基于改进差别矩阵的属性约简增量式更新算法

龙 浩[1,2] 徐 超[3]

(中国矿业大学计算机科学与技术学院 徐州 221008)[1]

(徐州工业职业技术学院信息管理技术学院 徐州 221000)[2] (武汉大学计算机学院 武汉 430072)[3]

摘 要 针对目前基于差别矩阵的属性约简算法需要耗费大量的时间和空间,粗糙集中求属性核和属性约简更新效率低以及有关属性约简的增量式更新算法目前还比较少等问题,提出了一种基于改进差别矩阵的属性约简增量式更新算法。该算法在更新差别矩阵时,仅须插入某一行及某一列,或删除某一行并修改相应的列,因而可有效地提高核和属性约简的更新效率。然后在分析新增对象 x 与原决策系统对象的关系的基础上,给出了属性约简增量更新算法。理论与实验分析表明,提出的算法提高了属性约简的更新效率,明显降低了时间和空间复杂度。

关键词 差别矩阵,属性约简,粗糙集,决策系统

中图法分类号 TP181 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.6.053

Incremental Updating Algorithm for Attribute Reduction Based on Improved Discernibility Matrix

LONG Hao[1,2] XU Chao[3]

(School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221008, China)[1]

(Information Management Department, Xuzhou College of Industrial Technology, Xuzhou 221000, China)[2]

(School of Computer, Wuhan University, Wuhan 430072, China)[3]

Abstract In order to solve the problem that the attribute reduction algorithm based on discernibility matrix spends a lot of time and space and the efficiency of the attribute core and the attribute reduction update of the rough set are slow, what is more, it lacks the incremental updating algorithm for attribute reduction, this paper proposed an incremental updating algorithm for attribute reduction based on the discernibility matrix. When the algorithm updates the discernibility matrix, it only needs to insert a row and a column, or delete a row and modify the corresponding column, which can effectively improve the updating efficiency of core and attribute reduction. We analyzed the relationship of the new object x with the original decision system object, giving out the updating algorithm of the attribute reduction increment. Theoretical and experimental analysis shows that the proposed algorithm can improve the updating efficiency of attribute reduction, reducing the time and space complexity significantly.

Keywords Discernibility matrix, Attribute reduction, Rough set, Original decision system

1 引言

在基于粗糙集理论的知识获取研究中,属性约简是最核心的组成部分之一。属性约简的结果会对最终形成的规则产生直接的影响,许多学者已对属性约简的算法进行了大量的研究[1],但这些研究几乎都是针对静态数据的,而在实际应用中,数据库往往都是动态变化的,因此许多研究者建议数据库知识发现算法应该是增量式的。增量式的规则获取算法和增量式的属性约简算法已经开始得到研究。目前的增量式属性约简算法大致可分为两大类:一类是获取属性约简簇集的增量式算法;另一类是获取一个属性约简的增量式算法[2,3]。这两类算法都使用了对象之间分辨属性来处理,属于代数观下的属性约简方法。一些文献提出的一些增量式属性约简方法虽然能够得到信息系统的最小约简,但只能求出绝对约简(不包含决策属性),而在实际应用中,大部分数据都是具有决策属性的[4,5]。有的文献给出了一种基于 Skowron 分辨矩阵的属性约简的增量模型,但不能保证得到一个 Pawlak 约简,也不能处理不相容决策表。有的文献中给出的算法可以处理不相容决策表,得到一个完备的 Pawlak 约简,但不能得到最小约简。总之,快速的约简算法及其增量版本问题仍是粗糙集理论的研究热点之一[6-9]。

现在已经有很多学者提出了属性约简的增量算法,很多高效属性约简算法都是从核开始的,而求解核的主要目标是求解属性约简,因此如何以核的高效更新算法为基础进行高效属性约简算法研究成为了本文的主要目标。本文采用以改进差别矩阵为基础的属性约简增量式更新算法,首先提出对

