文章编号: 1006-2475(2009)03-0123-04

CFE: 一种基于连分数的动态 XML编码

曾志民, 江 弋, 张东站

(厦门大学计算机科学与技术系, 福建 厦门 361005)

摘要: 论述了一种基于连分数的动态 XML编码, 首先介绍了 CFE编码的概念, 在此基础上把 CFE应用到区间编码和前缀编码, 接着对 CFE编码的更新算法进行了阐述, 最后进行实验对比, 说明 CFE编码是可行的。

关键词: 动态 XML; 编码模式; 连分数

中图分类号: TP311 文献标识码: A

CFE: A Continued Fraction Based on Encoding for Dynamic XML Data

ZENG Zhi-min, JIANG Yi, ZHANG Dong-zhan

(Department of Computer Science and Technology, Xiamen University, Xiamen 361005, China)

Abstract: This paper introduces a continued fraction based encoding for dynamic XML data. Firstly presents what is CFE encoding. Then applies it to region encoding and prefix encoding. And proposes an algorithm for dynamic update XML data for CFE encoding. Finally, the result shows that CFE encoding is effective.

Key words: dynamic XML; labeling scheme; continued fraction

0 引 言

由于 XML的灵活性和自描述性, XML已经成为 Internet上数据交换的事实标准。基于 XML的查询和处理的研究引起学术界和工业界的广泛关注。XML查询语言主要有 XPath[1]、XQuery[2]等, 这些查询语言共同的核心技术之一就是利用路径表达式来表示和检索用户所指定的结构模式, 实现 XML的结构查询, 其基础是元素或属性在 XML文档中的父子关系或祖先-后代关系的判断。为了有效地支持 XML的结构查询, 研究者提出了各种编码算法, 包括静态编码算法[3-7]和动态编码算法[8-12], 对 XML文档中的元素进行惟一标识以迅速判定两个元素之间是否父子关系、祖先-后代关系或它们的文档次序(document order)。

本文提出的动态编码算法 CFE具有以下几个特点: (1) CFE支持 XPath的所有结构化查询, 可以确定任意两个 XML节点之间的祖先-后代关系, 父子关系和文档次序; (2) CFE可以在任意两个相继的 CFE码之间插入一个新的 CFE码, 所以当更新 XML文档时, CFE可以完全避免重新标识现有的节点; (3) CFE与区间编码模式、前缀编码模式成正交关系, 可以与之结合提高更新性能; (4)本文对提出的算法进行了相应的模拟实验, 实验结果表明 CFE算法具有良好的查询和更新性能, 具有实际的可行性。

1 CFE编码

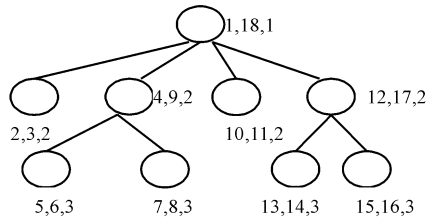
定义 1 已知 i, n为正整数, 且 1≤ i≤ n, 那么数字 i的编码为 cfcode(i) = [0, n+2- i]。其中 cfcode是用连分数进行表示, 即 $[a, b, c, \cdots] = a + \cfrac{1}{b + \cfrac{1}{c + \cdots}}$。

1.1 CFE的应用

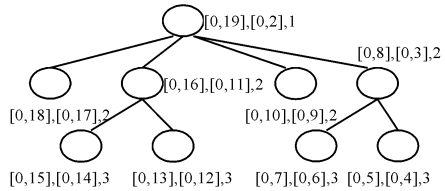
CFE编码与前缀编码、区域编码等成正交关系。当把 CFE编码应用到区域编码[3]的时候, 称之为

收稿日期: 2008-11-18
基金项目: 国家自然科学基金资助项目(50604012)
作者简介: 曾志明(1984-), 男, 福建莆田人, 厦门大学计算机科学与技术系硕士研究生, 研究方向: XML, 数据库, 数据仓库; 江弋, 男, 副教授, 研究方向: 数据库技术和数据挖掘; 张东站, 男, 讲师, 博士, 研究方向: 数据库理论与应用, 矿业信息系统。

Range-CFE。在 Range-CFE中,每个节点用(startCF, endCF, level)来标识,其中 startCF、endCF是简单连分数,图 1(b)是 Range-CFE的一个实例。



