融合粒子群算法改进 XML数据智能清洗策略

刘 波[1],杨路明[1],邓云龙[2]
(1.中南大学 信息学院,湖南 长沙 410083; 2.中南大学 湘雅附三医院,湖南 长沙 410013)

摘 要:针对 XML数据质量问题,以 XML键为基础、借助多模板隐马尔可夫模型信息抽取策略与粒子群算法构建新的 XML数据清洗方法;为了提高 XML相似性数据并行检测效率,尝试利用波函数对粒子群算法进行相应优化.对比其他 XML数据清洗算法,一系列仿真实验表明改进的 XML数据清洗方法不仅自适应学习功能强、人工参与程度低、计算量小,而且时间性能有 94%左右提升.

关键词: XML键;粒子群算法;数据清洗;隐马尔可夫模型
中图分类号: TP311.13 **文献标识码**: A **文章编号**: 1673-4785(2008)03-0226-08

An intelligence data cleaning strategy for XML database using PSO

LIU Bo[1], YANG Lu-ming[1], DENG Yun-long[2]
(1. College of Information Science and Engineering, Central-south University, Changsha 410083, China; 2. The 3rd Xiangya Hospital, Central-south University, Changsha 410013, China)

Abstract: To improve XML data quality, this paper proposes a new XML data cleaning method based on XML-keys, the information draw-out strategy of multiple templates, the hidden Markov model (HMM), and particle swarm optimization (PSO). To improve parallel efficiency when detecting similar XML records, a wave function is employed to improve the PSO algorithm. A series of simulations indicated that, compared with other XML data cleaning algorithms, the improved XML data cleaning algorithm has a more powerful adaptive learning capability, requires less human interaction, and reduces computational time by about 94%.

Keywords: XML key; particle swarm optimization; data cleaning; hidden Markov model

目前 Web上已经积累了大量的 XML数据,这些参差不齐的数据形成了许多“脏数据”,它们会阻碍商业应用,因此需要对它们进行挖掘、清洗,这是提高数据质量的关键.由于国外信息化程度较高,对数据清洗的需求较为迫切,因此当前的研究大多集中在国外[1-2];随着国内信息化的快速发展,对数据清洗的研究也逐步展开,并取得了骄人的成果[3-6].当前数据清洗的研究主要集中如下: 1)特殊域清洗,主要解决某类特定应用域的数据清洗,这是目前研究得较多的领域,也是应用最成功的一类; 2)与特定应用领域无关的数据清洗,主要集中在清洗重复的记录,如郑仕辉提出的基于 XML相似重复记录检测方法[4]和陈伟提出的 XML相似重复数据的清理方法[5]等; 3)数据清洗框架,如陆凤霞提出的开放式数据清理框架[6],王桐提出的基于改进粒子群优化的结构聚类方法[7], Richi Nayak根据语义和上下文相似性对 XML文档进行分级、智能聚类等操作[8]; 4)数据分析工具,如 Riera-Ledesma与 Salazar-González针对数据数清洗时局部错误数据提出的分枝切割算法[9]和启发式求解算法[10]等; 5) ETL工具,如 Lee根据数据挖掘过程的学习环境提出的诊断、预测与合成模型[11]等.但这些研究或多或少存在不完备的地方,主要表现如下: 1)数据清洗的研究主要集中在字符型数据上,识别其他字段之间的关系异常还不成熟、实用,还需探索更灵活的清洗手段和更实用的清洗算法; 2)尽管检测重复记录受到很大的关注,采取了许多措施,但遭遇海量数据时,耗时太多,检测效率与检测精度并不令人满意; 3)

收稿日期: 2007-06-25.
基金项目:湖南信息职业学院科技创新资助项目 (108652006011);湖南省教育厅科研基金资助项目 (05c671).
通讯作者:刘 波. E-mail: ltbo99@yahoo.com.cn.

大多数数据清洗工具都是针对特定的领域,其应用受到一定的限制.虽然特定领域的数据清洗仍是应用的重点,但较通用的清洗解决方案会受到越来越多的关注;4)国产的数据清洗工具还很少,其主要是研究重复记录的清洗问题,目前还很少研究关于不完整数据、错误数据的清洗问题;5)目前数据清洗的研究主要集中在结构化数据上,而 XML的通用性、自描述性等特征使它在互联网上得到了广泛的应用,其相应的 Web数据越来越多,而对它的数据管理研究却滞后.

针对以上分析,本文将以 XML键为基础讨论构建 XML数据清洗的过程.

