文章编号:1001 - 9081(2008)10 - 2537 - 04

XML 文档压缩技术比较研究

张 胜[1,2],舒 坚[1],包晓玲[2]
(1. 南昌航空大学 计算机学院,南昌 330063; 2. 南昌航空大学 无损检测技术教育部重点实验室,南昌 330063)
(zwxzs168@126.com)

摘 要:XML 已经成为互联网上信息交换和信息表示的事实标准。然而 XML 文档中包含大量重复出现的标签和结构等冗余信息,导致 XML 文档在查询处理和数据交换时付出更高的代价,特别在带宽和资源受限的设备上显得更为突出。压缩技术是解决这一问题的重要途径。搜集了近几年提出的各种 XML 压缩方法,从压缩率、压缩与解压时间、内存消耗、查询性能等方面比较分析了六个具有代表性的 XML 压缩技术,最后简要归纳了各自的优点和存在的不足,并探讨未来努力的方向。

关键词:XML 压缩;查询处理;Web 应用
中图分类号: TP391;P208 **文献标志码**:A

Comparison of XML compression techniques

ZHANG Sheng[1,2], SHU Jian[1], BAO Xiao-ling[2]
(1. *School of Computing, Nanchang Hangkong University, Nanchang Jiangxi* 330063, *China*;
2. *Key Laboratory of Nondestructive Test, Ministry of Education, Nanchang Hangkong University, Nanchang Jiangxi* 330063, *China*)

Abstract: XML is a de facto standard for exchanging and presenting information on the Web. However, XML data is also recognized as verbosity since it heavily inflates the data size due to the repeated tags and structures. The data verbosity problem gives rise to many challenges of conventional query processing and data exchange. The hindrance is more apparent in bandwidth- and memory-limited devices. Compression techniques are the important way to overcome the verbosity problem. Multifarious XML-conscious compression methods were collected, and six XML-conscious compression technologies were compared and analyzed in terms of compression ratio, compression and decompression times, memory consumption, and query performance. Their advantages and shortcomings were discussed, and then further work of XML-conscious compression was pointed out.

Key words: XML compression; query processing; Web applications

0 引言

随着互联网技术和移动通信技术的不断发展,越来越多的 XML 数据将在 Web 应用中产生和交换。然而 XML 数据由于具有自我描述的特性,使其存在大量的信息冗余,增加了在 Web 中数据处理、存储、交换的成本,在一定程度上阻碍了 XML 应用的发展。研究、开发 XML 数据压缩技术,是减少 XML 数据冗余的有效途径。

表 1 XML 压缩技术

压缩技术	底层压缩方案	支持查询	解析器	使用平台
XMill	Gzip, Bzip	否	SAX	Unix/Windows(C++)
XMLPPM	PPM	否	SAX	Unix/Windows
Millau DDT	Differential DTD tree, Gzip	否	DOM	Portable(Java)
XMLZip	Gzip	否	DOM	Portable(Java)
XComp	Zlib, Huffman coding	否	SAX	Unix (VC++)
XGrind	Huffman coding	是	SAX	Unix/Windows(C++)
XPRESS	Huffman coding, reverse arithmetic encoding	是	SAX	Portable(Java)
XQzip	SIT(Structure Index tree), Gzip	是	SAX	Windows(C++)
XPACK	Binary encoding, Gzip	是	Xerces	Windows (Java)
XQuec	Huffman coding, ALM algorithm	是	SAX	Linux(Java)
XBzip	PPMdi, XBW, FM-INDEX	是	DOM	Windows(C++)
XCQ	Gzip, bzip2	是	SAX	Windows(Java)

虽然 XML 文档可以采用通用文本压缩技术(如:Gzip,Bzip2,WinZip 等)进行压缩,但这样会丧失 XML 文档固有的优势(如:结构特征、语义特征等)。自从 Hartmut Liefke 和 Dan Suciu 于 2000 年提出第一个 XML 压缩技术 XMill[1] 以来,国外许多学者在

收稿日期:2008 - 04 - 09;**修回日期**:2008 - 08 - 01。 **基金项目**:国家自然科学基金资助项目(60773055);江西省教育厅科技计划资助项目(GJJ08223);南昌航空大学学院基金资助项目(EA200606198)。

