第 37 卷 第 15 期
Vol.37 No.15
计 算 机 工 程
Computer Engineering
2011 年 8 月
August 2011
• 软件技术与数据库 •
文章编号：1000—3428(2011)15—0034—03
文献标识码：A
中图分类号：TP311

基于结构标记树的 XML 可查询压缩方法

魏东平，徐瑞敏，贾 楠
(中国石油大学计算机与通信工程学院，山东 东营 257061)

摘 要：针对支持查询的 XML 数据压缩方法存在的路径和数据重复等问题，通过去除 XML 数据中的重复路径，简化 XML 数据结构，提出结构标记树的概念及其生成算法，设计一种基于结构标记树的可查询 XML 数据压缩方法 SSTQC，对 XML 数据进行压缩和组织查询。SSTQC 一次扫描 XML 文档，具有较好的的压缩性能和查询效率。

关键词：XML 数据；数据压缩；查询处理；重复路径；结构标记树

XML Queryable Compression Method Based on Structure Sign Tree

WEI Dong-ping, XU Rui-min, JIA Nan
(Institute of Computer and Communication Engineering, China University of Petroleum, Dongying 257061, China)

【Abstract】Aiming at some problems of present XML queryable compressors, this paper proposes the definition and detailed algorithm of structure sign tree, which can simplify the structure of XML data by removing the repeated paths. On the basis, a new XML queryable compressor SSTQC is put forward to compress XML data and organize queries. SSTQC requires only a single pass over the XML document, and it has excellent compression performance and better query efficiency.

【Key words】XML data; data compression; query processing; repeated paths; Structure Sign Tree(SST)
DOI: 10.3969/j.issn.1000-3428.2011.15.009

1 概述

XML 数据的自描述性使其存在大量的冗余信息，因此，XML 数据压缩技术便成为提高 XML 数据管理的一种有效途径。如何利用 XML 的特征进行高效的压缩，同时又不会对查询处理带来过多的冗余操作，是 XML 压缩的 2 个关键问题。本文对现有的可查询 XML 数据压缩方法进行比较分析，本文提出一种新的压缩方案——基于结构标记树的可查询压缩方法 SSTQC，基于结构标记树对 XML 数据进行压缩和组织查询，并进行了实验验证。

2 相关研究

XGrind[1]是第一个支持查询处理的 XML 压缩方法，采用非自适应的 Huffman 编码。XGrind 只针对冗余标签，没有解决重复路径和重复数据问题，压缩数据时需要 2 次扫描 XML 数据，并且无法对所有的复杂查询进行直接解析。

XPress[1-2]采用倒装算术编码，增强了对查询的支持，有效解决了复杂路径中的后代查询问题，并根据数据特征选择使用字典编码或 Huffman 编码。同样，XPress 需要扫描 XML 数据 2 遍，不能满足数据流环境的要求。

XQzip[3]合并等价的子树，消除 XML 数据中的重复结构得到结构索引树，数据项用 gzip 压缩成一些物理块。但是结构索引树中仍然存在重复路径问题，当 XML 数据较大时，构造过程将占用过多的资源。

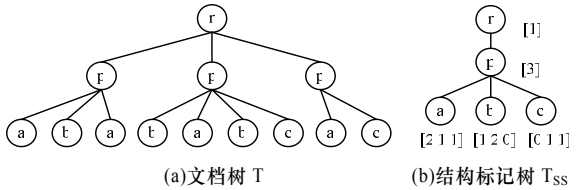
XQueC[4]用更灵活的方式将 XML 的结构信息和数据项分离，但单独压缩每个数据项导致压缩率的降低，为支持查询，建立了结构树和索引树，这些结构信息连同指向单独压缩的数据项的指针产生了很大的空间开销。