1　基本定义与 XML键的获取

定义 1　脏数据[3-4]指不符合数据仓库或上层应用逻辑规格的数据,清洗过程中识别脏数据后,将会丢弃或转换,用 DirtyData表示;

定义 2　清洗检查指检测出干净数据或脏数据的过程,用条件函数 cond: Data→boolean表示: cond (data) = true表示数据项 data验证为脏数据 (data∈DirtyData); cond(data) = false表示数据项 data验证为干净数据 (data∈CleanData);

定义 3　数据清洗[3-4]指从各种原始数据中抽取出干净数据的过程,可以形式化表示为 Data-Clean: RawData→CleanData;

定义 4　XML文档树指一棵 XML文档树被定义为 $T = (V, \mathrm{chl}, \mathrm{lab}, \mathrm{val}, V_r)$,其中 V为 XML文档树 T中的节点集合,V_r为根结点,chl表子节点的集合,val表从集合 V到 $E \cup S \cup A$(E为元素名称集合,S:指代 #PCDATA,A:属性名称集合)的映射函数.

针对 XML树 T及其对应的架构,一个 XML关键字就是一个对 $(H, \{p_1, p_2, \cdots, p_n\})$,其中 H是一个路径表达式 Paths(T),$\{p_1, p_2, \cdots, p_n\}$是一个简单路径表达式的集合,其约束关系如下:取出任意 2个节点 $(n_1, n_2) \in [[H]]$,对应 2个节点集合对 $(n_1[[p_i]], n_2[[p_i]])$,这 2个集合分别是从 n_1、n_2沿着路径 p_i所到达的节点的集合.

定义 5　XML候选键约束指给定 DTD $D = (E, A, P, R, r)$,任意的 XML文档树 $T \models D$,Paths(T)为文档树 T上的路径集合,是 XML函数依赖 (XML functional dependency, FD)集,K是 $\mathrm{FD}_{\mathrm{XML}}$集 $\sum$的候选键,当且仅当

1) K是 Paths(T)中元素构成的集合;

2)存在路径 P',P''使得

① P',P''和 K构成 Paths(T)的一个划分;

② $\forall P'_i, P'_i \in P', K \rightarrow P'_i$;

③ $\forall P'_j, P'_j \in P'', \exists \{p'_{x1}, \cdots, p'_{xn}\}$,使得 $\{p'_{x1}, \cdots, p'_{xn}\} \cup K \rightarrow P'_j$;

3) K的任何真子集都不满足 1)和 2).

根据上述的定义及约束条件,给出一个求解 $\mathrm{FD}_{\mathrm{XML}}$集 $\sum$的候选键算法:

输入:一个 $\mathrm{FD}_{\mathrm{XML}}$集 $\sum$,路径集 Paths(T);

输出: $\sum$的一个候选健 K;

Finding a Candidate Key for XML ($\sum$, Paths(T)).

1)初始化:{LP←左部路径集;RP←右部路径集;DP←双部路径集;EP←外部路径集};

2) $P'' \leftarrow EP$;

3) if $DP = \Phi$ then {$K \leftarrow LP$; return(K);}

else{$K \leftarrow LP \cup${所有右部路径的左部路径}; $P' \leftarrow RP$;

$\sum \leftarrow \sum$ - {所有右部路径出现过的 $\mathrm{FD}_{\mathrm{XML}}$的约束};

while $\sum \neq \Phi$ do

{for each (每个 $\mathrm{FD}_{\mathrm{XML}}$: $p_{x1}, \cdots, p_{xn} \rightarrow p_{y1}, \cdots, p_{ym}$ in $\sum$中)

{$K \leftarrow K \cup \{p_{x1}, \cdots, p_{xn}\}$;

$\sum \leftarrow \sum$ - {$p_{x1}, \cdots, p_{xn} \rightarrow p_{y1}, \cdots, p_{ym}$};

for each ($p_{y1}, \cdots, p_{ym}$的 $\mathrm{FD}_{\mathrm{XML}}$: Φ in RP)

{$K \leftarrow K \cup${Φ的左部路径};

$\sum \leftarrow \sum$ - Φ;

for each ($\forall p_{y1}, \cdots, p_{ym}$的 $\mathrm{FD}_{\mathrm{XML}}$: Φ' in LP)

{ $\sum \leftarrow \sum$ - Φ'; }

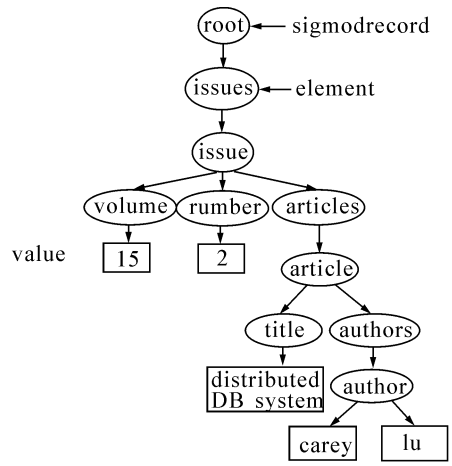
}}

}}

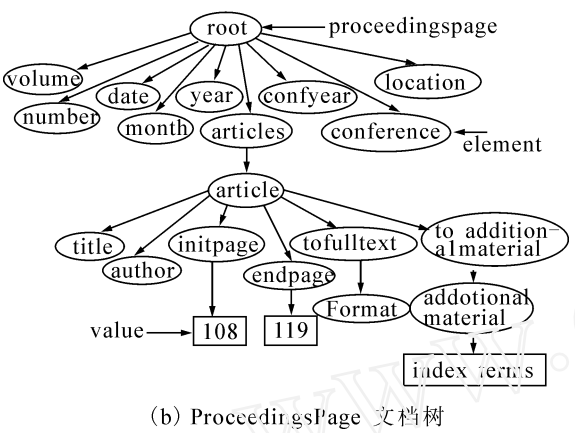
4) return(K).