到稿日期:2014-06-17 返修日期:2014-08-17 本文受国家自然科学基金重点项目(91118003),国家自然科学基金面上项目(61170022),江苏省高校"青蓝工程"优秀青年骨干教师培养对象资助。

龙 浩(1984—),男,讲师,主要研究方向为粗糙集、粒计算、软件工程、数据挖掘,E-mail:longhhao@163.com(通信作者);徐 超(1980—),男,博士,副教授,主要研究方向为粗糙集、软件工程、并行分布式处理、可信软件、嵌入式系统等。

其差别矩阵进一步的改进。改进后的算法可通过快速更新差别矩阵，在核增量式更新算法的基础上，有效地利用原有的属性约简进行属性约简增量更新，因而可提高属性约简的更新效率。该算法在更新差别矩阵时，仅须插入某一行及某一列，或删除某一行并修改相应的列，因而可有效地提高核和属性约简的更新效率。

2 粗糙集理论的相关定义

信息系统的数据通常以关系表的形式表示，故也简称为信息表。关系表的行对应对象，列对应对象的属性，对象的信息通过指定对象的各属性值来表达。容易看出，一个属性对应一个等价关系，一个表可以看作是定义的一个等价关系族，即知识库，知识约简可以转化为属性约简。

定义 1 在信息系统 $T=(U,A,V,f)$ 中，如果 $A=C\cup D$，且 $C\neq\emptyset,D\neq\emptyset,C\cap D=\emptyset$，则称 T 是一个决策系统（简称决策表），C 是条件属性集合，D 是决策属性集合。

属性集合 A 分为条件属性集 C 和决策属性集 D。对于 $B\subseteq A$，无差别关系 $IND(B)$ 定义为 $IND(B)=\{(x,y)\in U^2 \mid \forall a\in B, f(x,a)=f(y,a)\}$。通过 $IND(B)$ 将 U 划分为若干个类 E_i，$1\leqslant i\leqslant |U/IND(B)|$。

假设条件属性集合 C 中有 m 个属性：$C_1,C_2,\cdots,C_m$，其值域为有限离散集合，并用 $card(C)$ 或 $|C|$ 表示集合 C 的基，即集合的元素个数。决策属性只有一个 D，其取值范围为 $1,2,\cdots,k$。由 D 导出的等价类构成 U 的一个划分 $U/D=\{\psi_1,\psi_2,\cdots,\psi_k\}$，其中，$\psi_i=\{x\in U \mid f(x,D)=i\}$，$i=1,2,\cdots,k$。

定义 2 在信息系统中，若一些对象具有相同的条件属性值而具有不同的决策属性值，则称这类对象为不一致的，否则为一致的。即有 U 中的两个对象 $x\in U,y\in U$，若（$\forall a\in C) f(x,a)=f(y,a)$ 且 $f(x,D)\neq f(y,D)$，则称 x 和 y 为不一致的（C-不一致的），否则称 x 和 y 为一致的（C-一致的）。

定义 3 设 $X\subseteq U$ 为论域的一个子集，$P\subseteq C$，X 的关于 P 的下近似为 $\underline{PX}(U)=\{x\in u:[x(U)]_P\subseteq X\}$；其中，$[x(U)]_P$ 表示 U 中所有与 x 在关系 $IND(P)$ 下是等价的元素构成的集合。

定义 4 设 $P\subseteq C$，对划分 $U/D=\{\psi_1,\psi_2,\cdots,\psi_k\}$ 的 P-近似精度为 $\gamma_P(U)=\sum_{i=1}^{k}|\underline{P}\psi_i X_i|/|U|$，其中记 $P(U)=\sum_{i=1}^{k}|\underline{P}\psi_i X_i|$。

定义 5 设 $P\subseteq C$，若 $\gamma_P(U)=\gamma_C(U)$，且不存在 $R\subset P$，使得 $\gamma_R(U)=\gamma_C(U)$，则称 P 为 C 的一个（相对于决策属性 D 的）属性约简。称满足 $\gamma_P(U)=\gamma_C(U)$ 的条件属性子集 P 为候选属性约简，所有 C 的属性约简的交集称为 C 的核（简称核），记为 $Core(C,U)$。

