基于遗传算法的数据交换问题求解方法

唐智勇，马武彬，黄宏斌，邓 苏
（国防科学技术大学 信息系统重点实验室，长沙 410073）

摘 要： 数据交换问题是信息集成中的关键问题之一。针对数据交换问题的基本定义描述、算法以及求解过程，提出改进的数据交换问题求解过程，解决当某类依赖条件下无解或不能在多项式时间内求解问题。该方法通过对约束条件有依据地弱化修正，使得能够在多项式时间内求出近似解，并最大限度地满足用户对于目标数据库的需求，最后进行了实验验证。

关键词： 信息集成；数据交换；chase 方法；普通解法；遗传算法
中图分类号： TP393 **文献标志码：** A **文章编号：** 1001-3695(2012)06-2043-04
doi:10.3969/j.issn.1001-3695.2012.06.010

Solving data exchange problem based on genetic algorithm

TANG Zhi-yong, MA Wu-bin, HUANG Hong-bin, DENG Su
(*C4ISR Key Laboratory, National University of Defense Technology, Changsha* 410073, *China*)

Abstract: Data exchange is one of the key problems of information integration. This paper proposed an improved chase approach in the special background of application for the data exchange. Solving the problem that finding the most approximately solution in polynomial time for data exchange problem by amending the dependence condition appropriately, this solution must be also satisfied by users' needs for target database as much as possible.

Key words: information integration; data exchange; chase approach; universal solution; genetic arithmetic

0 引言

本文所研究的数据交换问题(data exchange problem)主要是指在不同数据库模式下的数据交换问题[1]。数据交换是数据库理论领域非常重要的基础理论方法，目的在于研究和发展不同数据库之间数据转移的基础理论和算法。其应用价值在于既能够使得数据库之间保持实时的映射联系，又能够完成模式集成以后底层的数据抽取工作，对于企业将传统的数据集成应用于实践具有重大意义。

数据交换具体是指在源模式到目标数据库模式映射确定以后，物化出一个新的目标数据库，使其既能够满足源数据库模式到目标数据库模式的约束条件，同时满足目标数据库模式的内部约束条件。

文献[1]首次提出了数据交换的概念，分析了一般求解过程和交换中的查询回答。该文给出了求解方法，并对其复杂度进行了分析，并分析了数据交换问题的有解条件，提出了该问题有解时所满足的条件。

目前对于数据交换问题的求解一般利用 chase 方法[2~5]。Chase 方法能够有效地求解数据交换问题。该方法将约束条件依次应用到初始的源数据实例上，使得数据库实例解法能够满足所有的约束。然而传统的 chase 过程存在以下问题：chase 过程求解数据交换问题时不断地产生新的空值元组来满足约束条件，出现无限循环的迭代过程。

文献[6]分析了 chase 方法的有解条件，并将其扩展到了分层约束的范畴，即在弱循环依赖定义的基础上，利用依赖查询过程来避免 chase 过程中的无限循环。文献[7]在分析分层 chase 条件的基础上，提出了安全条件与诱导限制条件，将 chase 过程的有解限制条件进一步扩大。但是该文也不能证明 chase 过程有解条件的完备性，所以，仍然可能存在约束条件，无法利用 chase 过程求解。

目前对于数据交换问题的普通解法大多采用 chase 过程方法，但是现今还没有人能够保证对于给出的所有数据交换问题能够有解。在某些实际情况中，有时必须要生成一个可行的目标数据库，例如针对问题背景[8]，在信息的聚焦服务过程中，要处理各个数据源的集成数据，必须要根据用户的需求来生成一个可能并不能够严格地满足约束条件的解。在这种需求条件下，本文利用遗传算法的优化思想，将约束条件进行有目标的退化，以生成数据交换问题的最优解。

本文首先简要介绍数据交换问题及 chase 方法，然后改进 chase 方法，建立依赖条件的目标模型，并利用遗传算法求解。

1 问题描述

对于源数据库 S_i 上的源模式 σ_i，存在目标模式 τ，以及 m 是从 σ_i 到 τ 的映射，其中 m 满足约束条件 Σ，需要创建在模式 τ 下的数据库 T，使得 $(S_i,T)\in m$，即有 $(S_i,T)\models\Sigma$（$\models$ 代表满足约束）。这时，数据库 T 就被称做在约束条件 Σ_i 下对于 S_i 的解决方法。图 1 为数据交换问题描述。

收稿日期：2011-10-20；修回日期：2011-12-05

作者简介：唐智勇(1978-)，男，湖南祁东人，博士研究生，主要研究方向为信息集成；马武彬(1986-)，男，重庆人，博士研究生，主要研究方向为信息集成、Web 服务等(mawubin417@163.com)；黄宏斌(1975-)，男，江苏如皋人，副教授，博士，主要研究方向为信息集成、综合处理；邓苏(1963-)，男，湖南长沙人，博导，主要研究方向为信息集成.