(a)区域编码模式



(b)Range-CFE编码模式

图 1 应用 CFE到区域编码模式

类似地, CFE也可以应用到前缀编码模式,仍然保持文档次序。当 CFE应用到前缀编码时,称为 Prefix-CFE。

例子 1: 图 2显示的是把 CFE应用到前缀编码模式,由于根节点是惟一的,不需要进行标识。根节点有 4个孩子,从左到右分别用[0, 5], [0, 4], [0, 3], [0, 2]进行标识,节点[0, 4]有 2个孩子,从左到右分别用[0, 4]. [0, 3], [0, 4]. [0, 2]进行标识。

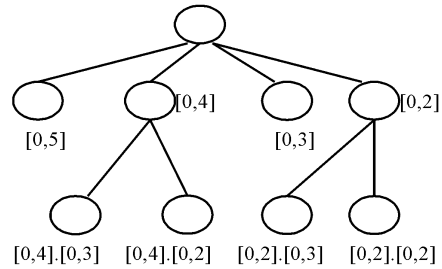


图 2 Prefix-CFE编码模式

1.2 判断祖先-后代关系和文档次序

定理 **1** Range-CFE中,节点 A是节点 D的祖先,当且仅当 A的标识($startCF_A$, $endCF_A$, $level_A$), D的标识($startCF_D$, $endCF_D$, $level_D$)满足以下不等式:

$$startCF_A < startCF_D < endCF_D < endCF_A$$

定理 **2** Range-CFE中,节点 A是节点 D的父亲节点,当且仅当 A是 D的祖先,且 A节点的标识($startCF_A$, $endCF_A$, $level_A$), D节点的标识($startCF_D$, $endCF_D$, $level_D$)满足以下等式:

$$level_A - level_D = 1$$

定理 **3** Prefix-CFE中,节点 A是节点 D的祖先,当且仅当 A的标识是 D的标识的前缀。节点 A是节点 D的父亲,当且仅当 A的标识是 D的标识的前缀且节点 A的标识的分隔符的个数比节点 D的标识的分隔符的个数少 1。

定理 **4** Range-CFE中,节点在节点 D的前面,即节点 A的文档次序小于节点 D,当且仅当 $startCF_A < startCF_D$。

定理 **5** Prefix-CFE中,设节点 A的标识 label(A) = $a_0, a_1, a_i, x_{i+1}, \cdots, x_s$,节点 D的标识 label(D) = $a_0, a_1, a_i, y_{i+1}, \cdots, y_t$, $x_{i+1} \neq y_{i+1}$,节点 A出现在节点 D的前面,即节点 A的文档次序小于节点 D,当且仅当 x_{i+1}为空或 $x_{i+1} < y_{i+1}$。

1.3 编码的存储

CFE编码采用 Utf-8格式存储。对于 Range-CFE,譬如一个节点的标识为"[0, 19], [0, 2, 8], 1",那么存储格式为"19 2, 8 1"。存储遵循以下规则:(1)每个不同的项之间用空格分开。(2)对于同一个连分数,开头的 0不存储。(3)对于同一个连分数,不同的项用逗号隔开。

对于 Prefix-CFE,存储规则为:(1)不同的连分数之间用空格分开。(2)同一个连分数内的不同项用逗号分隔。(3)对于同一个连分数,开头的 0不存储。譬如一个节点的标识为"[0, 4]. [0, 3, 2]",那么存储格式为"4 3, 2"。

2 CFE对 XML文档更新的处理

对于动态 XML文档,由于频繁插入、删除节点,如何最大限度地降低更新的代价,是一个很重要的问题。CFE编码的一个重要特点就是可以在任意的两个 CFE码之间插入一个新的 CFE码,而不需要重新标识已有的节点。推论 1说明了如何在任意的两个 CFE编码之间插入一个新的 CFE码,我们称之为 CFE更新算法 GetInsertedCode。

推论 **1** (CFE更新算法 GetInsertedCode)已知 A, B为简单连分数, $0 < A < B < 1$, $A = [0, a_2, \cdots, a_i, x_{i+1}, \cdots, x_s]$, $B = [0, a_2, \cdots, a_j, y_{j+1}, \cdots, y_t]$, $x_{i+1} \neq 1$, $y_{j+1} \neq 1$, $x_{i+1} \neq 1y_{j+1}$,当满足以下任一条件时,得到的连分数 C满足 $A < C < B$:

(1) $s = j \geq j+1$, $C = [0, a_2, \cdots, a_j, y_{j+1} + 100]$;

(2) $s \geq j+1$, $t = j$, $C = [0, a_2, \cdots, a_j, x_{j+1} + 100]$;

(3) $s \geq j+1$, $t \geq j+1$, $|x_{j+1} - y_{j+1}| > 1$, $C = [0, a_2, \cdots, a_j, (x_{j+1} + y_{j+1})/2]$;

(4) $s = j+1$, $t = j+1$, $|x_{j+1} - y_{j+1}| = 1$, j为偶数, $C = [0, a_2, \cdots, a_j, x_{j+1}, 100]$;

(5) $s \geq j+2$, $t = j+1$, $|x_{j+1} - y_{j+1}| = 1$, j为偶数。若 x_{j-2}

= 2, 则 C= [0, a_2, …, a_j, x_{j+1}, 1, 100]; 否则 C= [0, a_2, …, a_j, x_{j+1}, x_{j+2} - 1];

(6) s≥ j+ 1, t≥ j+ 2, $|x_{j+1} - y_{j+1}| = 1$, j为偶数, 则 C= [0, a_2, …, a_j, y_{j+1}, y_{j+2} + 100];

(7) s= j+ 1, t= j+ 1, $|x_{j+1} - y_{j+1}| = 1$, j为奇数, 则 C= [0, a_2, …, a_j, y_{j+1}, 100];

(8) s= j+ 1, t≥ j+ 2, $|x_{j+1} - y_{j+1}| = 1$, j为奇数。若 y_{j+2} = 2, 则 C= [0, a_2, …, a_j, y_{j+1}, 1, 100]; 否则 C= [0, a_2, …, a_j, y_{j+1}, y_{j+2} - 1];

(9) s≥ j+ 2, t≥ j+ 1, $|x_{j+1} - y_{j+1}| = 1$, j为奇数, 则 C= [0, a_2, …, a_j, x_{j+1}, x_{j+2} + 100];

例子 2　(a)当 A= [0, 1], B= [0, 1, 3]时, 此时 s = j= 2, t= 3≥ j+ 1, 符合情况 1, 所以 C= [0, 1, 103], 显然 A< C< B; (b)当 A= [0, 1, 3, 5], B= [0, 1, 3]时, j= 3, s= 4≥ j+ 1, t= 3= j 符合情况 2, 所以 C= [0, 1, 3, 105]; (c)当 A= [0, 1, 3], B= [0, 1, 9]时, j = 2, s= 3≥ j+ 1, t= 3≥ j+ 1, $|x_3 - y_3| > 1$, 符合情况 3, 所以 C= [0, 1, 6]。

2.1 Range-CFE编码模式对 XML更新的处理

对 XML文档的更新包括插入或删除一个 XML元素、文本、属性或者是更改一个文本。删除或更改操作都不涉及到编码, 只有插入操作才涉及到新节点的编码。对 XML节点的插入可分为四种情况: (1)插入的节点没有兄弟节点; (2)插入的节点只有左边兄弟节点; (3)插入的节点只有右边兄弟节点; (4)插入的节点左右两边都有兄弟节点。以下分四种情况进行讨论。

(1)节点 A的标识为 ($startCF_A$, $endCF_A$, level), 没有子节点。现在往 A节点插入一个子节点 D。D的标识计算如下:

$startCF_D$ = GetInsertedCode ($startCF_A$, $endCF_A$)。

$endCF_D$ = GetInsertedCode ($startCF_D$, $endCF_A$)。

(2)设 A节点的最右边的子节点为 B, 现在给 A节点添加一个子节点 D, 插入点在节点 B的右边。那么:

$startCF_D$ = GetInsertedCode ($endCF_B$, $endCF_A$)。

$endCF_D$ = GetInsertedCode ($startCF_D$, $endCF_A$)。

(3)设 A节点的最左边的子节点为 B, 现在给 A节点添加一个子节点 D, 插入点在节点 B的左边。那么:

$startCF_D$ = GetInsertedCode ($startCF_A$, $startCF_B$)。

$endCF_D$ = GetInsertedCode ($startCF_D$, $startCF_B$)。

(4)设 A, B为连续的兄弟节点, A在 B的左边, 假设在 A, B之间插入一个新的节点 D, 那么:

$startCF_D$ = GetInsertedCode ($endCF_A$, $startCF_B$)。

$endCF_D$ = GetInsertedCode ($startCF_D$, $startCF_B$)。

2.2 Prefix-CFE编码模式对 XML更新的处理

定义 2　Prefix-CFE编码中, 每个节点的编码由 parentlabel和 selflabel两部分构成。譬如节点 u是节点 p的子节点, 那么 label(u) = label(p) + "." + selflabel(u), parentlabel(u) = label(p), 其中 selflabel(u)是个连分数。

Prefix-CFE对 XML节点的插入同样也分为四种情况:

(1)设 A节点没有任何子节点, 现在往 A节点插入一个子节点 D。D的标识 label(D) = label(A) + "." + [0, mid]。mid是任意大于 1的正整数, 可以根据 A节点的子节点个数和 D的位置选择一个最佳的 mid值。

(2)设 A节点的最右边的子节点为 B, 在节点 B的右边给 A节点添加一个子节点 D, 已知 selflabel(B) = [0, a_2, a_3, …, a_n]。

如果 n= 2且 $a_2 > 2$, 那么 label(D) = label(A) + "." + [0, a_2-1]。

如果 n= 2且 $a_2 = 2$, 那么 label(D) = label(A) + "." + [0, 1, 2]; 否则, label(D) = label(A) + "." + [0, a_2, a_3 + 1]。

(3)设 A节点的最左边的子节点为 B, 在节点 B的左边给 A节点添加一个子节点 D, 那么 label(D) = lablel(A) + "." + [0, tempB + 1], 其中 selflabel(B) = [0, a_2, a_3, …, a_n], tempB = a_2。

(4)设 A, B为连续的兄弟节点, A在 B的左边, 其父节点为 P, 在 A和 B之间插入一个新的节点 D, 那么 label(D) = label(p) + "." + GetInsertedCode (selflabel(A), selflabel(B))。

3　实验分析

本文对四种动态编码方法 OrdPath[9]、QED[10]、Range-CFE、Prefix-CFE, 在查询时间、更新性能等方面都做了实验比较。所有编码都用 C#实现, 实验平台为 2.02GHz Celeron处理器, 512MB内存, 操作系统为 Windows XP Professional, XML测试数据集来自于文献[13], 各数据集的特点见表 1。

表 1　测试数据集

数据集	主题	文件数	XML元素的最大扇出	XML元素的最大深度	数据集的节点总数
D1	Club	12	47	3	2928
D2	Movie	490	38	4	26044
D3	ednunds	1190	162	3	234400
D4	Shakespeare's plays	37	434	5	179689

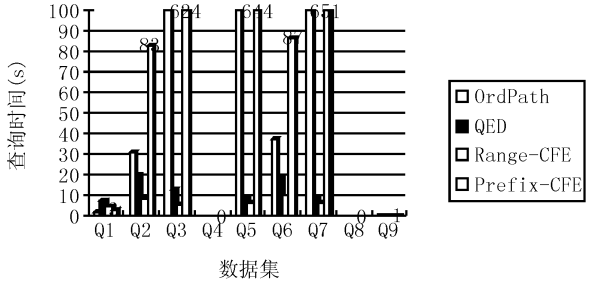


图 3 查询时间比较

3.1 查询时间

本实验采用 Shakespeare's plays(D4)数据集来测试四种编码的查询性能。测试的查询语句和返回的节点数见表 2。从图 3可以看出, Range-CFE的查询性能最好, QED次之, Prefix-CFE的性能最差,说明 CFE比较适合用在区域编码上,不适合用于前缀编码。

表 2 查询语句

查询		返回节点数
Q1	/play/act[4]	37
Q2	/play/act[5]//preceding::scene	611
Q3	/play/act/scene/speech[2]	730
Q4	/play/*/*	1938
Q5	/play/act//speech[3]/preceding-sibling::*	3093
Q6	/play//act[2]/following::speaker	18406
Q7	/play//scene/speech[6]/following-sibling::speech	26705
Q8	/play/act/scene/speech	30933
Q9	/play/*//line	107833