为了方便计算，用 $R(U)$ 表示由信息系统（或决策表）的所有属性约简集，$CR(U)$ 表示所有候选属性约简集。

定义 6 如果属性 $a\in C$ 满足 $\gamma_{C-a}(U)<\gamma_C(U)$，则称属性 a 为不可缺少的，否则，称属性为冗余的。

性质 1 属性 $a\in Core(C,U)$ 当且仅当 a 是不可缺少的属性。

性质 2 对任意给定的属性 a，有 $\underline{C-\{a\}}\psi_i\subseteq\underline{C}\psi_i$ 成立，其中，$i=1,2,\cdots,k$。

由性质 2 可以看出，对任意给定的属性 a，有 $|\underline{C-\{a\}}\psi_i|\leqslant|\underline{C}\psi_i|$ 成立，其中，$i=1,2,\cdots,k$。

引理 1 对任意 $P\subseteq C$，$\underline{P}\psi_i\cap\underline{P}\psi_j=\emptyset$，其中，$i\neq j$ 且 $i\in[1,k]$，$j\in[1,k]$。

引理 2 对属性 $a\in C$，$\gamma_{C-a}(U)<\gamma_C(U)$ 成立当且仅当 $i\in[1,k]$ 使得 $|\underline{C-\{a\}}\psi_i|\leqslant|\underline{C}\psi_i|$。

属性约简的关键是在保持分类能力不变的前提下，对知识进行约简。即当属性减少时，能保证 $P(U)=C(U)$ 不变，亦即相对应的 $\gamma_P(U)=\gamma_C(U)$。

3 改进的差别矩阵与核增量更新算法

为克服通过求出所有的不可缺少属性来确定核方法的不足，文献[9]提出了简洁的利用改进差别矩阵来确定核的方法，从而使算法的效率在改进后更加高效。本文对其进行分析，提出差别矩阵的进一步改进。

以下是文献[9]给出的差别矩阵构造算法。

定义 7 对给定的信息系统 IS，定义差别矩阵 $M1=\{m_{ij}\}$ 为

$$m_{ij}=\begin{cases}\{a\in c: f(x_i,a)\neq f(x_j,a), f(x_i,D)\neq f(x_j,D), \\ \quad x_i\in U_1, x_j\in U_1\} \\ \{a\in C: f(x_i,a)\neq f(x_j,a), x_i\in U_1, x_j\in U_2'\} \\ \emptyset\end{cases} \tag{1}$$

其中，$U_1=\bigcup_{i=1}^{k}\underline{C}\psi_i$，$U_2=U-U_1$，$U_2'=delrep(U_2)$。

下面的算法 1 给出 $delrep(U_2)$ 的描述。

算法 1

```
输入：U，U1，U2
输出：U2'
begin
  U2'=∅
  for ∀x∈U2 do
    if 不存在 y∈U2' 使得 ∀a∈C,f(x,a)=f(y,a) 且 f(x,D)≠f(y,
      D) then
      U2'=U2'∪{x}
    return U2'
end
```

从中可以看出，在差别矩阵 $M1$ 中，虽然矩阵空间为 $|U_1|\times|U_1\cup U_2'|$，但是有一部分元素是重复的，即 $x_i\in U_1,x_j\in U_1$，$i<j$ 与 $x_i\in U_1,x_j\in U_1,i>j$ 所对应的 m_{ij} 是重复的。因此，为提高后面算法的计算效率，考虑在不影响计算结果的基础上增加矩阵空间来换取计算的效率，把矩阵空间大小调节为 $|U_1\cup U_2'|\times|U_1\cup U_2'|$，此时矩阵为方阵，这仍比普通的差别矩阵（空间为 $|U|\times|U|$）要小。由此给出引理 3。

