 在空间中的位置。在此空间中定义一个距离函数 $d:\boldsymbol{X}\times\boldsymbol{X}\to\mathbb{R}$，它满足两个 Agent 间的距离越近，函数值越小。计算集合 $\boldsymbol{X}$ 中任意两 Agent 的距离，可形成距离矩阵 $\boldsymbol{D}=[d(\boldsymbol{X}_i,\boldsymbol{X}_j)]_{i,j=1,2,\cdots,N}$。假设每个 Agent 能感知一个半径为 R 的范围内的其他 Agent，并与他们分别连接一条边，那么 N 个 Agent 即可建立一个有权无向图 $G(\boldsymbol{X}(t),E(t),d)$。

定义 1　对于一个有 N 个可移动 Agent 的自治系统，其形成的有权无向图 $G(\boldsymbol{X}(t),E(t),d)$ 为

$$\left.\begin{aligned}&\boldsymbol{X}(t)=\{\boldsymbol{X}_i(t),i=1,2,\cdots,N\}\\&E(t)=\bigcup_{i=1}^{N}E_i(t)\\&E_i(t)=\left\{e\left(\boldsymbol{X}_i(t),\boldsymbol{X}_j(t)\right)\middle|j\in\Upsilon_i(t)\right\}\\&\Upsilon_i(t)=\left\{j\middle|d\left(\boldsymbol{X}_i(t),\boldsymbol{X}_j(t)\right)\le R,\boldsymbol{X}_j(t)\in\boldsymbol{X}\right\}\end{aligned}\right\}\tag{1}$$

其中，每个 Agent $\boldsymbol{X}_i$ 对应图 G 中的一个顶点，$\Upsilon_i(t)$ 为 Agent $\boldsymbol{X}_i$ 的邻集。

随着 Agent 在空间中移动，其位置坐标将不断随时间发生变化，这就使得在它的瞬时 R 邻域内，与他有联系的其他 Agent，在每一时刻都不尽相同，也就是说，它的边集 $E_i(t)$ 和邻集 $\Upsilon_i(t)$ 都是时变的。这样，由 N 个可移动 Agent 形成的有权无向图 $G(\boldsymbol{X}(t),E(t),d)$，其形状也将随时间变化。

初始时，需要从集合 $\boldsymbol{X}$ 中选择 $\beta=\eta\cdot N$ 个 Agent 作为虚拟领导，形成虚拟领导集合 V：

$$V=\arg\max_{j\in\{1,2,\cdots,N\}}\beta\left(\left\{\mathrm{Deg}_j(0),j\in\{1,2,\cdots,N\}\right\}\right)\tag{2}$$

这里，函数 $\arg\max\beta(\bullet)$ 表示取度最大的 β 个 Agent 作为虚拟领导。然后，每个 Agent $\boldsymbol{X}_i$ 从虚拟领导集合 V 中，选择 $\gamma\le\beta$ 个作为自己的虚拟领导，从而形成其总的邻集 $\Gamma_i(t)$：

$$\left.\begin{aligned}&\Gamma_i(t)=\Upsilon_i(t)\cup\Psi_i(t)\\&\Psi_i(t)=\arg\min_{j\in V}\gamma\left(\left\{d\left(\boldsymbol{X}_i(t),\boldsymbol{X}_j(t)\right),j\in V\right\}\right)\end{aligned}\right\}\tag{3}$$

进一步假设每个 Agent $\boldsymbol{X}_i$ 通过一个势函数 $\phi(\bullet)$ 形成一个局域势场，此势函数 $\phi(\bullet)$ 是度 $\mathrm{Deg}(\bullet)$ 和距离 $d(\bullet,\bullet)$ 的函数。如果 Agent $\boldsymbol{X}_j$ 是 Agent $\boldsymbol{X}_i$ 的邻居，即它们之间有一条连边，那么 Agent $\boldsymbol{X}_j$ 在 $\boldsymbol{X}_i$ 处形成的场，由下面的公式计算：

