基于BWT的文本压缩算法研究

李彦军,苏红旗,杨 峰,李述迪,姚书科
(中国矿业大学 机电学院,北京 100083)

摘 要:为了理解高效的文本压缩算法,探究文本压缩的原理和意义,对基于BWT的字符串轮转理论进行了深入的理解;游程编码作为一种简单有效的压缩方式,在数据压缩中有十分广泛的应用,文本压缩的研究对于多媒体的压缩研究有着十分重要的意义。把BWT结合游程编码对选定的文本信息进行了压缩比较,实验证明了该算法的高效性和实用性。同时对基于BWT压缩算法的发展趋势进行了展望及分析。

关键词:BWT;压缩算法;文本压缩;游程编码
中图分类号:TP301.6 **文献标识码**:A **文章编号**:1673-629X(2009)05-0089-04

Research of BWT Compression Algorithm Based on Text

LI Yan-jun, SU Hong-qi, YANG Feng, LI Shu-di, YAO Shu-ke
(China University of Mining and Technology, Beijing 100083, China)

Abstract: In order to understand the text efficient compression algorithm, inquiring text and meaning of the principle of compression, the paper is based on the string transformation has conducted in-depth understanding, the run-length coding as a simple and effective compression in data compression has a very wide range of application, compressed version of the multimedia compression research is very important. Combining the BWT and the run-length coding the text of selected information is compressed, the experiment proved that the algorithm is efficient and practical. At the same time, BWT compression algorithm based on the development trends and analysis of the prospect.

Key words: BWT; compression algorithm; text compression; run-length coding

0 引 言

数据压缩分为有损压缩和无损压缩两大类,而无损压缩的迅速发展催生了诸如:Huffman编码,算术编码,L-Z系列编码,动态马尔科夫压缩编码(DMC),PPM编码和Burrows-Wheeler Transform(BWT)编码算法的产生。1994年Michael Burrows和David Wheeler在《ABlock-sorting Lossless Data Compression Algorithm》一文中共同提出的通用数据压缩法:Burrows-Wheeler Transformation,它基于数据块的字母矩阵进行轮换,从而调整了压缩元素之间的相对顺序来提高压缩的比率和压缩速度的。同目前现有的文本压缩算法相比,它具有仅次于PPM的压缩比率,但是在速度上有绝对的优势[1]。尽管基于LZ的方法的速度很快,但在文件较小的情况下性能会很差,相比较用BWT具有比较理想的效果。

收稿日期:2008-08-18
基金项目:国家科技支撑计划(2006BAK02B01)
作者简介:李彦军(1981-),男,河北石家庄人,硕士研究生,研究方向为图像处理与仿真;苏红旗,副教授,博士,研究方向为数据采集与压缩、图像处理;杨 峰,副教授,博士,研究方向为图像处理。

1 文本压缩

1.1 数据压缩

数据压缩的目的就是用较少的数码来表示原始的信号数据。在实际的应用中数据压缩的方法按照不同的压缩原理有不同的方法,根据解码后数据与原始数据是否完全一致进行分类,数据压缩可以分为无损压缩和有损压缩两大类[2]。再按照压缩原理的不同,常见的压缩方法又可以分类如图1所示。

1.2 文本压缩

数据压缩分为通用数据压缩和多媒体压缩两类,在图像出来之前,文本压缩是数据压缩的一个重要方向,由于文本的自身特点,文本压缩包括:编码、压缩、解压、反编码、恢复原始文档等部分[3],如图2所示。

2 BWT变换

2.1 BWT的核心思想

Burrows和Wheeler设计的BWT算法与以往所

有通用压缩算法的设计思路都迥然不同,现有比较著名的压缩算法都是处理数据流模型的,一次读取一个或多个字节,然而对于成块的数据处理比较棘手。BWT 使得处理成块的数据成为可能,这种算法的核心思想是对字符串轮转后得到的字符矩阵进行排序和变换,它的基本原理是将原始字符串每次左移(或者右移)一位,将新的字符串以行为单位按照字母表顺序重新排列成矩阵,最后一列呈现出字母内聚现象,然而 BWT 只是将字符顺序进行转换,并没有进行压缩,这种极有规律性的字母内聚的特点能够使其他文本压缩算法大大简化,同时也能得到较好的压缩比[4]。例如考虑一般的文本应用如英语文本中 the 这个字符串经常会被用到,经过 BW 转换之后,所有的 t 都被移动到最后并且聚合在一起,这样变换之后的字符串用通用的统计压缩模型(Huffman coding、LZ 算法、PPM 算法、游程编码等)等进行压缩就能得到更好的压缩比[5]。

