# DRR:一种多维案例检索优化算法研究

汪璟玢 胡 烜
(福州大学数学与计算机科学学院 福州 350108)

**摘 要** 基于本体的案例检索系统中,由于数据库中的案例数量随着时间的推移而成倍增加,案例检索的效率不断降低。提出了一种多维案例检索算法——DRR,该算法通过将多维空间案例点降维成二维空间点,利用一个二维空间点来代表类案例点组成的集合,并对此二维空间点建立R树空间索引,通过两级检索的方法,加速了检索效率和准确率。实验证明,该方法不仅提高了案例检索的准确率,还极大地提高了案例检索的效率。

**关键词** 案例检索,R树索引,相对点,相对向量
**中图法分类号** TP391.4 **文献标识码** A

## DDR:A Multidimensional Case Retrieval Optimization Algorithm

WANG Jing-bin HU Xuan
(College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China)

**Abstract** In ontology-based case retrieval system, because the number of cases in the database can double increases with time, the efficiency of case retrieval continues lower. This paper presented a multi-dimensional case retrieval algorithm-the DRR(Dimensionality Reduction of R-Tree), the algorithm dimensionality reduces the case point of the multi-dimensional space to two-dimensional space point, uses a two-dimensional space points to represent the collection on behalf of the class case points, and biulds R-tree spatial index for this two-dimensional space point accelerates the retrieval efficiency and accuracy by the two search methods. It is proved that the method not only improves the accuracy of case retrieval, but also greatly improves the efficiency of case retrieval.

**Keywords** Case retrieval, R-tree index, Relative point, Relative vector

## 1 引言

基于案例推理(Case Based Reasoning,CBR)的方法是一种通过访问知识库中过去对同类问题的求解过程与结果,并从新案例中获取新知识以适应当前问题的策略,是人工智能领域的一项重要推理方法[1]。在基于案例推理的系统中,案例检索的质量关系到整个系统的质量,即案例检索的效率及案例检索结果的相关度[2]。案例库中需要存储大量的案例来满足用户对知识的需求,然而随着案例库中案例的增长,案例检索效率会逐渐降低,这种现象称为沼泽现象[3]。在以往的案例检索中,每一次的案例比对相似度都是按需一次遍历,即每一次的检索都是全文的匹配搜索,这在案例库逐渐增长到一定规模时查询效率就会降低。耿焕同(2005)等[4]针对这种情况,提出了将聚类算法应用于案例库的维护。冯征(2008)[5]提出了一种基于K-Means的聚类算法。刘长征(2010)等[6]提出特征加权C均值聚类算法(WF-C-means)和基于聚类的案例检索方案来建立索引,由于通过C均值聚类算法调整所有属性的权值都包含在差异定义中,检索和新案例相似的案例所采取的差异定义变得非常精确和客观。乔丽(2011)等[7]提出一种改进K-means聚类的案例检索算法,解决了由噪声引起的聚类误差,通过收集用户的相关反馈信息对案例的能力进行评价,并将案例的能力作为选取样本案例的规则。

对于K-means聚类,样本数K的选择尤为重要,如果过小,则每个簇集会很大,如果过大,则样本案例很多,这些都会对查询效率造成影响;而噪声数据对于K-means聚类的查询准确性有很大的影响。

针对以上问题,本文提出了一种多维案例优化算法DRR:首先将案例库中的多维案例空间点通过降维计算,进行二维空间聚类,再对二维空间聚类进行R-Tree索引;当前故障也通过降维计算,得到当前故障的二维表示;通过R树索引的查找,可以帮助我们快速地定位当前故障的二维聚类所在,然后通过KNN算法在这个二维空间聚类中的所有多维案例中找到与当前故障最接近的多维案例。