$$\left.\begin{aligned}&\phi\left(\boldsymbol{X}_i(t),\boldsymbol{X}_j(t)\right)\cdot\boldsymbol{n}_{ij}=\begin{cases}b\cdot\boldsymbol{n}_{ij}, & d\left(\boldsymbol{X}_i(t),\boldsymbol{X}_j(t)\right)>\theta\\0, & \text{其他}\end{cases}\\&b=\frac{\mathrm{Deg}_j(t)}{\left(d\left(\boldsymbol{X}_i(t),\boldsymbol{X}_j(t)\right)*d\left(\boldsymbol{X}_i(0),\boldsymbol{X}_j(0)\right)\right)^3}\end{aligned}\right\}\tag{4}$$

这里，$\boldsymbol{n}_{ij}=\boldsymbol{X}_j(t)-\boldsymbol{X}_i(t)$ 是一个指向 Agent $\boldsymbol{X}_j$ 的向量；$d\left(\boldsymbol{X}_i(t),\boldsymbol{X}_j(t)\right)$ 和 $d\left(\boldsymbol{X}_i(0),\boldsymbol{X}_j(0)\right)$ 分别表示 Agent $\boldsymbol{X}_i$ 与 $\boldsymbol{X}_j$ 当前的距离和初始时它们之间的距离；θ 表示分离阈值，即当 Agent $\boldsymbol{X}_i$ 与 $\boldsymbol{X}_j$ 之间的距离小于等于阈值 θ 时，Agent $\boldsymbol{X}_i$ 不受 Agent $\boldsymbol{X}_j$ 产生的场的作用。

Agent $\boldsymbol{X}_i$ 的邻集 $\Gamma_i(t)$ 中的所有邻居，在 $\boldsymbol{X}_i$ 处均产生一个场。由于场满足线性叠加原理，因此，$\boldsymbol{X}_i$ 处的总场之和为

$$\ddot{\boldsymbol{X}}_i(t)=\sum_{j\in\Gamma_i(t)}\phi\left(\boldsymbol{X}_i(t),\boldsymbol{X}_j(t)\right)\cdot\boldsymbol{n}_{ij}\tag{5}$$

Agent $\boldsymbol{X}_i$ 在所有邻居的作用下，将沿势场之和 $\ddot{\boldsymbol{X}}_i(t)$ 指向的方向，移动一段距离 l_i。我们假设 Agent $\boldsymbol{X}_i$ 移动足够缓慢，移动前后的速度始终保持为 0，这样，模型就不再需要 Reynolds 的对齐规则。并且，Agent $\boldsymbol{X}_i$ 移动的距离与 $\ddot{\boldsymbol{X}}_i(t)$ 的大小成正比，但不能超过数据点 $\boldsymbol{X}_i$ 与其邻域内的邻居的距离的加权平均 l_i^{ave}，

$$l_i=\alpha\cdot\sum_{j\in\Gamma_i(t)}\phi\left(\boldsymbol{X}_i(t),\boldsymbol{X}_j(t)\right)=\alpha\cdot l_i'\tag{6}$$

这里，α 是一个比例因子，它由下面的公式计算：

$$\left.\begin{aligned}&\alpha=\begin{cases}l_i^{\mathrm{ave}}/l_i', & l_i'/l_i^{\mathrm{ave}}>1\\1, & \text{其他}\end{cases}\\&l_i^{\mathrm{ave}}=\frac{\sum_{j\in\Upsilon_i(t)}\mathrm{Deg}_j(t)\cdot d\left(\boldsymbol{X}_i(t),\boldsymbol{X}_j(t)\right)}{\sum_{j\in\Upsilon_i(t)}\mathrm{Deg}_j(t)}\end{aligned}\right\}\tag{7}$$