例　下面选用英文的 ACM SIGMOD2007的 XML数据集进行举例分析[12], ACM SIGMOD数据集是由 53卷 XML格式的 ACM SIGMOD论文组成的,其文档结构如图 1(a)所示.

为了减少单个键值的操作冲突,一般选择相应一组 XML键构成相应集合,如图 1(a)对应键组合{sigmodrecord→issues→issue→volume, sigmodrecord→issues→issue→number, sigmodrecord→issues→issue→articles→article→title, sigmodrecord→issues→issue→articles→article→authors→author}.



(a) sigmodrecord文档树



(b) ProceedingsPage 文档树

图 1 ACM SIGMOD数据集合中的两个 XML文档的 DOM树

Fig 1 The DOM tree of two XML documents extracted from ACM SIGMOD dataset

2 多模板隐马尔科夫模型与 XML文档向量化

虽然 XML键组合有利简化 XML数据的提取,但 XML数据本身的多样性与数据清洗需要知识库作为蓝本,对于动态的 XML数据难以建立一个固定的知识库样本,因此利用 XML键组合构建隐马尔可夫模型 (HMM)的多模板功能灵活地实现 XML数据的抽取就是一个较好地选择.

一个隐马尔可夫模型[13-14]是一个五元组: $(\Omega_X, \Omega_O, A, B, \pi)$,其中: $\Omega_X = \{q_1, q_2, \ldots, q_N\}$表状态的有限集合; $\Omega_O = \{v_1, v_2, \ldots, v_M\}$表观察值的有限集合; $A = \{a_{ij}\}$, $a_{ij} = p(X_{t+1} = q_j \mid X_t = q_i)$表转移概率; $B = \{b_{ik}\}$, $b_{ik} = p(O_t = v_k \mid X_t = q_i)$表输出概率; $\pi = \{\pi_i\}$, $\pi_i = p(X_i = q_i)$表初始状态分布,那么 $\lambda = \{A, B, \pi\}$是给定 HMM的参数,则应用 HMM主要解决如下问题:评估、解码与学习问题,而 XML数据抽取需要解决 HMM中的学习与解码问题,因此整个操作划分为三大步骤: 1)以 XML键组合训练数据为几个类; 2)训练 HMM参数; 3)使用学习到的 HMM参数进行 XML数据抽取.

1)基于马尔可夫链模型,以 XML键组合训练数据为几个类.

若第 k个已标记的训练 XML文档序列用以下的转移概率矩阵 A_k表示: $A_k = (p_{kij})$,其中

$$p_{kij} = \frac{S_{kij} + \alpha_{kij}}{\sum_{j=1}^{n} (S_{kij} + \alpha_{kij})}, \quad \alpha_{kij} = \frac{\beta}{n \times n}. \tag{1}$$

式中: S_{kij}是训练 XML文档序列中从标记状态 i转移到标记状态 j的次数,通常取 $\beta = n$, n是模型状态数.

$$\pi_i = \frac{\mathrm{Init}(i)}{\sum_{j=1}^{N} \mathrm{Init}(j)}, \ 1 \leq i, j \leq N. \tag{2}$$

式中: $\mathrm{Init}(i)$表所有训练序列中,初始状态为 i的序列个数.

$$\alpha_{ij} = \frac{C_{ij}}{\sum_{k=1}^{N} C_{ik}}, \ 1 \leq i, j \leq N. \tag{3}$$

式中: C_{ij}是所有训练序列中从状态 S_i转换到状态 S_j的次数.

$$b_j(V_k) = \frac{E_j(V_k)}{\sum_{k=1}^{M} E_j(v_k)}, \ 1 \leq j \leq N, \ 1 \leq k \leq M. \tag{4}$$

式中: $E_j(V_k)$是所有训练序列中状态 S_j释放单词 V_k的次数.

若第 k个训练序列用矩阵 A_k表示,第 l个训练序列用矩阵 A_l表示,那么这 2个训练 XML文档序列之间的距离可用以下公式来计算:

$$D(A_k, A_l) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} p_{kij} \log \frac{p_{kij}}{p_{lij}} + \sum_{i=1}^{n}\sum_{j=1}^{n} p_{lij} \log \frac{p_{lij}}{p_{kij}}}{2 \times n}. \tag{5}$$

那么对任意 2个马尔可夫链,当它们具有相同的动态特征时,它们之间的距离为零. 当它们之间的动态特征差异越大,距离值就越大. 这样,就能够得到任意 2个训练 XML文档序列之间的距离,基于这种距离,通过调节距离的门槛值,可以控制得到的分类个数.

