西安电子科技大学
学位论文独创性（或创新性）声明

秉承学校严谨的学风和优良的科学道德，本人声明所呈交的论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果；也不包含为获得西安电子科技大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中做了明确的说明并表示了谢意。

申请学位论文与资料若有不实之处，本人承担一切的法律责任。

本人签名：郑智　　日期 2010.12.21

西安电子科技大学
关于论文使用授权的说明

本人完全了解西安电子科技大学有关保留和使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权单位属西安电子科技大学。本人保证毕业离校后，发表论文或使用论文工作成果时署名单位仍然为西安电子科技大学。学校有权保留送交论文的复印件，允许查阅和借阅论文；学校可以公布论文的全部或部分内容，可以允许采用影印、缩印或其他复制手段保存论文。（保密的论文在解密后遵守此规定）

本人授权西安电子科技大学图书馆保存学位论文，并同意将论文在互联网上发布。

本人签名：郑智　　日期 2010.12.21

导师签名：　　日期 2010.12.21

摘要

聚类算法是一种广泛应用于数据挖掘、机器学习、图像处理等若干领域的重要技术。进化聚类算法是聚类算法中的重要分支，本文旨在提出两种新的进化聚类算法，即混合属性进化聚类算法和混合策略进化聚类算法。本文对聚类问题和进化计算做了简要的介绍，详细描述和讨论了提出的两种进化聚类算法，并在人工数据集和UCI数据集上对算法进行了性能测试和分析。

本文在第二章提出了针对混合属性数据集的进化聚类算法。它基于K原型算法，应用进化计算的框架和算子求得混合属性数据集的合理划分，因此可以被看做是一种进化K原型算法。作为一种基于划分的聚类算法，K原型算法是针对混合属性数据集的著名算法。然而，由于它采用K均值的迭代方式，所以它对初始原型敏感且容易陷入局部最优。进化计算具有全局搜索能力，因此本文采用进化计算操作K原型算法以克服原始算法的缺陷。对人工数据集和UCI数据集的实验表明，本章提出的混合属性进化聚类算法比原始的K原型算法性能更优。

本文在第三章提出了基于多种群的混合进化聚类算法，它属于混合策略进化聚类算法的一种。此算法采用多种群策略和一种新的抽取策略以传递父代个体的有用信息给子代。父代个体是从每个候选种群中选出的一个最好个体。整个数据集被建模成一幅无向图，因此本算法使用基于图的KWNC标准作为适应度函数选择个体。此外，进化中还使用了约简策略，以加快运行速度。对于进化收敛后的不同情形，本章设计了相应的终止方案以得到最终聚类结果。对人工数据集和UCI数据集的实验表明，本章算法能够找到比嵌入其中的候选算法和进化K均值算法更优的聚类结果。

本文在第四章提出了基于多种群和图搜索的混合进化聚类算法，它也属于混合策略进化聚类算法的一种。此算法采用与第三章算法类似的框架，适应度函数，抽取策略以及终止策略。不同点在于本章用三种不同算法产生层级聚类候选种群，增大了其多样性；此外两种不同的基于图的搜索策略被用于寻找更合理的数据划分。这些方法都大大提高了算法的性能。对人工数据集和UCI数据集的实验表明，本章算法不仅具有第三章算法的优点，并且其对比于其它算法的优势大大增强。最后本章对重要的参数做了分析以更好地阐述算法的特点。

本文工作得到国家自然科学基金(批准号：60703107)、国家863项目(批准号：2009AA12Z210)、教育部新世纪优秀人才支持计划(批准号：NCET-08-0811)，陕西省青年科技新星支持计划(批准号：2010KJXX-03)，和中央高校基本科研业务费重

点项目（批准号：K50510020001）资助。

关键词：聚类 进化计算 混合属性 混合策略 多种群 基于图的搜索

Abstract

Clustering is an important technology which has been studied wildly in many areas such as data mining, machine learning, image processing and so on. Evolutionary clustering algorithm is a kind of new and significant methods in clustering study. This thesis proposes a novel evolutionary clustering algorithm for mixed type data and two new hybrid schemes evolutionary clustering algorithms. First of all, the explanation of clustering problem and evolution computation is given. Then the proposed algorithms are discussed one by one with experiments and analysis.

In the second chapter, a novel unsupervised evolutionary clustering algorithm for mixed type data is proposed. It is based on the K-prototype algorithm and applys evolutionary framework and operators to get reasonable partition of mixed type datasets. As a partitional clustering algorithm, K-prototype algorithm is a well-known one for mixed type data. However, it applies K-means paradigm, so it is sensitive to initialization and converges to local optimum easily. Global searching ability is one of the most important advantages of evolutionary algorithm (EA), so an EA framework is introduced to help KP overcome its flaws. Experiments on synthetic and UCI datasets show that EKP is more robust and can generate much better results than KP.

Chapter three proposes a new hybrid evolutionary clustering algorithm with multi-population, which is a kind of hybrid schemes evolutionary clustering algorithm. It applies multi-population strategy and a new extraction procedure to transmit valuable information from parents in different candidate populations to offspring. Dataset is modeled as a graph and the graph-based KWNC criterion is used as the fitness function to select individuals. Reduction strategy is also aplied to speed up the whole algorithm. When evolution converges, different strategies for different evolutionary results are desinged to terminate the algorithm and get the final result. In experiments, artificial datasets and UCI datasets are used to test the proposed algorithm. The results show that the proposed algorithm is possible of finding better clustering results than the candidate clustering algorithms embedded in it and a traditional evolutionary K-means algorithm.

In chapter four, another hybrid schemes evolutionary clustering algorithm is proposed, i.e., evolutionary clustering algorithm with multi-population and graph-based search. This algorithm applies the similar framework, fitness function, extraction

scheme and termination as the one we proposed in chapter three. The diversity of hierarchical clustering candidate population is enhanced by using three different hierarchical clustering algorithms to generate it. Two different graph-based searching strategies are used to find more reasonable dataset partition, named neighborhood-based search and cluster linkage-based search. All of these designs enhance the performance of the algorithm. The experiments of artificial datasets and UCI datasets show that the algorithm proposed not only has the same advantages as the one we proposed in chapter three, but also performes much better compared with other algorithms. Moreover, we give out necessary analysis of some important parameters to explain the algorithm further.

This work was supported by the National Natural Science Foundation of China (Grant Nos. 60703107), the National High Technology Research and Development Program (863 Program) of China (Grant No. 2009AA12Z210), the Program for New Century Excellent Talents in University (Grant No. NCET-08-0811), the Program for New Scientific and Technological Star of Shaanxi Province (Grant No. 2010KJXX-03), and the Fundamental Research Funds for the Central Universities (Grant No. K50510020001).

Keyword: Clustering, Evolutionary Computation, Mixed Type Data, Hybrid Schemes, Multi-population, Graph-based Search

目 录

第一章 绪论

1.1 背景介绍

1.1.1 聚类的描述

聚类是一种将数据集划分为我们所期待的不同组（称为簇）的算法。在机器学习，数据挖掘和模式识别等领域，聚类是一项非常重要的研究。无论在科学研究还是在日常生活中，我们常常会遇到需要根据数据属性的不同来划分一组数据的问题，而且这些数据是没有标记的。这就是所谓的聚类问题。因此聚类算法的目标就是找出隐藏在数据集中的数据分布内在特征和规律，以达到对数据集实现合理划分的目的。不同的领域对聚类的描述和定义各不相同，文献[1]中给出了一个典型的描述：

聚类的最终目标是将数据项分配给有限的包含 K 个子集（簇）的系统中。通常（但不总是）子集不是相互交叉的，并且它们的并集等于去除可能异常值的数据集的总和。这个描述可由式 1.1 表示。

$$X = C_1 \cup ... \cup C_k \cup C_{outliers}, C_i \cap C_j = 0, i \neq j \tag{1.1}$$

对聚类问题的研究已经进行了几十年，研究者们设计了许多种类的有效算法。其中一些旨在找到特定的数据分布，如 K 均值算法[2]和 GAPS 算法[3]。K 均值算法试图找出在属性空间中数据集的局部最优球形划分；而 GAPS 算法则试图找出数据集的全局最优点对称分布划分。还有一类著名的聚类算法是层级聚类算法[4]。这类算法依据特定的准则判断如何逐步合并数据项或分离整个数据集，进而达到聚类的目的。我们在 1.2.1 小节中列举了具有代表性的聚类算法。聚类算法一般都以迭代的方式进行，找到各自评判标准下的最优划分。传统聚类方法一般只能找到局部最优解，而作为启发式算法的进化聚类算法可以找到全局或近似全局最优解。

1.1.2 进化计算简介[5]

进化算法（Evolutionary Algorithms，简称 EAs），也称进化计算（Evolutionary Computation，简称 EC），是模拟自然界中生物进化机制而产生的一种计算理论[5]。进化算法是一种鲁棒性较强的计算机算法，使用面较广，尤其适用于求解复杂系统优化问题。进化计算的框架描述如下：

步骤 1：产生初始种群。

步骤 2：评价初始种群的适应度。

步骤 3：个体变异、重组、搜索等操作改变种群。

步骤 4：评价改变后种群的适应度。

步骤 5：个体选择、复制等操作产生下一代种群。

步骤 6：若满足终止条件，则停止进化，输出结果；否则转到步骤 3，继续进行进化操作过程；

目前存在多种多样的进化计算，它们普遍具有如下特点[5]：

（1）算法的操作对象都是由多个个体所组成的一个集合，即种群；

（2）每个个体都有一个对系统环境的适应度，这个适应度是算法对每个个体优劣程度的一种度量；

（3）算法都要进行选择或者复制操作，以便能够将当前种群中具有较高适应度的个体更多地保留到下一代种群中；

（4）算法都通过个体重组，变异等进化操作，以及对个体所加入的一些微小变动，来增进种群的多样性。

（5）算法所模拟的生物进化过程都是一个反复迭代的过程。在种群的这个迭代进化过程中，个体的适应度和种群中所有个体的平均适应度都不断地得到改进，最终可得到一个或几个具有较高适应度的个体，它们对应于问题的最优解或近似最优解；

（6）算法所模拟的进化过程均受随机因素的影响，所以它们不易于陷入局部最优点，并都能以较大的概率找到全局最优点；

（7）算法都具有一种天然的并行结构，均适合于在并行机或局域网环境中进行大规模复杂问题的求解。

以上特点决定了进化算法是一种全局优化算法，具有良好的自适应性和鲁棒性，因此可以应用于许多复杂的问题。聚类问题就是其中的一种。

1.2 聚类算法的研究现状和发展前景

1.2.1 具有代表性的聚类算法

随着聚类在科技和人们生活领域重要性的提高，研究者们在几十年间设计了许多种类的聚类算法。我们可把当前的聚类算法总结为如下几种[1],[6]（存在一些算法可能同时属于下列分类中的几种）：

1. 层级聚类算法

这类算法主要包括凝聚算法和划分算法两种，前者依据一定的规则通过不断合并簇达到聚类的效果，而后者则通过不断分裂大的簇达到聚类的效果。在 3.1.2

小节有对这一算法的较详细的介绍。代表算法有 single-linkage[7]，complete-linkage[8]，CURE[9]，以及 BIRCH[10]等等。

2. 基于均方误差的方法

这类算法通过把数据集划分成若干子集来实现聚类的目的。通常这类聚类算法使用迭代的方式进行，根据指定的度量函数不断改变数据项的分簇从而不断改进聚类效果。这一类算法中具有代表性的例子有 K 均值算法[2]以及 genetic K-means 算法[11]等等。

3. 基于密度的方法

这类算法中距离某个数据项最近的那些数据项（即邻域）起着重要的作用。这种算法中簇被定义为相互联通的密集分支，这一分支的形状是由数据项分布决定的，哪里数据项分布密集，这一分支就延展到哪里。因此基于密度的聚类算法可以找到任意形状的簇。这种算法的代表例子有 DBSCAN[12]，DENCLUE[13]以及 OPTICS[14]等等。

4. 基于网格的方法

这种方法从潜在的属性空间中获得数据集的拓扑结构。为了减少运算，这些算法使用了超矩形片段（类似于有限网格）的概念。数据空间被划分成了具有网格属性的片段。这样，数据集的划分便可以从数据项在空间划分形成的矩形片段中的情况获得。这种算法的代表性例子包括 CLIQUE[15]，STING[16]以及 FC[17]等等。

5. 基于图论的划分法

由于图论的概念和特点非常适用于描述聚类问题，很多算法都是基于图而设计的。通常图中的节点对应于数据项，而图中的边对应于它所连接的两个数据项（节点）之间的相似程度。这样，根据构造的图所表述的信息，我们就可以对数据集进行划分了。这种算法的代表性例子有一些层级聚类算法，CLICK[18]以及 CAST[19]等等。

6. 模糊聚类算法

在模糊聚类算法中，一个数据项可以属于不止一簇。事实上，每个数据项可以依据一定的隶属度属于任何一簇。这种设定在很多场合是非常有用的，比如簇与簇之间的边界很模糊或有交叠的情况。此外，这种设定也有助于找到更复杂和隐蔽的数据关系。这种算法的代表性例子有 FCM[20]和 FCS[21]等等。

7. 基于组合搜索的方法

聚类可以被看做一种优化问题，给定一个数据集，聚类算法实质上旨在组织数据项以使它们依据一定的评判标准被划分到不同的簇中。这些问题通常具有 NP 难的复杂度且搜索空间非常巨大。很多启发式搜索技术可以找出优化问题的全局最优解或近似全局最优解，因而可以用在聚类问题上。通常人们会采用一些传统的方法作为局部搜索方法以找到聚类优化问题的局部最优解，同时配合全局搜索

方法如进化计算来进一步找到全局最优解。这类算法的代表性例子有 GKM[11]，GGA[22]以及[23]等等。本文所介绍的三个算法就是属于这类算法，同时兼具一些其它种类算法的优点。

此外，还有基于神经网络的聚类算法，基于核的聚类算法，基于约束的聚类算法以及结合机器学习的半监督聚类算法等等。有时人们会遇到一些特殊的数据，如高维数据，大规模数据，时间或空间序列数据等等，研究者们也为这些问题设计了相应的聚类算法。

1.2.2 进化聚类算法[24]

作为一种有效的全局优化算法，进化计算已经成功的应用在如数据挖掘和机器学习等多种领域[24]。聚类问题可以被视为一种特殊的 NP 难划分问题[25]。进化计算广泛地被视为一种针对 NP 难问题的有效自适应算法，它可以在合理的时间内找到此类问题的近似最优解[26]。研究者们已经设计了许多有效的进化聚类算法，并获得了满意的解[27-29]。在传统进化聚类算法中，聚类问题被当做一种优化问题。它们通过优化衡量聚类质量的目标函数（适应度函数）来得到满意的解，其中目标函数指引了进化的走向[30]。根据适应度函数的个数，进化聚类算法可以被分为单目标算法和多目标算法。目前大多数的文献介绍的是单目标进化聚类算法，同时也有一些文献[30-31]展示了多目标进化聚类算法的优势和效果。

与其它种类的聚类算法类似，很多进化聚类算法中分簇数是实现确定好的。但在近些年，也有很多研究者设计了自动确定分簇数的进化聚类算法，特别是在实际应用领域[32-33]。研究者设计了专门针对聚类问题的进化路径、编码方式、交叉算子、变异算子以及适应度函数。在进化计算的支撑下，这些聚类算法可以解决多种的数据聚类问题，我们可以找到很多相关的工作[34-38]。研究者不仅仅关注设计新的进化策略，也关注设计新的评判标准（包括适应度函数）[39]或者距离度量[33][40]。

近几年，进化聚类算法已经广泛地应用于图像处理[41-43]，生物信息学[44]，以及软件系统[45]等邻域。上述研究为我们展示了进化聚类算法在理论分析和实际应用中的巨大潜力。

1.3 论文内容与结构安排

本文主要介绍了三个进化聚类算法，其中第一个是混合属性进化聚类算法，第二、三个是混合策略进化聚类算法。本章作为绪论主要介绍了聚类问题，进化计算的基础知识以及进化聚类的相关背景。论文在第二章详细介绍了一种针对混合属性数据集的进化聚类算法；第三章主要介绍基于多种群策略的进化聚类算法；

第四章介绍了基于多种群和基于图搜索的进化聚类算法。本文在第二、三、四章首先介绍了相关的背景算法，然后给出本文所提出的算法的基本框架。接下来详细的讨论了算法的理论部分和实施细节。这三章都包含了实验部分，文中首先给出实验设定：包括算法参数的设置，测试数据集的详细信息；然后列出了实验结果以及对它的详细分析。最后，在每章的结尾本文给出对这一章的总结。第五章对全文作出了总结。

第二章 混合属性进化聚类算法

本章介绍了一种针对混合属性数据集的进化聚类算法(Evolutionary Clustering Algorithms for mixed type data)，简称ECM算法。此算法用进化框架操作K-prototype算法（简称KP算法），从而得到比KP更优的聚类效果。因此ECM可以视为一种进化K原型算法。

本章的安排如下：2.1节介绍了混合属性数据集；2.2节介绍了本章算法；2.3节展示了实验结果和分析；2.4对本章做了全面的总结。

2.1 混合属性数据集及K原型算法

常见的数据集按照其属性不同可划分为三种：数值型数据集，种类型数据集和混合型数据集。最常见的是数值型(Numerical)数据集，其中的数据都以数值描述。如UCI数据集iris. 它有四维属性，分别为花萼的长度、花萼的宽度、花瓣的长度及花瓣的宽度。这四个属性均由数值表示，代表了各数据项中记录的宽度与长度各是多少厘米。另一种常见数据集为种类型(categorical)数据集。这类数据集的属性由一些单词，布尔数据或者编号表示，如UCI中soybean disease数据集。在这个数据集中，属性包括日期，植物的性状等等。需要注意的是，在数值型数据中，在同一维属性里，两个数据点的这一维属性的值相差越远，代表这两个数据点在这一维的差异越大。然而在种类型数据中，这一规则并不一定成立。以描述数据点的国别属性为例，如果国家A，B，C，D分别用1，2，3，4表示，1在数值上距离4最远，但是国家属性A和D的区别却不一定比其它组合大。

如果一个数据集既包含数值属性，又包含种类属性，则它可被称为混合属性数据集。混合属性数据集在生活中是普遍存在的，如一个描述人口的数据集里既可能出现描述年龄，体重的数值属性，也可能出现描述民族，性别的种类属性。对于聚类问题，需要将数据集的各种属性都作为输入进行计算。因此，一个重要的问题是如何用数值表示种类属性和不同种类属性间的差异，以及如何将数据集的数值属性和种类属性统一起来描述整个数据集。

本章首先介绍一种有效的针对混合属性数据集的聚类算法——K 原型。这一算法由Huang于1997年提出[46]。K原型算法由K均值[2]算法拓展而成，是一种基于划分的聚类算法。K原型算法可视为K均值算法和K模式算法[47]的加权组合。它的主要流程如算法2.1所示：

算法 2.1 K 原型算法

输入： 聚类类别数，簇数。

输出： 聚类结果。

步骤 1： 为每类初始一个原型。

步骤 2： 对于每个数据项，计算它与各个原型之间的距离，把这个数据项划分到与其距离最小的原型所代表的簇中，并更新这一簇的原型。

步骤 3： 如果与当前数据项距离最近的原型并不是这一数据项所属簇的原型，则把这一数据项移到离它最近的原型所属的簇中，并更新这一数据项移动前和移动后所属的簇的原型。