作者简介:张胜(1968 -),男,湖北罗田人,副教授,博士,主要研究方向:数据挖掘、GIS、Web 信息处理、无线传感器网络; 舒坚(1964 -),男,江西靖安人,教授,主要研究方向:计算机网络与分布式系统、嵌入式系统、无线传感器网络; 包晓玲(1973 -),女,湖北黄冈人,工程师,主要研究方向:数据压缩、数据库技术、信息处理。

XML数据压缩方面做了大量工作,提出了一些针对XML文档的压缩技术[2-12],简要汇总如表1所示。国内这方面研究较少[13-14]。

根据是否支持压缩后的XML文档查询,可将XML数据压缩技术分为两大类:不支持查询的XML压缩技术和支持查询的XML压缩技术。不支持查询的XML压缩技术主要有:XMill、Millau、XMLPPM、XMLZip、XComp等。支持查询的XML压缩技术主要有XGrind、XPress、XQzip、XPACK、XQueC、XBZip、XCQ等。

本文选取比较有代表性的几个XML压缩技术,比较相应的压缩比、压缩与解压时间、内存消耗、查询处理等性能。并在此基础上归纳、总结现有XML压缩技术的不足及未来的发展趋势,旨在为开发和应用XML压缩技术的同行提供相关的技术性能和研究思路。

1 设定测试环境

工作环境:操作系统为Windows 2000,CPU:Pentium(R)4,2.93 GHz,内存512 MB。

比较对象:XMill,XMLPPM,XMLZip,XPress和XGrind。这些XML压缩技术是目前比较有代表性的,且找到了相应的源代码。为了便于说明问题,选择Gzip作为参考标准。

测试集:选用具有代表性的XML数据集(如表2所示)来进行实验。它们包含了不同类型的XML数据格式与结构。其中,XMark[15]是用xmlgen程序生成的标准XML文档,模拟"拍卖行为"的数据库,具有深层嵌套的大量元素和属性,许多元素值是长文本格式的段落。Weblog[16]是真实的Web服务日志,转换成XML格式,具有相对规则的结构。SwissProt[17]主要描述了DNA序列,具有最小冗余度的数据。DBLP[18]是一个参考书目录数据库,具有相对规则的结构。Shakespeare[19]是一个做过标注的莎士比亚戏剧语料库,包含有大量长文本格式的段落。Baseball[20]是参加1998年"Major League"的每一个棒球队的所有队员信息及相关比赛记录的统计表,包含有大量的数值类型的数据。

表2 XML数据源

测试数据集	文档大小/MB	嵌套深度	元素数目	属性数目
XMark	101.0	4	2873293	621490
Weblog	32.0	3	641037	0
SwissProt	21.0	4	618412	336073
DBLP	40.0	3	1107711	118028
Shakespeare	15.3	5	358044	0
Baseball	16.8	6	243766	1

2 压缩性能比较

测试的压缩性能为:压缩率、压缩与解压时间、内存消耗量。测试的结果及相关分析如下所述。

2.1 压缩率

表示压缩率通常有两种形式:CR_1和CR_2,定义如下:

$$CR_1 = \frac{\text{sizeof(compressed file)} \times 8}{\text{sizeof(original file)}} \quad (1)$$

$$CR_2 = \left(1 - \frac{\text{sizeof(compressed file)}}{\text{sizeof(original file)}}\right) \times 100\% \quad (2)$$

CR_1是指压缩后文档实际所占空间大小的比例,单位为b/B,数值越小表示压缩性能越好;CR_2是指压缩后文档相较于原文档减少量的百分比,数值越大表示压缩性能越好。

测试结果和结论:图1显示各种XML压缩技术对上述6种数据源进行压缩后的结果,图中的数据采用CR_1表示。从中可以看出:

1)总的来看,不支持查询压缩技术(XMill、XMLPPM和XMLZip)的压缩率比支持查询压缩技术(XGrind和Xpress)的压缩率小,有较好的压缩性能。

2)XMill、XMLPPM和XMLZip的压缩率比Gzip的压缩率小。其中,XMLZip与Gzip的压缩率非常接近,而XMLPPM的压缩率是最为出色的。

3)XGrind和Xpress的压缩率比Gzip的压缩率大。除Baseball数据集外,两者的压缩率接近。

对于结论1),主要是因为不支持查询压缩技术充分利用了XML文档中包含有大量冗余的结构信息,在压缩预处理过程中对XML文档进行重新组织,减少了冗余,故有较好的压缩性能。与之相对应,支持查询压缩技术虽然同样考虑了XML文档中大量冗余的信息,但为了能对压缩文档进行直接查询,XGrind和Xpress在压缩过程中保存了输入XML文档的结构以及结构与数据项之间的对应关系,导致支持查询压缩技术的压缩率较不支持查询压缩技术的压缩率差。