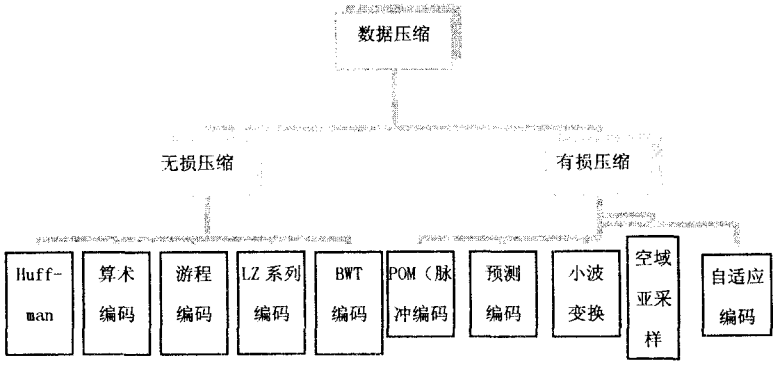


图 1 常见的数据压缩方法

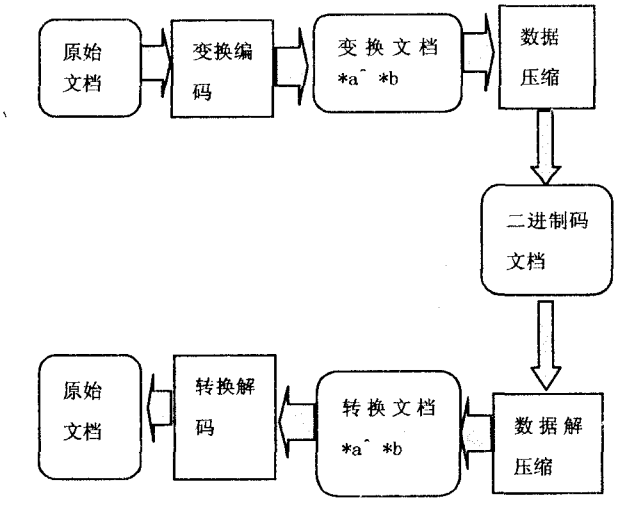


图 2 数据压缩的流程

2.2 BWT 变换的实现

具体的算法如下:

(1)输入一字符串,长度记为 n。

(2) 以该字符串为基础,进行依次的左移操作。这些左移的新字符串和原来的字符串构成一个轮换矩阵 $M(x)$。

(3) 按照字母顺序对 $M(x)$ 的行进行行排列,得到新的矩阵 $M'(x)$

(4) 变换后的第一行记做 F,最后一行记做 L,且找到原始的字符串在的行,记做 R。

(5) 字符串的 BWT 为 F,L 和 R。

以例子进行以下操作:

D	R	D	O	B	B	S

第一步 原始字符串

D	R	D	O	B	B	S
R	D	O	B	B	S	D
D	O	B	B	S	D	R
O	B	B	S	D	R	D
B	B	S	D	R	D	O
B	S	D	R	D	O	B
S	D	R	D	O	B	B

第二步 进行循环移动得到的轮换矩阵

B	B	S	D	R	D	O
B	S	D	R	D	O	B
D	O	B	B	S	D	R
D	R	D	O	B	B	S
O	B	B	S	D	R	D
R	D	O	B	B	S	D
S	D	R	D	O	B	B

第三步 新矩阵及 F 和 L

其中 F = (BBDDORS) 红色 L = (OBRSDDB) 紫色 R = 4

2.3 BWT 逆变换的实现

上述的 BWT 中不难发现每一列所包含的字符就是原设计字串中所含的字符,只不过是顺序不同罢了。也就是说只要记录任何一列,都能够将其严格按照字母表的顺序排列出 F 来。然而这样还是没有办法还原出原字符串,如果能找到一列与 F 有什么内在关联,说不定能够从中得到什么启示。我们知道,每一行都是左移而得到的,也就是说每行的最后一个字符都是由第一个字符左移得到的,那么在移动之前,最后一个字符就应该排在第一个字符之前。换句话说 L 列中的每个字符都是对应 F 列中字符的前序[6]。这个关系就成了能只存最后一列的信息而还原出原字符串信息的重要理论。也就是说在前面的 BWT 转换的最后其实已经记录了一个对应的转换关系[7]。

图 3 就是一个对应关系:$T[]$ = (1,6,4,5,0,2,3)。

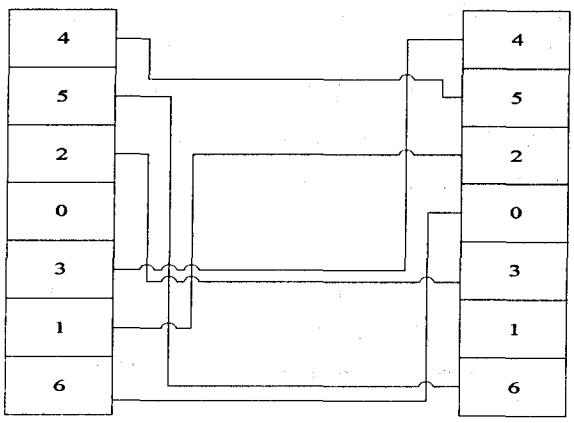
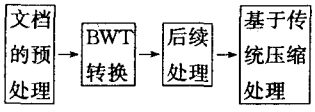