引理 3 对 $U_1=\bigcup_{i=1}^{k}\underline{C}\psi_i$，$U_2=U-U_1$，$U_2'=delrep(U_2)$，$x_i\in(U_1\cup U_2')$，$x_j\in(U_1\cup U_2')$，则 $i=1,2,\cdots,|U_1\cup U_2'|$，$j=1,2,\cdots,|U_1\cup U_2'|$，$m_{ij}=m_{ji}$。

根据引理 3，新的矩阵是一个方阵，去除其中的重复项，对差别矩阵进行进一步改进，对 $i\geqslant j$ 部分（即矩阵左下三角）的元素省略不计算，只需计算出 $i<j$ 的部分。

本文提出对差别矩阵的定义进行进一步的改进，如定义 8 所示。

定义 8 对给定的信息系统 IS，定义差别矩阵 $M2=$

$\{m_{ij}\}$为

$$m_{ij}=\begin{cases}\{a\in c: f(x_i,a)\neq f(x_j,a), f(x_i,D)\neq f(x_j,D), \\ \quad x_i\in U_1, x_j\in U_1, i<j\} \\ \{a\in C: f(x_i,a)\neq f(x_j,a), x_i\in U_1, x_j\in U_2', i<j\} \\ \{a\in C: f(x_i,a)\neq f(x_j,a), x_i\in U_2', x_j\in U_1, i<j\} \\ \emptyset, x_i\in U_2', x_j\in U_2', i<j \\ \emptyset\end{cases} \quad (2)$$

其中，$U_1=\bigcup_{i=1}^{k} C_{\psi_i}$，$U_2=U-U_1$，$U_2'=delrep(U_2)$，$i,j=1,2,\cdots,|U_1+U_2'|$。

下面的算法将使用定义8中的差别矩阵$M2$定义进行计算。

通过更新差别矩阵以后，可以得到一个新的差别矩阵，从而可由定理2与其对应的算法2来求核，但是这样必须重新遍历整个差别矩阵才能得到新的核$Core(C,U\cup\{x\})$。可以引入一个定义$SA(C)$，用它标记只含单个属性的元素m_{ij}的属性$a(a\in m_{ij}, |m_{ij}|=1)$，及其重复出现的次数$k$。

定理1 对于信息系统IS，若有$m_{ij}\in M2$，且m_{ij}为单个属性，则有$Core(C,U)=\cup\{m_{ij}\,|\,m_{ij}\in M2\}$，即当且仅当$m_{ij}$为单个属性集合时，该属性属于核$Core(C,U)$，$Core(C,U)$是$M2$中单个属性集合$m_{ij}$的并集。

若新增加元素x到信息系统IS时，只需要得到$(U_1\cup U_2'\cup\{x\})$的差别矩阵，便可由定理1进行求核。因此核的增量式更新本质上就是差别矩阵的更新问题。对$M2$的更新分为下列3种：

(1)若x与U_2'不一致，由$M2$的定义可知$M2$不变。

(2)若x与$U_1\cup U_2'$一致，则在$M2$中增加对象x对应的行和列，$U_1=U_1\cup\{x\}$，由于$M2$中规定$i<j$，则只需增加并计算最后一列的数据。

(3)若x与U_1不一致，即$\exists x_i\in U_1$使得x和x_i是不一致的，则删除对象x_i所在的行，删除x_i所在的列，$U_2'=U_2'\cup\{x_i\}$，$U_1=U_1-\{x_i\}$。然后把x_i作为一个新的元素加入到矩阵中，为矩阵增加一行一列，由于$i<j$的限制，因此只需计算最后一列的数据，最后一行为空。

当新增对象x时，动态更新差别矩阵并相应地对$SA(C)$中单个属性出现的次数进行修改。$SA(C)$的相应修改有以下3种情况：

(1)当$M2$保持不变时，$SA(C)$保持不变。

(2)当$M2$增加新列时，将新列中新出现的单个属性a($a\in C$)及出现的次数k组成对(a,k)插入到$SA(C)$；如果出现已经存在于$SA(C)$中的单个属性，则将其相应的计数k加1。