3 SSTQC 的主要思想

3.1 相关概念

定义 1(结构标记树 SST) 相对于原 XML 文档 T，结构标记树 T_{SS} 是一棵无序树，并且满足以下性质和条件：(1)T_{SS} 中不存在重复路径。(2)T_{SS} 中的每一个节点对应 T 中一个节点集合，且 T 中的每一个节点在 T_{SS} 中都有唯一的节点与之对应。(3)T_{SS} 中的任一节点 u 都对应一个唯一的标记(亦称计数器)S(u)=[u1,u2,…,uk]。根节点 r 的 S(r)=[1]，对于 T_{SS} 的一条路径/a/b，S(a)=[a1,…,ai]，S(b)=[b1, b2,…,bj]，其中，a1+a2…+ai 是节点 a 集合中元素的个数，b1+b2…+bj 是节点 b 集合中元素的个数，并且 j=a1+a1…+ai，即 b1,b2,…,bj 分别对应 T 中每个节点 a 下的孩子 b 的个数。

举例说明：图 1(a)是 XML 文档 T，T 中存在 3 条/r/p、4 条/r/p/a、3 条/r/p/b 和 2 条/r/p/c，去除重复路径后，得到图 1(b)中的结构标记树 T_{SS}。r 是根节点，所以，S(r)=[1]；3 条重复路径/r/p 合并后，S(p)=[3]；3 个 p 节点下孩子 a 的数目分别是 2、1、1，即 S(a)=[2,1,1]。同理，S(b)=[1,2,0]，S(c)=[0,1,1]。



(a)文档树 T (b)结构标记树 T_{SS}

图 1 XML 文档树及其结构标记树

定义 2(<) 在 XML 文档中有 2 个兄弟节点 a 和 b，如果 a 出现在 b 之前，则称 a<b；如果 b 出现在 a 之前，则称 b<a。

定义 3(环) 如果 XML 文档中出现以下 2 种情况之一：(1)节点 p 有 2 类孩子节点 a 和 b，其中有 a<b 且 b<a；(2)节

作者简介：魏东平(1965－)，男，副教授，主研方向：数据库技术，软件工程；徐瑞敏、贾 楠，硕士研究生
收稿日期：2011-01-07 **E-mail：**xurm1986@sina.com

点 p1、p2 有相同的路径，其中，p1 的 2 个孩子存在 a<b，而 p2 的 2 个孩子存在 b<a，则称 XML 文档中存在环。图 1(a)中同时存在这 2 种环。

3.2　SSTQC 的结构

如图 2 所示，对于给定的 XML 文档，用 SAX 解析器进行解析编码，将结构数据(标签名和属性名)和内容数据(元素内容和属性值)分离，结构数据在 SST 构建模块中生成 SST，内容数据则根据数据类型的不同，分发到不同的容器中进行压缩；生成的 SST 进一步处理，得到结构树和标记两部分；最后，通过数据压缩手段得到压缩文件。对于查询过程，由查询解析器解析查询语句，根据语句查询类型，解压得到 SST 和相关索引，通过部分解压缩实现查询处理的目的。图中，实线对应的是压缩过程，虚线对应的是查询处理。