2)对每一类训练数据,依次利用式 (2)～(4)训练初始概率矩阵 π、转移概率矩阵 A以及状态释放概率 B.

3)使用训练好的模型抽取 XML数据时,结合每一个初始概率矩阵 π、转移概率矩阵 A和统一的释放概率矩阵 B,使用马尔可夫模型的韦特比算法

找出最优的标记序列,并从这些最优标记序列中选择一个概率最大的序列作为最终标记序列. 即对于每一种分类模板使用韦特比算法一次,这样每一个模板都会产生一个标记序列,从所有这些模板产生的最优标记序列中选出概率最大的序列作为最终输出结果.

虽然这样抽取 XML数据比较灵活,但有一种情形必须考虑,就是缩写词的处理,它是 XML数据清理中不可回避的问题,如中南大学信息科学与工程学院,可以叫中大信息学院,中大信息院等,都是同一个单位,在此参考文献[15]提出的动态缩写发现算法进行处理,毕竟在专业领域内,缩写的形式和内容存在一定的规律,模式比较固定,因此应用启发式规则或知识库来选择真正的缩写形式是可行的.

利用多模板的隐马尔可夫模型构建 XML数据的清洗知识库后,就可以对海量 XML数据进行相应清洗了,为了降低清洗的难度,在此首先需要把 XML文档树进行向量化.

定义 6　XML文档向量化指从 XML数据库中按照 XML键组合抽取 n组且每组包含属性长度 k个 XML文档而构成一个参照集 RS, $RS=\{(r_1,\cdots,r_k)^1,(r_{k+1},\cdots,r_{2k})^2,\cdots,(r_{(n-1)k},\cdots,r_{nk})^n\}$,其中$(r_i,\cdots,r_t,\cdots,r_{i+k-1})$的选取尽可能的按照差异最大化的原则,这个差异由 XML文档间距来评定. 通过$(r_i,\cdots,r_t,\cdots,r_{i+k-1})$将 XML数据库中的每一个 XML文档 D映射为一个 k维的距离向量 V_D, V_D的第 t维坐标可以通过计算 D和 r_t之间的语义距离得到, $V_D[t]=\mathrm{ed}(D,r)$,这样通过参照集 RS,每个 XML文档 D可被映射为 n个 k维距离向量 V_D,再将这 n个距离向量 V_D的平均值作为每个 XML文档 D在 k维坐标平面上的投影 $V_D[\bar{r}_1,\cdots,\bar{r}_k]$,这样就可以降低 XML维度,有利 XML操作.

向量化的主要目的是将一个较长的记录进行分解,根据属性大小分成更小的信息片断,对来自 2个或多个 XML数据库的数据可以将所有可能匹配的记录排在一起,组成一个序列,然后应用以下步骤匹配相应记录:

1)产生键值:在记录序列里,取每个记录的相关字段或字段的一部分按求解 XML候选键算法生成每个记录的键.

2)排序:在记录序列里应用第一步产生的键值对记录进行排序.

3)比较、合并与剔除:在排好序的记录里,利用多模板的隐马尔可夫模型提取的知识库在记录序列里按选定方向移动,在某个距离范围内选择相似点,除去相同元素、修补欠缺元素、更正错误元素等,达到清除目的.

因此本文研究的数据清洗过程用图 2表示.

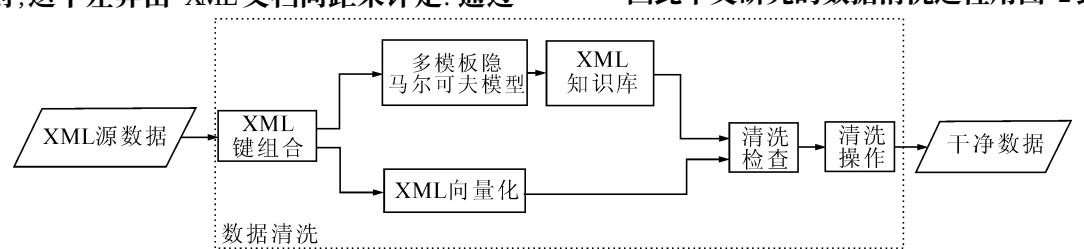


图 2　XML数据清洗过程

Fig 2　The data cleaning's process of XML

这样设计的主要优点是:

1)无需对 XML数据源数据进行假设,可以直接对 XML文档进行数据处理;

2)加强了 XML数据清洗的功能,由于 XML键的参与,数据抽取与量化有了方向,数据清洗可以更加灵活和贴近原始数据.

为了更快捷地利用抽取的知识库对量化的 XML数据进行相似性比较,特引入并优化粒子群算法参与 XML数据清洗过程.