## 2 多维案例检索优化:DRR算法

### 2.1 DRR算法模型与主要思想

如图1所示,在基于DRR算法的检索模型中,通过将多维案例空间点降维成二维案例空间点,对大量的案例进行聚

到稿日期:2012-10-19 返修日期:2012-12-20 本文受空间数据挖掘与信息共享教育部重点实验室开放研究基金(201006),2011年福建省科技拥军基金(JG2011005),2012年福建省科技拥军基金(JG2012003),2012年福建省自然科学基金(2012J01168)资助。
汪璟玢(1973—),女,硕士,副教授,主要研究方向为空间数据管理、网络数据库、智能技术,E-mail:wjbcc@263.net;胡 烜(1985—),男,硕士,主要研究方向为空间数据管理、智能技术。

类并建立 R 树索引。对于当前故障检索，首先对其进行降维处理，通过 R-Tree 索引查找，得到与当前故障的二维表示最接近的案例聚类(中间结果集)；然后针对中间结果集进行进一步的过滤精选，通过 K-NN 相似度计算方法获得与当前故障相似度最高的多维案例并输出查询结果。对于新案例的学习，相较于以往的案例库学习，DRR 算法无需进行案例的修正，只要案例库中的案例不与新案例完全一样，就添加新案例；而对于 R-Tree 索引，如果新案例所属的案例聚类已存在，则只需在聚类中加入新案例；若新案例所在聚类在索引中不存在，则对其进行降维并重新建立 R-Tree 索引。

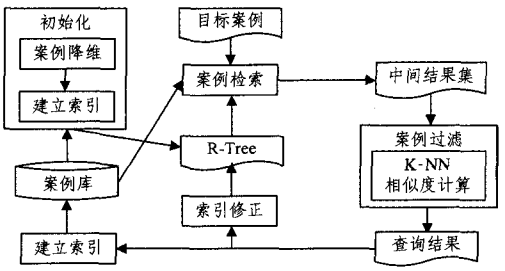


图 1　DRR 算法模型图

## 2.2　案例及案例库的表示

在基于案例推理系统中，设案例库 $CS=(c_1,c_2,c_3,\cdots,c_n)$ 为一个由 $n$ 个案例组成的非空有限集合，$\exists c_i(0\leqslant i\leqslant n)$，$c_i\in CS$。

案例库 $CS$ 中，案例通常采用基于特征向量的表示方法。可设案例 $c=(f_1,f_2,\cdots,f_i,\cdots,f_m)$ 为一个非空有限集合，其中 $f_i(1\leqslant i\leqslant m)$ 是 $c$ 的一个特征项，特征项用来描述案例的某一属性。

**定义 1**　$\forall a,b$，若 $a,b\in CS$，则案例空间两点的距离为欧拉距离 $R_{ab}$：$R_{ab}=\sqrt{\sum(a_i-b_i)^2}\ (1\leqslant i\leqslant m)$。

**定义 2**　$\forall a,b$，若 $a,b\in CS$，则案例空间两点的向量夹角为 $\theta_{ab}$：$\theta_{ab}=\cos^{-1}\dfrac{\sum a_i\cdot b_i}{\|b\|\cdot\|b\|}\ (1\leqslant i\leqslant m)$。

假设一个 $m$ 维的全局参考点为 $O(O_1,O_2,O_3,\cdots,O_m)$，一个全局参考向量为 $\vec{N}(N_1,N_2,N_3,\cdots,N_m)$。根据定义 1 和定义 2，计算案例空间中的点 $c$ 与 $O$ 的相对距离 $R_{cO}$(本文简记为 $R_c$)，点 $c$ 与参考向量 $\vec{N}$ 之间的夹角 $\theta_{cO}$(本文简化记为 $\theta_c$)，则 $m$ 维的案例点 $c$ 可以表示成二维空间点 $C(R_c,\theta_c)$，如图 2 所示。