Agent $\boldsymbol{X}_i$ 向 $\ddot{\boldsymbol{X}}_i(t)$ 指向的方向移动一段距离 l_i 后，其坐标按下面的公式更新：

$$\begin{aligned}\boldsymbol{X}_i(t+1) &= \boldsymbol{X}_i(t)+\alpha\cdot\ddot{\boldsymbol{X}}_i(t)\\ &= \boldsymbol{X}_i(t)+\alpha\cdot\sum_{j\in\Gamma_i(t)}\phi\big(\boldsymbol{X}_i(t),\boldsymbol{X}_j(t)\big)\cdot\boldsymbol{n}_{ij}\end{aligned}\tag{8}$$

当系统中所有 Agent 都更新位置坐标之后，模型完成一次迭代。如果所有 Agent 的移动距离之和不大于阈值 ε，即 $\sum_{i=1}^{N} l_i \le \varepsilon$，那么模型停止迭代。

3 算法与分析

假设一个未标记的数据集 $\boldsymbol{X}=\{\boldsymbol{X}_1,\boldsymbol{X}_2,\cdots,\boldsymbol{X}_n\}$，每个数据点有 p 个属性，分布在一个 p 维度量空间中。在本文的算法中，数据集中的每个数据点被考虑为一个可以在整个空间中移动的 Agent，并分别对应图 G 的一个顶点。

3.1 算法描述

步骤1　选择距离函数 $d:\boldsymbol{X}\times\boldsymbol{X}\to\mathbb{R}$，设定作用域 R，根据定义 1 形成的有权无向图 $G(\boldsymbol{X}(t),E(t),d)$；

步骤2　计算每个数据点 $\boldsymbol{X}_i$ 的初始度 $\mathrm{Deg}_i(0)$，设定参数 η、γ 和分离阈值 θ，根据式(2)形成虚拟领导集合 V；

步骤3　每个数据点 $\boldsymbol{X}_i$ 根据式(3)得到总的邻集 $\Gamma_i(t)$；

步骤4　每个数据点 $\boldsymbol{X}_i$ 根据式(4)分别计算 R 邻域内每个邻居在此处产生的场 $\phi(\boldsymbol{X}_i(t),\boldsymbol{X}_j(t))\cdot\boldsymbol{n}_{ij}$，并由式(5)求得 $\boldsymbol{X}_i$ 处的总场 $\ddot{\boldsymbol{X}}_i(t)$；

步骤5　每个数据点 $\boldsymbol{X}_i$ 根据式(6)计算移动的距离 l_i；

步骤6　每个数据点 $\boldsymbol{X}_i$ 根据式(8)更新自己的位置 $\boldsymbol{X}_i(t+1)$；

步骤7　根据距离函数 $d(\bullet,\bullet)$，重新计算距离矩阵 $\boldsymbol{D}(t)=[d(\boldsymbol{X}_i(t),\boldsymbol{X}_j(t))]_{i,j=1,2,\cdots,N}$ 和每个数据点的度 $\mathrm{Deg}_i(t)$；

步骤8　如果所有 Agent 的移动距离之和 $\sum_{i=1}^{N} l_i \le \varepsilon$，那么算法终止，否则，转回步骤 3。

3.2 算法分析

数据点 $\boldsymbol{X}_i$ 在聚类的过程中，不仅受其 R 邻域内的所有邻居的影响，而且还要受到选择的 γ 个虚拟领导的作用。每个邻居在 $\boldsymbol{X}_i$ 处产生一个指向此邻居的场，由式(4)可知，当邻居与数据点 $\boldsymbol{X}_i$ 的距离小于分离阈值 θ 时，此时，数据点 $\boldsymbol{X}_i$ 不受此邻居产生的场的影响，那么数据点 $\boldsymbol{X}_i$ 必然向远离此邻居的方向移动，这样就体现了 Reynolds 的分离规则。所有邻居在 $\boldsymbol{X}_i$ 处产生的总场 $\ddot{\boldsymbol{X}}_i(t)$ 由式(5)计算，数据点 $\boldsymbol{X}_i$ 将要移动的方向与 $\ddot{\boldsymbol{X}}_i(t)$ 的方向一致，并且，移动的距离与总场的大小成正比，这可以看作是 Reynolds 聚集规则的体现。