3　粒子群算法与 XML数据清洗算法

粒子群算法[16]是近年来被广泛关注和研究的一种智能优化算法,最初由 Kennedy和 Eberhart于 1995年提出并成功地应用于函数优化,后来进行了大量地拓展. 它是对鸟群觅食过程和聚集的模拟,是一种全局优化算法,现在有许多改进版本,其优化表达式如下:

$$V_i^{t+1}=\omega\times V_i^t+c_1\times r_1\times(p_{bdi}-X_i^t)+c2\times r_2\times(p_{gd}-X_i^t),\quad(6)$$

$$X_i^{t+1}=X_i^t+\alpha V_i^{t+1},\quad(7)$$

$$\omega=\omega_{max}-(\omega_{max}-\omega_{min})\times\exp\left(1-\mathrm{sqrt}\left(\frac{run_{max}}{run}\right)\right).\quad(8)$$

式中: r为区间 $[0,1]$上的随机数,ω为惯性权重,ω

$\in[0.2, 1.2]$,其中 ω_{max}为最大惯性权重值, ω_{min}为最小惯性权重值, run为当前迭代次数, run_{max}为算法迭代总次数, α为控制速度的约束因子, c_1 是认知系数, c_2 是社会系数, $c_1 + c_2 = 4$,在此对 ω、α、c_1、c_2 统称为权重因子.

那么若抽取的 XML 文档知识库规模为 k,利用 PSO 算法进行相似性测量过程如图 3所示.

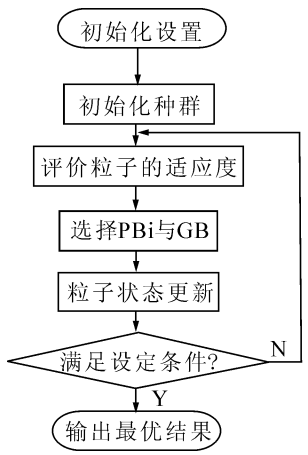


图 3 PSO操作过程

Fig 3 The operating process of PSO

在此过程中,利用定义 6将向量化的 XML 知识库投影为二维坐标平面上的样本点,同时将二维平面进行等间距网格划分,并将待检测已向量化的 XML 数据粒子随机分散于一个平面上, 即随机赋给每个粒子一对 (x, y)坐标,同时以本粒子初始对应坐标为中心, r为观察半径,利用式 (9)或 (10)计算此模式在观察半径范围内的粒子相似度,同时依次检测该网格周围的邻域网格是不是大于最少样本点数,若大于就将网格的坐标加入到邻域连接缓冲列表的尾部,否则就将网格内的所有样本点丢弃,这样依次计算已形成的每一个文档类的检测中心.

在运用 PSO 检测相似重复记录之前,需要先对数据进行一些处理,主要处理操作包括:

1)字段分裂. 依据 XML 键组合的结构,利用多模板的隐马尔可夫模型抽取各个部分;

2)验证和改正. 根据知识库及相关领域知识验证提取值的正确性,若发现错误,则加以改正;

3)数据标准化. 将同一类型的数据用统一的格式来表示,比如日期、电话号码、性别等.

4)在计算过程中,可分别针对数字字符与字符串分别套用不同的公式计算相应的距离值,图 4表示其检测过程示意图.

数字匹配问题:可采用将全部的数字字符一一比较的方法来得到标识距离,用公式

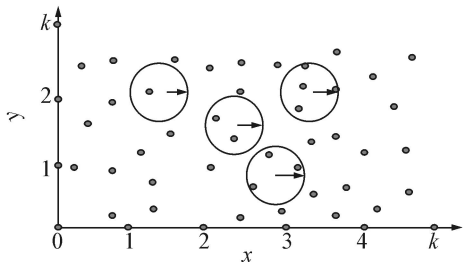


图 4 粒子相似性检测划分过程

Fig 4 The similarity checking and plotting process of particle

$$dex = 1 - \frac{\sum_{i=1}^{\min(|a|,|b|)} d(a_i, b_i) + \max(|a|, |b|) - \min(|a|, |b|)}{\min(|a|, |b|)} \tag{9}$$

计算距离, a_i、b_i 表示 2个标识对应的字符, $d(a_i, b_i)$ 表示对应字符间的距离,若 $a_i = b_i$,则 $d(a_i, b_i) = 1$, 否则 $d(a_i, b_i) = 0$; $\max(|a|, |b|) - \min(|a|, |b|)$ 表示多余的字符相应的距离.

字符串匹配问题:字符串属于长信息字段,可再分解,然后根据标识的重要性分配权值,利用

$$des = 1 - \frac{|a \cap b|}{\min(|a|, |b|)} \tag{10}$$

来计算标识距离,字段距离为标识距离的加权和,其中 a、b都是标识, $|a \cap b|$标识 a、b中相同的字符数: $|a|$表示 a的长度, $|b|$表示 b的长度.

由于粒子群算法的时间复杂度为 $O(nc \times n^3)$, 对海量 XML 文档进行检测,其代价较高,因此需要根据其量化过程对粒子群操作进行优化,并建立波函数的粒子群 (WPSO)并行处理机制.

