一种全新的全文索引模型——后继数组模型*

刘学文, 陶晓鹏, 于 玉, 胡运发

(复旦大学,上海 200433)
E-mail: liuxuewen2001@yahoo.com
http://www.fudan.edu.cn

摘要: 提出了一种新的全文索引模型——后继数组模型,它结合了目前多个主流全文检索模型(倒排表模型、Pat数组模型等)的优点,提高了空间效率和时间效率,并得到了理论和实验的证明.

关 键 词: 全文索引;倒排表;Pat 数组;后继数组
中图法分类号: TP311 文献标识码: A

全文检索是文本数据库(它被定义为管理大量文本的系统)研究的核心.而全文检索的首要问题是全文索引模型的选择.目前主流的全文索引模型有倒排表模型和 Pat 数组模型[1]等,但它们都有自身的缺点.本文针对它们的不足,提出了一种全新的全文索引模型,称为后继数组模型.这种模型能有效地改进已有模型的不足,并且还可以作为文本挖掘的基础数据模型(统计型索引),而不仅仅是全文检索的模型(检索型索引).我们认为,这种模型才是真正符合未来全文数据库需要的.

1 全文索引模型概述

目前,比较流行的全文检索索引有倒排表模型和 Pat 数组模型.Pat 数组模型将文本看成一组字符串的有序叠合,用户输入的检索字符串就不会被分解成单个字符的集合,而是直接检索字符串.

例 1:我们用一个简单文本来说明上述两种模型的表示形式和使用方法.简单文本见表 1.表 1 中第 1 行表示文本的位置号,第 2 行表示简单文本,两行进行对照,能够方便地得到每个字符在文本中的位置(文本的起始位置为 0).

Table 1 A simple text

表 **1** 一个简单文本

0	1	2	3	4	5	6	7	8	9	10	11
a	b	c	d	e	A	b	d	e	a	b	c

对于倒排表模型,若检索字符串“cd”,从表头(如图 1 所示)为“c”的表项中得到其位置为 2,11,“d”的位置为 3,7,两者位置相差 1 的只有 c 位于位置 2,便得到检索结果.

对于 Pat 数组模型,从每个字符到文本尾字符形成一个字符串(这种字符串有时称为半无限串).如,表 1 中的简化文本包含了 12 个字符,也包含了 12 个字符串,如下所示:

(0) abcdeabdeabc; (1) bcdeabdeabc; (2) cdeabdeabc; (3) deabdeabc; (4) eabdeabc;
(5) abdeabc; (6) bdeabc; (7) deabc; (8) eabc; (9) abc; (10) bc; (11) c.

我们用字符串中首字符的位置标记该字符串,对这些字符串做字典排序,排序结果是(9)(0)(5)(10)(1)(6)(11)(2)

* 收稿日期: 2000-03-15; 修改日期: 2000-07-26

作者简介: 刘学文(1975－),男,江苏响水人,硕士,主要研究领域为数据库,知识库;陶晓鹏(1973－),男,四川成都人,博士,主要研究领域为数据库,知识库;于玉(1942－),上海人,教授,主要研究领域为数据库,知识库;胡运发(1940－),上海人,教授,博士生导师,主要研究领域为数据库,知识库.

(7)(3)(8)(4).将排序结果保存在一个数组中就得到了文本 Pat 数组,见表 2.

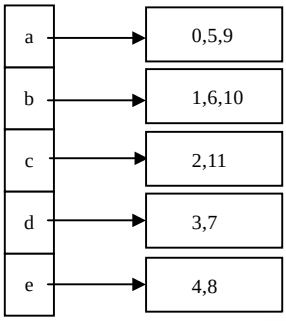

Fig.1 A simple inverted list model of a simple text

图 1 简单文本的倒排表模型表达形式

Table 2 A Pat array model of a simple text

表 2 简单文本的 Pat 数组模型表达形式

0	1	2	3	4	5	6	7	8	9	10	11
9	0	5	10	1	6	11	2	7	3	8	4

表 2 中第 1 行表示 Pat 数组的下标(以后各表类似,不再说明),第 2 行表示 Pat 数组值,每个值记录一个文本位置.

2 后继数组模型的设计

建立了一个文本,实际上就确立了一组字符的排列顺序(以后我们称为排列).文本文件记录、存放这组字符排列顺序.另外,当我们对一个文本建立了 Pat 数组索引之后,这组字符又确立了一个排列顺序.Pat 数组索引文件记录、存放这组排列顺序.因此,在 Pat 数组文本模型中,文本中的每个字符同时在两个排列中担任角色.由于目前 Pat 数组模型将这两个排列分别放置在不同文件中,而在创建和使用过程当中都要频繁穿插这两个排列(查询时使用 Pat 数组排列,读取和显示时使用文本排列),因此导致频繁的文件打开(关闭)操作,继而导致系统效率降低.我们的思路是在一个文件(或说一个材料)中保存这两种排列.

2.1 初始模型

我们用一个的简单例子加以说明.设有文本“bdac”,它的 Pat 数组确定的排列是:2,0,3,1.我们的初始改进模型用一个三元组排列来表示,三元组形如〈CharTextLoc,Char,NextCharPatLoc〉,其中 Char 标记字符, CharTextLoc 标记字符(同时也是以该字符为首的半无限串)在文本中的位置,NextCharPatLoc 标记字符在文本中的下一个字符在 Pat 数组中的位置.如,上例的三元组排列是〈2,a,2〉〈0,b,3〉〈3,c,Null〉〈1,d,0〉,其中 Null 是结束标记.如果再附加一个文本头在 Pat 数组中的位置指针,如 TextHeadPatLoc=1,则能够很容易地恢复文本.我们将这个初始改进模型称为带字符(即半无限串的头字符)的 Pat 数组模型(pat array with Char,简称 C-Pat).