(3)当删除M2中的某行时，若删除行中的某单个属性a，其在该行中出现的次数为k_1，则对$SA(C)$中的项$(a,count)$进行如下修改：如果$count>k_1$，则只修改$(a,count)$为$(a,count-k_1)$，否则删除该项；同理，修改差别矩阵中的列，并对$SA(C)$进行修改。

由此本文给出定理2，作为求核依据。

定理2 对于信息系统IS，若记$SA(C)=\{(a,count)\,|\,a\in m_{ij}, |m_{ij}|=1, m_{ij}\in M2, count$为$a$作为单个属性元素的值在$M2$中出现的次数，且$count\geqslant 1\}$，$Core(C,U)=\{a\,|\,(a,count)\in SA(C)\}$，其中$M2$为定义8中的差别矩阵。即当且仅当$m_{ij}$只包含单个属性时，其值$a$属于核$Core(C,U)$。

由上述推理，本文给出核增量更新算法，如算法2所示。

算法2

输入：$U_1=\bigcup_{i=1}^{k} C_{\psi_i}$，$U_2=U-U_1$，$U_2'=$delrep($U_2$)，差别矩阵M2，SA(C)

输出：M2($U\cup\{x\}$)及相应的核Core(C,$U\cup\{x\}$)

```
begin
  if x 与 U1∪U2'一致 then{
    在 M2 中增加最后一行 row 和最后一列 column,并由定义 8 得到矩阵元素的值。由于 M2 中规定 i<j,则只需增加并计算最后一列的数据。
    U1=U1∪{x};
    j=|U1∪U2'|;//设置列标为最后一列
    for i=1 to |U1∪U2'| //遍历最后一列
      if |mij|=1 then {
        if(mij的单一属性 a)∈SA(C) then
          修改 SA(C)相应项为(a,count+1);
        else
          添加(a,1)到 SA(C);
      }
  }
  else if x 与 U1 不一致 then{
    在 U1 中找到与 x 不一致的对象 xi,得到 xi 所在的行标 i;
      for j=i+1 to |U1∪U2'|{ //遍历 xi 所在行,且 i<j 的元素
        if |mij|=1 then {
        If(mij的单一属性 a)∈SA(C) then {
          if count-1>0 then
            修改 SA(C)相应项为(a,count-1);
      else
        删除 SA(C)相应项为(a,count);
        }
      }
    }
    j=i;
    for t=1 to j { //遍历 xi 所在列,且 i<j 的元素
      if |mij|=1 then {
      if(mij的单一属性 a)∈SA(C) then {
        if count-1>0 then
          修改 SA(C)相应项为(a,count-1);
        else
          删除 SA(C)相应项为(a,count);
        }
    }
    }
    删除对象 xi 所在的行;
    删除对象 xi 所在的列;
    U1=U1-{xi};
    U2'=U2'∪{xi};
    现在矩阵少了一行一列。把 xi 作为新元素,添加到矩阵中,为矩阵增加一行一列。其实质是根据定义 8 重新计算最后所在列的值;
    j=|U1∪U2'|;
    for i=1 to |U1∪U2'|{ //遍历"xi"所在列即最后一列的元素
      if |mij|=1 then {
      if(mij的单一属性 a)∈SA(C) then
        修改 SA(C)相应项为(a,count+1);
      else
        添加(a,1)到 SA(C);
```

```
    }
  }
}
else if x 与 U2'
不做任何处理;
由定理 3,遍历 SA(C)即可求出 Core(C,U∪{x});
end
```

4 属性约简算法及其增量更新算法

基于差别矩阵和逻辑运算的属性约简算法可以得到决策表的所有可能的属性约简结果,它实际上是将对属性组合情况的搜索演变成为逻辑公式的化简,从而简化问题。但是由于矩阵中的 m_{ij} 很多是重复的,导致逻辑公式化简时的计算量很大。

考察差别矩阵,不难发现,如果矩阵中存在一个元素,其取值为包含单属性元素的集合,则表明该属性是区分这两个样本的属性。差别矩阵中的这些元素所包含的属性组成的属性集合其实就是该决策表系统的相对属性核,可以将这些属性取出,同时将差别矩阵中包含核属性的元素的值修改为 $\emptyset$,从而得到一个新的矩阵 $M3$。