步骤 4： 重复**步骤 3** 直到没有数据项改变所属的簇，**终止。**

K 原型算法中的原型相当于 K 均值算法中的中心，它是所属簇的代表。由于混合属性数据具有数值型和种类型两种属性，因此原型也包括这两种属性。K 原型算法中数据点与原形之间的距离计算公式如式 2.1 所示。

$$d(X,Q)=\sum_{i=1}^{N}\left(X^i-Q^i\right)^2+r\sum_{j=1}^{C}\delta\left(X^j,Q^j\right) \tag{2.1}$$

式中 N 是数据集数值属性的维数，C 是数据集种类属性的维数；X 是当前数据项，Q 是 X 所属簇的原型；X^i 和 Q^i 分别表示 X 和 Q 的第 i 维数值型属性；X^j 和 Q^j 分别表示 X 和 Q 的第 j 维种类型属性；r 是平衡两种属性对聚类结果影响的参数. 由式 2.1 可见，K 原型算法的前半段采用了与 K 均值算法相同的方法处理数值型属性。其中函数 δ 定义如式 2.2 所示。

$$\delta\left(X^j,Q^j\right)=\begin{cases}0 & \text{if } X^j=Q^j\\ 1 & \text{if } X^j\neq Q^j\end{cases} \tag{2.2}$$

式 2.2 表示 K 原型算法对种类型属性的处理采取“匹配判断”的方法，如果相比较的属性相同，则输出 0，否则，输出 1。

其中簇原型的设定如下：1.对于数值型属性，原型数值型部分为当前簇各数据项数值型属性的均值；2.对于种类型属性，原型种类型部分为当前簇各数据项种类型属性中出现频率最高的属性代表符号。由于 K 原型算法采用了 K 均值算法的框架，因此它不可避免地像 K 均值算法一样对初始值敏感且容易陷入局部最优。因此，本章用进化计算的方式操作 K 原型算法，以克服它原有的缺陷。

2.2 混合属性进化聚类算法

本节主要介绍一种新的针对混合属性数据集的进化聚类算法，即 ECM。首先

本章将介绍ECM的算法流程，然后分两小节讨论ECM的编码方式和适应度函数，以及进化算子。

2.2.1　混合属性进化聚类算法的框架

此算法的框架如算法2.2所示。

算法2.2　混合属性进化聚类算法

输入：　数据集（规模为N），簇数K。

输出：　聚类结果。

步骤1：　**初始化。**这一步设定了终止准则，参数；并以两种随机方法产生一个具有多样性的初始种群。

步骤2：　**交叉。**在个体上，模拟二进制交叉作用于数值型属性部分，单点交叉作用于种类型属性部分。

步骤3：　**变异。**在个体上，多项式变异作用于数值型属性部分，均匀变异作用于种类型属性部分。

步骤4：　**K原型搜索。**将改进的K原型算法用于个体上，以搜索更优的簇原型，并得到合理的数据划分。

步骤5：　**评估和选择。**计算经过K原型搜索的个体的适应度。在计算后使用联赛选择和精英保留策略产生下一代种群。

步骤6：　**终止测试。**如果终止条件满足，找到适应度值最大的个体并输出其对应的结果，**终止**；否则转到**步骤2**。

下面的几小节将详细讨论上述算法的各个细节。

2.2.2　编码与适应度函数

本算法使用了基于原型的表示方法，使用实值编码方法将个体编码为簇的原型。给定分簇数K，数据集数值型属性维数N和种类型属性维数C，本章算法将个体存储在一个大小为$K\times(N+C)$的矩阵中。其中第i行代表第i簇的原型，前N列存储了各原型的数值型属性部分，余下的C列存储了各原型的种类型属性部分。由于簇的原型一般不可能分布在距离数据集很远的地方，为防止在进化操作中产生距离数据集分布过远的无意义搜索，本章事先限定初始种群分布在数据集范围内，或距离较近的区域内。对于种类型属性部分，本章用不同的整数表示不同的属性，因此，在进化操作中，个体内存储的值也仅限于这些设定好的整数，也不存在介于两个相邻整数之间的中间量。在进化中，算法不会产生新的种类型属性，

只以进化的方式寻找不同维种类型属性在同一个体内的组合。本章的编码方式可由表 2.1 表示：

表 2.1 ECM 编码方式

	n1	n2	n3	c1	c2	c3	c4
P1	1.23	12.45	-1.25	1	3	4	7
P2	5.42	5.23	-6.32	2	4	6	3
P3	3.24	0.56	-4.23	1	2	10	8

图 2.1 展示了一个三簇问题的编码方式，设定数据集具有三维数值型属性和四维种类型属性。图中 P1 到 P3 分别代表三簇对应的原型；n1 到 n3 分别对应原型的三维数值型属性；C1 到 C4 分别对应原型的四维种类型属性。

本文使用基于划分的适应度函数。在适应度函数计算前，本章首先执行了 K 原型搜索作为一种局部搜索从而获得对数据集的合理划分。然后适应度函数基于这个划分计算出来。依据 K 原型算法的代价函数，本章定义 ECM 的适应度函数如式 2.3 所示。

$$fitness = \frac{1}{\sum_{i=1}^{K}\sum_{j=1}^{CN_i} dis\left(X_{i_j}, P\right)} \tag{2.3}$$

$dis(X_{i_j}, P_i)$ 为式 2.1 所表达的形式。为方便下文的说明，本章将其用进化中所使用的符号重写如式 2.4 所示。

$$dis(X_{i_j}, P_i) = \sum_{g=1}^{N}\left(X_{i_j}^{g} - P_i^{g}\right)^2 + r\sum_{f=1}^{C}\delta\left(X_{i_j}^{f}, P_i^{f}\right) \tag{2.4}$$

其中 N 是数据集数值型属性的维数，C 是数据集种类型属性的维数。$X_{i_j}^{g}$ 是类 i 中数据项 j 的第 g 维数值型属性，P_i^{g} 是类 i 的原型的第 g 维数值型属性，$X_{i_j}^{f}$ 是类 i 中数据项 j 的第 f 维种类型属性，P_i^{f} 是类 i 的原型的第 f 维种类型型属性，r 是平衡数值型属性和种类型属性对聚类结果影响的重要权值，它越大，则种类型属性在聚类中越重要。

2.2.3 进化操作

本小节主要介绍 ECM 的初始化方法，两个重要进化算子，即模拟二进制交叉算子、多项式变异算子，以及带精英保留的联赛选择策略。

（1）. 初始化

为保证种群的多样性，这里使用两种随机方法来产生初始种群。在第一种方法在数据集中随机选择 K 个数据项作为簇的 K 个初始原型，将这 K 个数据项录入 2.2.2 小节中的矩阵里，便可得到一个个体。在第二种方法中，对于数值型属性，

在每一维的范围内随机产生一个数作为一个原型的当前维数值型属性值；对于种类型属性，本章在每一维的分布中随机选择一个属性作为一个原型的当前种类型属性。这里对第二种初始方法给出一个例子如下。

假设数据集具有两维数值型属性和两维种类型属性，它们的分布范围分别是$[2.23,5.63]$、$[6.56,5.13]$，和$\{1,2,3,4,5,6\}$、$\{1,2,3,4\}$。在前两个范围内随机产生两个数，设为3.21和6.23；在后两个范围内随机选取两个值设为2和4，则生成的初始原型为$\{3.21,6.23,2,4\}$。重复这一过程K次，便可得到K个初始原型，进而生成一个随机初始个体。

在实际情况中，有些数据集的最优原型是其中的某些数据项，而另一些数据集的最优原型则分布在数据项以外的地方。因此上述的两种初始方法可以产生一个具有多样性的种群，尽可能的涵盖两种情况，进而加速进化的收敛并使得ECM的鲁棒性更好。参数IP控制了两种方法产生的个体在初始种群中各占的比例。它的值越大，则有越多的初始个体产生于数据集本身。

一共有八个参数需要在这一步确定，即分簇数、公式2.4中的权值r、种群规模、在初始种群中随机选取数据集中的数据项生成初始个体的比例（IP）、交叉概率、变异概率、模拟二进制交叉中的参数η、多项式变异中的参数n。停机准则设置为进化达到一定的代数。

（2）. 交叉算子

本文采用模拟二进制交叉[48]处理个体的数值型属性部分，用单点交叉处理个体的种类型属性部分。

模拟二进制交叉是一种针对实值编码染色体的有效交叉算子，它模拟了单点交叉算子作用于二进制串的工作机制。首先，产生一个在0到1之间的随机数u。然后依据式2.5计算纵坐标β，这样在概率曲线下从0到β的面积等于随机数u。

$$\beta=\begin{cases}(2u)^{\frac{1}{\eta+1}} & \text{if } u_i \le 0.5\\ \left[\dfrac{1}{2(1-u)}\right]^{\frac{1}{\eta+1}} & \text{Otherwise}\end{cases} \tag{2.5}$$

产生的子代值的概率分布如式2.6所示。

$$P(\beta)=\begin{cases}0.5(\eta+1)\beta^{\eta} & \text{if } \beta \le 1\\ 0.5(\eta+1)\dfrac{1}{\beta^{\eta+2}} & \text{otherwise}\end{cases} \tag{2.6}$$

其中参数η依概率控制产生的子代值到父代值之间的距离。它的值越大，则子代值将以较大的概率靠近其父代值，反之则以较大的概率远离之。

子代值依式2.7和式2.8产生。

$$x^{(1,t+1)}=0.5\left[(1+\beta)x^{(1,t)}+(1-\beta)x^{(2,t)}\right] \tag{2.7}$$

$$x^{(2,t+1)}=0.5\left[(1-\beta)x^{(1,t)}+(1+\beta)x^{(2,t)}\right] \tag{2.8}$$

其中$x^{(1,t)}$和$x^{(2,t)}$是第t代的两个个体，$x^{(2,t+1)}$和$x^{(1,t+1)}$是由前两个父个体产生的第$t+1$代的两个子个体。

本章使用单点交叉处理个体的种类型属性部分，交叉点随机选取。由于ECM的编码是矩阵，每一行代表一个原型，因此本章在个体的每一行都应用上述两种交叉。一个个体的第i行将与另一个个体的第i行进行交叉。如表2.2和表2.3所示，假定数据集具有两维数值型属性，三维种类型属性。两个父代个体如表2.2所示，如果$\eta=2.0\,u=0.3$，一到三行种类型数据的交叉点分别为2，2，和3。则本章的交叉算法将产生两个如表2.3所示的子代个体。

表2.2 两个父代个体

Parental Individual 1				
4.3200	1.2500	1	4	7
2.4600	-2.3400	5	8	6
4.6500	1.2900	7	2	3

Parental Individual 2				
3.2600	4.0300	1	5	2
-4.2300	5.3600	2	3	3
1.5600	4.3600	3	7	3

表2.3 两个子代个体

Child 1				
4.2397	1.4607	1	5	2
1.9529	-1.7563	5	3	3
4.4158	1.5227	7	2	3

Child 2				
3.3403	3.8193	1	4	7
-3.7229	4.7763	2	8	6
1.7942	4.1273	3	7	3

个体也可被编码成一个长度为$K\times(N+C)$的向量，这样每段长度为$N+C$的片段代表一个原型，则本章的交叉策略可视为一种多点交叉。

（3）. 变异算子

ECM对个体的数值型属性部分采用多项式变异[49]。类似于模拟二进制交叉，多项式变异生成的个体值由一个参数δ依一定的概率分布产生。δ的计算如式2.9所示。

$$\delta=\begin{cases}(2u)^{\frac{1}{n+1}}-1 & \text{if } u<0.5\\ 1-\left[2(1-u)\right]^{\frac{1}{n+1}} & \text{if } u\geq 0.5\end{cases} \tag{2.9}$$

这里u是一个在0到1之间的随机数；n被设定为一个参数，它的值越大，则

子代值将在更靠近父代值的区域产生。这一分布概率如式 2.10 所示。

$$P(\delta)=0.5(n+1)(1-|\delta|)^n \tag{2.10}$$

父代个体经变异后得到的子代个体由式 2.11 计算得出。

$$x^{t+1}=x^t+\delta(\max(x)-\min(x)) \tag{2.11}$$

这里 x^t 是第 t 代作为父代的个体，x^{t+1} 是第 $t+1$ 代变异后的子代个体。$\max(x)$ 和 $\min(x)$ 分别是属性的最大值和最小值向量.

本章对个体的种类型属性部分采用均匀变异，被选中进行变异操作的基因将被置换为数据集同一维中的另一属性值。置换值为排除被置换值的集合中以相同的概率 $1/(C-1)$ 随机选取的任一值。与交叉类似，上述两种变异均作用于选中个体矩阵的每一行上。

（4）. 选择算子

本章采用带有精英保留的联赛选择策略，每代中适应度值最大的一个个体将不参加选择操作而直接进入下一代。为了避免局部最优，本章将模拟二进制交叉的参数 η 和多项式变异的参数 u 设置的相对较小，通常为 2.0。因此，某些子代个体会在距离父代个体较远的地方产生，这有可能会产生较差的个体。然而本章关注的是种群内最好的个体，它会被精英保留策略保留在种群中。因此，算法允许种群中存在上述可能产生的较差个体，因为它们是避免局部最优和以更大的几率寻找最优解的必要牺牲。

2.3 实验结果与分析

2.3.1 实验设定

为全面的展示ECM的性能，本章应用人工数据集和实际生活数据集进行测试。其中人工数据集由数值型人工数据集 sizes5 的一部分和 UCI 机器学习数据库[50]中的数据集 soybean(small)的一部分合成。实际生活数据集为 UCI 数据集 zoo。

如本章开头所述，ECM 是由 K 均值和 K 模式算法加权组合而成，因此本章也为进化 K 均值(EKM1)和进化 K 模式(EKM2)算法设计了实验，以帮助说明 ECM 算法。这两个算法采用与 ECM 相同的进化策略。完整的 sizes5 数据集用于测试 K 均值和 EKM1 算法。它是一个 2 维数据集，包含 1000 个数据项，标准分类为 4 类，如图 2.1 所示。种类型属性数据集 Soybean(small)用于测试 K 模式和 EKM2 算法，它包含 47 个数据项。它的原始数据集包含 35 维属性，但 17 维都是相同的，这意味着这些维上数据项之间没有区分。因此本章将这些维的属性去除，留下其余的 21 维。此数据集的标准分类为 4 类，表示四种不同大豆疾病的集合。K 均值和 K 模式的结果也分别与 EKM1 和 EKM2 的结果做了比较。

基于 EKM1 和 EKM2，本章测试了 ECM（即前两个算法的加权组合）。与前面的实验类似，本章也比较了 K 原型算法和 ECM 的结果。测试 ECM 的人工数据集含有 100 个数据项，它的数值型属性为 sizes5 数据集中每类数据的前 25 个数据项的属性，它的种类型属性为 soybean(small)数据集中每类数据依次抽取 25 个数据项的属性(如数目不足 25，则从头在这一类开始重复抽取)。Sizes5 和 soybean(small)数据集具有合适的复杂性和分布，因此分别被广泛地应用于测试数值型和种类型聚类算法。本章的数据集合成方法使得这个人工数据集具有部分 sizes5 和 soybean(small)的特点，并使之具有混合属性，其标准分类为 4 类。此外，本章使用了 zoo 数据集，它描述的是动物分类。数据集的第一维是动物的名字，最后一维是标准分类号。由于不同的动物具有不同的名字，标准分类号应该是未知信息。所以本章把这两维属性放弃不用。本章将第二维属性，即腿的数目作为数值型属性，其它属性作为布尔型的种类型属性。则本章使用的 zoo 数据集具有 1 维数值型属性和 15 维种类型属性。它包含 101 个数据项，标准分类为 7 类。Jie Li, Xinbo Gao, Licheng Jiao 提出过算法 FWKMo[51]，文中此算法的测试数据集也是 zoo，且数据集处理方法与本文相同，因此这一算法也作为 zoo 数据集的对比算法之一，试验结果来自源文献。

2.3.2 实验结果与分析

实验参数设定如表 2.4 所示。图 2.2，2.3，2.4 分别是 EKM1 在 sizes5，EKM2 在 soybean(small)以及 ECM 在人工合成数据集上的代表性收敛曲线。曲线分别记录了 100 代中每代的最好，最差和平均适应度值。

表 2.4 实验参数设定

参数	值
进化代数	100
种群规模	40
交叉概率	0.8
变异概率	0.01
初始种群里随机从数据集中选取数据项生成初始个体的比例(IP)	0.1
模拟二进制交叉参数η（存在于 ECM 和 EKM1 中）	2.0
多项式变异参数n（存在于 ECM 和 EKM1 中）	2.0
平衡数值型和种类型属性的参数r(只在 ECM 中)	0.5

从图 2.2 到 2.4 可以看出所有的算法均在 85 代前收敛。图 2.2 中的最大值曲线

光滑上升，图 2.3 中的曲线程阶梯状上升。图 2.4 中的曲线比图 2.3 中的光滑但比图 2.2 中的曲线更具阶梯状。这表明了 ECM 具有 EKM1 和 EKM2 的部分特点。从图 2.2 中可以看到算法有时会产生"坏"个体，它们表现为向下的尖峰。由于本章采取了精英保留策略，为了避免算法陷入局部最优，交叉概率和变异概率设定的相对较大；且模拟二进制交叉和多项式变异中的参数设置的较小以产生距离父代个体较远的子代个体。图 2.2 里平均值和最差值曲线中向下的尖峰由此产生。但本章只关注种群中的最好解，因此这不会对算法的聚类结果造成不良影响。

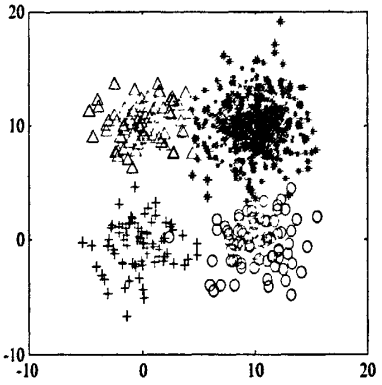


图 2.1　sizes5 数据集及其标准分布

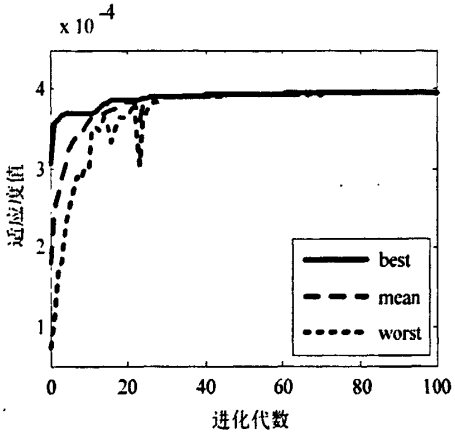


图 2.2　EKM1 进化收敛曲线

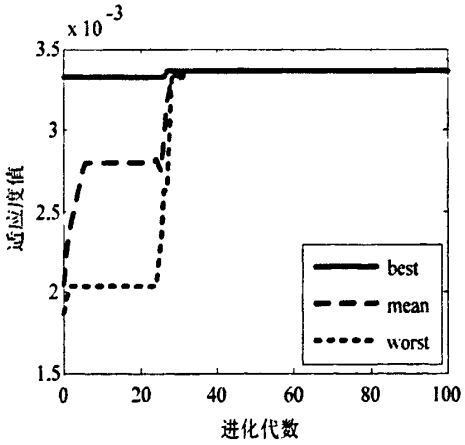


图 2.3　EKM2 进化收敛曲线

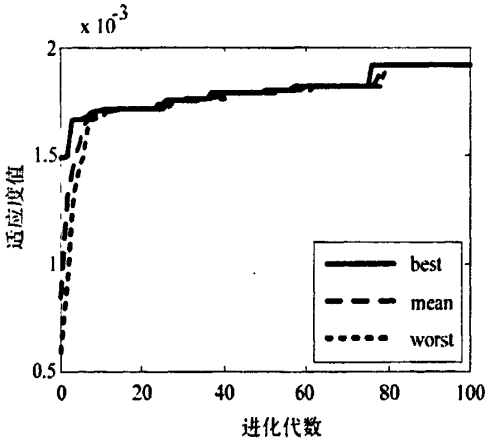


图 2.4　EMC 进化收敛曲线

表 2.5 展示了包含上述三个实验的所有实验结果（正确率）的统计值。算法实验参数如表 2.4 所示，其中除 zoo 数据集的分类数为 7 类之外，其它数据集均为 4 类。Zoo 数据集实验中的 r 为 15.0，IP 为 0.5。在实验中我们发现可依据数值型属性各维变化范围的和与分类型属性各维变化范围的和做为参照来设定参数 r 的值，原则是使这两种属性的变化范围相近。本章比较了 K 原型，以及其组成算法 K 均值、K 模式和以它们作为搜索方式的进化聚类算法的性能。因为这些算法均具有