定义 1 [1] 数据交换问题。已知$(S,T,\sum^{st},\sum^{t})$,I 是模式 S 的实例,寻找实例 J,使得解 J 满足 $\langle I,J\rangle \models \sum^{st}\cup\sum^{t}$,且 $J\models\sum^{t}$。

$S=\{S_1,S_2,\cdots,S_n\}$ 代表源模式,$T=\{T_1,T_2,\cdots,T_n\}$ 代表目标模式,两个模式之间无关联关系。I、J 分别是模式 S、T 的实例。实例 I、J 代表实际的数据,这些数据广泛地分布在底层的各个数据库中。数据可以分为常量和变量,分别用 Var 和 Const 表示。源数据以常量的形式存在,目标数据则通常包含常量和变量两种。$\sum^{st}$、$\sum^{t}$ 分别代表两种约束规则。源数据库和目标数据库通过约束规则联系在一起。

定义 2 [1] 数据源到目标的约束(source to target dependency, STD)。STD 是目标数据库与源数据库之间的依赖关系,用 $\sum^{st}$ 表示,一般形态为 $\forall x(\varphi_s(x)\rightarrow\chi_T(x))$。其中:$\varphi_s(x)$ 是源数据库上的规则描述,具有自变量 x;$\chi_T(x)$ 是目标模式上的规则描述;$x=\{x_1,\cdots,x_n\}$ 是 n 维关系数据自变量。

定义 3 [1] 目标内部约束(target dependency, TD)。TD 是指在目标数据库的内部数据库的依赖,利用 $\sum^{t}$ 表示。TD 包括两类依赖,即元组生成依赖(tuple generate dependence,TGD)和等式生成依赖(equation generate dependence,EGD)。TGD 的普通形式是 $\forall x(\varphi_T(x)\rightarrow\chi_T(x))$。其中:$\varphi_T(x)$ 和 $\chi_T(x)$ 是目标数据库上的规则描述。EGD 的普通形式为 $\varphi_T(x)\rightarrow(x_i=x_j)$。其中:$\varphi_T(x)$ 是目标数据库上的规则描述,$x_i,x_j\in x$。

定义 4 约束偏好。指用户对于各个约束条件的偏好。每一个依赖对应一个偏好值,通过用户的实际需求来决定。如果有 n 个依赖条件,则利用 $\sigma_1,\sigma_2,\cdots,\sigma_n$ 来表示用户对于依赖条件的偏好值。

定义 5 普通解法(universal solution)。对于一个数据交换问题$(S,T,\sum^{st},\sum^{t})$,在模式 S 上,实例 I 的解 J 是一个普通解法,当且仅当:对于该问题的每一个解 J',都有 $\exists h: J\rightarrow J'$ 是同态映射。

2 改进的 chase 方法

目前普通解法的求解一般采用 chase 方法。

2.1 Chase 过程 [4]

Chase 过程按照某种特定方法来构造序列 $A_0^{\Sigma},A_1^{\Sigma},\cdots,A_n^{\Sigma}$。最终使得 $A_n^{\Sigma}\models\sum$。详细过程可以参考文献[1~4,9]。

假设依赖条件:$\varepsilon=R(x,y)\rightarrow\exists zR(y,z)$,源数据库实例 $I=\{R(a,b)\}$,运用该方法可得$\{R(b,X),R(X,X_1),R(X_1,X_2)\cdots\}$,chase 过程将一直循环下去,无法终止。