3.2 更新性能

假设 Club(D1)、Movie(D2)、edmunds(D3)、Shakespeare's plays(D4)中元素是有序的。本文测试了以下 4种情况各种编码的更新性能; 在 Club中任意两个相邻的元素间插入一个新的元素; 在 Movie中任意两个相邻的元素间插入一个新的元素; 在 edmunds中任意两个相邻的元素间插入一个新的元素; 在 Shakespeare's plays中任意两个相邻的元素间插入一个新的元素。从图 4可以看出, Range-CFE的更新性能最好, QED次之, OrdPath相比 Prefix-CFE要好一点。

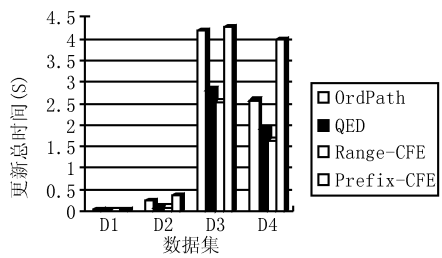


图 4 更新性能

4 结束语

本文提出的基于连分数的 XML编码算法 CFE,不仅支持在任意的两个 CFE码之间插入一个新的 CFE码,而且 CFE与前缀编码、区域编码等成正交关系。通过把 CFE应用于前缀编码、区域编码,本文提出了 Range-CFE编码模式和 Prefix-CFE编码模式。Range-CFE和 Prefix-CFE都支持任意两个 XML节点的祖先-后代关系、父子关系以及文档次序的快速判断。应用这两个编码模式,当任意插入或删除 XML节点时,都不需要重新标识已有的节点。最后,实验结果表明 Range-CFE编码模式的更新和查询处理性能明显优于现有的编码算法。后续工作将探讨如何在 CFE编码上建立索引以更加适合 XPath查询的需要。

参考文献:

[1] Berglund A, Boag S, Chamberlin D, Fernandez M F, Kay M, Robie J, Simon J. XML Path Language (XPath) 2.0. W3C Working Draft[DB/OL]. http://www.w3.org/TR/2002/WD-xpath20-20020816/, 2002-08-16.

[2] Li Q, Moon B. Indexing and querying XML data for regular path expressions[C]//The 27th Int'l Conf. on Very Large Data Bases. Roma: Morgan Kaufmann Publishers, 2001: 361-370.

[3] Zhang C, Naughton J, DeWitt D, et al. On supporting containment queries in relational database management systems[C]//Proc of the ACM SIGMOD Conf. C. Santa Barbara: ACM Press, 2001: 425-436.

[4] Wan C X, Liu Y S. X-RESTORE: Middleware for XML's relational storage and retrieve[J]. Wuhan University Journal of Natural Science, 2003, 8(1A): 28-34.

[5] Tatarinov I, Viglas S, Beyer K S, Shanmugasundaram J, Shekita E J, Zhang C. Storing and querying ordered XML using a relational database system[C]//Proc. of the 2002 ACM SIGMOD International Conference on Management of data. Madison: ACM Press, 2002: 204-215.

[6] Cohen E, Kaplan H, Milo T. Labeling dynamic XML trees[C]//SPDS. Madison: ACM Press, 2002: 271-281.

[7] Wu X, Lee M L, Hsu W. A prime number labeling scheme for dynamic ordered XML trees[C]//Proc. of 19th International Conference on Data Engineering (ICDE). Boston: IEEE Computer Society, 2004: 66-78.

[8] Amagasa T, Yoshikawa M, Uemura S. QRS: A robust numbering scheme for XML documents[C]//Proc. of 19th International Conference on Data Engineering (ICDE). Bangalore: IEEE Computer Society, 2003: 705-707.

(下转第 129页)

3 实验结果及分析

我们收集了约 4GB的 800首歌曲, 其中 700首是独唱的流行歌曲, 50首二重唱的歌曲, 50首合唱歌曲。使用 200首歌曲作为训练集, 其余 600首作为测试集。分类的准确度采用分类精度 (accuracy)来测量:

$$\text{分类精度} = \frac{\text{分类正确的样本 clip数}}{\text{样本 clip总数}} \times 100\%$$

分类结果分析如表 1所示。

表 1 不同类型歌曲分类结果

歌曲类别		分类精度
独唱	快节奏	89%
	慢节奏	94%
二重唱		66%
合唱		83%

从试验结果可以看出, 对于慢节奏的独唱歌曲分割效果最好, 其次是快节奏的独唱歌曲, 合唱歌曲效果也不差, 但是, 二重唱误差较大。分析其原因是由于二重唱中, 演唱者之间的语音参数存在较大差别, 从而干扰了对演唱片段与间奏片段的两类划分。