随机性，所以本章把每个实验都独立运行 30 次以获得统计结果。表 2.5 中的黑体字表示进化聚类算法的实验结果。

从表2.2可以看出，对于sizes5数据集，K均值算法可以以较小概率产生97.60%的优质结果。但由于此算法易陷入局部最优，它得到平均正确率只有 73.13%，最差值为 50.30%。相比之下，EKM1 得到了很好的稳定的结果，最差的一个也高达 97.40%。与 K 均值算法类似，在 soybean(small)数据集上 K 模式算法偶尔可以得到 100%的优质结果，但是它很容易陷入局部最优，其平均正确率只有 74.68%，最差值为 48.94%。相比之下，EKM2 得到了更好且稳定的结果，最差值为 91.49%，平均值为 99.09%。

表 2.5 原始聚类算法和进化聚类算法正确率比较

数据集	聚类算法	平均值	最好值	最差值
Sizes5	**EKM1**	**97.56%**	**97.60%**	**97.40%**
	K 均值	73.13%	97.60%	50.30%
Soybean (small)	**EKM2**	**99.09%**	**100%**	**91.49%**
	K 模式	74.68%	100%	48.94%
人工合成数据集	**ECM**	**98.00%**	**98.00%**	**98.00%**
	K 原型	86.93%	98.00%	63.00%
Zoo	**ECM**	**75.28%**	**90.10%**	**63.37%**
	K 原型	67.82%	83.17%	50.50%
	FWKPo	87.13%	87.13%	87.13%

K 原型为 K 均值算法和 K 模式算法的组合算法，以它为搜索方法的进化聚类算法大大的提升了聚类质量。对于人工合成数据集，ECM 在所有 30 次独立运行后都得到了 98.00%的正确率。而 K 原型算法虽然偶尔也可以得到这样的结果，但它频繁的产生诸如 63%的低正确率，因而其平均正确率只有 86.93%。这个结果和 ECM 的 98.00%的平均正确率相差甚远。对于数据集 zoo，ECM 的最佳正确率比 K 原型高 7%，平均正确率和最差正确率也均比 K 原型高 10%左右。FWKPo 的平均实验正确率高于 ECM 的平均正确率，但是不能得到 ECM 获得的较高的正确率，如 90.10%。

图 2.5 展示了 K 原型和 ECM 算法在人工合成数据集上的 30 次独立实验正确率。图 2.6 展示了 K 原型和 ECM 算法在 zoo 数据集上的 30 次独立实验正确率。在两幅图中，ECM 的曲线均比对应的 K 原型的曲线处于更高的位置，这表明了 ECM 可以获得更好的聚类正确率。在图 2.5 中，ECM 对应的曲线完全平坦，这表

明了 ECM 算法在人工数据集上性能非常稳定。对于 zoo 数据集，ECM 对应的曲线比 K 原型对应的曲线更平滑一些。这些都表明了 ECM 算法比 K 原型算法具有更好的鲁棒性。

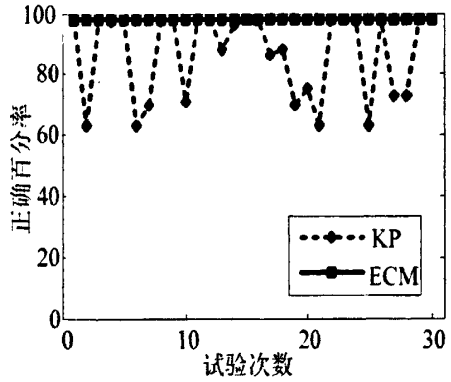


图 2.5 人工合成数据集实验正确率比较

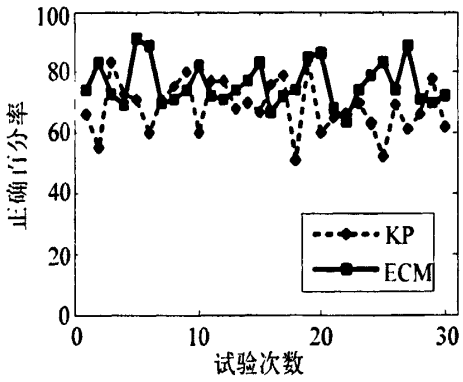


图 2.6 zoo 数据集实验正确率比较

表 2.6 ECM 和 K 原型算法在 zoo 数据集上的最好结果

标准分类	C1	C2	C3	C4	C5	C6	C7
(数据项数目)	C1	C2	C3	C4	C5	C6	C7
Mammals(41)	**41**	**0**	**0**	**0**	**0**	**0**	**0**
	37	0	4	0	0	0	0
Aves (20)	**0**	**19**	**0**	**0**	**1**	**0**	**0**
	0	20	0	0	0	0	0
Crawlers(5)	**0**	**0**	**1**	**1**	**3**	**0**	**0**
	0	0	0	1	4	0	0
Fishes(13)	**0**	**0**	**0**	**13**	**0**	**0**	**0**
	0	0	0	13	0	0	0
Amphibians (4)	**0**	**0**	**0**	**0**	**4**	**0**	**0**
	0	0	0	0	4	0	0
Insectology(8)	**0**	**0**	**2**	**0**	**0**	**6**	**0**
	0	0	0	0	0	0	8
Molluscs(10)	**0**	**1**	**3**	**0**	**0**	**0**	**7**
	0	0	0	0	0	0	10

表 2.6 展示了表 2.5 中针对 zoo 数据集 ECM 和 K 原型所得到的最好结果的详细信息。左数第一行是 zoo 数据集的标准分类。黑体字 **C1** 到 **C7** 是 ECM 的结果，

非黑体字 C1 到 C7 是 K 原型的结果。从表中可以看出 ECM 可以将 mammal 类精确的分出，然而 K 原型算法将其中的 4 个数据项错分在 crawler 类中；对于 aves 类，ECM 错分了 1 个数据项，但是 K 原型将其全部正确分出；K 原型算法丢失了整个 crawler 类，但 ECM 只忽略了其中的 1 个数据项；K 原型算法丢失了整个 insectology 类，但 ECM 只忽略了其中的 6 个数据项；K 原型算法和 ECM 算法都得到了完整的 amphibians 和 fishes 类；K 原型算法得到了完整的 molluscs 类，但 ECM 丢失了其中的 3 个数据项。Zoo 数据集应该被分成 7 类，ECM 的结果含有所有的 7 类，但 K 原型算法丢失了整个 C6 类。总体来说，ECM 明显比 K 原型算法更优秀。

由试验结果可见，本章提出的算法与原始 K 原型算法相比有了很大的提高，并可得到比 FWKPo 更优的结果。这种性能改善的代价是进化操作增加了算法的复杂度。由于我们并未采用复杂度很高的进化操作，并且嵌入进化框架的 K 原型算法是改进后的批量操作 K 原型算法。因此算法的运行时间是可以接受的。

2.4 小结

本章提出了一种针对混合属性数据集的无监督聚类算法，简称 ECM。首先，本章简要介绍了混合属性数据集和 K 原型算法，它们是理解 ECM 的基础。接下来详细介绍了算法的主要流程和各个进化操作及算子。本章设计了实验以验证 K 原型及其组成部分（K 均值和 K 模式）相对于进化聚类算法的表现。所有这些进化聚类算法都一一和原始算法做了比较，结果显示本章设计的进化框架大大的提高了原来算法的性能。

本算法使用了 K 原型的方法，通过引入一个权值参数来平衡数值型和种类型属性对聚类效果的影响。在未来的研究中，如何自动调节这个权值将是个比较有意义的课题。此外数值型属性和种类型属性可以被看做两个目标，因此多目标算法或许可以解决上述问题。

ECM 对数值型属性采用了欧氏距离平方的度量方法；对种类型属性采用了汉明距离的度量方法。如何构造或选择更好的度量方法也将是一个有意义的研究课题。

第三章　基于多种群的混合进化聚类算法

本章提出了一种基于多种群的混合进化聚类算法（Hybrid Evolutionary Clustering Algorithm with Multi-population），简称 HECM。此算法采用了多种群的思想，由不同的候选算法产生优质个体，进而得到数据集的合理分布。一种新的抽取算子被应用在从每个候选种群中选取的最好个体上，从而获得这些最好个体内的有效信息，并基于这些信息重新产生新的候选种群。整个数据集被建模成一幅无向图，本章采用了受图像处理算法启发得到的适应度函数，并采取了约简策略以加速算法。本章采用四个人工数据集和五个 UCI 数据集测试了 HECM。实验证明 HECM 能够获得比传统进化 K 均值算法和嵌入其中的候选算法更好的聚类正确率。此外，本章还对 HECM 中的数据集属性规范化策略进行了实验对比。

本章的安排如下：3.1 节详细地介绍了 HECM 算法；3.2 节展示了实验测试和结果分析；3.3 节对本章做了总结。

3.1 基于多种群的混合进化聚类算法

本节将详细介绍 HECM 的理论和设计部分，包括算法框架、候选算法介绍、编码与适应度函数、进化算子以及终止策略。

3.1.1　算法框架

首先，整个数据集被建模成一个无向图 $G(V,E)$。图中的节点是数据集中的数据项；连接节点与节点的边的权值 $w(i,j)$ 为数据项 i 到 j 之间的欧氏距离。适应度函数基于此图定义。HECM 算法采用一种与传统进化聚类算法不同的进化框架，采取多候选种群策略和一种崭新的抽取策略，以结合不同候选聚类算法的结果得到数据的划分。每一个候选种群由一种候选算法产生。抽取操作应用在从每个候选种群中选出的最优个体上，以有效地将它们的信息传递给下一代个体。变异采用随机的方法。当进化收敛后，可能会出现两种情况。第一种情况是所有的数据项都由进化过程做出最终划分。在这种情况下，进化的结果作为最终的聚类结果。另一种情况是进化并未对所有的数据项进行最终判断，此时本章设计了全局 K 均值算法[52]启发的改进 K 均值算法来获得最终聚类结果。在实际应用中，此改进 K 均值算法比全局 K 均值算法要高效许多。为了方便地操作数据集，本章采用实值编码技术以沟通数据集和进化操作。此外，本章在初始化和抽取操作之后采取了数据集约简措施以提高算法的运行速度。这一算法框架可以嵌入多种不同的候选聚类算法，候选种群数目应 ≥ 2。本章以 K 均值和 complete-linkage 算法为候选算

法产生两个候选种群。HECM 算法的框架如下:

算法 3.1 基于多种群的混合进化聚类算法

输入: 数据集，簇数。

输出: 聚类结果

步骤 1: **预处理。**原始数据集被聚成团以输入进化过程，此外下面各步骤所需的一些信息也在这一步产生和存储，以避免后续的重复计算。

步骤 2: **初始化。**每个候选种群 P_i 由算法 i 产生，分别用 K 均值算法和 complete-linkage 算法初始化两个候选种群，称为 KM 种群和 HC 种群。

步骤 3: **变异。**这里采取随机变异策略。

步骤 4: **抽取。**根据 KWNC 适应度值，从每个候选种群中选出最好的个体，然后选出这两个最好个体所判断的数据集划分的重叠部分。重叠部分中的数据项将由它们的均值代替。对于非重叠部分，如果需要，它里面的数据项将被重新聚团至一定数目。

步骤 5: 如果没有非重叠数据项存在，则输出当前的数据集划分作为最终的聚类结果；否则，转到**步骤 6**。

步骤 6: 如果重叠部分中所包含的数据项的数目连续 G 代都没有上升，则转到**步骤 7**；否则用**步骤 4** 中产生的数据集生成下一代 KM 和 HC 种群，然后转到**步骤 3**。

步骤 7: 以进化得到的数据划分作为初始信息，使用改进的 KM 算法得到最终的聚类结果，**终止**。

3.1.2 候选算法介绍

（1）. K 均值算法[2]

K 均值算法是一种非常经典的聚类算法，它已被广泛的应用在很多领域。它通过把数据项分配给由"中心"表示的簇得到数据集的划分。K 均值算法寻找到的数据集划分在属性空间趋于球型。与其它算法相比，K 均值算法的复杂度比较低，因此运行速度快，尤其适合处理大规模问题。K 均值算法的主要缺点包括对初始中心敏感和容易陷入局部最优。它的主要步骤可以归纳如下:

算法 3.2 K 均值算法

输入: 数据集（数据集的大小为 N），簇数 K。

输出： 聚类结果

步骤 1： 初始化 K 个聚类中心，即 $\{C_1, C_2, ..., C_K\}$。

步骤 2： 对于每个数据项 $x_i\ (i=1,2...N)$，计算它与所有聚类中心的距离，即 $dis(x_i, C_j)\ (i=1,2...N; j=1,2...K)$，把当前数据项 x_i 分配给距离它最近的中心所代表的簇中。

步骤 3： 对于每一簇，计算其中数据项的平均值作为它的中心。与之前的中心比较，如果没有中心移动位置，则代表算法收敛，**终止**并输出结果；否则转到**步骤 2**。

（2）. 层级聚类算法[4]

层级聚类算法也是一种非常著名的聚类算法，它通过一步步将相似的数据项凝聚到一起使得数据集的划分逐渐达到指定簇数以达到聚类的目的。也存在一些层级聚类算法通过一步步划分数据集到指定的簇数以达到聚类目的。前者比较常见，在多数情况下效果好于后者，因此本章采用前者。层级聚类算法为整个数据集构建了一种树结构，仅需从不同的层级切断树的"边"即可得到不同簇数的聚类结果。这种操作在很多情况下是很方便的。层级聚类算法的主要缺点是一旦一个数据项被划分，则绝不可能在后续的步骤中改变原有的决策。层级聚类算法的主要步骤可归纳如下：

算法 3.3　层级聚类算法

输入： 数据集（规模为 N），簇数 K。

输出： 聚类结果。

步骤 1： 把每个数据项 $x_i\ (i=1,2...N)$ 作为一簇 $c_i\ (i=1,2...N)$。

步骤 2： 计算每对簇之间的距离 $dis(c_i, c_j)(i=1,2...k; j=1,2...k)$，$k$ 是当前簇数。

步骤 3： 把最相近的两簇合并为一簇，$k=k-1$。

步骤 4： 如果 $k=K$，终止并输出结果；否则转到**步骤 2**。

不同层级聚类算法的不同点主要在于如何判断两簇之间的距离，最常见的判断方式有三种：单连接（single-linkage）[7]，全连接（complete-linkage）[8]，和平局连接（average-linake）[52]。单连接中两簇之间的距离是两簇之间离得最近的一对数据项之间的距离；全连接中两簇之间的距离用两簇之间离得最远的一对数据项之间的距离表示；平局连接则以两簇之间数据项的平均距离表示.在本章中，HC 种群由全连接算法产生。

3.1.3 编码与适应度函数

本章采用实数编码，编码长度为数据集的大小。染色体中第 i 个基因位上的码代表数据集中第 i 个个体的簇标。图 3.1 说明了这种编码方式。图中是一条染色体，数据集的大小为 10。这串编码表示数据项 1，2，3，6，8 属于簇 1；数据项 4，5 属于簇 2；数据项 7，9，10 属于簇 3。 每次抽取操作过后，进化中的数据集规模有可能改变（因为存在把每簇的重合部分约简成一个数据和把非重合部分重新聚团的操作），因此染色体的长度也是可能改变的。根据实验可以观察到，染色体的长度大体上随着进化的收敛迅速下降，偶尔也会存在小范围的波动。这也意味着进化处理的数据集总体来说越来越小。

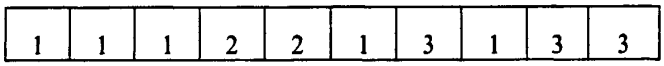


图 3.1 编码方式说明

本章采用 K-way Ncut 作为适应度函数，它用来从每个候选种群中选择出最优的个体进行后续的抽取操作。这个评判标准是为图像分割问题提出的[53]，在这项研究中图像被建模成一幅图，进而图像分割问题便转化成寻找图的最优划分问题。优化 KWNC 可以提供一幅图的 K 划分问题的最优近似解。在本章提出的算法中，数据也被建模成一幅图，因此也可以从图划分的角度去划分数据集。以 KNWC 作为适应度函数可以帮助找到最优的数据划分。最优的个体将为抽取操作提供最多的有用信息。KWNC 定义如式 3.1 所示：

$$KWNC=\frac{cut(A_1,V-A_1)}{assoc(A_1,V)}+\frac{cut(A_2,V-A_2)}{assoc(A_2,V)}+\ldots+\frac{cut(A_k,V-A_k)}{assoc(A_k,V)} \tag{3.1}$$

其中 A_i 是数据集 V 的第 i 个片段，K 是事先设定好的聚类数。在式 3.1 中，*cut* 和 *assoc* 定义如式 3.2 和式 3.3 所示：

$$cut(A,B)=\sum_{u\in A,v\in B} w(u,v) \tag{3.2}$$

$$assoc(A,V)=\sum_{u\in A,t\in V} w(u,t) \tag{3.3}$$

这里 $A\cup B=V$，$A\cap B=\varnothing$。$cut(A,B)$ 衡量了 A 与 B 的之间的不同；$assoc(A,V)$ 衡量了 A 与整幅图之间的不同。用 *assoc* 来规范化 *cut* 可以防止仅含有很少量数据项的集合被分为一簇。本章设定 $w(i,j)$ 是数据项 i 和数据项 j 之间的欧氏距离，则 KWNC 适应度应该被最大化。也就是说，KWNC 值越大的个体所表示的数据划分越合理。

3.1.4 进化操作

本小节主要介绍 HECM 中的进化操作和注意事项，包括预处理、初始化、变

异以及抽取。

（1）. 预处理

为了加速进化的运行速度，本章首先采取“聚团”的方式把原始数据集用 single-linkage 算法凝聚成 $M(K<M<N)$ 个小“团”。在接下来的进化过程中，这些“团”被视为单个的数据项，称为凝聚数据项。凝聚数据项由包含在其中的原始数据项的均值表示。本章称 M 为凝聚数。使用这种方法，嵌入进化框架中的 K 均值算法和 complete-linkage 算法所处理的凝聚数据集大小仅有 M。当 $M<<N$ 时，这种处理方法可以使得算法的搜索空间显著的下降。由实验章节可以看出当 $M<<N$ 时，聚类的结果并未收到明显的影响。图 3.2 说明了这个凝聚过程。给定 M=3 图中原始数据项 1 到 8，9 到 13 以及 14 到 20 分别被凝聚成了 3“团”。