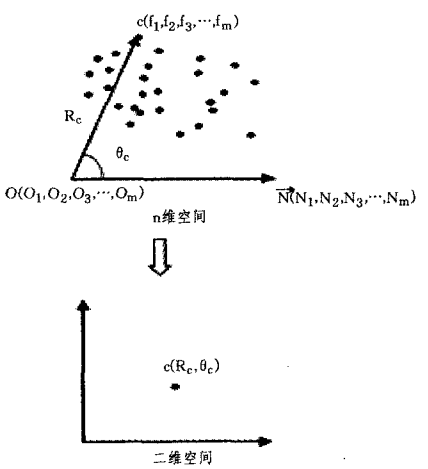


图 2　案例空间点降维

在 $m$ 维案例空间点上，会有若干与全局参考点 $O(O_1,O_2,O_3,\cdots,O_m)$ 的相对距离相同，且与全局参考向量 $\vec{N}$ $(N_1,N_2,N_3,\cdots,N_m)$ 具有相同夹角的点，这样就得到了一个二维空间的案例聚类。

**定义 3**　设二维空间 $S$ 中的一个点 $D(R_d,\theta_d)$ 表示一个点集 $(d_1,d_2,\cdots,d_i,\cdots,d_n)$。$d_i\in CS$，其中 $d_i$ 的二维空间表示均为 $D(R_d,\theta_d)$。

**定义 4**　设目标案例 $e(e_1,e_2,e_3,\cdots,e_m)$，$\forall a(a_1,a_2,a_3,\cdots,a_m)a\in CS$；$a(a_1,a_2,a_3,\cdots,a_m)$ 的二维空间表示为 $A(R_a,\theta_a)$，$e(e_1,e_2,e_3,\cdots,e_m)$ 的二维空间表示为 $E(R_e,\theta_e)$。若在二维空间 $S$ 中 $E(R_e,\theta_e)$ 与 $A(R_a,\theta_a)$ 最接近，则说明案例点 $a$ 是目标案例 $e$ 在案例空间 $CS$ 中的相似度最高的点。

证明：根据图 2，在 $m$ 维案例空间 $CS$ 中由参考点 $O(O_1,O_2,O_3,\cdots,O_m)$、当前故障点 $e(e_1,e_2,e_3,\cdots,e_m)$、相似点 $a(a_1,a_2,a_3,\cdots,a_m)$ 3 点确定了一个平面，并形成了一个三角形 $\triangle Oae$。其中 $R_e$ 是 $e$ 与 $O$ 之间的相对距离(三角形边 $Oe$ 的长度)，$R_a$ 是 $A$ 与 $O$ 之间的相对距离(三角形边 $Oa$ 的长度)，而夹角 $\triangle\theta$ 可以通过 $\vec{a}$ 与 $\vec{N}$ 之间的夹角 $\theta_a$ 和 $\vec{e}$ 与 $\vec{N}$ 之间的夹角 $\theta_e$ 相减得到：$\triangle\theta=|\theta_a-\theta_e|$。

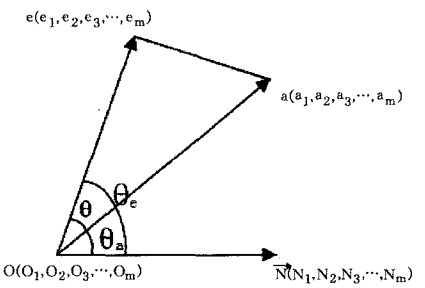


图 3　案例点在 $m$ 维案例库 $CS$ 中的表示

要想证明当前故障 $e$ 与案例点 $a$ 相似度最高，即需计算 $e$ 与 $a$ 在案例空间 $CS$ 之间的欧拉距离 $R_{ea}$ 最小(三角形边 $ea$ 的长度)。

由余弦定理可得，在 $m$ 维案例空间 $CS$ 中：

$$R_{ea}=\sqrt{R_e^2+R_a^2-2R_aR_e\cos\Delta\theta}$$

$$R_{ea}=\sqrt{R_e^2+R_a^2-2R_aR_e\cos\Delta\theta}$$

$$\Rightarrow R_{ea}=\sqrt{R_e^2+R_a^2-2R_aR_e+2R_aR_e-2R_aR_e\cos\Delta\theta}$$

$$\Rightarrow R_{ea}=\sqrt{(R_a-R_e)^2-2R_aR_e(1-\cos\Delta\theta)}$$

$$\Rightarrow \lim_{\substack{R_a\to R_e\\ \Delta\theta\to 0}}\sqrt{(R_a-R_e)^2-2R_aR_e(1-\cos\Delta\theta)}$$