算法中引入虚拟领导，主要是为了解决边界上的数据点移动距离小，而导致算法收敛速度降低的问题。对于边界上的数据点，它们一般所处区域密度较低，数据点的分布比较稀疏，因此，在其 R 邻域内的邻居点很少，甚至为 0，由式(5)可知，它受到的局域作用就很小。又因为移动距离与局域作用成正比，所以它的移动距离也相应很小。而数据集中的其他数据点受到的作用均大于边界上数据点，因而有较大的移动距离，并很快向某个中心聚集，这就进一步加大了边界点与其 R 邻域内邻居的距离，由式(4)可知，边界点受到的局域作用会进一步减小，从而导致其移动距离也同时减小。而其他数据点的快速移动，加剧了边界点移动距离减小的问题，最终导致边界点的 R 邻域内没有任何邻居点，从而使得此边界点不受任何作用而停留在某个位置不再移动。但是，如果事先设定一组虚拟领导，即使某个边界点的 R 邻域内没有任何邻居点，它仍然会在 γ 个虚拟领导的作用下移动，而不会由于缺乏其他数据点作用而停滞不前。这样，虚拟领导的引入，不仅加速了边界点向中心移动，而且加快了算法的收敛速度。

4 讨论

在这一节，首先讨论在作用域 R 变化的情况下，聚类类数如何变化；然后对算法在不同 R 下的收敛速度进行分析。

4.1 作用域 R 对聚类类数的影响

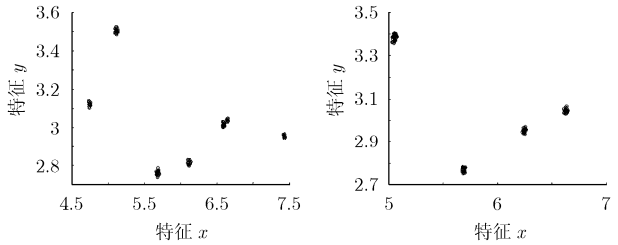
作用域 R 表明每个数据点能感知的邻域的大小，在实验中，对于不同的数据集，要直接给出作用域 R 的值是相当困难的，为简化计算，通过引入一个变量 k，来间接获得作用域 R 的值：

$$R=\mathrm{median}\left(\mathrm{sort}\big(\boldsymbol{D}(0)\big)\cdot[0\ \cdots\ 0\ \overset{k}{1}\ 0\ \cdots\ 0]^{\mathrm{T}}\right)\tag{9}$$

这里，$\mathrm{sort}\big(\boldsymbol{D}(0)\big)$ 表示对初始距离矩阵按行升序排列；函数 $\mathrm{median}(\bullet)$ 表示取向量的中位数。这样，作用域 R 的大小可以方便的通过 k 来调节。

变量 k 的大小直接控制了作用域 R 的大小。对于同一数据集，不同的 R 值部分决定了算法得到的聚类类数，一般来说，随着 R 的增大，算法得到的聚类类数减少。例如，当变量 k 取值较小时，在数据点 $\boldsymbol{X}_i$ 的 R 邻域内，邻居个数也必然较少，从连通性的角度来看，各个数据点的 R 邻域少有重叠，形成的连通域较小。相反，当变量 k 取值较大时，作用域 R 随之增大，这就使得数据点 $\boldsymbol{X}_i$ 能观察到更广

的范围，在其 R 邻域中的邻居必然比 R 较小时的多。这样，各个数据点的邻域就有更多的重叠部分，从而形成更大的连通域，最终就能形成几个大的聚类。图 1 展示了对于同一数据集，在不同的作用域 R 下，得到的聚类类数的变化。当变量 $k=5$ 时，得到 6 个聚类，而当 $k=15$ 时，数据点聚合为 4 个聚类。因此，在不能事先确定聚类类数的情况下，可以根据实际情况，通过调整变量 k，来获得不同的聚类类数。