对于结论2),在不支持查询压缩技术中,XMLPPM的压缩率之所以最小是因为它采用了MHM(Multiplexed Hierarchical Modeling)技术和PPM(Prediction by Partial Match)技术进行编码。而XMill和XMLZip都是采用Gzip作为底层压缩技术。文献[3]通过实验证明PPM比Gzip具有好的压缩率,且采用MHM模型进一步提高压缩率。

对于结论3),XGrind和Xpress均采用哈夫曼编码作为底层压缩技术,且还要额外保存XML文档的结构信息,故压缩率较差。至于在Baseball数据集外中Xpress比XGrind有较大的优势,可能是因为Baseball中主要是数值类型的数据,而逆算术编码更适合数值类型数据的压缩。

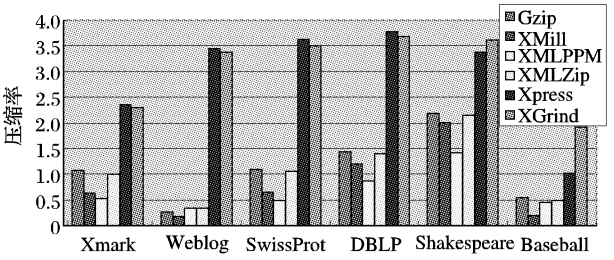

图1 压缩率比较

2.2 压缩与解压时间

图2和3分别显示不同压缩技术对6个数据集的平均压缩与解压时间。实验表明:

1)所有的XML压缩技术与Gzip相比都需要更长的压缩时间,其中XMill所用压缩间最短。解压时间除在Xmark数据集上Xmill和XMLZip较Gzip短以外,其他数据集上都比Gzip解压时间长。

2)XMLPPM、XPress、XGrind与其他三个XML压缩技术相比需要更长的压缩时间。在解压时间上,除Weblog数据集外,XMLPPM需要的时间比其他5个XML压缩技术都长。

针对结论1),XMill是5个XML压缩技术中压缩时间最短的,但仍较Gzip稍长,主要是因为XMill在使用Gzip压缩输入XML文档之前要将结构信息从XML文档中分离出来,并根据语义的不同将数据项重组到不同的容器中,然后使用Gzip对各个

容器分别进行压缩。因此,比直接使用 Gzip 压缩花更多的时间。同样,在解压的时候要执行相反分过程,即:先对数据容器进行解压,然后将数据项合并到原有的位置,并对原有的文档结构进行重构。故一般情况下 XMill 解压比 Gzip 花的时间稍长。然而图 3 表明 XMill 在对 Xmark 数据集解压缩时间比 Gzip 短,出现这种情况可能是因为 XMill 压缩后形成的文档比 Gzip 压缩后形成的文档小很多,使得 XMill 解压时花更少的时间读取压缩后的磁盘文件。XMLZip 压缩时间比 Gzip 长,主要是因为采用了 DOM 技术来解析 XML 文档,而在压缩过程中需要将 XML 文档转换成 DOM 树;在解压缩时,又需要将 DOM 树转换成 XML 文档,这一操作需要较长时间。

针对结论 2),XMLPPM、XPress、XGrind 的压缩与解压时间相对更长。主要是因为 XMLPPM 建立在 PPM 编码基础之上,而 PPM 尽管可以提供更好的压缩率,但它是一种相对较慢的编码方式。通过 PPM 技术解压也非常耗时,故比其他 XML 压缩技术解压时间更长。XGrind 采用了 Huffman 编码,为获得好的压缩率,需要对 XML 文档进行二次扫描来收集文档相关统计信息,故压缩时间差不多是 Gzip 的两倍。Xpress 采用了逆算术编码方案对 XML 文档中元素的树路径进行编码,压缩与解压时间比 XGrind 略好。