由推导公式可得，当当前故障 $e$ 的 $R_e$ 无限趋近于案例库中点 $a$ 的 $R_a$，并且 $e$ 与 $a$ 之间的夹角 $\Delta\theta$ 无限趋近于 0(即 $\theta_e$ 无限趋近于 $\theta_a$)时，在二维空间 $S$ 中的表示即为 $E(R_e,\theta_e)$ 与 $A(R_a,\theta_a)$ 这两个点无限接近，则此时获得极限值 $R_{ea}$ 将无限趋近于 0，即证明此时案例库 $CS$ 中 $e$ 与 $a$ 之间的相似度最高。

## 2.3　案例库建立 R-Tree 索引

**定义 5**　设点 $D(R_d,\theta_d)$ 是二维空间 $S$ 中的一个点，点 $D$ 的最小外包矩形 $MBR$ 为 $M_d$，其中 $M_d$ 为以 $(R_d-\Delta r,\theta_d-\Delta\theta)$，$(R_d+\Delta r,\theta_d+\Delta\theta)$ 为两对角点的矩形范围。

点 $D(R_d,\theta_d)$ 在 R-Tree 索引中的表示为一个叶节点 $D'(ID,M_d)$，其中 ID 是案例点的标识 ID，$M_d$ 是点 $D$ 的最小外

包矩形，对空间中的所有点建立 R 树索引。

### 2.4 DRR 算法描述

#### 2.4.1 案例检索

**定义 6** 当前故障 $e(e_1,e_2,e_3,\cdots,e_m)$ 在二维空间 $S$ 中的表示为 $E(R_e,\theta_e)$，点 $E$ 的查询范围矩形 $M_e$ 是以 $(R_e-\Delta sr,\theta_e-\Delta s\theta)$、$(R_e+\Delta sr,\theta_e+\Delta s\theta)$ 为两对角点的矩形，其中 $\Delta sr,\Delta s\theta$ 为点 $E$ 的查询范围，查询矩形越大，则查询的准确性越高。设二维空间 $S$ 中的案例点集 $D$ 在 R 树中的表示为 $M_d$，若 $M_d$ 与 $M_e$ 相交，则说明案例点集 $D$ 为点 $E$ 在二维空间 $S$ 中的相似点集，由此查询得到中间结果集 $R(R_1,R_2,R_3,\cdots,R_n)$。

假设 R 树索引的节点为 $T$，新查找当前故障为 $e(e_1,e_2,e_3,\cdots,e_m)$，查找同 $e$ 相似度最高的案例集的算法的描述如下：

1)计算 $e$ 与全局参考点 $O(O_1,O_2,O_3,\cdots,O_m)$ 的相对距离 $R_e$，计算 $\vec{e}$ 与全局参考向量 $\vec{N}(N_1,N_2,N_3,\cdots,N_m)$ 的夹角 $\theta_e$，所以案例 $e$ 在二维空间 $S$ 中的映射点是 $E(R_e,\theta_e)$。

2)计算 $E(R_e,\theta_e)$ 在 R 树索引中的查询范围矩形 $M_e$。

3)如果 $T$ 不是叶节点，那么检查 $M_e$ 是否同 $M_t$ 相交，如果相交，那么递归地检查 $E$ 是否同 $T$ 的子结点相交；如果不相交，则放弃该结点的查找。

4)如果 $T$ 已经是叶节点，则判断 $T$ 中的所有记录是否同 $M_e$ 相交。这样便可以查找到 R 树中所有同 $E$ 相交的记录，并且查找是沿着树的分支向下查找，不会去遍历树中的每一条记录。