总体来说, SVM 对于歌曲分割的正确率较高, 特别是对于独唱歌曲, 由于二重唱在歌曲中所占比例较小。因此, 此方法实际应用价值比较高。

4 结束语

从歌曲中切分出间奏片段与演唱片段是提取歌曲主旋律中极其重要的一个环节, 对后续的特征提取有很大影响。本文提出一种基于支持向量机的切分方法, 经试验证明效果良好, 能有效完成切分任务。下一步研究方向在两个方面: 一是提高对二重唱的切分准确率; 二是应用于不含人声的纯乐曲的主旋律提取中。

参考文献:

[1] Wold E, Blum T, Keislar D, Wheaten J. Content-based classification, search and retrieval of audio[J]. IEEE Multimedia, 1993, 3(3): 27-36.

[2] 卢坚, 陈毅权, 孙正兴, 等. 语音 /音乐自动分类中的特征分析 [J]. 计算机辅助设计与图形学报, 2003, 14(3): 233-237.

[3] 白亮, 老松杨, 陈剑赟, 等. 基于支持向量机的音频分类与分割 [J]. 计算机科学, 2005, 32(4): 87-91.

[4] Vanpnik V. Nature of Statistical Learning Theory[M]. John Wiley and Sons, Inc, New York in Preparation, 1996.

[5] Scheirer E, Slaney M. Construction and evaluation of robust multifeature music/speech discriminator[C] //Proc. of ICASSP 97. Vol II, 1997: 1331-1334.

[6] 边肇祺, 等. 模式识别 [M]. 北京: 清华大学出版社, 1988.

[7] Jonathan T Foote. Content-based retrieval of music and audio[C] //Multimedia Storge and Archiviing Systems II, Proc. of SPIE. Volume 3229, 1997: 138-147.

[8] Li S H. Content-based classification and retrieval of audio using the nearest feature line method[J]. IEEE Transactions on Speech and Audio Processing, 2000, 8(5): 619-625.

[9] Pfeiffer S, Fisher S, Effelsberg W. Automatic audio content analysis[C] //Proc. of the Fourth ACM Int Conf on Multimedia, 1996: 21-30.

[10] Saunders J. Real-time discrimination of broadcast speech/music[C] //Proc. of ICASSP96. Vol II, Atlanta, May, 1996: 99-105.

[11] Cortes C, Vapnik V. Support vector machine learning[J]. Machine Learning, 1995, 20: 273-297.

[12] Moreno P J, Rifkin R. Using the fisher kernel method for audio classification[C] //Proc. of ICASSP2000. Vol IV. 2000: 2417-2440.

[13] Zhang Tong, Kuo C J. Heuristic approach for generic audio data segmentatoin and classification[C] //Proc. of the 7th ACM Int Conf on Multimedia, Orlando, 1999: 67-76.

(上接第 126页)

[9] O'Neil P, O'Neil E, Pal S, Cseri I, Schaller G. ORDPATHs: Insert-friendly XML node labels[C] //Weikum G, Konig A C, DeBloch S, et al. Proc. of the ACM SIGMOD Int'l Conf. on Management of Data (SIGMOD). Paris: ACM Press, 2004: 903-908.

[10] Li C, Ling T W. QED: A novel quaternary encoding to completely avoid re-labeling in XML updates[C] // CIKM. Bremen: ACM Press, 2005: 501-508.

[11] Min Jun Ki, Lee Jihyun, Chung Chin Wan. An efficient encoding and labeling for dynamic XML data[C] //Proc. of 12th International Conference on Database Systems for Advanced Applications (DASFAA). Bangkok, Thailand: Springer-Verlag, 2007: 715-726.

[12] Changqing Li, Tok Wang Ling, Min Hu. Efficient processing of updates in dynamic XML data[C] //ICDE. Atlanta, Georgia: IEEE Computer Society, 2006: 13-23.

[13] 路燕, 张亮, 汪卫, 张彪, 施伯乐. 一种新的 XML文档编码机制 [J]. 计算机研究与发展, 2004, 41(3): 500-503.