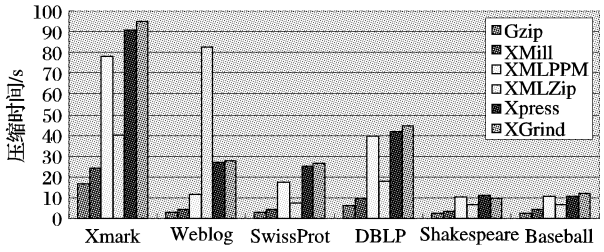

图 2　压缩时间比较

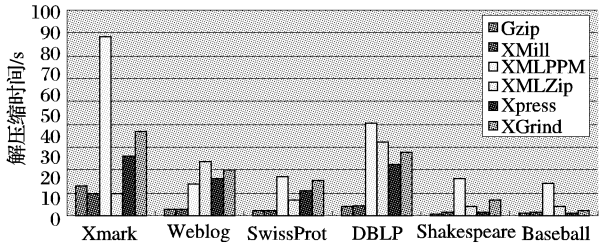

图 3　解压缩时间比较

2.3　内存消耗

图 4 显示不同压缩技术对上述 6 种 XML 数据集的内存消耗情况。可以看出,Gzip 占有的内存最少,固定为3MB,且与输入 XML 文档的大小无关;XMill,Xpress,XGrind 大约占 8 MB 内存;XMLPPM 所占内存比 XMill 多 3 MB,约 11 MB;XMLzip 占有内存量最大,且与输入 XML 文档的大小有关。

XMill 在压缩过程中,使用了一个固定的内存窗口(默认值为 8 MB),当内存消耗达到固定值时,XMill 将压缩文档写入硬盘,然后继续进行压缩操作。XGrind 使用哈夫曼编码来进行压缩,因此只需要较少的内存对哈夫曼模型进行存储,因此总的内存占有量不是很多。XPress 的内存消耗量与 XGrind 相似。它们主要区别在于,XPress 采用了逆算术编码,而 XGrind 采用了哈夫曼编码。XMLPPM 为 XML 文档建立了 4 个 PPM 模型,这些模型对内存都有一个固定值的限制,因此整体对内存的消耗量较少。XMLZip 使用 DOM 对 XML 文档进行解析,需要建立 DOM 树,这一过程与 XML 文档大小成正比,更准确地说与 XML 文档中节点的多少成正比,因此需要占有更大的内存。

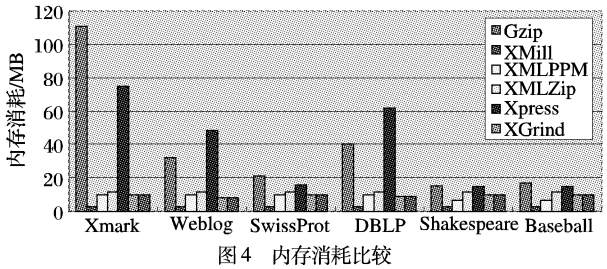

图 4　内存消耗比较

3　查询性能

支持查询压缩技术的查询覆盖如表 3 所示。由于其中大部分压缩技术的源代码没有公开,所以对查询性能的比较只能引用相关参考文献的描述。

XGrind、XPress 都支持 XQuery、Xpath 等查询语言,是因为它们都采用了同型转换策略来保持原有的 XML 文档结构,但是它们却不支持复杂的查询处理。

由文献[6]可知,XGrind 的查询性能比 XMill 好 2 ~ 3 倍,因为 XMill 需要将文档解压后再执行查询处理。通过文献[7]可知,XPress 的查询性能是 XGrind 的 2.83 倍,而在文献[8]通过实验表明,XQzip 的查询性能是 XGrind 的 12.84 倍。XQueC 没有跟上述压缩算法进行查询性能的比较,但是与 Galax 进行了比较[21],结果表明 XQueC 的查询性比 Galax 出色。在文献[11]中,XBzipIndex 与 XGrind、Xpress、XQzip 进行了比较,结果显示 XBzipIndex 压缩率与查询性能都很出色。在文献[12]中指出 XCQ 也具有较好的查询性能。

表 3　XML 压缩技术查询覆盖

压缩技术	查询覆盖
XGrind	Exact-match, Prefix-match, partial-match, Xpath Axes: Child attribute
XPress	Exact-match, Prefix-match, Xpath Axes: Child descendant attribute
XQzip	Exact-match, Prefix-match, multi-pradicate and aggregation, deeply nested predicate, Xpath Axes: Child descendant attribute ancestor parent
XPACK	Exact-match, Prefix-match, partial-match, Xpath Axes: Child attribute
XQueC	Exact-match, Prefix-match, fuzzy-match, Xpath Axes: Child descendant attribute Selections, Joins, aggregations, nested query; supporting proximity and wildcards
XBzip	Patrent, child, block of children, rank and select query
XCQ	Exact-match, Prefix-match, multi-pradicate and aggregation, deeply nested predicate, Xpath Axes: Child descendant attribute ancestor parent