5)找到与 $M_e$ 相交的所有中间结果集 $R(R_1,R_2,R_3,\cdots,R_n)$。

#### 2.4.2 案例过滤

对中间结果集 $R$ 进行过滤，计算 $e(e_1,e_2,e_3,\cdots,e_m)$ 与中间结果集中所有案例点的相似度，并得到相似度最高的案例点。

中间结果集 $R$ 可能出现如图 4 所示的情况。

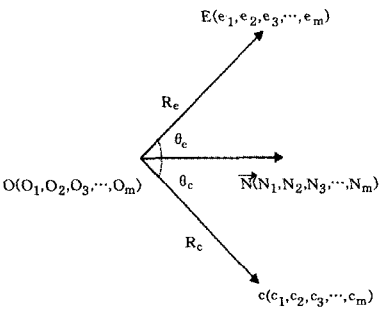


图 4 案例检索中间结果集

此时，对于案例 $c$，$R_e-\Delta sr\geqslant R_c\leqslant R_e+\Delta sr,\theta_e-\Delta s\theta\geqslant\theta_c\leqslant\theta_e+\Delta s\theta$，案例 $c$ 是中间结果集 $R$ 中的一个案例点，但是案例 $c$ 并不是当前故障 $e$ 的相似案例。由于降维而导致二维空间点集中的案例点差异变小，从而得到的中间结果集就会存在类似 $c$ 案例的冗余案例点，这时就需要对中间结果集进行过滤，计算 $e(e_1,e_2,e_3,\cdots,e_m)$ 与中间结果集中所有案例点的相似度，并得到相似度最高的案例点。

本文采用 K 最近邻（K-Nearest Neighbor Algorithm，KNN）算法来进行相似度计算。它将案例的特征矢量看作高维空间中的点，然后在问题空间中寻找与当前故障相匹配的点，将超过相似度阈值的案例返还给用户，其一般过程描述如下：

输入待检索当前故障 $e$，输出与案例 $e$ 相似的已有案例对象。结果集 $R$ 中包含 $k$ 个案例，各案例由 $m$ 个属性描述，即 $x_i=\{x_{i1},x_{i2},\cdots,x_{ij},\cdots,x_{im}\}$，$i=1,2,\cdots,k;j=1,2,\cdots,m$，计算已有案例 $x_i$ 与当前故障 $e$ 之间的欧拉距离 $d(x_i,e)$：

$$d(x_i,e)=\sqrt{\Sigma(x_{ij}-e_j)^2}\ (1\leqslant j\leqslant m)$$

基于此，易知已有案例 $x_i$ 与当前故障 $e$ 之间的相似性：

$$\mathrm{SIM}(x_i,e)=1-d(x_i,e)$$

#### 2.4.3 案例学习与 R-Tree 索引修正

对于新案例 $e(e_1,e_2,e_3,\cdots,e_m)$，如果案例库 $CS$ 中存在案例 $c\ (c_1,c_2,c_3,\cdots,c_m)$，并且 $c_i=e_i(1\leqslant i\leqslant m)$，则说明新案例在案例库中已存在，无需新增案例与索引修正。

若不存在以上情况，则需对新案例进行案例学习：

1)将新案例 $e$ 插入到案例库 $CS$ 中。

2)计算新案例 $e$ 与全局参考点 $O$ 的相对距离 $R_e$、点 $e$ 与全局参考向量 $\vec{N}$ 之间的夹角 $\theta_c$，则 $m$ 维的案例点 $e$ 可以表示成二维空间点 $E(R_e,\theta_e)$。

3)在二维空间 $S$ 中，对于新增的点 $E(R_e,\theta_e)$，判断此点在空间中是否已存在，若存在，说明此聚类已存在，则只需在此聚类中新增案例点的标识 ID；若空间中不存在此点，说明新增了一个聚类点，则需要重新调整 R-Tree 索引。

## 3 实验

将 DRR 算法进行仿真实验，将算法与传统检索方法、基于粗糙集的 K-means 算法进行比较，案例库采用某故障诊断系统数据，测试数据集大小分别为 30000、50000、80000、100000、200000、300000、500000，单位为个。算法通过 eclipses 实现，JVM 最大内存 512M。