在量子时域空间的框架下,粒子的状态由一个波函数 $\psi(x, t)$来描述,在三维空间中粒子的波函数描述为

$$|\psi|^2 dxdydx = Q dxdydz. \tag{11}$$

式中: $|\psi|^2$ 波函数的概率密度, 并且满足 $\int_{-\infty}^{+\infty} |\psi|^2 dxdydz = \int_{-\infty}^{+\infty} Q dxdydz = 1$,那么粒子群算法可优化成如下形式:

$$P = (a\ p_{id} + b\ p_{gd}) / (a + b), \tag{12}$$

$$L = (1/g).\ abs(x_{id} - p), \tag{13}$$

$$x_{id} = p \pm L.\ (\ln(1/u)). \tag{14}$$

式中: p_{id}是第 i个粒子在解空间的第 d维所找到的最优解, x_{id}是第 i个粒子在解空间第 d维的当前值, p_{gd}是所有粒子在解空间中第 d维找到的最优解, a、b、u都是 (0, 1)之间的随机数,而参数 g的选取将直接影响到算法的性能,由文献 [16]给出的参数控制

与选取的方法,只要 g满足条件: $g > \ln\sqrt{2}$就能保证算法的收敛.

因此 XML数据清洗的主要算法如下:

输入:一个 FD_{XML}集 $\sum$,路径集 Paths(T);

输出: 一个干净的 FD_{XML}集 $\sum'$;

Data Cleaning for XML ($\sum$, Paths(T)).

1) Finding a Candidate Key for XML ($\sum$, Paths (T)),获取 XML键组合 $\sum_{keyset}$.

2) Picking up a Knowledge Lib for XML ($\sum$, $\sum_{keyset}$),获取动态知识库 $\sum_{XMLlib}$.

3)利用 XML知识库 $\sum_{XMLlib}$利用 WPSO算法匹配向量化的 XML数据集.

①在给定的范围内初始化种群粒子的位置 X_0和速度 V_0;

②利用式 (9)或 (10)计算每个粒子的适应度 f_i,并将粒子群的当前位置设为 p_{id},将适应度最优位置的粒子设为 p_{gd};

③根据步骤 3.2的结果利用式 (2)～(14)计算 P、L、x_{id};

④用个体粒子的速度产生选择该粒子更新位置方程的数据:

$$rand\_q = 1/(1 + |(v_{max} - v_{id})/(v_{id} - v_{min})|). \quad (15)$$

⑤由步骤 3.2产生的数据选择更新粒子位置的方程:

if rand_q > 0.5 $x_{id} = p + L \times (\ln(1/u))$,

else $x_{id} = p - L \times (\ln(1/u))$.

⑥根据步骤 ②中 x_{id}的大小判断是否更新粒子的速度:

if $x_i^t > x_i^{t-1}$,

if $x_{id} > X_{max}\{x_{id} = X_{max}, v_{id} = -v_{id}\}$,

else if $x_{id} < X_{min}\{x_{id} = X_{min}, v_{id} = -v_{id}\}$.

在更新粒子速度时,即使粒子的速度超出了预设的范围,无需采取 $v_{id} = v_{max}$、$v_{id} = v_{min}$这种策略,这样就避免了分母中 $(v_{max} - v_{id})$和 $(v_{id} - v_{min})$这 2项为零,而式 (15)分母取绝对值,保证其结果处于 0和 1之间,也保证算法能有效运行.

4) for each (p_i in $\sum$) { //收集数据成 $\sum'$}.

本算法中的波函数粒子群算法 (WPSO)相对一般粒子群算法 (PSO)主要改进方面是:

1)它适合 XML多维数据的平面处理,而向量的投影有利降低 PSO的维度灾和简化计算规模;

2)利用个体粒子的速度产生一个介于 (0, 1)之间的数来代替原算法中的由计算机随机产生的数,以选择该粒子位置的更新;

3)如果个体粒子的位置超出了预设的范围,采取让该粒子向相反方向飞行的策略. 为了尽可能的减少计算量和更好的引导粒子向最优解靠拢以增强粒子的收敛性,规定只要当前代粒子的适应值优于上一代,就不再更新粒子的速度 V_i^{t+1}与位置 X_i^{t+1},这样减少计算工作量.

本算法虽然由 4个部分组成,但时间性能主要由 WPSO决定,相对 PSO,虽然循环代码没有多少变化,但由于采用不同的更新与投影方法,其迭代次数却大大减少.

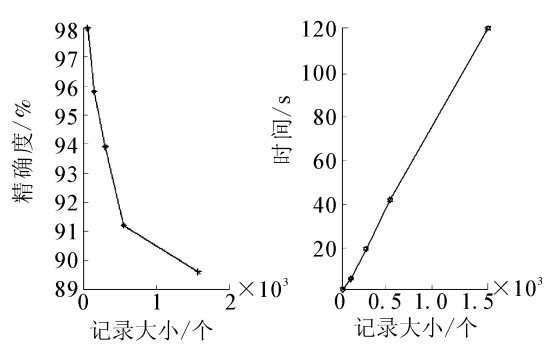
4　实验仿真

4.1　数据集与实验设计

实验使用 C-M1.73 GHzCPU, 1 GB内存, 80 GB SATA5400转硬盘的笔记本,在 Win2000专业 sp4版操作系统上利用仿真软件 Mathlab2007a和 C#开发工具进行仿真测试.