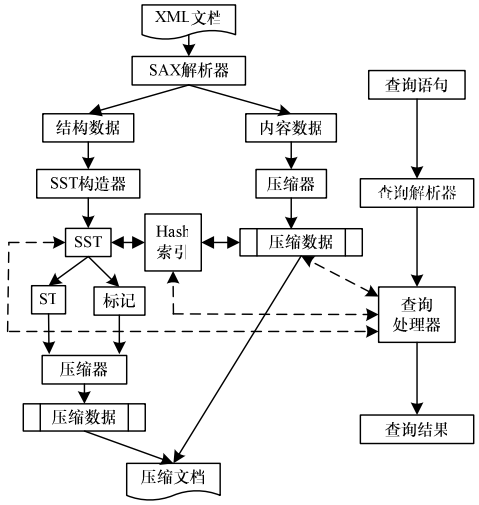


图 2　SSTQC 结构

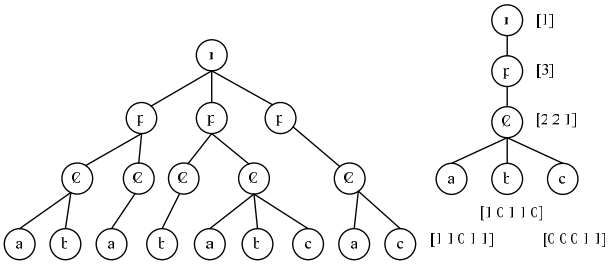
3.3　SST 的构建

SST 是一棵去除了重复路径的无序树，没有考虑元素事件出现的先后顺序，仅依赖该元素第一次出现的位置及其发生频率。由于环的存在，因此建立的 SST 进行还原时，元素将聚集出现，与原文档相比，压缩后的数据遭到破坏，结构内容混乱，这就违背了进行压缩的本意。

3.3.1　构造思路

将 XML 数据看作复合多样的图，利用图论的知识，可以发现并处理 XML 文档中的环。

改进策略：将 XML 文档中某一父亲节点及其孩子作为图的顶点，在解析过程中根据节点出现的顺序依次创建边。如果图中出现环路，则表示 XML 文档存在环，此时，为该父亲节点添加一个虚拟的子节点 Ø，将出现环路之前的孩子节点作为该虚拟节点的孩子，见图 3(a)。通过增加路径分支，避免环的出现，得到结构标记树 T_{SS}。



(a)文档树 T　　(b)结构标记树 T_{SS}

图 3　增加虚拟节点后的文档树及其结构标记树

3.3.2　SST 算法

SST 是一种简单可靠的表示 XML 文档结构的方法。

SAX 一次线性扫描 XML 文档，对解析的每个 SAX 开始/结束标记事件，调用 SST 构造程序，建立 SST。为每个树节点定义 4 个节点指针：parent，previousSibling，nextSibling，firstChild，指针很大程度上加速了 SST 构造和查询中节点的定位。指针占用的空间并不重要，因为一个 SST 通常很小。

算法　SST_Construction(SAX_Event)
输入　XML 文档 T
输出　结构标记树 T_{SS}

```
SST_Construction(SAX_Event){
if(SAX_Event 是开始标签事件){
创建 T 的一个新节点 u;
if(元素 u 第一次在 T 中出现){
将 u 添加到 SST 中，初始化 u 的标记 S(u);
将 u 插入到 u.parent->graph 中，并创建新边;}
else{
if(u.parent->graph 的最后一个节点是 u)
S(u)最后一位计数上加 1;
else{//将 u 插入 Graph 中出现环路
if(u.parent 路径下不存在虚拟节点)
创建虚拟节点，并初始化计数为 2;
else u.parent 路径下虚拟节点的计数增加 1;
u 增加一位计数并初始化为 1，即 S(u)=+[,1];
u 之前的兄弟节点增加一位计数并初始化为 0;
}}}
else//SAX_Event 是结束标签事件
if(u 是 u.parent 的最后一个孩子)
结束 S(u)当前位的计数;}
```

完成后检查标记，确保相应位置上的数目是正确的，如不正确，常规节点添加前缀“0”，虚拟节点添加前缀“1”。

3.4　数据压缩

SAX 对 XML 文档进行解析，边建立结构标记树边将数据项分离出去。分离出去的数据并不是随便堆置，再利用数据压缩技术单独压缩每个数据项，这样常常导致压缩率的降低，同时，也给将来的查询工作增加了困难。为了避免这种情况的发生，将相同路径下或者 SST 中一个节点集合的数据放到同一容器中，这样，一方面保证了这些数据在一定程度上具有相似性，有助于压缩性能的提高；另一方面也避免了查询时的完全解压缩，只需根据需求进行部分解压即可，有助于查询处理效率的提高。