### 3.1 参数设置

案例空间维度 $m=10$，参考点 $O(O_1,O_2,O_3,\cdots,O_m)$ 为 0 (0,0,0,⋯,0)，参考向量为 $\vec{N}(1,0,0,\cdots,0)$，二维空间点的外包矩形大小为 $\Delta r=1,\Delta\theta=1$，案例相似度阈值为 0.9，而当前故障的查询矩形范围为 $\Delta sr=60,\Delta s\theta=30$。

### 3.2 实验结果与分析

#### 3.2.1 查准率实验

实验中测试案例的标准结果集已给出（即查询案例库中与测试案例相似度大于 0.9 的案例个数），查询结果越接近标准结果集，则说明查询的准确率就越高。从实验可以看出，K-means 算法对于噪声数据比较敏感，在查询 50000 的数据集时，由于噪声数据造成聚类的偏差从而影响了准确率。而噪声数据对于 DRR 算法查询结果的几乎没有影响，并且 DRR 算法的结果集准确率高，稳定性比 K-means 算法要高，如表 1 所列。

表 1 查准率实验

| 案例库大小 | 标准结果集 | K-means | | DRR | |
|---|---|---|---|---|---|
| | | 结果 | 准确率(%) | 结果 | 准确率(%) |
| 30000 | 237 | 216 | 91.13 | 236 | 99.57 |
| 50000 | 410 | 308 | 75.12 | 405 | 98.78 |
| 80000 | 662 | 641 | 96.82 | 656 | 99.09 |
| 100000 | 829 | 807 | 97.34 | 817 | 98.55 |
| 200000 | 1630 | 1434 | 87.97 | 1606 | 98.52 |
| 300000 | 2585 | 2235 | 86.46 | 2547 | 98.64 |

#### 3.2.2 查询效率实验

从表 2 和图 5 可以看出，K-means 算法和 DRR 算法的查

（下转第 125 页）

and Distributed Computing,1992,12(16):276-289
[2] Baxter J, Patel J H. The LAST algorithm: a heuristic-based static task allocation algorithm[C]//Proceedings of the 1989 International Conference on Parallel Processing. 1989:217-222
[3] McCreary C, Gill H. Automatic determination of grain size for efficient parallel processing[J]. Communication of ACM, 1989, 32(9):1073-1078
[4] Yang Y, Chen K. Temporal data clustering via weighted clustering ensemble with different representations[J]. IEEE Transactions on Knowledge and Data Engineering, 2011, 23(2):307-319
[5] Kim S J, Browne J C. A General Approach to Mapping of Parallel Computation upon Multiprocessor Architectures[C]//Proceedings of International'l Conference on Parallel Processing. 1988:1-8
[6] Wu M Y, Gajski D D. Hypertool: A programming aid for message-passing systems[J]. IEEE Transations on Parallel and Distributed Systems, 1990, 1(3):330-343
[7] Yang T, Gerasoulis A. DSC: scheduling parallel tasks on an unbounded number of processors[J]. IEEE Transactions on Parallel and Distributed Systems, 1994, 5(9):951-967
[8] Sarkar V. Partitioning and scheduling parallel programs for multiprocessors[M]. MIT Press, Cambridge, MA, 1989
[9] Lam Y M. Integrated task clustering , mapping and scheduling for heterogeneous computing systems[J]. International Journal of Computer Science&Information Technology, 2012, 4(1):127-146
[10] Sih G C, Lee E A. A Compile-time scheduling heuristic for interconnection-constrained heterogeneous processor architectures [J]. IEEE Transactions On Parallel and Distributed System, 1993, 4(2):175-187
[11] Koziris N, Romesis M, Tsanaksa P, et al. An efficient algorithm for the physical mapping of clustered task graphs onto multiprocessor architectures[C]// Proceeding Eighth Eurmicro Workshop Parallel and Distributed Processing. 2000:406-413
[12] Bambha N K, Bhattacharyya S S. Joint Application Mapping/Interconnect Synthesis Techniques for Embedded Chip-Scale Multiprocessors [J]. IEEE Transactions On Parallel and Distributed System, 2005, 16(2):99-112

(上接第 88 页)