分别选用英文的 ACM SIGMOD[12]中 2006、2005、2002、1999年 4个年度 XML数据集进行实验分析,每个 ACM SIGMOD数据集都由数千篇不同 XML格式的 ACM SIGMOD论文组成的,其文档结构类似图 1格式,实验选用 1 165个文档由表 1所示.

这 4个数据集都有分类信息,在 ACM SIGMOD文档中,每个文档通常属于多个分类,而属于同一个分类的 XML文档具有较强的相似性,因此在实验中采取随机选取 3次训练样本,测试结果取平均,并分别采用基于规则、聚类与本算法进行 XML数据清洗实验评价.



(a)记录大小与相对精度比　(b)记录大小与时代价

图 5　提取不同规模的 XML数据及花费时间图

Fig 5　The precision and time of different bulk of XML data

4.2　实验结果分析

4.2.1　多模板的隐马尔可夫模型对 XML数据的抽取能力

本实验利用 XML键组合作为模板,利用隐马尔

可夫模型提取 XML数据,根据 XML数据规模统计准确提取的数据及花费时间.

4.2.2 WPSO与 PSO对 XML数据的检测能力

为了检测各算法对 XML数据的检测能力,以表 1提取的数据为基础投影到二维平面上,针对不同数据规模进行测试,实验结果如表 3所示.

表 1 实验中使用的数据子集

Table 1 Data subsets used in the experiment

	2006(ACM - 1)	2005(ACM - 2)	2002(ACM - 3)	1999(ACM - 4)	XML键个数
IndexTermsPage	0	0	0	920	13
OrdinaryIssuePage	45	45	30	51	9
ProceedingsPage	16	16	17	17	14
SigmodRecord	3	3	1	1	6

表 2 提取不同规模的 XML数据及花费时间

Table 2 Picking up different bulk of XML data and corresponding time

记录大小	文档数	有数记录数	花费时间/s
52	2	51	1.1958
143	7	137	5.7953
297	13	279	19.5516
578	22	527	49.0597
1572	36	1409	121.4384

表 3 XML数据相似性实验

Table 3 The result of XML data similarity test

记录大小	最终距离(pso/wpso)	迭代次数(pso/wpso)	花费时间(pso/wpso)/s
52	93.2479/94.8851	5029/2803	93.2479/27.2749
143	362.9409/363.3528	3729/1356	38.0556/15.3254
297	655.4254/655.8966	3564/976	50.9508/15.3275
578	1082.1/1080.0	1447/289	50.7136/30.0117
1573	3180.9/3180.3	2293/521	152.0110/52.3560

从实验结果可知由于 WPSO主要针对 XML多维数据进行投影操作,对比一般优化的 PSO算法,在最终距离相近情况下,其迭代次数与时间花费却至少降低一倍以上,而且随着数据量的加大,效果越明显.

4.2.3 不同清洗策略对比实验

为了更好的对比 XML数据清洗算法,本文特与文献[15-16]提出的规则清洗与聚类清洗算法在相同条件下进行对比实验,为了达到更好的清洗效果,针对不同数据规模加入不同噪音的测试数据,对应秩序是规则清洗、聚类清洗与粒子清洗,实验结果如表 4,图 6所示.其对应的平均准确率是 94%,90.5%、

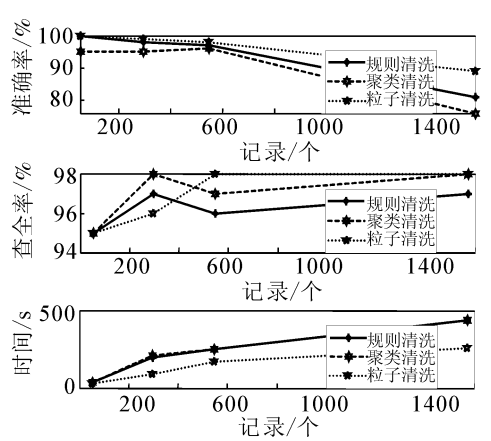


图 6 XML相似重复数据清洗对比图

Fig. 6 The result of XML similarity data cleaning

表 4 XML相似重复数据清洗实验结果

Table 4 The result of XML similarity data cleaning

记录规模	重复记录	检测结果	正确检测	准确率/%	查全率/%	时间/s
52	20	20/19/20	19/18/19	100/95/100	95/95/95	37.2/38.7/29.5
297	130	127/123/129	123/121/124	98/95/99	97/98/96	195.5/207.3/91.4
578	359	350/346/352	335/336/345	97/96/98	96/97/98	252.8/247.3/171.5
1573	539	434/410/481	423/401/472	81/76/89	97/98/98	438.2/433.7/259.6

96.5%,平均查全率是96.25%、97%、96.75%,平均节省时间率是3.26%、2.45%、94.29%,由此可知本文算法的主要优势,随着记录规模的扩大,其时间性能上表现更明显.