图 3　BWT 的转换关系

3　基于 BWT 的压缩算法

基于 BWT 的压缩算法的基本流程如下:



3.1　游程编码

游程编码又称"运行长度编码"或"行程编码",是一种统计编码,该编码属于无损压缩编码。对于二值图有效。它的基本原理是:用一个符号值或串长代替具有相同值的连续符号(连续符号构成了一段连续的"行程"。行程编码因此而得名),使符号长度少于原始数据的长度。例如:555555777773332222111111,行程编码为:(5,6)(7,5)(3,3)(2,4)(1,7)。可见,行程编码的位数远远少于原始字符串的位数。在对图像数据进行编码时,沿一定方向排列的具有相同灰度值的像素可看成是连续符号,用字串代替这些连续符号,可大幅度减少数据量。

3.2　基于 BWT 的游程文本压缩编码实现

BWT 只是转换轮换字符串的顺序,其实并没有进行压缩,但是变换后的字母之间的内聚性给利用通用的压缩算法带来了好的前提条件。这里将 BWT 的轮换与游程编码压缩算法相结合来达到真正的压缩效果。

压缩的具体步骤如下:

1) 输入字符串 S,长度为 N。

2) 将 S 依次左移一位,得到 n 个字符串,组成 $N\times N$ 的矩阵。

3) 将 2) 中的矩阵以行为单位按照字母表的顺序重新进行排列得到矩阵 M。

4) 取 M 的第一列 F 与最后一列 L,以及 S 在 M 中的行序号 I。

5) 将 L 的第一个字符放入比较字符 CH 中,依次比较 F 中的字符。若与 CH 相同,计数器 p 加 1;否则将 CH 与 p 写入一维数组 a 中,将当前新字符写入 CH 中,计数器 p 归零;如此循环直至所有字符比较完毕。

解压缩的过程如下所示:

6) 读数组 $a[]$ 中第一个单元内容写入 CH,第二个单元内容赋给计数器 p,循环次数为 p,将字符 CH 写入字符串数组 L 中 p 次。循环完毕后继续读数组 a 中后两个单元内容,如此反复,直到 a 中所有内容被写到 L 中。

7) 将 L 中的内容严格按照字母表顺序排列后写到字符串数组 F 中。

8) 取 F 中第 i 个字符入比较字符 CH 中,依次与 F 中 1 到 i 个字符比较,记录该字符是第 q 次出现,然后顺次将 CH 与 L 中各字符比较,若相同计数器 p 加 1,当 $p=q$ 时停止,将该字符当前在 L 中的序号写到向量 T 的第 i 个位置。$i+1$ 继续循环直到 F 中所有字符都与 L 中字符有关系 T。

9) 依次将 $F[I],L[T[I]],\cdots,L[T[T[T\cdots[T[I]]]\cdots]]$(有 $n-1$ 个 T) 写入字符串 $S1$ 中。

10) 输出是 $S1$ 即为源字符串 S。

4　实验结果及分析

以 MicrosoftVisualC++6.0 语言为基础,在笔者的电脑上进行了数据压缩。实验数据:选取了长度均为 1000Bytes 的不同文件(英文)进行压缩实验,试验结果见表 1。

表 1　实验结果:压缩比

出现重复字符次数	重复字符串平均长度	压缩比(%)
10	4	98.01
10	6	96.12
10	10	92.06
20	4	96.33
20	6	92.15
20	10	84.37
30	4	94.25
30	6	88.16
30	10	76.32
50	4	90.11
50	6	76.21
50	10	56.83

由表 1 容易得出在文本中字符重复出现次数相同的情况下,重复字符串的平均长度越长,压缩比就越低;在重复字符串的平均长度相同的情况下,字符重复出现的次数越多,压缩比也越低。也就是说当随机字符串经过 BWT 转换后,L 列字母积聚性越好重复次数越多,压缩效果就越好。反之,当转换后字符分散,

重复次数较少时,压缩效果也不好。

此外对大小为 1M 的中文数据也进行了试验,发现效果没有英文的好,这是因为英文的重复和规律行比中文要好,而中文的信源远多于英文。

同时在实验的后期选用不同的实验数据与 PKZIP 进行了比较。实验结果见表 2。

表 2 不同材料的实验结果

文件名称	原始文件大小	PKZIP 文件大小	PKZIP Bits/Byte	Bwt 文件大小	BwtBits/Byte
书本	610, 856	209, 061	2.74	186, 592	2.44
文章	38, 105	13, 331	2.80	13, 159	2.76
新闻	377, 109	146, 010	3.10	134, 174	2.85