2.2 Chase 循环条件

本节分析如何判断 chase 过程循环的依赖条件以及修改方法。

首先解释一种生成依赖图的方法 [1]:$\sum$ 是 TGD 依赖。构建这样一个有向图:(a) 对于关系 R,将图中节点上的值用 R^i 表示,R^i 是 R 中的第 i 个元素;(b) 对于一般的 TGD 形式:$\varphi(x)\rightarrow\exists y(\varphi(x,y))$,源为 S,目标为 T。按如下方法构建边:

a) 当在 S^i 和 T^j 位置上同时存在 $x\in x$,则从元素 S^i 向 T^j 连接有向边。

b) 将第一条规则中连接的起点 x 与每一个 $y\in y$ 进行连接,并将该连接边标志为特殊边 $(x,y)=1$。

图 2 是两个约束条件有向图。

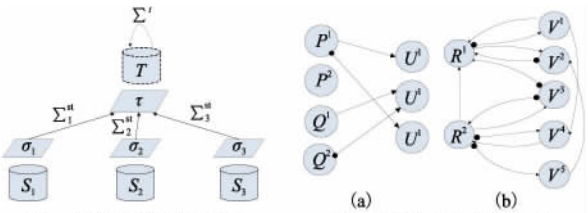

图1 数据交换问题描述 图2 依赖条件的有向图

概念 1 弱非循环约束条件。如果依赖图中不存在带有特殊边的环路,则该约束条件为弱非循环约束条件。

根据上述定义,从图 2(a) 可以看出,该约束是弱非循环约束条件;(b) 所代表的约束条件则不是弱非循环约束条件,存在数据交换问题的无解情况。

已经证明 [3],只要约束条件为弱非循环的,就存在普通解。只要找到一种合适的方法,使得所有的约束条件都能在满足用户需求的前提下转换为弱非循环依赖,那么就可以保证该问题有解。

下面给出判定约束条件是否为弱非循环依赖的算法。

算法 1 特殊有向边的查询算法

```
Procedure FindSpecialCycle
Input: dependency graph ∑G(v,e);
Output: cycle with special edge;
FindSpecialCycle(∑)
{
(1) FindCycle(u) { /* 基于 Tarjan 算法的深度优先搜索寻找特殊环,并生成多个存储环路节点的栈* /
(2) DFN[u] = Low[u] = ++Index;
(3) Stack.push(u);
(4) for each(u,v) in ∑G
(5) if (v is not visited) {
(6) FindSpecialCycle(v);
(7) Low[u] = min(Low[u],low[v])}
(8) else if (v in S)
(9) Low[u] = min(Low[u],DFN[v])
(10) end;
(11) if (DFN[u] = = Low[u])
(12) repeat;
(13) v = S.pop();
(14) Stack[i = ++1].push(v) until(u = = v)}
(15) for each Stack I
(16) if All(u,v) = = 0 in Stack[k]
(17) Stack[k].remove();
(18) end;
(19) if Stack I = null;
(20) return false;
(21) return true;
}
```

通过该算法,可以判断依赖条件是否为弱循环依赖,如果不是则返回 1;如果是,则返回 0,并得到该依赖有向图的所有带有特殊边的环。

下文将给出如何修改约束条件。

2.3 改进的 chase 方法

从本质上来讲,chase 的无限循环过程就是空值元素无限的增加过程。可以通过合理修改依赖条件,使其满足弱非循环依赖条件,从而避免无限循环增加的空值元素过程。

通过算法 2,基于该依赖图中的依赖条件将变成一个弱非循环约束条件。

算法 2 去除有特殊边有向环的算法

```
Input: Stack[n];
Output: WeaklyAcyStack I;
EraCycle(Stack I)
{
(1) for each Stack I
```

```
(2) for each edge //对于特殊环中的每一条边做一个遍历
(3) if ((u,v) =1)
(4) Stack Ⅰ. EraseSpecial(u,v); /* 对特殊边进行处理,并将相关约束进行重新构造* /
(5) RecordStack Ⅱ. push(u,v); //将取出的特殊边放入记录中
(6) else
(7) Stack Ⅰ. EraseNormal(u,v); /* 对普通边的处理,并将相关约束进行重新构造* /
(8) RecordStack Ⅱ. push(u,v);
(9) if(FindSepcialCycle(v))
(10) return true;
(11) i++;
(12) else
(13) EraCycle(FindSpecialCycle(v))
(14) end;
(15) end;}
```

EraseSpecial(u,v) 和 EraseNormal(u,v) 是两种不同的修改依赖条件的函数,其目的都是消除环路。主要包括以下三种不同的方法:

a) 去除特殊依赖边方法。其是指对导致目标模式中依赖条件产生新元组依赖进行删除。这个过程需要更改目标模式的结构,而且还会影响到其他依赖图形的形态。

b) 将特殊依赖边修改为普通的依赖边。其通过修改源模式到目标模式的依赖,将空值元属的增加过程转换为一个常量值的产生或者修改为目标模式中任何一个非空值元组的值。这个过程会导致目标数据模式的重复数据计算,并导致在物化过程中出现大量的数据冗余。

c) 修改普通边。其是通过去掉依赖条件中的一些非空值约束条件来完成的。该方法的缺点是会影响到目标模式的意义,可能会得出与用户期望差距较大的结果。

这些方法的选择需要一个合理的目标驱动。

2.4　修正依赖的目标

算法 2 是一个破除特殊依赖环的过程,该过程是在某些特殊任务的需求中必须要求生成一个解的前提下给出的,所以必须保障这个方法的合理性。即该过程如何保障去除特殊环后所得到的依赖能够最大近似地反映出原依赖。