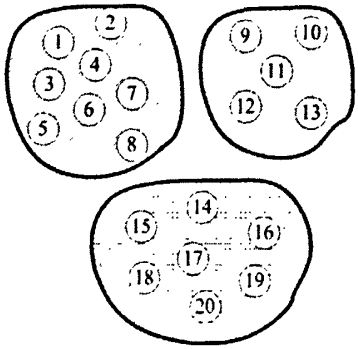


图 3.2 凝聚过程示例

在变异，抽取(部分)和搜索阶段，算法将返回原始数据集以达到精确操作的目的。因此，本章建立了一个索引，它表明了每个凝聚数据项包含着哪些原始数据项。表 3.1 是这个索引的一个例子，索引数据集如图 3.2 所示。原始数据集被凝聚成 3 个“团”，表 3.1 记录了原始数据项 1 到 8 属于第一个凝聚数据项（AD1）；原始数据项 9 到 13 属于 AD2；原属数据项 14 到 20 属于 AD3。

表 3.1　索引示例

索引号	原始数据号							
AD1	1	2	3	4	5	6	7	8
AD2	9	10	11	12	13	Null	Null	Null
AD3	14	15	16	17	18	19	20	Null

除此之外，本章在预处理中也计算了在基于邻域的搜索中要用到的邻域信息，并把它存储起来。这样此信息在后续迭代中就无需重复计算了。这个策略也加快了进化操作的速度。

（2）. 初始化

在这一步中， K 均值算法初始化大小为 KMPS 的一个候选种群，称为 KM 种群。本章随机选择 K 个凝聚数据项作为初始中心，然后用 K 均值算法将整个凝聚数据项聚为 K 簇。然后，聚类结果的簇标被编码成 KM 种群的个体。因为每个初始聚类中心都是随机选择的，因此各个 K 均值的聚类结果可能会不同。这一策略保证了 KM 种群的多样性。另一个候选种群由层级聚类算法 complete-linkage 产生，因此本章将聚类结果复制 HCPS 个，以产生一个大小为 HCPS 的初始种群。由于后续有变异操作，因此这个种群的多样行将在后续的进化操作中放大。这种多种群操作使得 K 均值算法的聚类结果和层级聚类算法的聚类结果能够在抽取之前进行同步独立进化。

（3）. 变异操作

本章采用了一种随机变异算子对当前 K 均值和层级聚类算法得到的数据划分做随机的扰动。这有助于避免局部最优以及增加候选种群的多样性。对选定的基因位，本章随机改变其上的编码，使之变成与之前不同的一个簇标。例如数据集将被聚为 4 簇，当前选定基因位上的编码为 1，代表此处的数据项被分为第一簇。那么本章将以随机的方式以相同的概率选取 2，3 或 4，并代替此基因位上的 1。变异概率应合理控制，过小会无助于种群多样性，过大会过分的破坏变异前的合理数据分布。

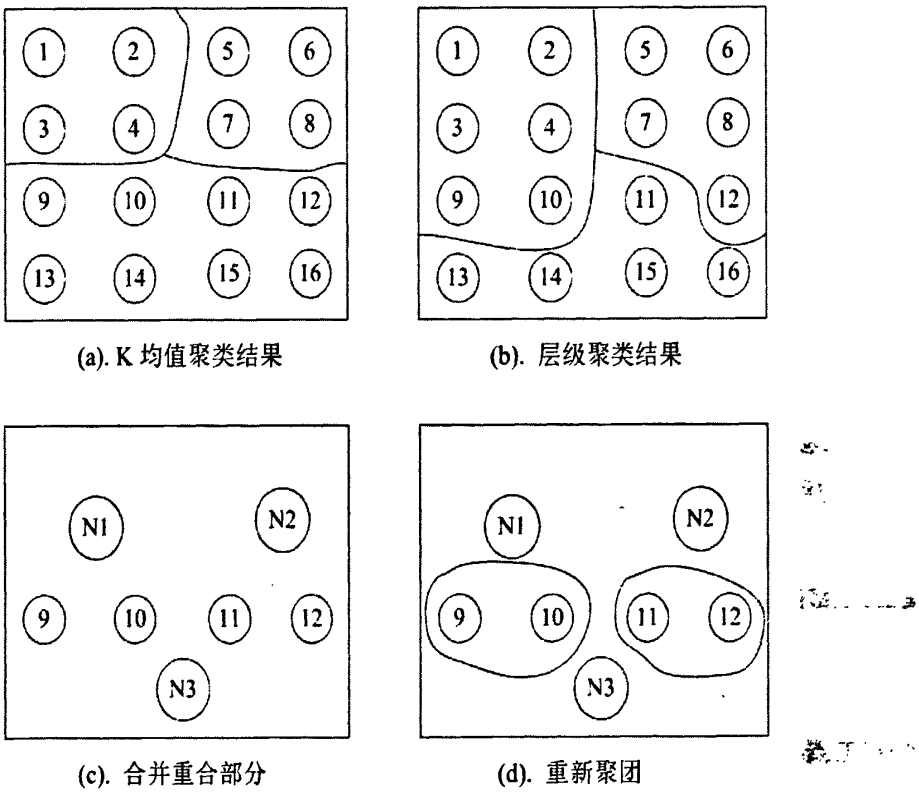
（4）. 抽取操作

抽取操作是 HECM 中非常重要的一部分。传统进化聚类算法应用交叉操作以交换父代个体所携带的信息并把它们传播到子代种群中。本章使用抽取操作从每个候选种群中的最佳个体里获得有用信息，并基于此信息产生下一代个体。因此本章的抽取操作可以被视为一种特殊的精确交叉操作，它可以有效地从父代个体中抽取有用信息，并可避免传统交叉操作的冗余计算。

首先，本章依据 KWNC 适应度值从两个候选种群中各挑选一个最优个体。如果这两个最优个体所携带的数据集划分信息中有重合的部分，即两个结果都显示某些数据项应该属于某一簇，那么本章认定这些重合部分的划分结果是合理的，并应予以保留。本章用每一簇重合区域中原始数据项的均值来表示这个区域。注意这里用前面章节提出的索引，由凝聚数据集返回原始数据集进行上述操作，以达到精确计算的目的。

图 3.3 展示了这一过程。假设图中 3.3(a)表示的是 KM 种群中最好的个体所记录的数据集划分结果。其中簇 1={ 1，2，3，4 }，簇 2={ 5，6，7，8 }，簇 3={ 9，10，11，12，13，14，15，16 }。图中 3.3(b)表示的是 HC 种群中最好个体所记录的数据集划分结果，其中簇 1={ 1，2，3，4，9，10}，簇 2={ 5，6，7，8，12 }，簇 3={ 12，13，14，15，16 }。这两个划分结果的重叠部分在簇 1 中为{ 1，2，3，

4}，在簇 2 中为{5，6，7，8}，在簇 3 中为{13，14，15，16}。因此本章把这 3 个重合部分合并成 3 个单独的数据项，即图 3.3(c)所示的 N1，N2 和 N3。如果非重合部分所包含的原始数据项比本章所期待的大，这里设定为大于 *M-K*（*M*=5，*K*=3），算法将把它们重新聚团成 *M-K* 块，如图 3.3(d)所示。



(a). K 均值聚类结果　(b). 层级聚类结果

(c). 合并重合部分　(d). 重新聚团

图 3.3　抽取操作示意

3.1.5　终止算子

如果进化收敛时，两个候选种群的最好个体所指示的数据分布完全重合，这意味着两种候选算法得出了相同的解。此时把这个解输出作为 HECM 的最终解。如果连续 G 代两个最佳个体的重合部分里包含的原始数据项数目不再上升，则进化收敛，并取出 G 代中原始数据项重合数目最大的结果作为一种改进 K 均值算法的初始条件进行计算，从而得出整体数据集的最终解。上述这种情况意味着在当前进化收敛规则的条件下两种候选算法不能对数据集的所有数据项都得出一致的判断，因此，本章偏重于一种算法从而得到最终结果。在实验中，本章采用一种改进的 K 均值算法完成这个工作。进化的结果用来决定这个算法的初始簇中心。这一改进算法受全局 K 均值算法启发，但实验中它的运行比全局 K 均值简洁很多。

设包含重合部分的簇数为 V，$V \le K$。本章把每簇中重合部分包含的数据项的均值作为当前簇的初始中心，这样就可以得到 K 均值算法的 V 个初始中心了。如

果 $V=K$，则只需要进行一次 K 均值聚类就可以得到最终结果了。这种情况在实验中常常出现。如果 $V<K$（在实验中这种情况偶尔发生且 V 的值一般为 $K-1$），且有 L 个非重合数据项，本章使用算法 3.4 来得到最终聚类结果。

算法 3.4　改进的 K 均值算法

输入： 由进化判断过的部分确定分簇的数据集，簇数 K。

输出： 聚类结果。

步骤 1： 把每簇中重合数据项的均值设定为一个初始簇中心，共可产生 V 个中心。

步骤 2： 依次从 L 个未重合的数据项中选择一个作为第 $V+1$ 个初始簇中心，这样可以得到 L 个数据集 $V+1$ 划分问题的解。

步骤 3： 从上述 L 个解中选取最好解，即每个数据项到它们的簇中心的欧氏距离和最小的解。产生这个解的初始簇中心（为某个非重合数据项）被视为第 $V+1$ 个初始簇中心。$V=V+1$。

步骤 4： 如果 $V=K$，输出最好结果，**终止**算法；否则，转到**步骤 2**。

从实验中可以观察到，与数据集的大小比起来，L 通常很小。所以这一终止操作并不会花费很多时间。

与传统进化 K 均值算法相比，HECM 的收敛代数较少，在下一章所述实验中，所有进化均可在 30 代以内收敛。HECM 的复杂度主要取决于嵌入其中的候选算法的复杂度。因本章中嵌入了层级聚类算法，其复杂度为平方级，因此本章中 HECM 的时间复杂度高于传统进化 K 均值算法。综合进化代数和时间复杂度，HECM 的运行时间和传统进化 K 均值算法相比持平或略长。

3.2 实验结果与分析

3.2.1 实验设定

本章采用四个人工数据集以及五个 UCI 数据集测试 HECM 的性能。为了全面的评估 HECM，本章将其与传统的进化 K 均值算法，嵌入 HECM 中的 K 均值算法和 complete-linkage 算法做以比较。HECM，进化 K 均值算法和 K 均值算法都是随机算法，每次运行的结果可能不同，因此本章把每个实验都独立运行 30 次，并比较统计结果。这九个数据集的信息列在表 3.2 中，图 3.4 为四个人工数据集及它们的标准分布。

表 3.2　测试数据集详细信息

数据集	数据集大小	属性维数	标准聚类数
sun and moon	500	2	2
three leaves	750	2	3
square1	1000	2	4
square4	1000	2	4
breast	277	9	2
pimaindians	768	8	2
wine	178	13	3
iris	150	4	3
zoo	101	16	7

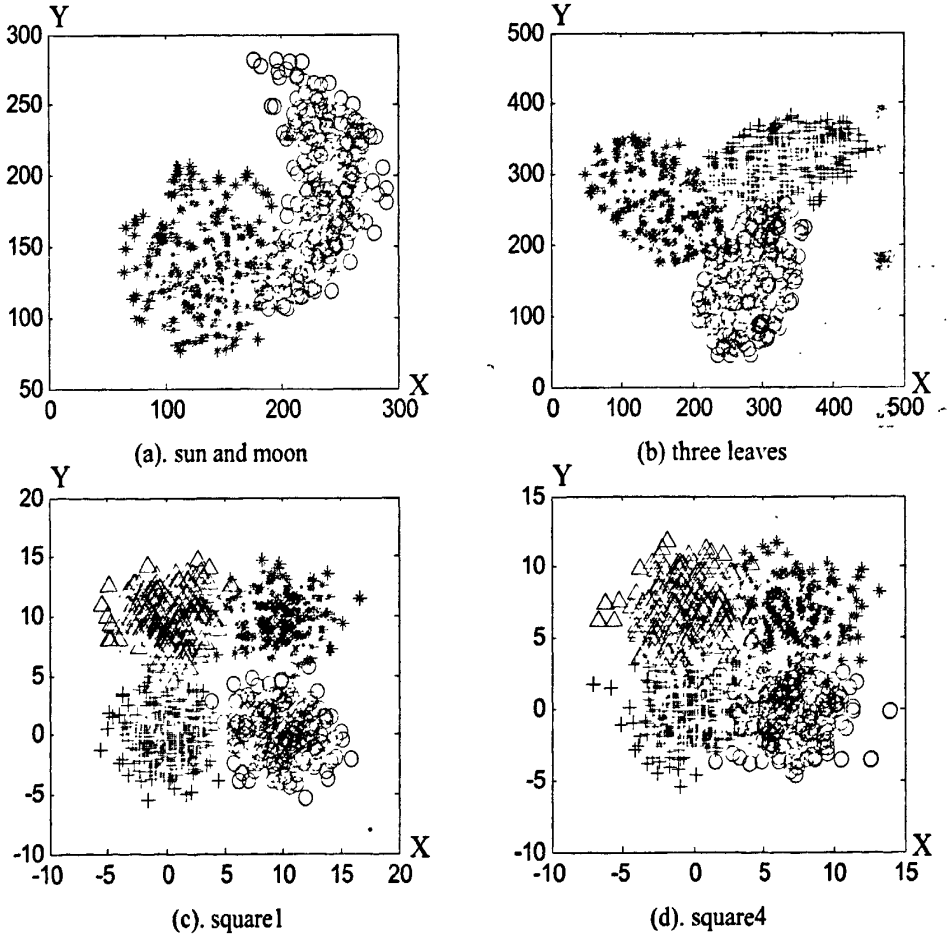


图 3.4　人工数据集及标准分布

在实验中，本章将所有的属性都进行了规范化处理，如式 3.4 所示：

$$f_{ij}^{*}=\frac{f_{ij}}{\max(F_i)-\min(F_i)} \tag{3.4}$$

这里 f_{ij} 是数据项 j 的第 i 维属性值，f_{ij}^{*} 是 f_{ij} 规范化之后的值。$\max(F_i)$ 和 $\min(F_i)$ 分别是整个数据集第 i 维属性的最大值和最小值。这个处理可以避免属性值变化大的属性淹没属性值变化小的属性对聚类的影响。为了说明这一处理的效果，本章分别对规范化和非规范化属性的数据集做了实验。

HECM 的实验参数设置如表 3.3 所示。如果在一代里，抽取操作遗留下的非重合数据项数目大于凝聚数（设为 M），则把它们重新凝聚成 M 个团。聚团这些非重合数据项的时候，重合数据项不参与其中。如果非重合数据项的数目小于 M，则本章不再重新聚团，并把凝聚数设为 $\lceil M/2 \rceil$。在当前设定下，试验基本可在 30 代内收敛。

本章中的对比算法——一种进化 K 均值算法[24]的参数设置如表 3.4 所示。它采用了实值编码，把个体编码成聚类中心，并使用模拟二进制交叉和多项式变异。所有这些设置都如文献[24]所示，它们都是进化计算和聚类中比较常见的策略。

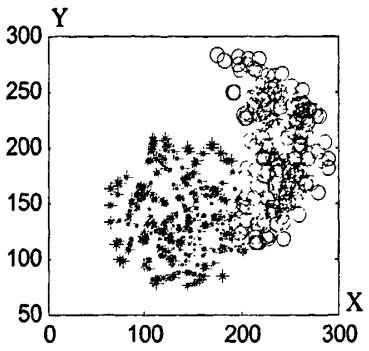
表 3.3 实验参数设置

参数	设置		
KM 种群规模	10		
HC 种群规模	10		
变异概率	0.1		
邻域大小	5		
凝聚数	sun and moon	three leaves	square1
	20	15	20
	square4	Breast	pimaindians
	20	20	20
	wine	Iris	zoo
	18	15	10
终止条件	连续 5 代抽取后重合数据项数目没有增加		