定义 9 对给定的信息系统 IS,定义差别矩阵 $M3=\{m_{ij}'\}$ 为

$$m_{ij}'=\begin{cases}\emptyset: \exists a\in Core(C,U), a\in m_{ij}, m_{ij}\in M2\\ m_{ij}: \forall a\in Core(C,U), a\notin m_{ij}, m_{ij}\in M2\end{cases}$$

在求核以及定义 9 的基础上,本文给出属性约简算法,如算法 3 所示。

算法 3

第一步 对差别矩阵 M3 中的所有坐标 i<j 且取值为非空集合的元素 $m_{ij}'(m_{ij}'\neq\emptyset)$,建立相应的析取逻辑表达式 L_{ij}:

$$L_{ij}=\bigvee_{a_k\in m_{ij}'} a_k, k=1,2,\cdots,|m_{ij}|$$

第二步 将所有的析取表达式 L_{ij} 进行合取运算,得到一个合取范式 L,即

$$L=\bigwedge_{m_{ij}'\neq\emptyset} L_{ij}$$

第三步 将合取范式 L 转换为析取范式的形式:

$$L'=\bigvee_i L_i$$

第四步 将 Core(C,U)的属性进行合取,得到合取表达式 LCore(C,U),得

$$LCore(C,U)=\bigwedge_{a_k\in Core(C,U)} a_k$$

第五步 将 LCore(C,U)加入到析取范式 L'的每个合取项中,得到

$$L'=\bigvee_i (L_i\wedge LCore(C,U))$$

第六步 析取范式 L'中的每个合取项,即为一个属性约简。

在求核以及对 $M2$ 化简的基础上,属性约简算法 3 变得更加高效。

新增对象 x 将引起 $C(U)$ 的改变,以此来求解属性约简。新增对象 x 对 $C(U)$ 的影响可分为以下 3 种情况:

(1)x 与 U_2 中的某个对象不一致;

(2)x 与 U_1 中的某个对象产生不一致;

(3)x 与 $U_1\cup U_2$ 中的对象是一致的。

5 实验结果

通过几组实验来验证算法的有效性,在内存为 4G,CPU 为 4 核 3.0GHz,操作系统为 Win7 的联想笔记本上,用 Java 实现了文献[5]、文献[9]及本文中的算法,记文献[5]、文献[9]的算法为 XU 算法和 YANG 算法。使用网络中提供的蘑菇数据库进行实验,该数据库有 8124 个对象,记录蘑菇的 23 种属性,其中第 1 列为决策属性,共 2 个决策分类。

将下载的蘑菇数据库看作决策表,并进行以下两组实验:(1)从 8124 个对象中选择 6000 个对象作为基准决策表(基准决策表的含义是指该决策表生成的差别矩阵作为算法 XU 和 YANG 的输入),从剩下的 2124 个对象中依次选择 500、1000、1500、2124 个对象作为增量,实验结果如图 1 所示;(2)由蘑菇数据库生成 7500 个对象,其中不一致对象数为 4000,从生成的 7500 个对象中选择 6000 条作为基准决策表,从剩下的 1500 个对象中依次选择 300、800、1100、1500 个对象作为增量,实验结果如图 2 所示。

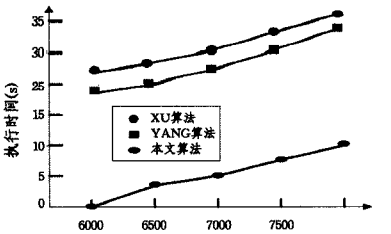

图 1 第一组实验的执行时间

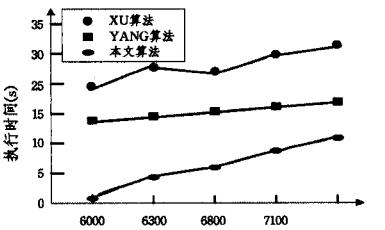

图 2 第二组实验的执行时间

从图 1 和图 2 可见,本文算法的性能优于 XU 和 Yang 算法的,这是因为本文的算法可充分利用已经求得的差别矩阵,通过对差别矩阵的更新来求解核,所以可提高求核的效率,起到高效的维护作用。此外,从图 2 可见,当决策表中的不一致对象增加时,本文算法的效率可有效提高,这是由于差别矩阵规模减小的原因,而且本文的算法也得到有效的改进,这是因为某些增量对象对求核没有作用,不修改差别矩阵。