4　结语

从以上的比较分析可以看出 XMill 拥有非常出色的压缩率,较有优势的平均压缩与解压时间、较低的内存消耗,获得最优的平均压缩性能。美中不足的是不支持对压缩文档的直接查询。XMLPPM 拥有非常出色的压缩率,但是它的压缩与解压时间较长,因此阻碍了它的应用。由于 XMLZip 与 Millau 压缩率比 Gzip 稍好,压缩时间较长,且内存占有量与 XML 文档大小成正比,因此它们的应用不如 XMill 和 XMLPP。XGrind 与 Xpress 的压缩率远比不上 Gzip,但是它们支持对压

缩文档的直接查询。XQzip、XQueC 和 XCQ 同时也支持查询操作,且查询性能优于 XGrind 与 Xpress。

尽管 XML 压缩技术取得了很大的进展,仍然存在一定的不足,还需在以下方面得到提高:1)有效压缩。有效的压缩需要有出色的压缩率、较短的压缩与解压时间以及较少的内存消耗。2)有效的查询引擎。可以直接对压缩文档进行查询,并支持复杂查询操作。3)较少的人工干预、提供友好的用户接口。要想 XML 压缩技术作为底层压缩工具来应用,就必须减少人工干预;要想 XML 压缩技术成为基本的、通用的压缩技术,就得像 WinZip 一样具有友好的用户接口。

此外,对 XML 压缩技术的研究涉及到一系列计算机技术,例如:数据库技术、信息检索技术、信息论、算法研究等。在未来 XML 压缩技术的研究中至少面临如下挑战:

1)支持查询压缩技术能够对压缩文档进行直接查询,能否提出一个有效的方法对压缩文档进行直接更新操作?

2)XQueC 和 XQzip 利用辅助结构可对压缩 XML 文档进行比较理想的查询处理。能否设计一个更有效的辅助结构(如:索引模型)来更好地支持压缩 XML 文档的查询操作?

3)在进行压缩文档查询时,能否建立一个智能分析模型用于分析不同用户的查询请求,来优化查询处理,提高查询效率?

4)XML 文档主要在 Web 中产生和应用,因此压缩技术必须考虑在线工作模式将是未来发展的方向。

参考文献:

[1] LIEFKE H, SUCIU D. XMill: An efficient compressor for XML data [C]// Proceedings of the ACM SIGMOD International Conference on Management of Data. New York, NY, USA: ACM, 2000: 153 - 164.

[2] SUNDARESAN N, MOUSSA R. Algorithms and programming models for efficient representation of XML for internet applications [C]// Proceedings of the 10th International WWW Conference. New York, NY, USA: ACM, 2001: 366 - 375.

[3] CHENEY J. Compressing XML with multiplexed hierarchical PPM models [C]// Proceedings of the IEEE Data Compression Conference. Washington, DC: IEEE Computer Society, 2000: 163 - 172.

[4] XMLZip - XML Solutions [EB/OL]. [2008 - 03 - 12]. http://www.xmls.com/.

[5] LI WEIMIN. XCOMP: An XML Compression Tool [D]. Waterloo, Canada: University of Waterloo, 2003.

[6] TOLANI P M, HARITSA J R. XGRIND: A query - friendly XML compressor [EB/OL]. [2008 - 03 - 12]. http://dsl.serc.iisc.ernet.in/publications/conference/xgrind.pdf.

[7] MIN J K, PARK M J, CHUNG C W. XPRESS: A queriable compression for XML data [EB/OL]. [2008 - 03 - 12]. http://islab.kaist.ac.kr/chungcw/InterConfPapers/sigmod2003_jkmin.pdf.

[8] CHENG J, NG W. XQzip: Querying compressed XML using structural indexing [EB/OL]. [2008 - 03 - 12]. XQzip: Querying compressed XML using structural indexing.

[9] DANIEL R, JAMES C, LING L. XPACK: A High-performance Web Document Encoding[EB/OL]. [2008 - 03 - 12]. http://faculty.cs.tamu.edu/caverlee/pubs/rocco05xpack.pdf.

[10] ARION A, BONIFATI A, COSTA J, *et al*. XQueC: Pushing queries to compressed XML data[EB/OL]. [2008 - 03 - 12]. http://www.vldb.org/conf/2003/papers/S35P04.pdf.