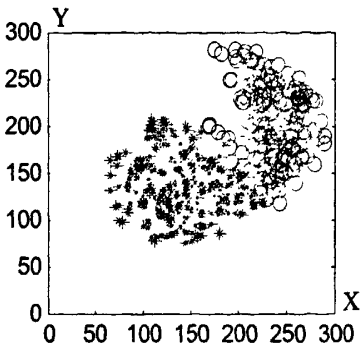
表 3.4 进化 K 均值算法参数设定

参数	设置
种群规模	20
变异概率	0.1
交叉概率	0.8
终止条件	连续 5 代适应度值没有提高

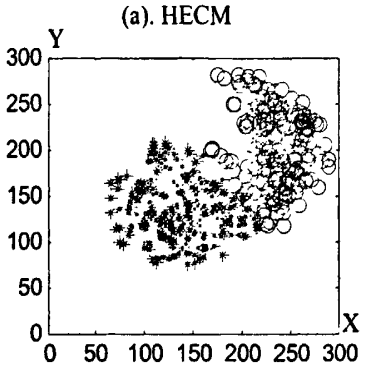
3.2.2　实验结果与分析



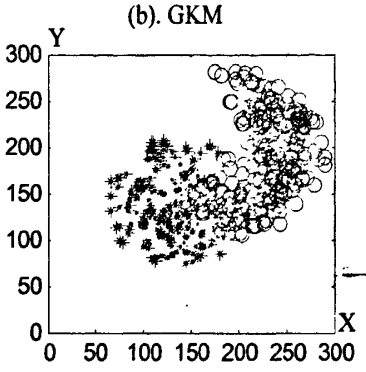
(a). HECM



(b). GKM

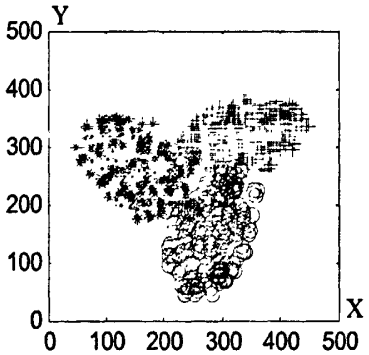


(c). KM

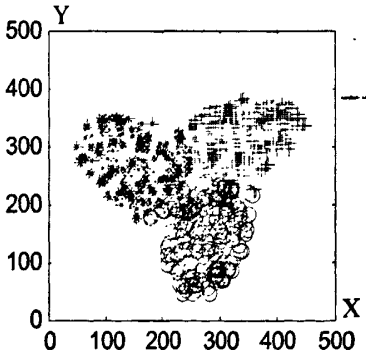


(d). CL

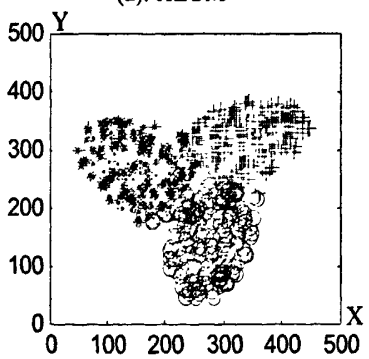
图 3.5　sun and moon 数据集典型实验结果



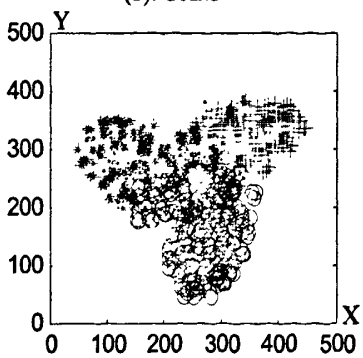
(a). HECM



(b). GKM

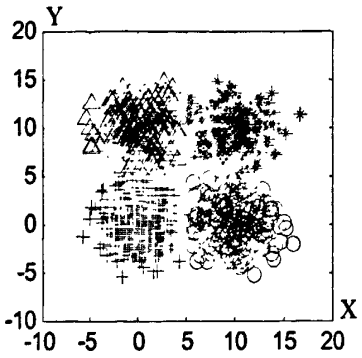


(c). KM

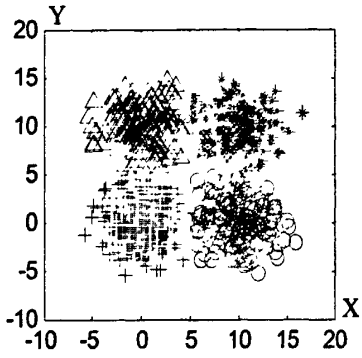


(d). CL

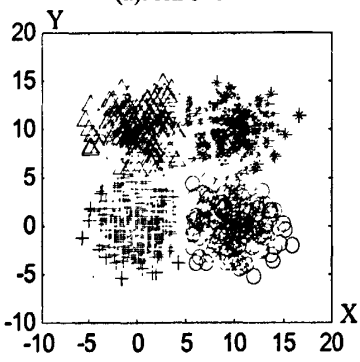
图 3.6　three leaves 数据集典型实验结果



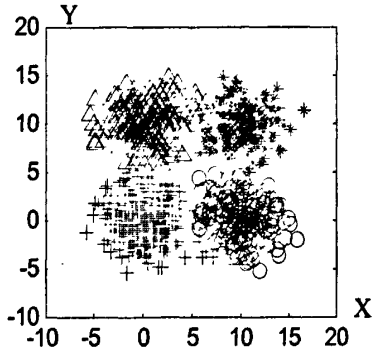
(a). HECM



(b). GKM

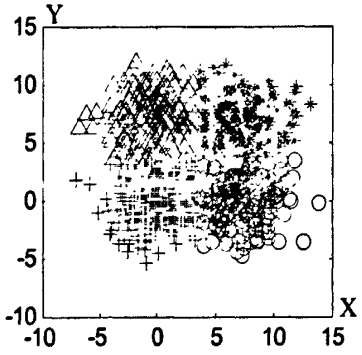


(c). KM

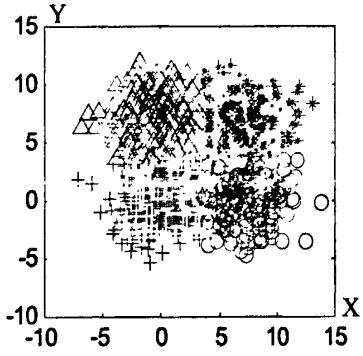


(d). CL

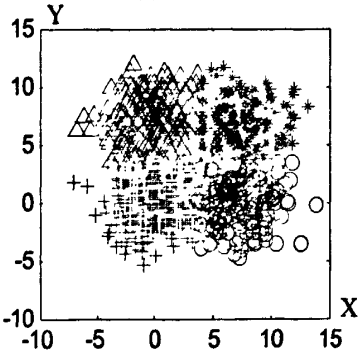
图 3.7 square1 数据集典型实验结果



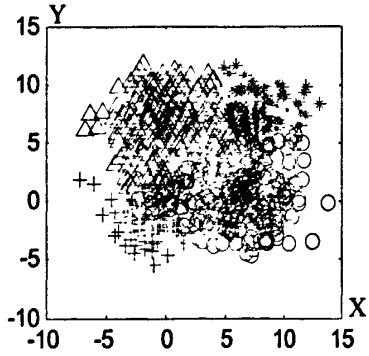
(a). HECM



(b). GKM

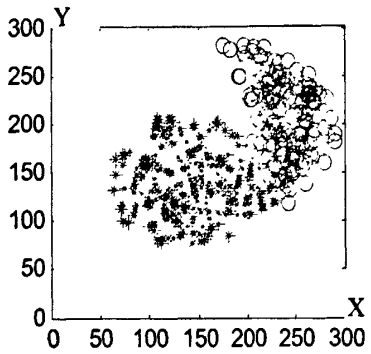


(c). KM

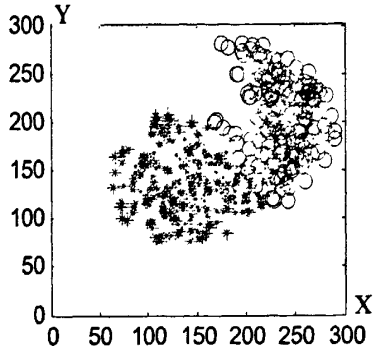


(d). CL

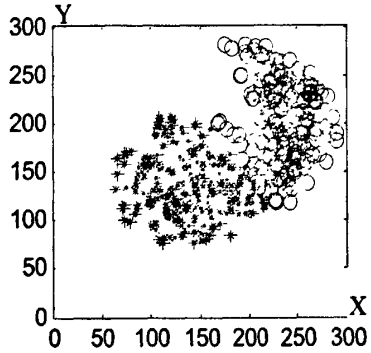
图 3.8 square4 数据集典型实验结果



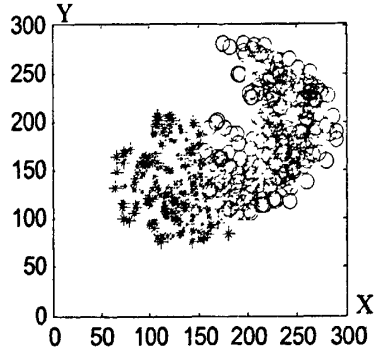
(a). HECM(N)



(b). GKM(N)

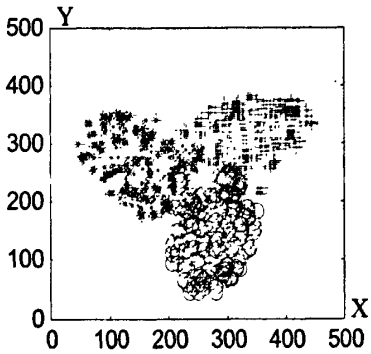


(c). KM(N)

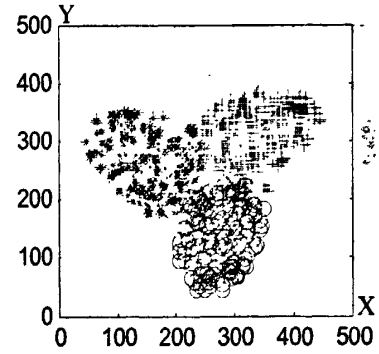


(d). CL(N)

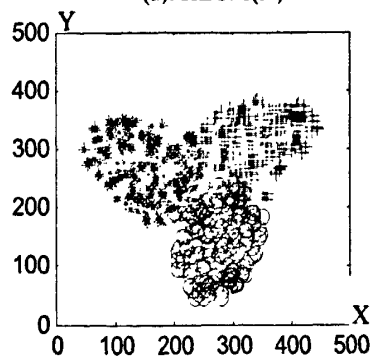
图 3.9　sun and moon 数据集典型实验结果（未进行特征规范化）



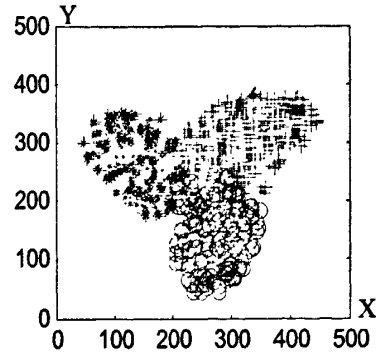
(a). HECM(N)



(b). GKM(N)

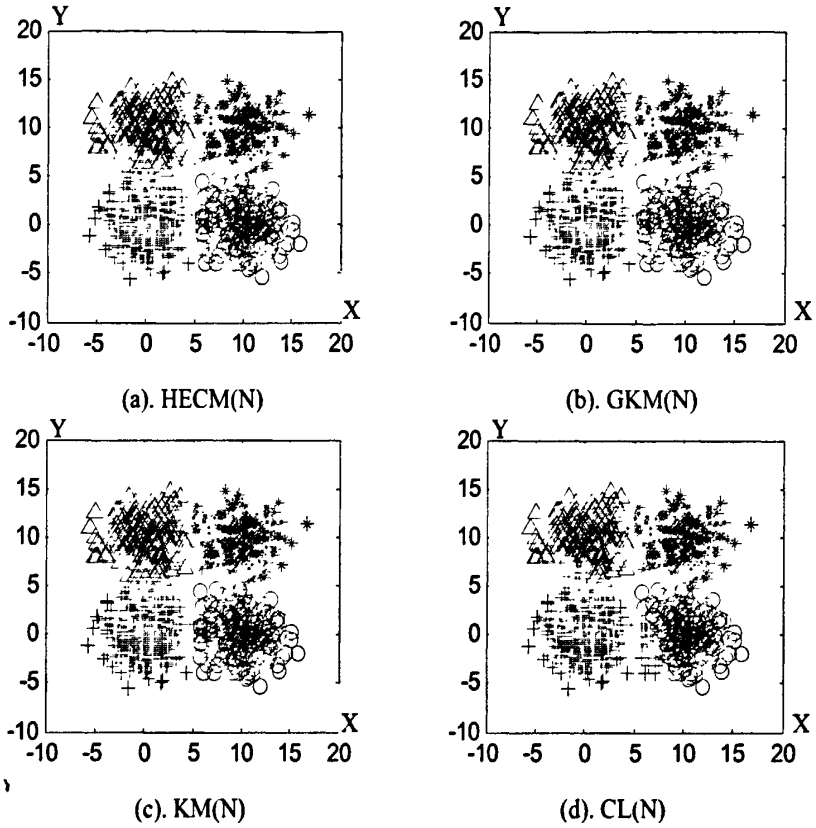


(c). KM(N)



(d). CL(N)

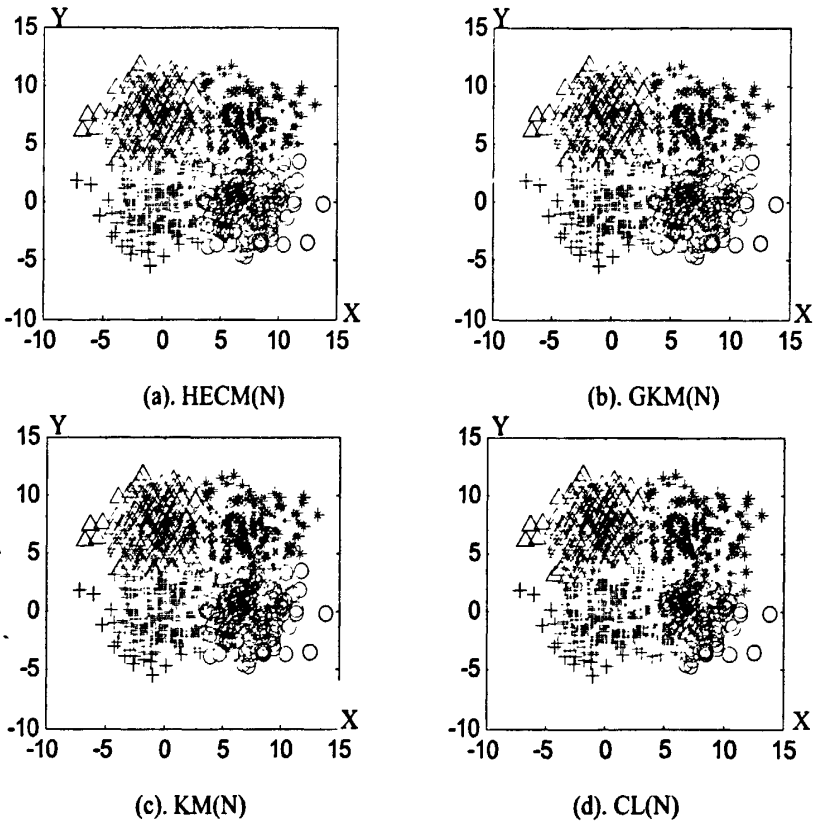
图 3.10　three leaves 数据集典型实验结果（未进行特征规范化）



(a). HECM(N) (b). GKM(N)

(c). KM(N) (d). CL(N)

图 3.11 square1 数据集典型实验结果（未进行特征规范化）



(a). HECM(N) (b). GKM(N)

(c). KM(N) (d). CL(N)

图 3.12 square4 数据集典型实验结果（未进行特征规范化）

本章算法和对比算法在4个2维人工数据实验上的代表性结果，如图3.5到图3.8所示。实验数据集依次为sun and moon、three leaves、sqaure1、square4。每组图中从(a)到(d)的四幅图分别是HECM，进化K均值（GKM），K均值（KM）和complete-linkage（CL）算法的代表性结果。图3.9到图3.12分别为未对特征进行规范化的实验结果。实验数据集依次为sun and moon、three leaves、sqaure1、square4。每组图中从(a)到(d)的四幅图分别是HECM(N)，进化K均值（GKM(N)），K均值（KM(N)）和complete-linkage（CL(N)）算法的代表性结果。

从图中可以看出，各种聚类算法结果的主要区别在于对不同簇的交界区域的判定上。以 sun and moon 数据集为例，图3.5中HECM算法将球形和月牙形两个区域较好的划分开，而其余对比算法则将球形区域的一部分误划分给了月牙形区域。在图3.6中，数据集的三个椭圆形区域被HECM算法较好的划分开。而其余对比算法均以更大的错误率将每个椭圆区域中的一部分误划分给其余的椭圆区域。在图3.7和图3.8中，HECM，GKM，以及KM算法的结果图相差不大。但是CL的效果与前几种算法相比较差。但值得注意的是，KM可能会产生较差的结果，本文在这里未作展示，所示结果是30次试验中比较好的结果。图3.9到图3.12所表现出的规律与图3.5到图3.8类似。但数据集特征规范化和未进行规范化所获得的聚类结果有所不同。如图3.12中CL的聚类结果右下角的簇的一部分误分到左下角的簇；而在图3.8中，聚类结果左下角的簇的一部分误分到右下角的簇。总体来说特征规范化处理后得到的聚类结果优于未规范化得到的结果。

这里列出了对每个数据集独立运行30次实验中的最大，最小和平均正确率值。表3.5展示了规范化属性后的结果（为方便对比，本章把未规范化属性的HECM算法的结果也列入），表3.6展示了未进行属性规范化属性的结果，其中的黑体字是每行中的最好结果。表中HECM表示本章所提出的算法，HECM（N）表示未进行属性规范化的HECM算法，GKM表示进化K均值算法，KM表示K均值算法，CL表示complete-linkage算法。从表3.5可以看出，HECMS最明显的效果是可以找到嵌入其中的候选算法和进化K均值算法所不能找到的合理数据划分。在sun and moon，three leaves，square1，iris，zoo数据集上的实验很好的证明了这一点， HECM得到了比其它算法都好的最大正确率。在数据集three leaves和zoo上，HECM还得到了最好的统计平均正确率。未对数据集属性进行规范化的HECM表现明显低于HECM，仅在breast数据集上获得了比规范化属性的HECM更好的最大正确率，二者的稳定性相当。与进化K均值相比，HECM的稳定性略逊一筹，下一章将说明如何使这类算法的稳定性增强。此外进化K均值算法仅在wine数据集上得到了最大正确率，在其它数据集上表现和HECM相当或稍差。K均值算法由于对初始中心敏感且容易陷入局部最优，因此在数据集breast，iris和zoo上性能不稳定，在人工数据集上表现比较稳定，它仅在pimaindians上获得了最大正确

表 3.5 对数据集属性进行规范化的 HECM 算法和其它算法的实验正确率对比

数据集	正确率值(%)	算法				
		HECM	*HECM(N)*	*GKM*	*KM*	*CL*
sun and moon	最大值	**97.00**	93.00	93.00	93.20	87.20
	平均值	92.53	92.27	92.85	**93.09**	87.20
	最小值	91.40	91.40	**92.26**	**92.26**	87.20
three leaves	最大值	**94.53**	91.70	89.73	89.33	78.00
	平均值	**92.56**	88.59	89.68	89.26	78.00
	最小值	**89.33**	83.33	88.27	89.07	78.00
square1	最大值	**99.20**	99.00	99.00	99.00	98.00
	平均值	98.62	98.20	**99.00**	**99.00**	98.00
	最小值	91.60	96.80	**99.00**	**99.00**	98.00
square4	最大值	**93.60**	93.50	**93.60**	**93.60**	72.90
	平均值	90.57	92.20	**93.60**	93.34	72.90
	最小值	70.50	86.70	**93.60**	93.10	72.90
breast	最大值	72.20	**75.81**	70.76	72.92	72.92
	平均值	60.36	**73.11**	52.22	61.83	72.92
	最小值	50.90	71.12	50.90	50.90	**72.92**
pimaindians	最大值	66.80	66.80	66.80	**66.93**	65.10
	平均值	65.46	66.38	**66.80**	**66.80**	65.10
	最小值	64.32	66.02	**66.80**	**66.80**	65.10
wine	最大值	95.51	62.92	**96.63**	95.51	93.26
	平均值	80.13	57.79	**95.00**	94.79	93.26
	最小值	55.62	55.06	**93.26**	93.26	93.26
iris	最大值	**95.51**	88.67	88.67	88.67	88.00
	平均值	80.13	70.04	**88.63**	81.13	88.00
	最小值	55.62	36.67	88.00	57.33	88.00
zoo	最大值	**92.08**	79.21	91.09	80.20	75.25
	平均值	**87.63**	76.11	72.51	67.10	75.25
	最小值	**75.25**	69.31	52.48	45.54	**75.25**

表 3.6　无属性值规范化处理的实验结果

数据集	正确率值(%)	算法			
		HECM(N)	*GKM(N)*	*KM(N)*	*CL(N)*
sun and moon	最大值	93.00	**93.40**	**93.40**	89.00
	平均值	92.27	**93.40**	**93.40**	89.00
	最小值	91.40	**93.40**	**93.40**	89.00
three leaves	最大值	**91.70**	89.47	89.47	89.87
	平均值	88.59	89.47	89.33	**89.87**
	最小值	83.33	89.47	89.20	**89.87**
square1	最大值	**99.00**	**99.00**	**99.00**	95.70
	平均值	98.20	**99.00**	**99.00**	95.70
	最小值	96.80	**99.00**	**99.00**	95.70
square4	最大值	93.50	93.50	**93.60**	87.70
	平均值	92.20	93.49	**93.50**	87.70
	最小值	86.70	**93.40**	**93.40**	87.70
breast	最大值	**75.81**	72.20	73.29	74.37
	平均值	73.11	71.97	64.78	**74.37**
	最小值	71.12	71.84	51.26	**74.37**
pimaindians	最大值	**66.80**	66.02	66.02	64.97
	平均值	**66.38**	66.02	66.02	64.97
	最小值	**66.02**	**66.02**	**66.02**	64.97
wine	最大值	62.92	**70.22**	**70.22**	67.42
	平均值	57.79	**70.22**	68.05	67.42
	最小值	55.06	**70.22**	56.74	67.42
iris	最大值	**88.67**	72.67	77.33	76.00
	平均值	70.04	64.00	60.56	**76.00**
	最小值	36.67	35.33	36.67	**76.00**
zoo	最大值	**79.21**	88.12	87.13	75.25
	平均值	**76.11**	67.82	64.89	75.25
	最小值	69.31	54.46	42.57	**75.25**

率。Complete-linkage 算法由于不具有随机性，因此稳定性最好，但是其寻找合理数据集划分的能力低于其它算法。总的来说，在这九个数据集上的实验中，HECM 的性能最好。