定义 6　依赖差别度。两个依赖关系分别在目标模式和源模式相同的条件下,利用 Differ($\sum^1$,$\sum^2$) 表示两个依赖关系之间的差别度。

依赖差别度的计算公式为

$$\mathrm{Differ}(\sum^1,\sum^2)=\frac{\sum R_1(N_i)-\sum R_2(N_j)}{\max(\sum R_1(N_i),\sum R_2(N_j))}+\sum(r_{1i}\sigma_i-r_{2i}\sigma_i)$$

其中: $R_1(N_i)$ 表示在 $\sum^1$ 中关于 N_i 元素的关系数量; r_{1i} 表示在 $\sum^1$ 中第 i 个关系,如存在则为 1,不存在则为 0; σ_i 表示第 i 个关系的权重,$\sum\sigma_i=1$。

在去除环算法的过程中,本文引入了两个依赖关系的差别度计算,可以用来检验该去除环的优化程度,即怎样在去环过程中,在能够得到解的前提下,通过依赖差别度来表示用户的满意程度,通过最优化该差别度来最大限度地满足用户需求。将该问题转换为以下最优化问题:

目标函数为

$$F(\sum)=\min(\sum(\mathrm{Differ}(\sum^i,\sum^j)))\quad i\neq j$$

约束条件有

$$f(\sum)=\mathrm{FindSpecialCycle}(\sum)=0 \tag{1}$$

在该目标函数的驱动下,利用进化算法可以得到最优的去环算法过程。将该方法移入到 chase 过程的无限空值循环的处理中,即可得到数据交换问题的最大近似解。

2.5　使用遗传算法的求解策略

遗传算法是一种进化算法,主要是通过群体的迭代来进行优化。

首先使用一个合适的基因组给问题编码。本文使用一个整数二维数组来表示基因组,每一个项目组为组成每一个环中可以更改依赖条件的方法数量,数组中的每个元素依次包含一种寻找替换边或者去边使得该环破解的方法。每一个方法 g 对应生成了一种约束图 $\sum$。

3　实验分析

实例 1　数据交换问题(S,T,$\sum^{st}$,$\sum^t$),S 包含两个二元关系和一个三元关系,$S=(P,Q,R)$,假设 $\sum^t=\varphi$,$\sum^{st}$ 由如下约束条件组成:

$\varepsilon_1:=(Px,y)\wedge Q(u,v)\rightarrow\exists zU(x,u,z)$

$\varepsilon_2:=P(x,y)\wedge R(x_1,y_1,z_1)\rightarrow\exists p\exists qV(x,y_1,p,q,z_1)$

$\varepsilon_3:=Q(u,v)\rightarrow\exists k\exists r\exists tV(k,r,u,v,t)$

$\varepsilon_4:=V(k,r,m,n,t)\rightarrow Q(k,r)$

$\varepsilon_5:=V(k,r,m,n,t)\rightarrow R(m,n,t)$

数据实例: $I=\{s(P),s(Q),s(R)\}$。其中:

$s(P)=\mathrm{Person}(\mathrm{PerId},\mathrm{PerName})$

$s(Q)=\mathrm{Equipment}(\mathrm{EquId},\mathrm{EquName})$

$s(R)=\mathrm{Weapon}(\mathrm{WeapId},\mathrm{WeapName},\mathrm{WeapFire})$

假设用户对于每一个约束的偏好值为

$$\lambda=(\sigma_1,\sigma_2,\sigma_3,\sigma_4,\sigma_5)=$$
$$(0.221,0.333,0.142,0.231,0.074)$$

可以看出该问题的约束条件不是弱循环约束。如果按照标准的 chase 过程,则该问题是无解的,即会陷入无限的目标变量生成循环中。

下面利用遗传算法来改进 chase 过程求解。假设每个个体适应度为 Fitness(i),被选中概率 P_i 为

$$P_i=\mathrm{Fitness}(i)/\sum_{i=1}^{\mathrm{topsize}}\mathrm{Fitness}(i) \tag{2}$$