(a) 作用域 $R(k=5)$　　(b) 作用域 $R(k=15)$

图 1 聚类类数与作用域 R (变量 k)的关系

4.2 作用域 R 对收敛速度的影响

算法的收敛速度也与作用域 R 或变量 k 的大小紧密相连。作用域 R 越大，数据点 $\boldsymbol{X}_i$ 邻域中的邻居就越多，那么，由式(5)可知，数据点 $\boldsymbol{X}_i$ 受到所有邻居的局域作用就越强。又由于数据点 $\boldsymbol{X}_i$ 受到其他邻居作用后的移动距离与总场的大小 $\ddot{\boldsymbol{X}}_i(t)$ 成正比，所以，每一次数据点 $\boldsymbol{X}_i$ 的移动距离也较大，在这种情况下，算法的收敛速度加快。但是，由于收敛速度太快，使得数据点探索的范围有限。相反，如果作用域 R 较小，那么与它有相互作用的邻居则不多，受到的局域作用较小，因此，移动距离也较小。这就使得数据点 $\boldsymbol{X}_i$ 有更多的机会在空间中探索，从而接触到更多的其他数据点，但是，这也减缓了算法的收敛速度。

因此，在选择作用域 R (变量 k)的大小时，应综合考虑收敛速度和探索范围之间的关系。对于同一数据集，在不同的作用域 R (变量 k)下，算法收敛速度的比较如图 2 所示。图中每一点表示一次移动之后，系统中所有数据点的移动距离之和 $\sum_{i=1}^{N} l_i \le \varepsilon$。

从图 2 可以看出，移动距离之和随作用域 R 增大而增大。特别是，当迭代次数为 1，变量 $k=14$ 时，系统中所有数据点的转移距离之和明显大于 $k=5$ 时的移动距离之和。这就使得算法的收敛速度随作用域 R 增加而加快。另一方面，虽然在不同作用域 R 下，算法的收敛速度有差别，但是从总体来看，算法本身仍具有较快的收敛速度。

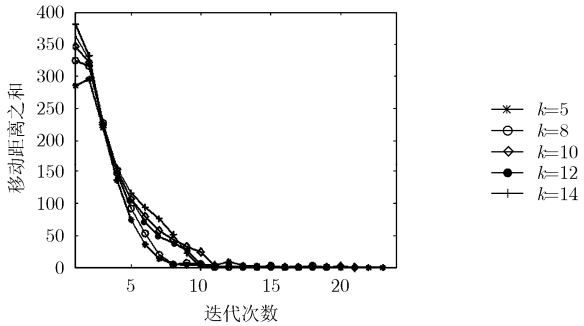


图 2 作用域 R (变量 k)对收敛速度的影响

5 实验

为了测试本文算法，我们从 UCI 数据库[9]中选择了 4 个数据集：Iris，Wine，Glass 和 Breast cancer Wisconsin，并在所选数据集上完成仿真实验。在这一节中，首先简要介绍了这 4 个数据集的情况和实验中参数的设定，然后给出算法的实验结果。

5.1 实验设定

实验用的这 4 个数据集，原始数据均有多个特征，分布在一个高维空间中。其样本点个数、特征以及类数等信息概括在表 1 中。另外，对于 Breast 数据集中丢失的属性，用一个随机数替换。最后，此算法用 MATLAB 6.5 编程实现，并在所有数据集上完成实验。

表 1 实验数据集

数据集	样本数	特征	类数
Iris	150	4	3
Wine	178	13	3
Glass	214	9	6
Breast	699	9	2

在所有实验中，数据集 $\boldsymbol{X}$ 中的所有数据点，都被考虑为可以在空间移动的 Agent，它们的初始位置直接从数据集中取值，移动规则由模型给出。数据集中数据点的距离测量，通过距离函数 $d(\bullet,\bullet)$ 计算得到，实验中，选用普通欧氏距离作为距离函数。另外，模型中虚拟领导的比例 $\eta=0.1$，即虚拟领导的总数 $\beta=0.1N$；每个 Agent 从虚拟领导集合中选择连接的虚拟领导个数 $\gamma=k/2\le\beta$；分离阈值 $\theta=0.1$。