由表 3.6 可以看出即使在不对属性规范化的时候，HECM(N)仍然具有前文叙述的特点，即可以找到比嵌入的候选算法的聚类结果更优的数据划分。此外，它也具有比传统 K 均值算法更好的性能。HECM 在 three leaves,，square1，breast，pimaindians，iris 和 zoo 数据集上都得到了最大正确率，GKM 在 sun and moon，square1，square4，wine 数据集上得到了最大正确率，KM 在 sun and moon，square1，square4，wine 上得到了最大正确率，而 CL 未在任何数据集上获得最大正确率。从稳定性上可以看到，由于 CL 不是随机算法，因此最稳定，其次是 GKM 和 HECM，KM 的稳定性最差。对比表 3.5 和表 3.6 可以看出，将数据集的属性规范化对所有算法的结果都有影响。大部分的影响是正面的，即聚类结果正确率有所提高，尤其是对 HECM 算法。有些数据集的正确率在这种处理下会下降，但总体来说，正确率上升的幅度比下降的幅度大，且正确率上升的结果占多数。规范化属性操作简单快捷，因此本章保留这一处理，在下一章也予以采用。

3.3 小结

本章提出了一种基于多种群的混合进化聚类算法，称为 HECM。HECM 使用了一种新的进化框架，旨在运用不同的候选聚类算法找出合理的数据划分。多种群策略和抽取策略是 HECM 实现这一目标的关键。整个数据集被建模成了一幅无向图，本章引用了一种基于图的适应度函数，用来在每个候选种群中选择一个最好的个体进行抽取操作。此外，本章也使用了约简方法来减少进化过程处理的数据集大小，进而提高算法的运行速度。本章使用了实数编码操作以方便进化。在进化收敛后，可能会出现两种情况：一.如果进化过程为所有的数据项都做了划分，则此结果作为 HECM 的最终聚类结果；二.如果进化收敛后还存在一些数据项未被划分簇，则本章设计了一种改进的 K 均值算法得到最终的聚类结果，而此时进化所做的判断作为此算法的初始条件。

对四个人工数据集和五个 UCI 数据集的测试中，HECM 可以找到比嵌入其中的候选算法找到的更合理的数据划分。此外，HECM 的效果也优于传统的进化 K 均值算法。需知 HECM 旨在提供一种有效的进化聚类算法框架，所以嵌入其中的候选算法并不仅仅局限于 K 均值和 complete-linkage。任何合理的算法组合都可以嵌入 HECM 框架中。此外，在进化计算的全局搜索框架下嵌入搜索策略也是现在一种比较流行的改进方法。下一章将实施这两种改进并分析结果。

第四章　基于多种群和图搜索的混合进化聚类算法

本章主要介绍一种基于多种群和图搜索的混合进化聚类算法（Hybrid Evolutionary Clustering Algorithm with Multi-population and Graph-based Search），简称 HECMS。HECMS 也是一种混合策略进化聚类算法，它采用上一章所述 HECM 的算法框架。但 HECMS 的种群策略与 HECM 不同，它扩展了 HC 种群的多样性。此外，HECMS 添加了十分重要的策略——两种基于图的搜索。实验显示，HECMS 比 HECM 性能更优越。

本章的安排如下：4.1 小节叙述了 HECMS 的算法框架和与 HECM 不同的部分——种群策略和基于图的搜索；4.2 小节描述了 HECMS 的实验结果和分析，并对重要的参数做了讨论。4.3 小节对本章进行了总结。

4.1 基于多种群和图搜索的混合进化聚类算法

本节详细的介绍了 HECMS 的算法框架，种群策略以及基于图的搜索。

4.1.1　算法框架

与上一章相同，整个数据集被建模成一个无向图 $G(V,E)$。图中的节点是数据集中的数据项；连接节点与节点的边的权值 $w(i,j)$ 为数据项 i 到 j 之间的欧氏距离。HECMS 算法采用与 HECM 相同的适应度函数，多候选种群策略和一种崭新的抽取策略，以结合不同候选聚类算法的结果得到数据的划分。每一个候选种群由一种候选算法产生。抽取操作应用在从每个候选种群中选出的最优个体上，以有效地将它们的信息传递给下一代个体。本章采用实值编码技术和随机变异方法。与 HECM 相同，进化收敛后，可能会出现的第一种情况是所有的数据项都由进化过程聚类，则进化的结果作为最终的聚类结果。另一种情况是部分数据项不能由进化决定划分，此时使用上一章提到的改进 K 均值算法来获得最终聚类结果。HECM 中的初始化和抽取操作之后的数据集约简措施也在 HECMS 中使用以提高算法的运行速度。本章以 K 均值和三种不同的层级聚类算法 complete-linkage，single-linkage 和 average-linkage 算法为候选算法产生两个候选种群。此外，本章引入了很重要的策略，即基于图的搜索，从而在当前数据集划分的基础上寻找更合理的划分结果。HECMS 算法的框架如下：

算法 4.1 基于多种群和图搜索的混合进化聚类算法

输入： 数据集，簇数。

输出： 聚类结果。

步骤 1： **预处理。**原始数据集被聚成团以输入进化过程，此外下面各步骤所需的一些信息也在这一步产生和存储，以避免后续的重复计算。

步骤 2： **初始化。**分别用 K 均值算法初始化 KM 种群；用 complete-linkage，single-linkage 和 average-linkage 算法初始化 HC 种群。

步骤 3： **变异。**这里采取随机变异策略。

步骤 4： **基于图的搜索。**这里使用了两种基于图的搜索策略，即基于邻域的搜索和基于簇连接的搜索。本章将搜索后的个体和之前的个体比较，取适应度值较大的一个放入种群中。

步骤 5： **抽取。**根据 KWNC 适应度值，本章从每个候选种群中选出最好的个体，然后找出这两个最好个体所判断的数据集划分的重叠部分。重叠部分中的原始数据项将由它们的均值代替。对于非重叠部分，如果需要，它里面的原始数据项将被重新聚团至一定数目。

步骤 6： 如果没有非重叠数据项存在，则输出当前的数据集划分作为最终的聚类结果；否则，转到**步骤 7**。

步骤 7： 如果重叠部分中所包含的原始数据项的数目连续 G 代都没有上升，则转到**步骤 8**；否则用**步骤 5** 中产生的数据集生成下一代 KM 和 HC 种群，然后转到**步骤 3**。

步骤 8： 以进化得到的数据划分作为初始信息，使用改进的 KM 算法得到最终的聚类结果，**终止。**

4.1.2 种群策略

本章也采用了双候选种群策略。与 HECM 类似，第一个候选种群——KM 种群用 K 均值算法以随机方式产生。第二个候选种群采用三种层级聚类算法初始化，分别为 single-linkage，complete-linkage 和 average-linkage。层级聚类算法的主要步骤可以参见 3.1.2 小节的内容。这三种算法的不同之处在于它们如何判断两簇之间的距离，single-linkage 把两簇之间离得最近的两个数据项之间的距离作为两簇之间的距离，complete-linkage 把两簇之间离得最远的两个数据项之间的距离作为两簇之间的距离，average-linkage 两簇之间数据项的平均距离作为两簇之间的距离。

由于层级聚类算法不具有随机性，每次对同一数据集的运行结果相同。因此本章首先用三种方法产生三个聚类结果，编码成三个初始个体。然后把这三个个

体复制生成 HC 种群，则种群大小 *HCPS=Times*×3，其中 *Times* 表示复制的次数。这样，HC 初始种群一共有三种个体，与 HECM 算法相比多样性增强了。在后续的步骤中，除了变异之外，还存在两种搜索过程可以使得种群的多样性随着进化增加。这将有利于算法搜寻到更加合理的数据划分。

4.1.3　基于图的搜索

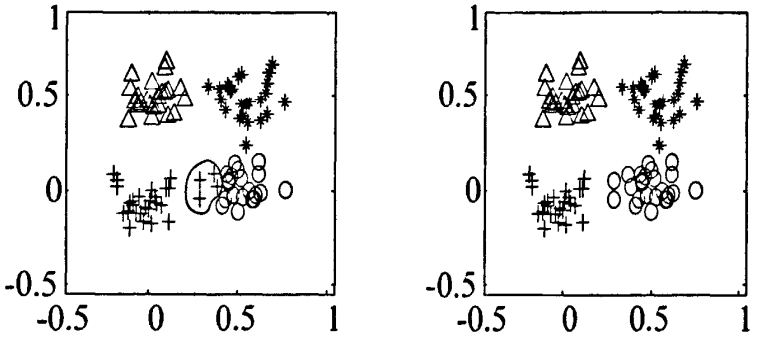
基于图的搜索是本章非常重要的步骤。数据集被建模成了一副无向图，所以本章设计了两种不同的基于图的搜索策略在图中寻找更好的数据划分。它们分别是基于邻域的搜索和基于簇连接的搜索。两种搜索将用于变异后的个体，它们分别来自 KM 种群和 HC 种群。为了精确地操作，此时由种群中凝聚数据集的聚类结果返回原始数据集，返回方法可参见第三章 3.1.4 小节的内容。例如，本章假定在图 3.2 中的数据集被分成了两簇，图中上半部分的两个凝聚数据项属于簇 1，而处于下半部分的凝聚数据项属于簇 2。在搜索阶段，利用索引返回原始数据集的分布，应为簇 1={ 1，2，3，4，5，6，7，8，9，10，11，12，13 }和簇 2={ 14，15，16，17，18，19，20 }。接下来，搜索将在这两个集合中的原始数据项上进行。

本章设置了两个参数来控制搜索过程，命名为搜索强度（SI）和搜索范围（SR）。SI 指的是种群中有多少比例的个体将被搜索，SR 指的是被搜索的个体所表示的数据集划分中，有多少比例的数据项将被搜索过程判断。被选中的个体将进行何种搜索则以 0.5 的概率随机决定两种之一。

基于邻域的搜索受文献[30]的启发而设计。邻域指的是数据项的邻域，一个数据项邻域中的内容是在图上距离这个数据项最近的几个数据项。邻域的范围，也就是它包含了多少数据项，由用户提前设定。例如，如果邻域范围是 2，则表明每个数据项的邻域只包含在图上离它最近的两个数据项。在选定进行搜索的个体中，随机在要进行搜索的数据项邻域中选择一个邻域数据项，如果被搜索的数据项与选定的邻域数据项不在同一簇，则把被搜索数据项移动到选定的邻域数据项所属的簇中。在一个邻域中，每个邻域数据项可能被选择的概率相同。图 4.1 说明了这一搜索的作用，其中 4.1(a)是未被搜索的数据集划分，4.1(b)是基于邻域搜索的一个可能结果。图中阴影部分表示变化了分簇的数据项，可以看出数据集的划分变得更加合理了。

基于簇连接的搜索过程执行如下：首先，计算将要被搜索的数据项与每簇之间在图上的连接关系，称为簇连接（cluster linkage），简称 *CLink*。被搜索的数据项将被移动到这个值最小的那一簇中。*CLink* 的定义如式 4.1 所示：

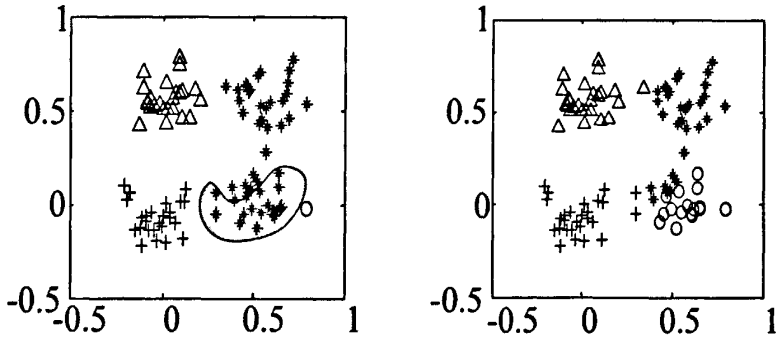
$$CLink_i^c = \sum_{j=1}^{N^c} dis(x_i, x_j^c) \Big/ N^c, \quad \left(x_i \neq x_j^c\right) \tag{4.1}$$



（a）搜索之前数据集划分　　（b）搜索之后数据集划分

图 4.1 基于邻域的搜索示意图

其中 $CLink_i^c$ 是数据项 x_i 对应于第 c 簇的簇连接；x_j^c 是第 c 簇中的第 j 个数据项；N^c 是第 c 簇中数据项的数目；$dis(x_i, x_j^c)$ 是 x_i 和 x_j^c 之间的欧氏距离。这一测量指标表示的是选定的数据项到某一簇中所有数据项在图上的平均距离。图 4.2 说明这种搜索的效果。其中 4.2(a)是 single-linkage 算法的结果，作为搜索前的数据集分布，4.2(b)是经过基于簇连接搜索的结果，搜索对每个数据项都进行了一次。从这两幅图中可以看到数据的划分变得合理了许多，阴影部分表示变化了分簇的数据项。



（a）搜索之前数据集划分　　（b）搜索之后数据集划分

图 4.2　基于簇连接的搜索示意图

算法分别计算了搜索前后个体的适应度值，如果搜索后个体的适应度值大于搜索前，则用它代替搜索前的个体进入后续的操作中，否则保持原来的个体不变。在后续的实验章节 4.2 中，文章通过测试参数验证了搜索的效果。

4.2 实验与结果分析

4.2.1　实验设定

同第三章一样，本章采用四个人工数据集以及五个 UCI 数据集来测试 HECM 的性能。为了全面的评估 HECMS，本章将其与 HECM，传统的进化 K 均值算法，嵌入 HECMS 中的 K 均值算法，single-linkage，complete-linkage 和 average-linakge 算法做以了比较。HECMS，进化 K 均值算法和 K 均值算法都是随机算法，每次运行的结果可能不同，因此本章把每个实验都独立运行 30 次，并比较统计结果。九个实验数据集的信息列在表 3.2 中，图 3.3 为四个人工数据集及它们的标准分布。在实验中，本章将所有的属性都进行了规范化处理，如 3.2.1 小节所述。

HECMS 的实验参数设置如表 4.1 所示。如果在一代里，抽取操作遗留下的非重合数据项数目大于凝聚数（设为 AN），则本章把它们重新凝聚成 AN 个团。聚团这些非重合数据项的时候，重合数据项不参与其中。如果非重合数据项的数目小于 AN，则本章不再重新聚团，并把凝聚数设为$\lceil AN/2 \rceil$。

表 4.1　实验参数设置

参数	设置		
KM 种群规模	10		
HC 种群规模	10		
搜索强度	1.0		
搜索范围	1.0		
变异概率	0.1		
邻域大小	5		
凝聚数	sun and moon	three leaves	square1
	20	15	20
	square4	breast	pimaindians
	20	20	20
	wine	iris	zoo
	18	15	10
终止条件	连续 5 代抽取后重合数据项数目没有增加		

本章中的对比算法——进化 K 均值算法的参数设置如表 3.4 所示，它的详细

设置请参见 3.2.1 小节。

4.2.2 实验结果与分析

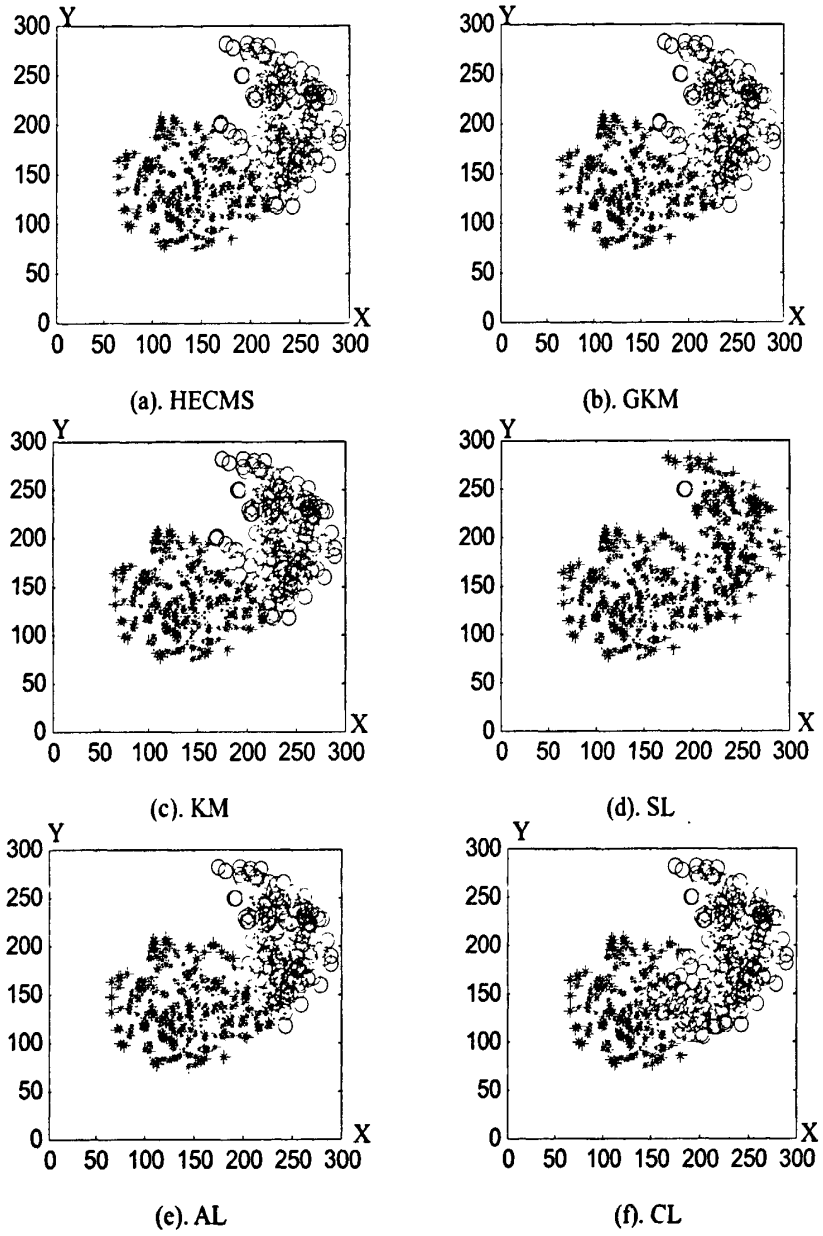
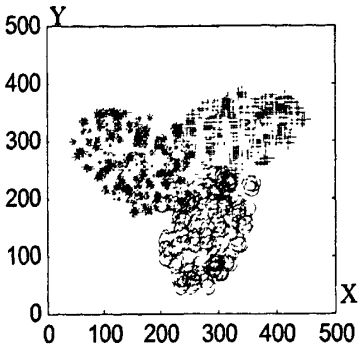
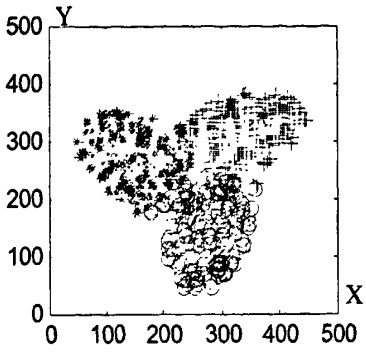