结束语 本文提出一种基于改进差别矩阵的属性约简增量式更新算法,该算法可通过快速更新差别矩阵,在动态求解核的基础上,利用原有的属性约简有效地进行属性约简的增量式更新。通过实验证明了该算法的正确性,并将其与文献[5,9]中的增量式更新算法进行对比,结果表明新算法的时间复杂度和空间复杂度都较优。新算由于法能有效地避免许多重复的计算工作,因此能节省大量的计算时间,显著地提高了动态更新属性约简的效率。

参 考 文 献

[1] Pawlak Z. Rough sets[J]. International Journal of Information and Computer Science,1982,11(5):341-356

[2] Pawlak Z. Rough set approach to multi-attribute decision analysis[J]. European Journal of Operational Research,1994,72(3):443-459

[3] Liu Qing. Rough Sets and Rough Reasoning [M]. Beijing: Science Press,2001

[4] Hu X H, Cercone N. Learning in relational databases: Arough set approach[J]. Computational Intelligence,1995,11(2):323-338

[5] 徐章艳,杨炳儒,宋威. 基于简化的二进制差别矩阵的快速属性约简算法[J]. 计算机科学,2006,33(4):65-68
Xu Zhang-yan, Yang Bing-ru, Song Wei. Fast attribute reduction algorithm based on simple binary discernibility matrix[J]. Computer science, 2006, 33(4): 65-68

[6] 葛浩,李龙澍,杨传健. 改进的快速属性约简算法[J]. 小型微型计算机系统,2009,30(2):308-312
Ge Hao, Li Long-shu, Yang Chuan-jian. Advanced Fast attribute reduction algorithm[J]. Micro computer system, 2009, 30(2): 308-312

[7] 刘宗田. 属性最小约简的增量式算法[J]. 电子学报,1999,27(11):96-98
Liu Zong-tian. An incremental algorithm for minimum attribute reduction[J]. Chinese Journal of Electronics, 1999, 27(11): 96-98

[8] Jelonek J, Krawiec K, Slowinski R. Rough set reduction of attributes and their domains for neural networks[J]. Computational Intelligence, 1995, 11(2): 339-347

[9] 杨明. 一种基于改进的差别矩阵的属性约简增量式更新算法[J]. 计算机学报,2007,30(5):815-822
Yang Ming. An incremental updating algorithm for attribute reduction based on improved discernibility matrix[J]. Chinese Journal of computers, 2007, 30(5): 815-822

(上接第 250 页)

用 6 种不同变异策略的原始 DE 以及 4 种有代表性的改进 DE 进行了比较。实验结果表明,SMSDE 的优化效果优于其他对比的算法。

参 考 文 献

[1] Storn R, Price K. Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces[J]. Journal of global optimization, 1997, 11(4): 341-359

[2] Epitropakis M G, Tasoulis D K, Pavlidis N G, et al. Enhancing differential evolution utilizing proximity-based mutation operators[J]. IEEE Transactions on Evolutionary Computation, 2011, 15(1): 99-119

[3] Wang J, Liao J, Zhou Y, et al. Differential evolution enhanced with multiobjective sorting based mutation operators[J]. IEEE Trans. Cybernetics, 2014, 46(12): 2792-2805

[4] Cai Y, Wang J. Differential evolution with neighborhood and direction information for numerical optimization[J]. IEEE Trans. Cybernetics, 2013, 43(2): 634-647