5　结束语

本文基于 XML键提出 XML数据清理新方法,利用多模板的隐马尔可夫模型抽取 XML数据构成动态知识库,结合波函数优化粒子群算法,简化 XML数据相似性判断过程,特别是对 XML数据进行向量化,有效地避开了粒子群算法的维度灾问题,有利数据并行比较,简化计算量,在并行处理与时间性能上有较大的改善,也为 XML数据清理的研究提供一些借鉴.

参考文献:

[1]RIERA L J, SALAZAR G J. A branch-and-cut algorithm for the continuous error localization problem in data cleaning [J]. Computers & Operations Research, 2007, 34(9): 2790-2804.

[2]ZHAO Q K, CHEN L, BHOWMICK S S, et al. XML structural delta mining: issues and challenges [J]. Data & Knowledge Engineering, 2006, 59(3): 652-680.

[3]郭志懋,周傲英. 数据质量和数据清洗研究综述 [J]. 软件学报, 2002, 13(11): 2076-2083.
GUO Zhimao, ZHOU Aoying. Research on data quality and data cleaning: a survey [J]. Journal of Software, 2002, 13(11): 2076-2083.

[4]郑仕辉,周傲英,张　龙. XML文档的相似测度和结构索引研究 [J]. 计算机学报, 2003, 26(9): 1116-1123.
ZHENG Shihui, ZHOU Aoying, ZHANG Long. Similarity measure and structural index of XML documents [J]. Journal of Computer, 2003, 26(9): 1116-1123.

[5]陈　伟,丁秋林. 一种 XML相似重复数据的清理方法研究 [J]. 北京航空航天大学学报, 2004, 30(9): 835-838.
CHEN Wei, DING Qiulin. Study on an XML approximately duplicated data cleaning method [J]. Journal of Beijing University of Aeronautics and Astronautics, 2004, 30(9): 835-838.

[6]陆凤霞,王静秋,王宁生. 一种开放式数据清理框架 [J]. 南京航空航天大学学报, 2006, 38(4): 459-463.
LU Fengxia, WANG Jingqiu, WANG Ningsheng. Open data cleaning frame [J]. Journal of Nanjing University of Aeronautics and Astronautics, 2006, 38(4): 459-463.

[7]王　桐,刘大昕. 一种基于改进粒子群优化的 XML结构聚类方法 [J]. 小型微型计算机系统, 2007, 28(5): 871-875.
WANG Tong, LIU Daxin. XML structural clustering based on the improved particle swarm optimization [J]. Journal of Chinese Computer Systems, 2007, 28(5): 871-875.

[8]NAYAK R, IRYADI W. XML schema clustering with semantic and hierarchical similarity measures [J]. Knowledge-Based Systems, 2007, 20(6): 336-349.

[9]RIERA L J, SALAZAR G J. A branch-and-cut algorithm for the continuous error localization problem in data cleaning [J]. Computers & Operations Research, 2007, 34(9): 2790-2804.

[10]RIERA L J, SALAZAR G J. A heuristic approach for the continuous error localization problem in data cleaning [J]. Computers & Operations Research, 2007, 34(8): 2370-2383.

[11]LEE C S. Diagnostic, predictive and compositional modeling with data mining in integrated learning environments [J]. Computers & Education, 2007, 49(3): 562-580.

[12]Sigmod. Sigmod Record [EB/OL]. [2007-05-29]. http://www.sigmod.org/record/xml/index.xml

[13]GEMELLO R, MANA F, SCANZIO S, et al. Linear hidden trans-formations for adaptation of hybrid ANN/HMM models [J]. Speech Communication, 2007, 49(10): 827-835.

[14]CHARITOS T, WAAL P R, GAAG L C. Convergence in Markovian models with implications for efficiency of inference [J]. International Journal of Approximate Reasoning, 2007, 46(2): 300-319.

[15]叶　舟,王　东. 基于规则引擎的数据清洗 [J]. 计算机工程, 2006, 32(23): 52-55.
YE Zhou, WANG Dong. Rules engine based data cleansing [J]. Computer Engineering, 2006, 32(23): 52-55.

[16]冯玉才,桂　浩,李　华,等. 数据分析和清理中相关算法研究 [J]. 小型微型计算机系统, 2005, 26(6): 1018-1022.
FENG Yucai, GUI Hao, LI Hua, et al. Research on related algorithms in data analysing and cleaning [J]. Journal of Chinese Computer Systems, 2005, 26(6): 1018-1022.

作者简介:

刘　波,男,1969年生,博士研究生,主要研究方向为软件工程与数据库技术.

杨路明,男,1947年生,教授,博士生导师,主要研究方向为信息系统与数据库技术,发表学术论文 40余篇.

邓云龙,男,1962年生,教授,博士生导师,博士,主要研究方向为软件心理学.获得科研教学成果多项.发表学术论文 50余篇.