询效率比传统的方法高很多。K-means 算法之所以会比传统算法效率高，是因为其对案例库进行了聚类。DRR 算法采用二级检索的方式，能够快速地锁定与目标案例接近的案例聚类，从而在效率上比传统算法和 K-means 算法都要快得多。传统检索方式每次都将所有数据读取内存进行查询比较，随着案例空间的数据逐渐增长，查询数据的时间也逐步增长，当数据到达一定数量时则会发生内存溢出。

表 2 几种算法检索效率比较表

| 案例库大小 | 传统算法(ms) | K-means(ms) | DRR(ms) |
|---|---|---|---|
| 50000 | 29.33 | 15 | 11 |
| 80000 | 31.78 | 16 | 13 |
| 100000 | 37 | 16 | 15 |
| 200000 | 67.67 | 51.67 | 36.67 |
| 300000 | 93.33 | 73.25 | 53 |
| 500000 | 内存溢出 | 内存溢出 | 235 |

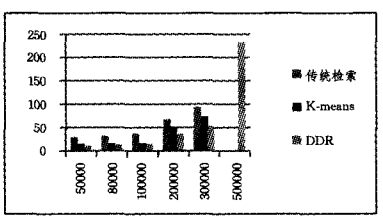


图 5 查询结果柱状图

K-means 和 DRR 算法将案例库中的数据读入，建立索引，以后的每次查询都是基于索引的查询，因此查询会比较快；K-means 索引建立的基础是整个案例库的数据，所以当数据到达一定数量的时候也会发生内存溢出；DRR 算法先对案例库中的记录通过降维的手段进行二维聚类，之后针对二维聚类结果建立 R 树索引，这样在内存中 R 树索引的空间比 K-means 算法的索引要小很多。因此在数据集很大的时候，DRR 算法依然可以实现查询，不仅不会出现内存溢出情况，而且能很好地解决 K-means 算法中由于样本点过大而造成的搜索效率下降问题，提高了大数据量案例库的检索效率。

**结束语** 基于 DRR 算法的案例检索算法将原案例库中的点降维计算映射成一个二维空间点表示，从而实现原案例库记录的二维空间聚类，再针对二维空间聚类建立 R 树索引；当前故障检索时，先通过 R 树索引找到与之最接近的案例二维聚类，再通过 KNN 算法进行精确的相似度计算，从而返回最接近新任务的案例。该算法通过降维聚类、二级检索的方法，加快了检索的效率，并且案例的二维聚类是基于业务无关的，避免了人工分类造成的分类错误。下一步工作将对算法进一步改进，考虑案例中特征项的权重对检索的影响，以提高案例检索的准确率和效率。

## 参 考 文 献

[1] Aamodt A, Plaza E. Case—based reasoning: Foundational issues, methodological variations, and system approaches[J]. AI Communications, 1994, 7(1):39-59
[2] Gu Yin-shan, Hua Qiang, Zhan Yan, et al. Case-base maintenancebased on representative selection for I-NN algorithm[C]// 2003 International Conference on Machine Learning and Cybemetics. 2003:2421-2425
[3] Derere L. Case-based reasoning: Diagnosis of faults in complex-systems throughreuse of experience[C]// Proceedings of InternationalTest Conference. 2000:77-105
[4] 耿焕同，肖明军，邹翔，等. 聚类算法在范例库维护中的应用研究明[J]. 计算机工程，2005，31(12)：166-168
[5] 冯征. 一种基于粗糙集的 K-Means 聚类算法[J]. 计算机工程与应用，2008，44(20)：141-142
[6] 刘长征，董冬. 基于特征加权 C 均值聚类算法的案例索引和检索[J]. 计算机应用与软件，2010，27(2)：111-114
[7] 乔丽，姜慧霖. 一种 k-means 聚类的案例检索算法[J]. 计算机工程与应用，2011，47(4)：185-187
[8] Guttman A. R-Trees: A dynamic index structure for spatialsearching[C]// Proc. ACM S IGMOD Conf, Ann. Meeting. 1984:47-57