[5] 薛羽,庄毅,顾晶晶,等. 自适应离散差分进化算法策略的选择[J]. 软件学报,2014,25(5):984-996
Xue Yu, Zhuang Yi, Gu Jing-jing, et al. Strategy selecting problem of self-adaptive discrete differential evolution algorithm[J]. Journal of Software, 2014, 25(5): 984-996

[6] Guo S, Yang C. Enhancing differential evolution utilizing eigenvector-based crossover operator[J]. IEEE Transactions on Evolutionary Computation, 2014, 19(1): 31-49

[7] Brest J, Greiner S, Boskovic B, et al. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems[J]. IEEE Transactions on Evolutionary Computation, 2006, 10(6): 646-657

[8] 杨振宁,唐柯. 差分进化算法参数控制与适应策略综述[J]. 智能系统学报,2011,6(5):415-423
Yang Zhen-yu, Tang Ke. An overview of parameter control and adaptation strategies in differential evolution algorithm[J]. CAAI Transactions on Intelligent Systems, 2011, 6(5): 415-423

[9] Qin A K, Huang V L, Suganthan P N. Differential evolution algorithm with strategy adaptation for global numerical optimization[J]. IEEE Transactions on Evolutionary Computation, 2009, 13(2): 398-417

[10] Wang Y, Cai Z, Zhang Q. Differential evolution with composite trial vector generation strategies and control parameters[J]. IEEE Transactions on Evolutionary Computation, 2011, 15(1): 55-66

[11] Zhang J, Sanderson A C. JADE: adaptive differential evolution with optional external archive[J]. IEEE Transactions on Evolutionary Computation, 2009, 13(5): 945-958

[12] Gong W, Cai Z, Ling C X, et al. Enhanced differential evolution with adaptive strategies for numerical optimization[J]. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 2011, 41(2): 397-413

[13] Wang Y, Li H-X, Huang T, et al. Differential evolution based on covariance matrix learning and bimodal distribution parameter setting[J]. Applied Soft Computing, 2014, 18: 232-247

[14] Xin B, Chen J, Zhang J, et al. Hybridizing differential evolution and particle swarm optimization to design powerful optimizers: a review and taxonomy[J]. IEEE Trans Systems Man Cybern Part C Appl Rev, 2011, 42(5): 744-767

[15] Ghosh S, Das S, Vasilakos A V, et al. On convergence of differential evolution over a class of continuous functions with unique global optimum[J]. IEEE Trans. Syst., Man, Cybern. B, Cybern., 2012, 42(1): 107-124

[16] Ding Y, Jiao Y-C, Zhang L, et al. Solving port selection problem in multiple beam antenna satellite communication system by using differential evolution algorithm[J]. IEEE Transactions on Antennas and Propagation, 2014, 62(10): 5357-5361

[17] 毕志升,王甲海,印鉴. 基于差分演化算法的软子空间聚类[J]. 计算机学报,2012,35(10):2116-2128
Bi Zhi-sheng, Wang Jia-hai, Yin Jian. Subspace clustering based on differential evolution[J]. Chinese Journal of Computer, 2012, 35(10): 2116-2128

[18] Das S, Suganthan P N. Differential evolution: A survey of the state-of-the-art[J]. IEEE Transactions on Evolutionary Computation, 2011, 15(1): 4-31

[19] Mezura-Montes E, Velázquez-Reyes J, Coello Coello C A. A comparative study of differential evolution variants for global optimization[C]// Proceedings of 8th Annual Conference on Genetic and Evolutionary Computation. ACM, 2006: 485-492

[20] Montgomery J, Chen S. An analysis of the operation of differential evolution at high and low crossover rates[C]// IEEE Congress on Evolutionary Computation. 2010: 1-8

[21] Liang J J, Qu B Y, Suganthan P N, et al. Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization[R]. Technical Report, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China, Nanyang Technological University, Singapore, 2013

[22] Derrac J, García S, Molina D, et al. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms[J]. Swarm and Evolutionary Computation, 2011, 1(1): 3-18