容器中存放的数据类型不尽相同，可选择的压缩方法也有很多，为了支持查询且不影响查询结果，采用的压缩方法都尽可能是无损压缩。比如，可以使用 gzip、Winzip 和 7-zip 等，还可以使用其他常见的数据压缩方法，本文采用最通用的 gzip 压缩 XML 相关数据。

3.5　查询处理

SSTQC 的查询处理步骤主要包括查询的解析、选取节点、获取数据和输出查询结果。

查询表达式首先经过查询解析器的解析，得到相关路径上的节点信息；解压还原 SST，判定节点关系，选取正确的节点；通过 SST 和压缩数据之间相关联的 hash 索引以及查询节点的标记，找出所求数据在容器中的位置；最后通过查询处理器判定分析，将压缩数据解压整理成合适的输出结果。

查询解析器将查询表达式分为路径查询和值查询 2 种，

不管是路径查询还是值查询，都需要解压还原 SST。为了避免每次查询时都反复解压同一数据的情况，建立一个缓存区，应用最近最久未使用 (Least Recently Used, LRU)算法，存放前面查询时已经解压好的数据，从而提高查询的效率。

4 实验测试和分析

由于部分压缩技术的源代码没有公开，选取 XGrind 和 XPress 2 种典型方法与 SSTQC 进行对比实验，XQzip 和 XQueC 则引用文献[3-4]中的描述。实验环境：CPU1.66 GHz Inter Core Duo T2300，内存 1.5 GB，操作系统是 Windows XP Professional。测试集选择具有代表性的 XML 数据集[5]，相关参数如表 1 所示。

表 1 测试数据集

数据集	大小/MB	元素个数	属性个数	最大深度
DBLP	127.0	3 332 130	404 276	6
SwissProt	109.0	2 977 031	2 189 859	5
Lineitem	30.8	1 022 976	1	3
Shakespeare	7.5	179 690	0	7

4.1 压缩率

压缩率 $p=\left(1-\frac{sizeof(compressed\_file)}{sizeof(original\_file)}\right)\times 100\%$ 表示压缩后的文档相对于原始文档减小的比例，数值越大，压缩性能越好[6]。为支持压缩文档直接查询，XGrind 和 XPress 保留了大量的结构信息及结构与数据项之间的对应关系，与 XML 文档的大小呈正比。XQzip 利用结构索引树技术消除等价子树，但仍存在重复路径，其压缩率约比 XGrind 好 16.7%。XQueC 保存的结构信息连同指向单独压缩的数据项的指针产生很大的空间开销，压缩率略逊于 XPress。而 SSTQC 采用 SST 消除重复路径，保留结构更小，由图 4 的对比结果可知，压缩率稍优于 XQzip。

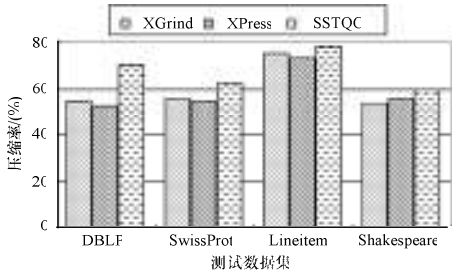


图 4 3 种方法的压缩率对比

4.2 时间性能

考虑到数据大小悬殊很大，下面的时间轴采用对数刻度，图 5 给出压缩时间对比。

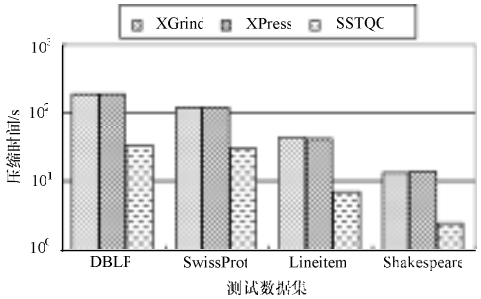


图 5 压缩时间对比

XGrind 和 XPress 需扫描 2 次 XML 文档，过程很慢，XQzip 压缩和解压缩速度是 XGrind 的 5.33 倍和 3.4 倍，XQueC 需要对 XML 数据进行预处理，时间性能比 XQzip 差一些。相比之下，SST 构造算法简单，保留结构小，压缩速度是 XGrind 的 5.37 倍。由于 SSTQC 解压时需还原 SST，因此解压速度是 XGrind 的 3.01 倍，稍逊于 XQzip，解压缩时间对比见图 6。