[11] FERRAGINA P, LUCCIO F, MANZINI G, *et al*. Compressing and Searching XML Data Via Two Zips [C]// WWW 2006. New York, NY, USA: ACM, 2006: 751 - 760.

[12] LAM W Y, NG W, WOOD P T, *et al*. XCQ: XML Compression and querying system [EB/OL]. [2008 - 03 - 12]. http://www.cs.ust.hk/~wilfred/paper/www03a.pdf.

[13] 王腾蛟,高军,杨冬青,等. 面向 XPath 执行的 XML 数据流压缩方法[J]. 软件学报, 2005, 16(5): 869 - 877.

[14] 钟世明,邵锐,张胜,等. 基于位置服务系统中 XML 数据流压缩方法[J]. 武汉理工大学学报:交通科学与工程版, 2006, 30(1): 29 - 32.

[15] SCHMIDT A R, WASS F, KERSTEN M L, *et al*. XMark: A benchmark for XML data management[EB/OL]. [2008 - 03 - 12]. http://www.vldb.org/conf/2002/S30P01.pdf.

[16] Log Files - Apache HTTP Server[EB/OL]. [2008 - 03 - 12]. http://httpd.apache.org/docs/logs.html.

[17] SWISS - PROT Protein Knowledgebase[EB/OL]. [2008 - 03 - 12]. http://www.expasy.ch/sprot/.

[18] DBLP[EB/OL]. [2008 - 03 - 12]. http://dblp.uni - trier.de/.

[19] BOSAK J. Shakespeare 2.00[EB/OL]. [2008 - 03 - 12]. http://www.cs.wisc.edu/niagara/data/shakes/shakspre.htm.

[20] HAROLD J R. Long Baseball Examples from the XML Bible[EB/OL]. [2008 - 03 - 12]. http://www.ibiblio.org/xml/examples/baseball/.

[21] MARIAN A, SIMEON J. Projecting XML documents[EB/OL]. [2008 - 03 - 12]. http://www.cs.rutgers.edu/~amelie/papers/2003/xmlprojection.pdf.

(上接第 2532 页)

[8] ROTH S, BLACK M J. Fields of experts: A framework for learning image priors [C]// IEEE Conference on Computer Vision and Pattern Recognition. Washington, DC: IEEE Computer Society, 2005, 2: 860 - 867.

[9] TAPPEN M F. Utilizing variational optimization to learn Markov random fields [C]// IEEE Conference on Computer Vision and Pattern Recognition. Washington, DC: IEEE Computer Society, 2007: 1 - 8.

[10] ZHU S C, MUMFORD D B. Prior learning and Gibbs reaction-diffusion[J]. IEEE Transaction on Pattern Analysis and Machine Intelligence, 1997, 19(11): 1236 - 1250.

[11] MOULIN P, LIU J. Analysis of multiresolution image denoising schemes using generalized-Gaussian and complexity priors[J]. IEEE Transaction on Information Theory, 1999, 45(3): 909 - 919.

[12] GEMAN D, REYNOLDS G. Constrained restoration and the recovery of discontinuities [J]. IEEE Transactions of Pattern Analysis and Machine Intelligence. 1992, 14(3): 367 - 383.

[13] HYVARIMEN A. Estimation of non-normalized statistical models by score matching [J]. Journal of Machine Learning Research. 2005, 6: 695 - 709.

[14] MACKAY D J C. 信息论、推理与学习算法[M]. 肖明波,席斌,许芳,等译. 北京: 高等教育出版社, 2006: 416 - 417.

[15] ROMBERG J K, CHOI H, BARANIUK R G. Bayesian tree structured image modeling using wavelet domain hidden Markov models[J]. IEEE Transactions on Image Processing, 2001, 10(7): 1056 - 1068.

[16] 刘芳,刘文学,焦李成. 基于复小波邻域隐马尔可夫模型的图像去噪[J]. 电子学报, 2005, 33(7): 1284 - 1287.

[17] SENDUR L, SELESNICK I W. Bivariate shrinkage functions for wavelet-based denoising exploiting interscale dependency[J]. IEEE Transactions on Signal Processing, 2002, 50(11): 2744 - 2756.

[18] PORTILLA J, STRELA V, SIMONCELLI W E. Image denoising using scale mixtures of Gaussians in the wavelet domain [J]. IEEE Transaction on Image Processing, 2003, 12(11): 1338 - 1351.