总之 BWT 转换后字符内聚性在很大程度上影响最终压缩的比率。只有转换后表现出良好内聚性的字符才能得到较好的压缩比率。

5 结束语

BWT 变换及逆变换的作用虽然不是直接的压缩,却给其他文本压缩方法提供了很好的基础。BWT 独特的可逆轮换方式使得原本没有规律的随机字符串呈现出局部的字符内聚性质,即使部分字符在转换中以扎堆的形式连续出现,这一重要特性方便了其它编码,节省了其他编码对原始字符串的处理过程,同时又能达到更高效的压缩比率改变了文本的固有属性,在一段程度上拓展了一些原有的压缩算法的应用范围。同时,BWT 算法本身由于包含了大量的排序操作,从而造成了较高的时间复杂度,相信现在和不久的将来会有不少学者在对此进行研究。同时基于 BWT 的多媒体数据压缩是需要进一步研究的方向。

参考文献:

[1] 黄 锐,唐继勇.文本类型气象雷达原始回波数据的无损压缩[J].通信与信息技术,2008(1):33-34.

[2] 祝 庚,钟嘉鸣.数据压缩与解码技术探讨[J].湘南学院学报,2002(5):20-22.

[3] Bunton S. On-Line Stochastic Processes in Data Compression [D]. USA: Dept. of Computer Science and Engineering, University of Washington, 1996.

[4] Burrows M, Wheeler D J. A Block-sorting Lossless Data Compression Algorithm[R]. SRC Research Report 124[s. l.]: Digital Systems Research Center, 1998.

[5] 白跃彬,卢华斌,韩庆绵,等.图像压缩技术及其进展[J].微机发展(现更名:计算机技术与发展),1996,16(4):97-98.

[6] Cleary J G, Teahan W J. Unbounded Length Contexts for PPM[J]. Thev Computer Journal, 1993, 36(5):32-36.

[7] Cormack G V, Horspool R N. Data Compressing Using Dynamic Markov Modeling[J]. Computer Journal, 1987, 30(6): 541-550.

(上接第 88 页)

表 1 DWR 和 JSON-RPC-Java 的特点比较

应用开发的技术支持	DWR	JSON-RPC-Java
后台服务的注册和访问控制方式	使用 dwr.xml 定义需要公开的服务,在前台 javascript 中用<script src="xxx">把服务引进到客户端;服务的周期管理可以在 XML 中配置	默认在 Session 的 Bridge 里注册要使用的 Java 对象,这种方式有很多限制,使用时需要注册 JSP 或 Servlet 程序;服务的周期管理较麻烦
客户端页面形式	HTML 页面,JSP 页面	JSP 页面
与原有 J2EE 框架整合程度	能与 Spring 和 Structs 等框架无缝整合	不能与 Spring 和 Structs 等框架的整合应用
调试控制台	可在控制台调试注册远程 javascript 对象的详细信息	不提供调试控制台
支持反向 AJAX 程度	支持	不支持
其它	提供 DWRUtils 工具类简化 DOM 操作	不提供任何简化 DOW 操作工具

5 结束语

利用 AJAX 框架,简化 AJAX 应用开发,构建高效 Web 应用已是热点研究课题。文中结合工程实际,研究了基于 AJAX 技术构建 Web 应用的两种开发模式,通过具体实例分析了相关框架集成和应用实现的技术要点。所实现的功能已取得了满意的应用效果和用户好评。

参考文献:

[1] 李 刚.基于 J2EE 的 Ajax 宝典[M].北京:电子工业出版社,2007.

[2] Garrett J J. Ajax: A New Approach to Web Applications[EB/OL]. 2005-02-18. http://www.adaptivepath.com/publications/essays/archiv es/000385.php.

[3] DWR: Easy AJAX for JAVA[EB/OL]. 2007-11-20. http://getahead.org/dwr/overview/dwr.

[4] 吴学义.基于 AJAX 的 B/S 架构及应用[J].吉林大学学报,2007,3(13):314-318.

[5] 李 扬,马光思.扩展与整合 Web 应用框架的研究与实践[J].计算机工程,2006,32(10):144-146.

[6] 蒋 纬,马光思.Spring 与其他框架整合及流程分析[J].计算机工程,2007,33(14):79-81.

[7] 纪 颖,马光思.使用 DAO 和业务代理联合模式整合 Web 应用框架[J].计算机技术与发展,2006,16(11):19-21.

[8] Dynamic Html and XML: The XMLHttpRequest Object[EB/OL]. 2005-06-24. http://developer.apple.com/internet/webcon tent/xmlhttpreq.html.