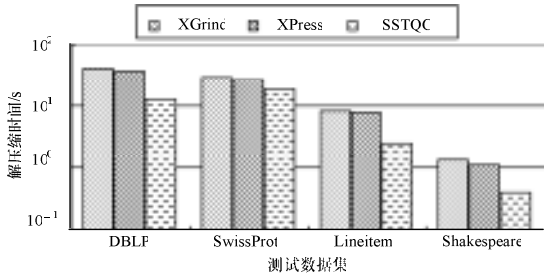


图 6 解压缩时间对比

4.3 查询性能

在 4 个数据集上分别指定 2 种查询，将 XGrind 和 SSTQC 进行对比。XPress 采用倒装算术编码，其查询性能是 XGrind 的 2.83 倍，XQzip 利用结构索引树，其查询性能是 XGrind 的 12.84 倍。SSTQC 采用 SST 能更有效地组织查询，但是当解压缓冲池为空时，需要解压还原 SST，如表 2 所示，SSTQC 的查询情况优于 XGrind 和 XPress，但比 XQzip 差；当解压缓冲池非空时，查询效果还是可取的。SSTQC 的这种查询更适合数据流的压缩，进行网络文件传输。

表 2 XGrind 和 SSTQC 查询对比情况

数据集	查询语句	查询时间/s	
		XGrind	SSTQC
DBLP	/dblp/mastersthesis/@key	49.70	10.54
	/dblp/article/cdrom	58.87	10.83
SwissProt	/root/Entry/@id	30.29	8.06
	/root/Entry/Ref/Comment	38.42	9.17
Lineitem	/table/T/L_TAX	6.94	2.23
	/table/T/L_COMMENT="slowly"	5.88	1.96
Shakespeare	/PLAYS/PLAY/TITLE	2.03	0.56
	/PLAYS/PLAY/ACT/SCENE/SPEECH/SPEAKER	2.34	0.98

5 结束语

本文通过研究现有的几种支持查询的 XML 压缩方法，提出了一种新的支持查询的压缩方法 SSTQC，对原文件进行一次扫描，充分利用 SST 增加对查询的支持，方法简单，压缩性能和查询性能也很可观。下一步工作是在底层压缩方案和查询处理方面继续改进，使其性能更加出色。

参考文献

[1] Tolani P M, Haritsa J R. XGRIND: A Query-friendly XML Compressor[C]//Proceedings of the 18th International Conference on Data Engineering. San Jose, California, USA: [s. n.], 2002.

[2] Min Jun-Ki, Park Myung-Jae, Chung Chin-Wan. XPRESS: A Queriable Compression for XML Data[C]//Proceedings of the ACM SIGMOD International Conference on Management of Data. San Diego, California, USA: [s. n.], 2003.

[3] James C, Wilfred N. XQzip: Querying Compressed XML Using Structural Indexing[C]//Proceedings of EDBT'04. Heraklion, Crete, Greece: [s. n.], 2004.

[4] Arion A, Bonifati A, Costa G, et al. XQueC: Pushing Queries to Compressed XML Data[C]//Proceedings of the 29th International Conference on Very Large Data Bases. Berlin, Germany: [s. n.], 2003.

[5] University of Washington. XML Data Repository[EB/OL]. (2010-02-12). http://www.cs.washington.edu/research/xmldatasets/.

[6] 张 胜, 舒 坚, 包晓玲, 等. XML 压缩方法的比较分析[J]. 计算机工程, 2009, 35(11): 36-28, 31.

编辑 张正兴