算法	概括	优缺点
AGNES	典型的凝聚式层次聚类	
DIANA	典型的划分式层次聚类	划分式层次聚类的复杂度比凝聚式的大得多，所以较为少用。
CURE	用到了kd-tree跟heap。 合并两个类的时候，先选若干well-scattered的点。从中挑出离中心最远的点，之后再挑离该点最远的点...如此得到一堆代表点，基于这些点去做层次聚类。 对于大数据：先随机抽样，再对样本进行分区，然后对每个分区局部聚类，最后对局部聚类进行全局聚类。	时间上最坏是：$O(n^2log(n))$ 若数据维度较小，可以降到：$O(n^2)$ 空间复杂度是：$O(n)$
ROCK	1.生成相似度矩阵。 2.根据相似度阈值得到邻居矩阵-A。 3.计算链接矩阵-L=A x A 4.计算相似性的度量（Goodness Measure），将相似性最高的两个对象合并。（用到了链接矩阵） ROCK算法首先用相似度阀值和共同邻居的概念，从给定的数据相似度矩阵中构建一个稀疏图，然后对该稀疏图使用分层聚类算法进行聚类	CURE算法不能处理枚举型数据，而ROCK算法是在CURE基础之上适用于枚举数据的聚结分层聚类算法。
Chameleon	1.由数据集构造成一个K-近邻图G_k 2.通过图的划分算法将图G_k划分成大量的子图，每个子图代表一个初始子簇 3.凝聚式层次聚类	Chameleon跟CURE和DBSCAN相比，在发现高质量的任意形状的聚类方面有更强的能力。但是，在最坏的情况下，高维数据的处理代价可能对n个对象需要$O(n^2)$的时间。
BIRCH	用到了$CF < n, LS, SS >$ CF-tree类似于B-树，有两个参数：内部节点平衡因子B，叶节点平衡因子L，簇半径阈值T。 1.自上而下选择最近的子节点 2.到达子节点后，检查最近的元组CF_i能否吸收此数据点 若能吸收，则更新CF值 否则考虑是否可以添加一个新的元组 如果可以，则添加一个新的元组 否则，分裂最远的一对元组，作为种子，按最近距离重新分配其它元组 3.更新每个非叶节点的CF信息，如果分裂节点，在父节点中插入新的元组，检查分裂，直到root	BIRCH优点： 1.节省内存。叶子节点放在磁盘分区 2. 在对树进行插入或查找操作很快。 3.一遍扫描数据库即可建树。 4.可识别噪声点。 5. 可作为其他聚类算法的预处理过程 BIRCH缺点： 1.结果依赖于数据点的插入顺序。 2.对非球状的簇聚类效果不好。 3.对高维数据聚类效果不好。 4.最后得出来的簇可能和自然簇相差很大。 5.在整个过程中算法一旦中断，一切必须从头再来。 6.局部性
*BUBBLE	把BIRCH算法的中心和半径概念推广到普通的距离空间	
*BUBBLE-FM	通过减少距离的计算次数，提高了BUBBLE算法的效率	
Probabilistic agglomerative clustering	距离度量用： $dist(C_1, C_2) = -log((P(C_1 \cup C_2))/(P(C_1)P(C_2)))$ 如果dist小于零，则合并两个簇。	易于理解 一般跟其他凝聚式层次聚类算法的效率差不多 但是：it outputs only one hierarchy with respect to a chosen probabilistic model; it cannot handle the uncertainty of cluster hierarchies.