5.2 实验结果

我们将提出的聚类算法，分别应用于前述的 4 个 UCI 数据集。针对每个数据集，分别取不同的作用域 R (变量 k)，再运行算法，可得到不同作用域 R (变量 k)下的聚类结果。对于同一数据集，由前部

分的分析可知，聚类类数随作用域 R 的减小而增大。当作用域 R 较小时，算法结束时得到的聚类类数很可能大于数据集指定的类数，那么将使用合并子程序，合并多余的类。合并时，首先找出包含最少数据点的类，将它与距离最近的类合并，重复这一过程，直到类数等于数据集指定的类数为止。算法得到的聚类结果，用聚类精度的形式表示，聚类精度定义如下：

定义 2[10] 令 cluster_i 为算法分配给数据集中样本点 $\boldsymbol{X}_i$ 的类标签，c_i 为样本点 $\boldsymbol{X}_i$ 在数据集中的实际类标签。那么，聚类精度为

$$\left.\begin{aligned}&\text{accuracy}=\frac{\sum_{i=1}^{n}\lambda\left(\text{map}\left(\text{cluster}_i\right),c_i\right)}{n}\\&\lambda\left(\text{map}\left(\text{cluster}_i\right),c_i\right)=\begin{cases}1, & \text{当}\text{map}\left(\text{cluster}_i\right)=c_i\\0, & \text{其他}\end{cases}\end{aligned}\right\}\tag{10}$$

其中映射函数 $\text{map}(\bullet)$ 将算法得到的类标签集，映射到数据集的实际类标签集。

图 3 是每个数据集在不同作用域 R 下运行后得到的结果，图中每个点表示算法的聚类精度。从图上可以看出，对于同一数据集，在不同作用域 R (变量 k)下，都得到了近似的结果。

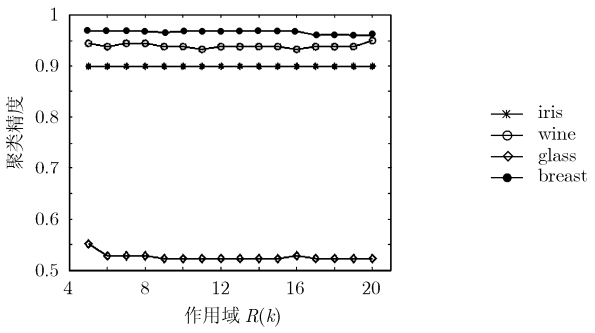


图 3 作用域 R (变量 k)对聚类精度的影响

同时，我们也与其他几种聚类算法(Kmeans[11]，PCA-Kmeans[11]，LDA-Km[11]和 DMVC[12])在同一数据集上的结果作了简单的比较，如表 2 所示。

表 2 不同算法聚类精度比较(%)

算法名称	Iris	Wine	Glass	Breast
本文算法	90	94.94	55.14	96.71
Kmeans	89.3	70.2	47.2	95.99
PCA-Kmeans	88.7	70.2	45.3	96.03
LDA-Km	98	82.6	51	–
DMVC	84	95.5	48.6	96.3