本文设计实动态惩罚函数和拉伸函数等,以验证对比结果。

定义如下约束满足罚函数:

$$H(g)=\sum_{i=1}^{K}f_i(\sum)\times m_i \tag{3}$$

其中:

$$m_i=\begin{cases}0 & f_i(g)=0\\ 1 & f_i(g)\neq 0\end{cases}$$

得到包含基因组 g 的适应度函数为

$$\mathrm{Fit}(g)=F(g)+rH(g) \tag{4}$$

其中: r 代表罚函数尺度系数,$r>0$。

本文利用动态适应度函数来检验算法性能。

$$\mathrm{Fit}(g,t)=F(g)+rH(g)\times\frac{t_{\max}-t}{t_{\max}} \tag{5}$$

其中: t 代表当前代数,$t_{\max}$ 代表最大遗传代数,计算 100 代结束。

进化算法在运行早期个体差异较大,采用轮盘赌方式选择可以取得较好效果。在进行到遗传算法的后期,优秀的个体在产生后代的优势将不再明显,从而可能会导致整个种群进化停滞。因此,需要对适应度函数作适当地变换。本文借鉴文献[10] 算法的思想,适应度变换方法如下:

$$\mathrm{Fit}=\frac{1}{\sqrt{\lambda}}\mathrm{e}^{-\lambda(1-\mathrm{Fit}(g,t))}\quad \lambda>0 \tag{6}$$

其中: λ 代表了基因复制的概率,其值越小,复制有较大适应度的个体的概率越大。文献[10]的算法表明: 当 λ 取值在 0.95 ~ 1.05 时,算法能够有较好的性能。

算法的主要参数取值如下: 保留两个最佳个体一直保持活跃整个世代,交叉概率 $P_c=0.7$,变异概率 $P_m=0.05$,最大迭代数 $t_{\max}=110$,种群大小 $p_{\text{size}}=30$,搜索精度 $\varepsilon=1.0\times10^{-5}$,复制的强制性系数 $\lambda=0.98$,匹配度 $md=1$。

采用静态的适应度函数方法,得出最小的近似度值为 0.68。整个遗传过程如图 3 所示。

取三种适应值的处理方法,对实验值分段取样,得到对比折线分析如图 4 所示。

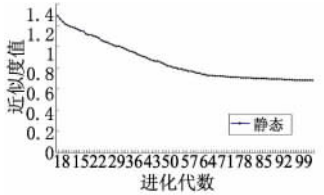

图3 静态适应度函数下进化过程

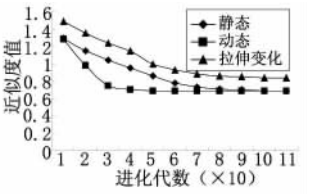

图4 近似度值分段取样对比分析

稳定以后,通过解码,得到该数据交换问题的极大近似依赖条件为

$$\varepsilon_1:=P(x,y)\ \wedge Q(u,v)\rightarrow\exists zU(x,u,z)$$
$$\varepsilon_2:=P(x,y)\ \wedge R(x_1,y_1,z_1)\rightarrow\exists p\exists qV(x,y_1,p,q,z_1)$$
$$\varepsilon_3:=Q(u,v)\rightarrow\exists tV(c_1,c_2,u,v,t)$$
$$\varepsilon_4:=V(k,r,m,n,t)\rightarrow Q(k,r)$$
$$\varepsilon_5:=V(k,r,m,n,t)\rightarrow R(c_3,c_4,c_5)$$

其中: (c_1,c_2,c_3,c_4,c_5) 为修改依赖条件后生成的常量。由此得到该数据交换问题的解为

$J=\{U(\mathrm{PerId},\mathrm{EquId},\mathrm{null}),V(\mathrm{PerId},\mathrm{WeapName},\mathrm{null},\mathrm{null},\mathrm{WeapFire}),V(c_1,c_2,\mathrm{EquId},\mathrm{EquName},\mathrm{null}),\mathrm{Equipment}(c_1,c_2),\mathrm{Weapon}(c_3,c_4,c_5)\}$

由普通解的定义可知,该解为数据交换问题的一个普通解。求解完毕。

4 结束语

数据交换问题逐渐成为信息集成问题中最为活跃的一个领域。本文主要研究了数据交换问题中的求解问题。针对利用 chase 过程无法求出某类问题的解,提出了一种引入进化计算的改进 chase 方法来获取问题的最大近似解,较好地解决了上述问题。

由于数据交换问题的求解研究仍然处于起步阶段,对于避免空值元组无限产生的过程还需要进一步研究。另外,在得出了普通解以后,如何保证在占用最小资源的情况下,能够完全表示该问题的所有解结构(该最小解称为该数据交换问题解的核),即数据交换求解问题的核计算问题,这些将是以后研究的重点。

参考文献:

[1] FAGIN R, KOLAITIS P G, MILLER R J, *et al*. Data exchange: semantics and query answering [J]. Theoretical Computer Science, 2005, 336(1): 89–124.