图 4.3 sun and moon 数据集典型实验结果

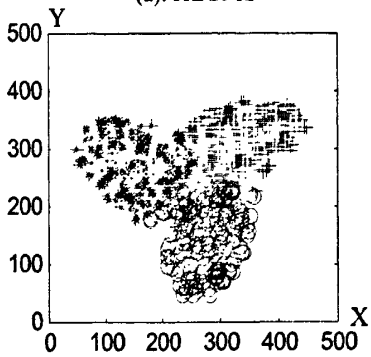
这里将各算法的典型结果展示如下。其中图 4.3 到图 4.6 分别是各种算法对 sun and moon, three leaves, square1 和 square4 数据集的典型结果。这几组图中，从图(a)到图(f)分别是算法 HECMS，进化 K 均值（GKM），K 均值（KM），single-linkage（SL），average-linkage（AL）和 complete-linkage（CL）的典型结果。



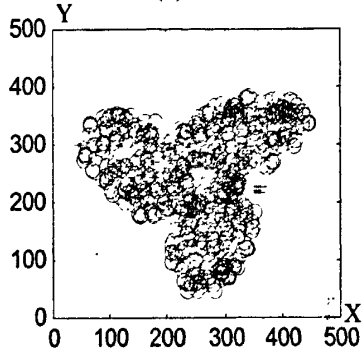
(a). HECMS



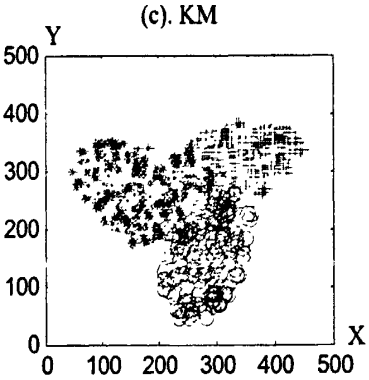
(b). GKM



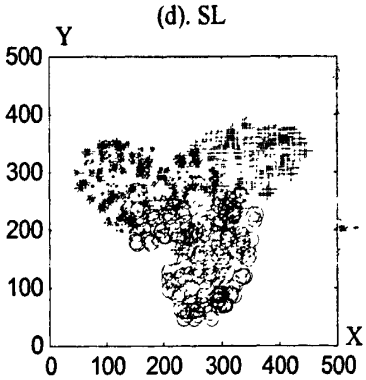
(c). KM



(d). SL

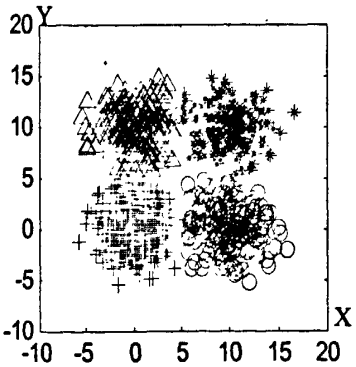


(e). AL

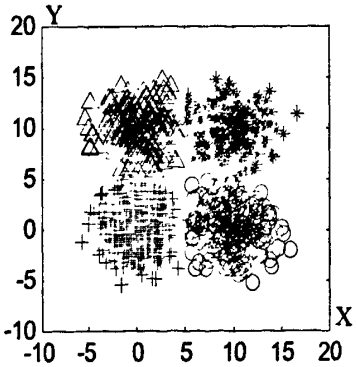


(f). CL

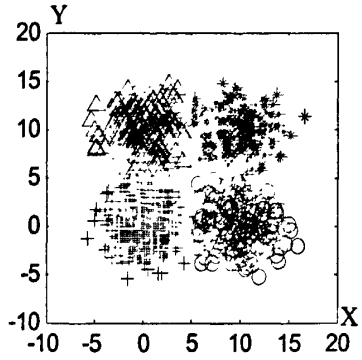
图 4.4 three leaves 数据集实验典型结果



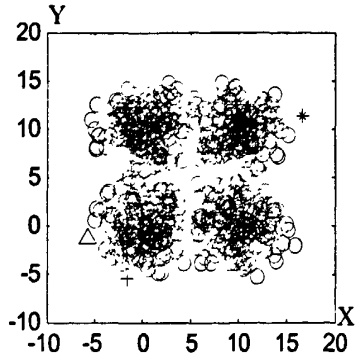
(a). HECMS



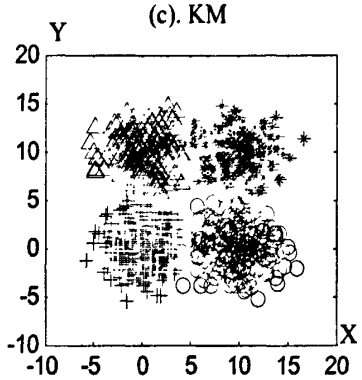
(b). GKM



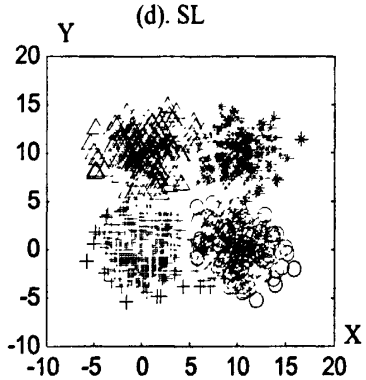
(c). KM



(d). SL

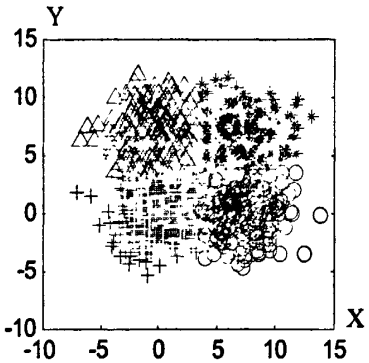


(e). AL

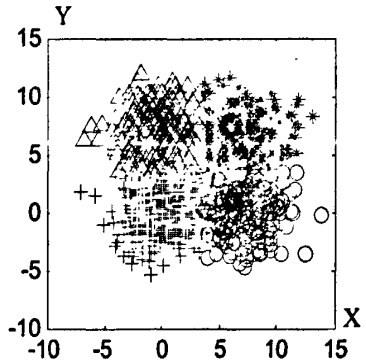


(f). CL

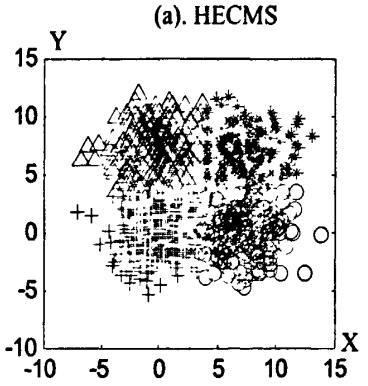
图 4.5 square1 数据集典型实验结果



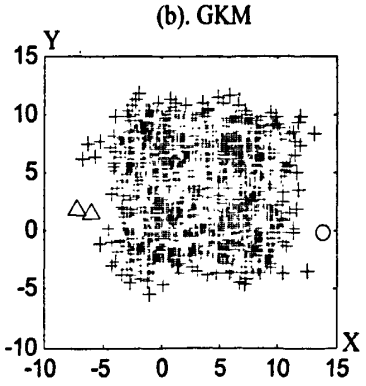
(a). HECMS



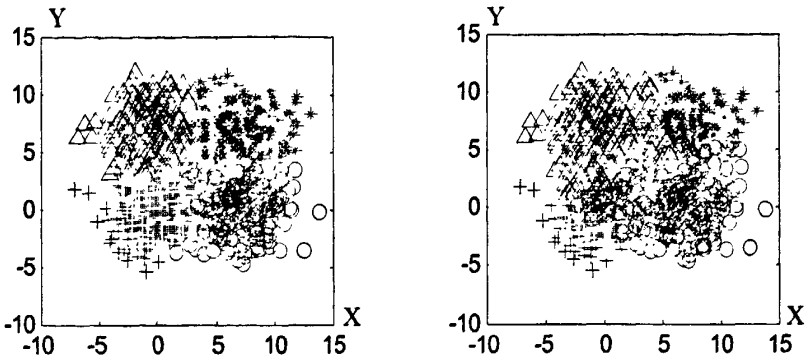
(b). GKM



(c). KM



(d). SL



(e). AL　　(f). CL

图 4.6 square4 数据集典型实验结果

对于 sun and moon 数据集，由图 4.3 可见 HECMS，GKM 和 KM 的效果较好，其中 HECMS 将球形区域误分给月牙形区域的数据项最少，GKM 和 KM 的效果类似。SL 效果很差，无法将两个区域分开，却将月牙形区域中比较孤立的两个数据项分成了单独的一簇。AL 将月牙形区域中的很多数据项误划分给球形区域，CL 则得到相反的结果。类似地，在图 4.4three leaves 数据集试验中，HECMS 得到了最好的聚类效果，GKM 和 KM 其次，AL 和 CL 效果相当，SL 的结果最差。在图 4.5 和图 4.6 中，即对 square1 和 square4 数据集试验中，可以看到 HECMS，GKM 和 KM 都能获得较理想的聚类结果。SL 的结果最差。对于 square1 数据集，AL 和 CL 的结果在中间部分数据项上有所差异，判定结果左右簇倾向相反。而对于 square4 数据集，AL 的效果好于 CL，因为 CL 结果中左上角和右下角的两簇大量吞噬了本来应属于右上角和左下角簇的数据项。值得注意的是，KM 的运行结果不稳定，这里展示的是其获得的比较好的结果，此外也存在一些效果较差的聚类结果。

本节展示了对每个数据集独立运行 30 次实验中的最大，最小和平均正确率值。表 4.2 为这些统计结果，其中的黑体字是每行中的最好结果。可以看出 HECMS 在所有数据集上都得到了最好的最大值。整体上讲，HECMS 比 GKM 的性能更好，只有在数据集 squanre4 和 wine 上 HECMS 的平均值低于 GKM。此外 HECMS 的稳定性也比 GKM 好，这可以体现在它们结果的最大值与最小值差值的大小上。以上对比可以说明本章所提出的进化聚类算法比传统的算法更具优势。K 均值算法的结果显示它偶尔可以得到比较好的值，但是非常的不稳定，尤其在数据集 breast，iris 和 zoo 上。SL 在大多数数据集上表现一般，在数据集 three leaves，square1，square4 和 wine 上表现很差。AL 在 sun and moon，square1，breast 数据集上得到了比较好的结果，而 CL 仅在 breast 数据集上得到的好的结果。总得来说，HECMS 在所有的实验里表现最好，并适用于多种不同类型的数据集。HECMS 最大的优点是可以找到 GKM 和嵌入其中的候选算法所找不到的合理数据划分。在对数据集

表 4.2 实验统计结果

数据集	正确率值(%)	算法					
		HECMS	*GKM*	*KM*	*SL*	*AL*	*CL*
sun and moon	最大值	**93.40**	93.00	93.20	52.60	93.00	87.20
	平均值	**93.31**	92.85	93.09	52.60	93.00	87.20
	最小值	**93.00**	92.26	92.26	52.60	**93.00**	87.20
three leaves	最大值	**92.67**	89.73	89.33	34.00	88.40	78.00
	平均值	**91.28**	89.68	89.26	34.00	88.40	78.00
	最小值	**89.33**	88.27	89.07	34.00	88.40	78.00
square1	最大值	**99.10**	99.00	99.00	25.20	98.70	98.00
	平均值	**99.01**	99.00	99.00	25.20	98.70	98.00
	最小值	**99.00**	**99.00**	**99.00**	25.20	98.70	98.00
square4	最大值	**93.70**	93.60	93.60	25.30	92.60	72.90
	平均值	93.50	**93.60**	93.34	25.30	92.60	72.90
	最小值	92.60	**93.60**	93.10	25.30	92.60	72.90
breast	最大值	**72.92**	70.76	**72.92**	68.23	**72.92**	**72.92**
	平均值	61.83	52.22	61.83	68.23	**72.92**	**72.92**
	最小值	50.90	50.90	50.90	68.23	**72.92**	**72.92**
pimaindians	最大值	**68.10**	66.80	66.93	65.23	65.23	65.10
	平均值	**66.92**	66.80	66.80	65.23	65.23	65.10
	最小值	64.45	**66.80**	**66.80**	65.23	65.23	65.10
wine	最大值	**97.19**	96.63	95.51	38.76	38.76	93.26
	平均值	94.78	**95.00**	94.79	38.76	38.76	93.26
	最小值	**93.26**	93.26	93.26	38.76	38.76	93.26
iris	最大值	**93.33**	88.67	88.67	66.00	88.67	88.00
	平均值	**89.00**	88.63	81.13	66.00	88.67	88.00
	最小值	88.00	88.00	57.33	66.00	**88.67**	88.00
zoo	最大值	**92.08**	91.09	80.20	67.33	75.25	75.25
	平均值	**84.98**	72.51	67.10	67.33	75.25	75.25
	最小值	66.34	52.48	45.54	67.33	**75.25**	**75.25**

three leaves，wine，iris 和 zoo 的实验结果上，可以很明显的看到这一点。与第三章的试验结果相比，HECMS 比 HECM 的效果好很多，这充分说明了本章中扩展 HC 种群多样性和两种基于图的搜索策略的积极作用。

表 4.3 列出了在对 square1，iris 的 30 次独立实验中，当进化收敛时非重合原始数据项的数目。其中参数 *SI=0.5* and *SR=0.5*。表 4.4 列出了相同实验里含有重合原始数据项的簇的数目。在表 4.3 和表 4.4 中，0 表示进化阶段就得到了最终聚类结果。本章可以看到在对数据集 square1 的实验中，大多数最终结果是靠改进后的 K 均值算法得到的；而在对数据集 iris 实验中，大多数的最终结果是进化后直接得到的。在两个实验中，进化收敛后留下的非重合原始数据项的数目与数据集的规模比起来是非常小的。并且重合原始数据项分布在所有的 *K* 簇中（square1 数据集 *K*=4，iris 数据集 *K*=3）。这意味着只需要进行一次 K 均值聚类就可以得到最终聚类结果了。在别的数据集实验中，也存在这样的规律，即含有重合原始数据项的簇数在绝大多数情况下等于标准分簇数，偶尔会出现等于 *K-1* 的情况。

表 4.3　square1 和 iris 数据集实验中进化收敛后非重合数据项个数

数据集	非重合数据项个数（个）									
square1	1	0	1	0	2	1	2	1	2	1
	0	0	1	1	0	0	1	1	1	1
	1	0	2	1	0	1	0	2	1	0
iris	0	0	0	0	0	0	1	1	0	0
	1	1	0	0	0	0	0	0	0	1
	0	1	0	0	0	0	0	1	1	2

表 4.4　square1 和 iris 数据集实验中进化收敛后含有重合数据项的簇数

数据集	含有重合数据项的簇数（个）									
square1	4	0	4	0	4	4	4	4	4	4
	0	0	4	4	0	0	4	4	4	4
	4	0	4	4	0	4	0	4	4	0
iris	0	0	0	0	0	0	3	3	0	0
	3	3	0	0	0	0	0	0	0	3
	0	3	0	0	0	0	0	3	3	3

本节还设计了实验来说明参数搜索强度（SI）和搜索范围（SR）对算法的影响。图 4.7 是 HECMS 在 wine 数据集上的 30 次运行结果展示，每条曲线对应不同的 *SI* 和 *SR* 值。可以看到在没有搜索的情况下（SI=0.0，SR=0.0），运行结果不稳定。有些结果可以高达 95.51%，但也存在另一些不好的解，甚至低于 70.00%。当 SI=0.5，SR=0.5 时，运行结果比之前稳定，但是仍然存在低于 60.00%的解。当 SI=1.0，SR=1.0 时，所有 30 个解都很好，且它们之间的差异不大，说明算法运行很稳定。此外，这 30 个解还包含之前两种参数设定下所没有得到的优质解。在实验过程中，可以发现 SI 和 SR 并非对所有数据集都是越大越好，比如 iris 数据集就是个例外。

图 4.8 展示了凝聚数对实验结果的影响，图 4.9 展示了在相同实验中运行时间的对比。其中测试数据集为 iris，数据集大小为 150，这里将凝聚数每次变化数据集大小的 0.1，对应数据集中的 15 个数据项。图中分别展示了 30 次独立实验中聚类正确率最大值，平均值和最小值对应不同参数的曲线。从图 4.8 中可以看出凝聚数对算法的影响不是很大，每条曲线上相邻两点之间的正确率变化小于 0.05。整体上看，图中仅仅是最大值曲线稍微有所下降，而平均值曲线和最小值曲线并未显著的变化，相隔两点之间的正确率变化基本小于 0.01。这主要归功于两点：一、变异，学习和抽取阶段返回原始数据集的精确操作；二、凝聚策略使得彼此靠近的数据点被归为同一个凝聚数据项，并且不会在后续候选聚类算法的操作中彼此分离。从图 4.9 中可以看出随着凝聚数的增大，运行时间逐渐上升，特别是凝聚数大于 105 之后。这是因为随着凝聚数变大，进化中处理的凝聚数据集大小增加了，而且数据分布更加复杂。最后三个参数设定的实验运行时间较之前提高了 1 倍到 6 倍左右，且 30 次实验运行的时间稳定性有很大下降。基于本章实验的得出的规律，本通常可以把凝聚数设定在原始数据集大小的 10%以下，这样既可以减少运行时间，也可以保证聚类效果。

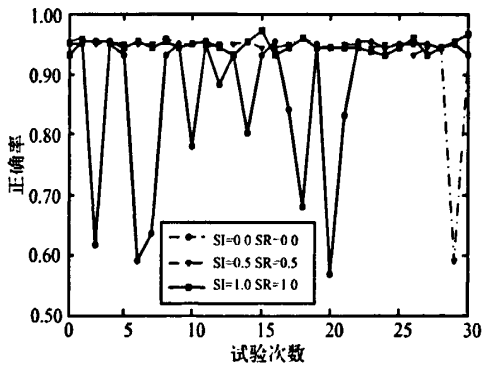


图 4.7 HECMS 在 wine 数据集上对应于不同参数 *S* 和 *SR* 的 30 次独立运行结果

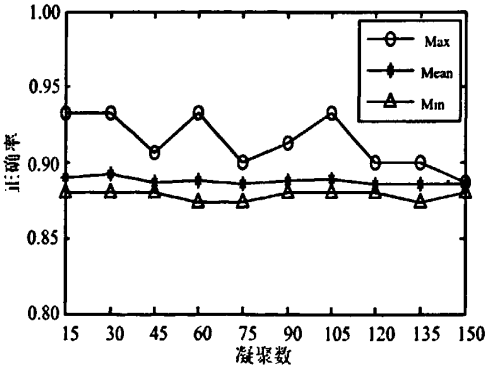


图 4.8 凝聚数对运行结果的影响

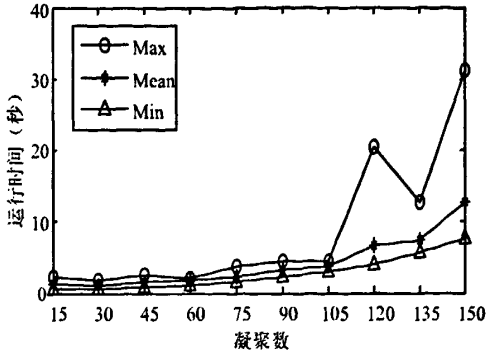


图 4.9　凝聚数对运行时间的影响

4.3 小结

本章提出了一种基于多种群和图搜索的混合进化聚类算法，称为 HECMS。HECMS 提供了一种有别于传统进化聚类算法的新框架，它旨在运用不同的候选聚类算法找出合理的数据划分。与上一章类似，HECMS 运用了一种独特的多种群策略和抽取策略来实现这一点。整个数据集被建模成了一幅无向图，本章设计了两种基于图的搜索以在当前候选算法聚类结果的基础上寻找更好的数据划分。本章引用了一种基于图的适应度函数，它用来在每个候选种群中选择一个最好的个体用于抽取。此外，本章也使用了约简方法来减少进化过程中处理的数据集大小，进而提高算法的运行速度。为了方便操作，HECMS 使用了实数编码作为沟通数据集和进化之间的桥梁。在进化收敛后，可能会出现两种情况：一.如果进化过程为所有的数据项都做了划分，则此结果作为 HECMS 的最终聚类结果；二.如果进化收敛后还存在一些数据项未被划分簇，则采用一种改进的 K 均值算法得到最终的聚类结果，而此时进化所做的判断作为此算法的初始条件。