从表中可以看出，具有虚拟领导的 Flocking 聚类算法在各个数据集上都展示出不错的性能。

6 结论

本文提出一种改进的带虚拟领导的 Flocking 模型，并基于此模型发展了一种数据聚类算法，此算法将数据集中的数据点考虑为可以在空间中移动的 Agent。在选定距离函数 $d:\boldsymbol{X}\times\boldsymbol{X}\to\mathbb{R}$ 之后，通过设定一个作用域 R，根据定义 1 即可生成数据点间的基本连接关系 $G(\boldsymbol{X}(t),E(t),d)$。然后，从数据集中选择 β 个数据点形成虚拟领导集合 V，每个数据点 $\boldsymbol{X}_i$ 从虚拟领导集合中选择离自己最近的 $\gamma\le\beta$ 个，分别建立指向这些虚拟领导的连接，从而形成总的邻集 $\Gamma_i(t)$。通过为每个数据点设计一个势函数 $\phi(\bullet)$，邻集 $\Gamma_i(t)$ 中的每个元素在 $\boldsymbol{X}_i$ 处产生一个场，数据点 $\boldsymbol{X}_i$ 在所有场的共同作用下，沿总的场所指向的方向移动一段距离 l_i。在数据点移动的过程中，有权无向图 $G(\boldsymbol{X}(t),E(t),d)$ 的形状随时间变化。初始时，所有数据点的移动距离之和较大，而当聚类形成时，所有数据点的移动距离之和趋于一个很小的值。在算法中，我们设定，当所有数据点的移动距离之和小于阈值 ε 时，算法结束。

模型为数据点的聚类提供了启发式，最后，属于同一类的数据点互相靠近而形成紧的聚类，而不同的聚类则相互远离。此算法能在事先不确切知道聚类类数的情况下，通过调整作用域 R (变量 k)来控制聚类类数的多少，即作用域 R (变量 k)越大，得到的聚类类数越少。本文在 4 个 UCI 数据集上对算法进行评估，实验结果表明，数据集中的数据点都能有效合理地被聚类，且算法在不同的作用域 R 下都有较快的收敛速度。另外，此聚类算法能对任意形状、大小、密度的数据集进行聚类，而且，聚类的结果具有稳定性。

参 考 文 献

[1] MacQueen J. Some methods for classification and analysis of multivariate observations. Proceedings of Fifth Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, USA, 1967, 1: 281–297.

[2] Ben-Hur A, Horn D, and Siegelmann H T, *et al*.. Support Vector Clustering. *Journal of Machine Learning Research*, 2002, (2): 125–137.

[3] Merwe V D and Engelbrecht A P. Data clustering using particle swarm optimization. Proceedings of IEEE Congress on Evolutionary Computation 2003, Canbella, Australia, 2003: 215–220.

[4] Handl J and Meyer B. Improved ant-based clustering and

sorting in document retrieval interface. Proceedings of the Seventh International Conference on Parallel Problem Solving from Nature, 2002, LNCS 2439: 913–923.

[5] Cui Xiao-hui, Gao Jin-zhu, and Potok T E. A flocking based algorithm for document clustering analysis. *Journal of Systems Architecture*, 2006, 52(8): 505–515.

[6] Olfati-Saber R. Flocking for multi-agent dynamic systems: algorithms and theory. *IEEE Transactions on Automatic Control*, 2006, 51(3): 401–420.

[7] Parrish J K, Viscido S V, and Grunbaum D. Self-organized fish schools: An examination of emergent properties. *Biological Bulletin*, 2002, 202(2): 296–305.

[8] Reynolds C W. Flocks, herds and schools: A distributed behavioral model. *SIGGRAPH Computer Graphics*, 1987, 21 (4): 25–34.

[9] Blake C L and Merz C J. UCI Repository of machine learning databases. http://archive.ics.uci.edu/ml/, 1998.

[10] Günes E. Language model-based document clustering using random walks. Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL, New York, 2006: 479–486.

[11] Ding C and Li T. Adaptive Dimension Reduction Using Discriminant Analysis and K-means Clustering. Proceedings of the 24th International Conference on Machine Learning, Corvallis, OR, USA, 2007: 521–528.

[12] Song L, Smola A, and Gretton A, *et al.*. A dependence maximization view of clustering. Proceedings of the 24th International Conference on Machine Learning, Corvallis, OR, USA, 2007: 815–822.

李　强：　男，1978 生，博士生，研究方向为模式识别、量子计算.

何　衍：　男，1973 生，副教授，研究方向为信息融合、多机器人协作.

蒋静坪：　男，1935 生，教授，博士生导师，研究方向为基于网络的控制、模式识别.