[2] FAGIN R, KOLAITIS P G, POPS L. Data exchange: getting to the core [J]. ACM TODS, 2005, 30(1): 147–210.

[3] NASH A, DEUTSCH A, REMMEL J. Data exchange, data integration, and the chase, UCSD Tech Report CS2006-0859 [R]. 2006.

[4] GOTTLOB G, NASH A. Efficient core computation in data exchange [J]. Journal of ACM, 2008, 55(2): 1–49.

[5] PICHLER R, SAVENKOV R. Towards practical feasibility of core computation in data exchange [J]. Theoretical Computer Science, 2011, 2(5): 368–379.

[6] DEUTSCH A, NASH A, REMMEL J. The chase revisited [C] //Proc of International Conference on Principles of Database Systems. 2008.

[7] MEIER M, SCHMIDT M, LAUSEN G. On chase termination beyond stratification [C] //Proc of International Conference on Very Large Database. 2009.

[8] 黄宏斌. 基于语义关系的战场信息资源聚焦服务方法及关键技术研究 [D]. 长沙: 国防科学技术大学, 2007.

[9] DEUTSCH A, TANNEN V. Mars: asystem for publishing XML from mixed and redundant storage [C] //Proc of International Conference on Very Large Database. 2008.

[10] GOLDBERG D E. Genetic algorithms in search, optimization and machine learning [M]. Boston: Addison-Wesley Publishing Company, 1989.

[11] 蒋哲远, 韩江洪, 王钊. 动态的 QoS 感知 Web 服务选择和组合优化模型 [J]. 计算机学报, 2009, 32(5): 1014–1025.

(上接第 2042 页)

5 结束语

本文提出了一种带有线性调整策略的人工蜂群算法,并将这种算法应用到 K-均值聚类中,实现了改进人工蜂群算法与 K-均值相结合的混合聚类算法。该聚类算法结合了人工蜂群算法鲁棒性较好和 K-均值收敛速度快的优点,与 K-均值相比,在算法稳定性和收敛精度方面都有了明显的提高,特别是对于数据之间的分类特性不明显情况,本文算法与遗传谱聚类和免疫 K-均值相比更具优势。最后,本文算法经实验验证是有效、可行的,具有较好的实际应用价值。

参考文献:

[1] 郑晓鸣, 吕士颖, 王晓东. 基于免疫粒子群优化的聚类算法 [J]. 计算机工程, 2008, 34(15): 179–184.

[2] 傅景广, 许刚, 王裕国. 基于遗传算法的聚类分析 [J]. 计算机工程, 2004, 30(4): 122–124.

[3] 徐义春, 董方敏, 刘勇, 等. 带平衡约束矩形布局优化问题的遗传算法 [J]. 模式识别与人工智能, 2010, 23(6): 794–800.

[4] 王会青, 陈俊杰, 郭凯. 遗传优化的谱聚类方法研究 [J]. 计算机工程与应用, 2011, 47(14): 143–145.

[5] 郑伟, 刘静, 曾建潮. 人工蜂群算法及其在组合优化中的应用研究 [J]. 太原科技大学学报, 2010, 31(6): 467–471.

[6] 郑伟, 刘静, 曾建潮. 具有混沌差分进化搜索的人工蜂群算法 [J]. 计算机工程与应用, 2011, 47(29): 27–30.

[7] KARABOGA D, BASTURK B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm [J]. Journal of Global Optimization, 2007, 39(3): 459–171.

[8] KARABOGA D. An idea based on honey bee swarm for numerical optimization [D]. Erciyes: Erciyes University, 2005.

[9] KARABOGA D, BASTURK B. On the performance of artificial bee colony (ABC) algorithm [J]. Applied Soft Computing, 2008, 8(1): 687–697.

[10] UCI. Date sets [EB/OL]. http://archive.ics.uci.edu/ml/datasets.html