对四个人工数据集和五个 UCI 数据集的测试中，可以发现 HECMS 可找到比嵌入其中的候选算法找到的更合理的数据划分。此外，HECMS 的效果也优于传统的进化 K 均值算法。在参数分析实验中，可以看到两种搜索策略的效果很明显，并且凝聚约简策略既可以保证算法的质量，又可以加快算法执行的速度。

若想提高 HECMS 的性能，有三点可以加强。第一点是选择合理的候选算法；第二点是构造更有效的抽取策略；第三点是加强搜索策略。

第五章. 总结与展望

聚类算法是数据挖掘，机器学习，图像处理等领域的一项十分重要的研究。它在科研和人们日常生活中都扮演了重要的角色。本文主要讨论了两种进化聚类算法，即混合属性进化聚类算法和混合策略进化聚类算法。第一章提供了必要的背景知识，主要包括聚类、进化计算和进化聚类算法。第二章介绍了混合属性进化聚类算法(ECM)，第三章和第四章介绍了两种混合策略进化聚类算法，即 HECM 和 HECMS。

本文在第二章首先介绍了混合属性数据集和 K 原型算法，它们是理解 ECM 的基础。接下来我们详细地描述了算法的主要流程和各项进化操作及算子。本章设计了实验以验证 K 原型及其组成部分（K 均值和 K 模式）相对于进化聚类算法的表现。所有这些进化聚类算法都一一和原始算法做了比较，实验结果表明我们设计的进化框架大大的提高了原来算法的性能。本章使用了 K 原型的方法，通过引入一个权值来平衡数值型和种类型属性对聚类效果的影响。在未来的研究中，如何自动调节这个权值将是个有意义的课题。另一种思路是把数值型属性和种类型属性看做两个目标，应用多目标算法解决混合属性问题。ECM 对数值型属性采用了欧氏距离平方的度量；对种类型属性采用了汉明距离度量。在以后的工作中引入更合理的度量方法也将是一个有意义的研究课题。

本文在第三章提出了一种基于多种群的混合进化聚类算法，称为 HECM。HECM 使用了一种新的进化框架，旨在运用不同的候选聚类算法找出合理的数据划分。多种群策略和抽取策略是 HECM 实现这一目标的关键。整个数据集被建模成了一幅无向图，因此我们引用了一种基于图的适应度函数在每个候选种群中选择一个最好的个体用于抽取。为提高算法的运行速度，我们在进化中也使用了数据集约简策略。此外，本章也使用了实值编码和随机变异技术。进化收敛后，可能会出现的第一种情况是进化过程为所有的数据项都做了划分，则此结果作为 HECM 的最终聚类结果；如果进化收敛后还存在一些数据项未被划分簇，则使用一种改进的 K 均值算法得到最终的聚类结果，而此时进化所做的判断作为此算法的初始条件。

测试中我们发现 HECM 可以找到比嵌入其中的候选算法找到的更合理的数据划分。此外，HECM 的效果也优于传统的进化 K 均值算法。HECM 旨在提供一种有效的进化聚类算法框架，所以嵌入其中的候选算法并不仅仅局限于 K 均值和 complete-linkage。如何选择合理的算法组合嵌入 HECM 框架是一个有价值的研究

课题。

本文在第四章中提出了一种基于多种群和图搜索的混合进化聚类算法，称为HECMS。HECMS 沿用了 HECM 的算法框架，把整个数据集建模成了一幅无向图。本文设计了两种基于图的搜索在当前候选算法聚类结果的基础上寻找更好的数据划分。与 HECM 相比，HECMS 在 HC 种群中嵌入了三种层级聚类算法，大大增加了种群的多样性。在测试中，我们发现 HECMS 可以找到比嵌入其中的候选算法得到的更合理的数据划分。此外，HECMS 的效果也优于传统的进化 K 均值算法和 HECM 算法。在参数分析实验中，我们可以看到两种搜索策略的效果很明显，并且凝聚约简策略既可以保证算法的质量，又可以加快算法执行的速度。若想提高 HECMS 的性能，有三点可以加强。第一点是选择合理的候选算法；第二点是构造更有效的抽取策略；第三点是设计性能更好的终止算法。

致谢

在硕士学习阶段即将结束之际，我谨以此文献给敬爱的母校，导师，家人，同学和朋友们。感谢你们给我提供了良好的学习和生活环境，也感谢你们给予我的巨大帮助和关怀。我所取得的每一份成就，都有你们的功劳。

首先，我要感谢西安电子科技大学智能感知与图像理解教育部重点实验室，它为我们提供了良好的科研环境和学术氛围。实验室也是一个温暖的大家庭，给予了我无微不至的关怀。接下来我要特别感谢我的导师焦李成教授，是他引领我进入了广博的科研领域。他渊博的学识、严谨的治学态度、对科学前沿敏锐的洞察力以及对科学事业无私的奉献精神都深深地影响了我。焦教授不仅是我硕士期间学习的导师，更是我迈向人生更高阶段的领路人。我还要感谢公茂果教授，他耐心的指导我克服了一个又一个困难，逐步走入科研正轨。公教授的博学多才、亲切和蔼以及严谨认真使我折服。我能够顺利地完成硕士研究生阶段的学习，离不开公教授的鼓励和帮助。此外，我要感谢智能所侯彪、王爽、缑水平、钟桦、李阳阳、刘若辰、马文萍等其他老师在科研工作中对我的指导和帮助，与他们交流和探讨使我获益匪浅。

我也要感谢杨咚咚、凤宏晓学长，马晶晶学姐，吴巧娣、刘芳、左益、邓倩倩、王倩、斯佳佳、逄菲菲、张娟、刘超、冯吭雨、赵富家等同学给予我的帮助和支持。与他们共同度过的研究生生活给了我许多启发，在日常生活中我们互相帮助，建立了牢固的友谊。

最后我要衷心感谢我的父母，感谢他们多年来对我成长的支持和关爱。没有家庭的支持，就没有我今天的成绩。

郑智

二〇一〇年十二月于西安电子科技大学

参考文献

[1] P. Berkhin, "A Survey of Clustering Data Mining Techniques," Grouping Multidimensional Data, J. Kogan, et al., Eds., ed: Springer Berlin Heidelberg, pp. 25-71, 2006.

[2] Hartigan, J.A., Wong, M.A., "A K-Means clustering algorithm," Applied Statistics, pp.100-108, 28 (1979).

[3] Sanghamitra Bandyopadhyay and Sriparna Saha, "A Point Symmetry-Based Clustering Technique for Automatic Evolution of Clusters," IEEE Transactions on Knowledge and Data Engineering, Vol. 20, No. 11, pp.1441-1457. Nov. 2008.

[4] Stephen C. Johnson, Hierarchical Clustering Schemes, Psychometrika ,Vol. 32~ No. 3, Sep. 1967.

[5] 周明.孙树栋.遗传算法原理及应用.北京.国防工业出版社.1999 年 6 月. pp. 22-23, 37-38, 46, 53-54, 56-57, 166-175

[6] Rui Xu; Wunsch, D., II, "Survey of clustering algorithms," IEEE Transaction on Neural Networks, Vol. 16, No. 3, pp. 645-678, May 2005.

[7] P. Sneath, "The application of computers to taxonomy," J. Gen. Microbiol.,Vol. 17, pp. 201–226, 1957.

[8] T. Sorensen, "A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its applicationto analyzes of the vegetation on Danish commons," Biologiske Skrifter, Vol. 5, pp. 1–34, 1948.

[9] S. Guha, R. Rastogi, and K. Shim, "CURE: An efficient clustering algorithm for large databases," in Proc. ACM SIGMOD Int. Conf. Management of Data, pp. 73–84, 1998.

[10] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An efficient data clustering method for very large databases," in Proc. ACM SIGMOD Conf. Management of Data, pp. 103–114, 1996.

[11] K. Krishna and M. Murty, "Genetic K-means algorithm," IEEE Trans.Syst., Man, Cybern. B, Cybern., Vol. 29, No. 3, pp. 433–439, Jun. 1999.

[12] M. Ester, H-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the 2nd ACM SIGKDD, pp. 226–231, Portland, Oregon, USA, 1996.

[13] A. Hinneburg and D. Keim. An efficient approach to clustering large multimedia

databases with noise. In Proceedings of the 4th ACM SIGKDD, pp. 58–65, New York, NY, USA, 1998.

[14] Michael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and J¨org Sander.Optics: Ordering points to identify the clustering structure. In Proceedings ACM SIGMOD International Conference on Management of Data, pp. 49–60, Philadelphia, Pennsylvania, USA, 1999.

[15] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In Proceedings of ACM SIGMOD, pp. 94–105, Seattle, Washington, USA, 1998.

[16] W. Wang, J. Yang, and R. Muntz. STING: a statistical information grid approach to spatialdata mining. In Proceedings of the 23rd Conference on VLDB, pp. 186–195, Athens, Greece, 1997.

[17] D. Barbará and P. Chen, "Using the fractal dimension to cluster datasets,"in Proc. 6th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, pp. 260–264, 2000.

[18] R. Sharan and R. Shamir, "CLICK: A clustering algorithm with applications to gene expression analysis," in Proc. 8th Int. Conf. Intelligent Systems for Molecular Biology, pp. 307–316, 2000.

[19] A. Ben-Dor, R. Shamir, and Z. Yakhini, "Clustering gene expression patterns," J. Comput. Biol., Vol. 6, pp. 281–297, 1999.

[20] F. Höppner, F. Klawonn, and R. Kruse, Fuzzy Cluster Analysis: Methods for Classification, Data Analysis, and Image Recognition. New York: Wiley, 1999.

[21] J. Bezdek and R. Hathaway, "Numerical convergence and interpretation of the fuzzy c-shells clustering algorithms," IEEE Trans. Neural Netw., Vol. 3, No. 5, pp. 787–793, Sep. 1992.

[22] L.O. Hall, B. Ozyurt, and J.C. Bezdek. Clustering with a genetically optimized approach. IEEE Trans. on Evolutionary Computation, Vol. 3, No. 2, pp.103–112, 1999.

[23] G.P. Babu and M.N. Murty. A near-optimal initial seed value selection in kmeans algorithm using a genetic algorithm. Pattern Recogn. Lett., Vol. 14, No. 10, pp.763–169, 1993.

[24] Zhi Zheng, Maoguo Gong, Licheng Jiao, Jingjing Ma, Qiaodi Wu, "Unsupervised Evolutionary Clustering Algorithm for mixed type data,", In Proceedings of The 2010 IEEE Congress on Evolutionary Computation, CEC2010, Barcelona, Spain,

July 18-23, 2010.

[25] E. Falkenauer, "Genetic Algorithms and Grouping Problems," NewYork: Wiley, 1998.

[26] Eduardo Raul Hruschka, Ricardo J. G. B. Campello, Alex A. Freitas, and Andr´e C. Ponce Leon F. de Carvalho, "A Survey of Evolutionary Algorithms for Clustering," IEEE Transaction on System, Man, and Cybernetics Part-C: Applications and Reviews, Vol. 39, No. 2, pp. 133-155, March 2009.

[27] A. A. Freitas, "A review of evolutionary algorithms for data mining," in Soft Computing for Knowledge Discovery and Data Mining, O. Maimon and L. Rokach, Eds. New York: Springer-Verlag, pp. 61–93, 2007.

[28] Maulik, U., Bandyopadhyay, S., "Genetic algorithm-based clustering technique," Pattern Recognition, Vol. 33, No. 9, pp.1455-1465, 2000.

[29] Pan, H., Zhu, J., Han, D., "Genetic algorithms applied to multiclass clustering for gene expression data," Genomics, Proteomics & Bioinformatics, Vol. 1, No. 4, pp. 279-287, 2003.

[30] Julia Handl and Joshua Knowles, "An Evolutionary Approach to Multiobjective Clustering," IEEE Transactions on Evolutionary Computation, Vol. 11, No. 1, pp.56-76. Feb. 2007.

[31] Ripon, K.S.N.; Siddique, M.N.H., "Evolutionary multi-objective clustering for overlapping clusters detection," IEEE Congress on Evolutionary Computation, 2009. pp. 976 – 982, 18-21 May 2009.

[32] Weiguo Sheng, Stephen Swift, Leishi Zhang, and Xiaohui Liu, "A Weighted Sum Validity Function for Clustering with a Hybrid Niching Genetic Algorithm," IEEE Transactions on System, Man, and Cybernetics-part B: Cybernetics, Vol. 35, No. 6, pp.1156-1167. Dec. 2005.

[33] Yongguo Liu, Mao Ye, Jun Peng and Hong Wu, "Finding the Optimal Number of Clusters Using Genetic Algorithms," IEEE Conference on Cybernetics and Intelligent Systems, pp.1325 – 1330, Sept. 2008.

[34] R. M. Cole. Clustering with genetic algorithms. M.S. thesis, Univ. Western Australia, Perth, W.A., 1998.

[35] S. Bandyopadhyay, U. Maulik, and A. Mukhopadhyay, "Multiobjective genetic clustering for pixel classification in remote sensing imagery," IEEE Transactions on Geoscience and Remote Sensing, Vol. 45, No. 5, pp. 1506–1511, May 2007.

[36] S. Bandyopadhyay and U. Maulik, "An evolutionary technique based on k-means algorithm for optimal clustering in RN," Information Science, Vol. 146, pp.

221–237, 2002.

[37] S. Das, A. Abraham, and A. Konar, "Automatic clustering using an improved differential evolution algorithm," IEEE Transactions on System, Man, and Cybernetics-part A: System, Humans, Vol. 38, No. 1, pp. 218–237, Jan. 2008.

[38] L. Sarafis, P. W. Trinder, and A. M. S. Zalzala, "NOCEA: A rule-based evolutionary algorithm for efficient and effective clustering of massive high-dimensional databases," Applied Soft Computation., Vol. 7, No. 3, pp. 668–710, Jun. 2007.

[39] M. C. Naldi and A. C. P. L. F. de Carvalho, "Clustering using genetic algorithm combining validation criteria," in Proc. 15th European Symposium on Artificial Neural Networks, Bruges, Belgium, pp. 139–147, 2007.

[40] Sriparna Saha and Sanghamitra Bandyopadhyay, "A Genetic Clustering Technique Using a New Line Symmetry Based Distance Measure," 15th International Conference on Advanced Computing and Communications. pp.365 – 370, 18-21 Dec. 2007.

[41] Sriparna Saha and Sanghamitra Bandyopadhyay, "Application of a Multiseed-Based Clustering Technique for Automatic Satellite Image Segmentation," IEEE Geoscience and Remote Sensing Letters, Vol. 7, No. 2, pp.306-308, Apr. 2010.

[42] Sanghamitra Bandyopadhyay, Ujjwal Maulik, and Anirban Mukhopadhyay, "Multiobjective Genetic Clustering for Pixel Classification in Remote Sensing Imagery," IEEE Geoscience and Remote Sensing, Vol. 45, No. 5, pp. 1506-1511. May 2007.

[43] Hou Zhenjie, Ma Shuoshi, Pei Xichun, and Zhang Wei, "Segmentation and Recognition of Marrow Cells Image Based on Wavelet and Genetic Cluster," Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, Vol. 1, pp.559-563, July 30 , Aug.2007.

[44] Patrick C. H. Ma, Keith C. C. Chan, Xin Yao, and David K. Y. Chiu, "An Evolutionary Clustering Algorithm for Gene Expression Microarray Data Analysis," IEEE Transactions on Evolutionary Computation, Vol. 10, No. 3, pp. 296-314, Jun. 2006.

[45] Adam Vanya and Steven Klusener, Nico van Rooijen, and Hans van Vliet, "Characterizing Evolutionary Clusters," 16th Working Conference on Reverse Engineering. pp.227-236, 13-16 Oct. 2009.

[46] Huang, Z., "Clustering large data sets with mixed numeric and categorical values," Proceedings of the First Pacific Asia Knowledge Discovery and Data Mining Conference, Singapore: World Scientific, pp. 21–34, 1997.

[47] Huang, Z., "Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values," Data Mining and Knowledge Discovery, Vol. 2, No. 3 pp.283–304, 1998.

[48] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," Complex Systems, Vol. 9, No.2, pp. 115–148, 1995.

[49] A. Hubert, "Comparing partitions," Journal of Classification, Vol. 2, pp.193-198, 1985.

[50] Blake, C.L., Merz., C.J.,"UCI Repository of Machine Learning Databases, " Technical report, University of California, Department of Information and Computer Science, Ir-vine, CA, 1998.

[51] Jie Li, Xinbo Gao, and Licheng Jiao, "A New Feature Weighted Fuzzy Clustering Algorithm," Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, Lecture Notes in Computer Science, vol. 3641, pp. 412–420, 2005.

[52] Han, J. and Kamber, M.. Data Mining – Concepts and Techniques. The Morgan Kaufmann Series in Data Management Systems, March 2006.

[53] Aristidis Likas, Nikos Vlassis, JakobJ. Verbeek, "The global k-means clustering algorithm," Pattern Recognition pp.451–461, 36 (2003).

[54] Jianbo Shi and Jitendra Malik, "Normalized Cuts and Image Segmentation.," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, No. 8, pp.888-905,Aug. 2000.

硕士期间的学术成果

[1] Licheng Jiao, Maoguo Gong, Shuang Wang, Biao Hou, **Zhi Zheng**, Qiaodi Wu. Natural and Remote Sensing Image Segmentation Using Memetic Computing. IEEE Computational Intelligence Magazine, Vol. 5, No. 2, May 2010: 78–91. SCI: 000276821200006, EI: 20101712890441.

[2] **Zhi Zheng**, Maoguo Gong, Jingjing Ma, Licheng Jiao, Qiaodi Wu. Unsupervised Evolutionary Clustering Algorithm for Mixed Type Data. In: Proceedings of The 2010 IEEE Congress on Evolutionary Computation, CEC2010, Barcelona, Spain, July 18-23, 2010. EI:11568007.

[3] 吴巧娣, 公茂果, 焦李成, **郑智**, 左益. 基于有向图编码的 Memetic 图像分割方法. 全国人工智能大会, 北京, 2010 年 8 月 20 - 23 日.

[4] **Zhi Zheng**, Maoguo Gong, and Jingjing Ma. Hybrid Evolutionary Clustering Algorithm with Multi-population and Graph-based Search. IEEE Symposium on Computational Intelligence in Memetic Computing, 2011 - Paris, France. Accepted. EI 检索源.

[5] Qiaodi Wu, Maoguo Gong, Jingjing Ma, **Zhi Zheng** and Licheng Jiao. A Neighborhood-based Difference Image Computing Approach for Change Detection in SAR Images. 2011 3rd International Conference on Machine Learning and Computing (ICMLC 2011), Singapore, February 26-28, 2011. Accepted. EI 检索源.

[6] Maoguo Gong, **Zhi Zheng**, Meng Ma, and Licheng Jiao. Global Clustering Algorithm with Density Exploring Distance Measure. 已完成.

[7] Maoguo Gong, **Zhi Zheng**, Licheng Jiao. Coevolutionary Memetic Algorithm for Clustering. 已完成.