假设要检索字符串“da”,类似于原来的 Pat 数组索引使用方法,采用二分法,但不同的是,由于 Pat 数组中元素已经带有它所指向的半无限串的第 1 个字符,因此,算法无须在比较前打开源文本,读取此半无限串即可直接比较第 1 个字符,如果第 1 个字符无法得到比较结果,继续比较下一个字符,下一个字符也无须读取源文本,根据三元组中的 NextCharPatLoc 字段得到下一个字符在 Pat 数组中的位置,再从中取得下一个字符的具体值.因此,带字符的 Pat 数组模型使用效率高于 Pat 数组模型.但是,它增加了相当大的空间开销.它用一个三元组代替了原 Pat 数组的一元(即半无限串的位置信息),空间开销大约增加了 1.5 倍.所以,我们需要对此模型继续加以改进.

2.2 改进1——文本位置信息的节省

我们注意到,在查询过程中,三元组的字符在文本中的位置字段(CharTextLoc)是没有用到的,但是它会影响查询结果的显示.解决方法是:Pat 数组的文本尾字符附带一些有关文本的信息,如文本号、文本总长度等等,那

么,当查询时发现结果以后,继续向后扫描,直到到达尾字符为止,从尾字符的信息就能得到该查询字符串所在的文本号和文本位置(文本总长度减去向后扫描的个数);或者是在文本的每个标点符号处附带文本信息,查询时,只要向后扫描到达最近的标点符号处,就能知道查询的结果.因此,我们改进的第 1 步就是省掉三元组中的 CharTextLoc 字段.节省后的模型称为带字符的 Pat 数组简约模型(concise Pat array with Char,简称 CC-Pat).它的空间开销是带字符的 Pat 数组模型的 60%,约为原 Pat 数组的 1.5 倍.这在目前的全文检索模型中是很优秀的指数,因为目前的系统的膨胀比大都在 2 以上.尽管如此,我们认为空间效率还是可以进一步提高的.

2.3 改进2——字符与后继信息的分离存储

我们将带字符的 Pat 数组简约模型的二元组拆开,分别保存在两个数据结构中,后继信息(即 NextCharPatLoc 字段)仍采用 Pat 数组的存储方式(称为后继数组),将字符(即 Char 字段)取出,存入另一个二元组数组(称为字符表)中.这个二元组的第 1 个组元保存头字符,第 2 个组元保存该字符在原 Pat 数组上的起始位置.这样就得到了本文最终的全文索引模型,我们称它为后继数组模型(SubSequence array).值得注意的是,它由两个数据结构组成,这一点使得它区别于以前的各种全文索引模型.采用两个数据结构的原因是让不同的数据以各自最高效的方式进行存储.

例 2:我们继续以表 1 的文本作为初始文本,它的索引模型见表 3 和表 4.

Table 3 A Pat array model with words of a simple text

表 3 简单文本的带字符的 Pat 数组模型表达形式

0	1	2	3	4	5	6	7	8	9	10	11
9,a,3	0,a,4	5,a,5	10,b,6	1,b,7	6,b,8	11,c,ϕ	2,c,9	7,d,10	3,d,11	8,e,0	4,e,2

符号ϕ是文本结束符,也可以视为半无限串结束符.它是一个特殊符号,本文假设它的字典序小于其他所有文本符号.

Table 4 A predigested Pat array model with words of a simple text

表 4 简单文本的带字符 Pat 数组简化模型表达形式

0	1	2	3	4	5	6	7	8	9	10	11
a,3	a,4	a,5	b,6	b,7	b,8	c,ϕ	c,9	d,10	d,11	e,0	e,2

简单文本的后继数组模型的表达形式见表 5 和表 6,其中表 5 是后继数组,表 6 是字符表.

Table 5 The part of the subsequence array in the subsequence array model of a simple text

表 5 简单文本的后继数组模型表达形式的后继数组部分

0	1	2	3	4	5	6	7	8	9	10	11
3	4	5	6	7	8	ϕ	9	10	11	0	2

Table 6 The part of character list in the subsequence array model of a simple text

表 6 简单文本的后继数组模型表达形式的字符表部分

a	b	c	d	e	ϕ
0	3	6	8	10	12

文本结束符没有在后继数组中出现.为了后续计算方便,我们给它赋值为后继数组的后续下标,本例为 12.显然,根据字符表能够很容易地得到每个字符在后继数组中的分布区域,因此后继数组模型能够完整地复原带字符的 Pat 数组简化模型.因此,带字符的 Pat 数组简化模型所具备的功能后继数组模型也具备.

3 后继数组模型的实现

我们提出了一种新的全文索引模型——后继数组模型,并且从理论上分析得出后继数组模型的创建效率远高于 Pat 数组.本节研究后继数组模型的具体实现方法,并用实验数据加以证明.

3.1 后继数组模型的数学基础

全文数据库中的文本和用户输入的检索条件都可以看成是字符串,分别称为文本串和检索串.设文本串为

t,那么 t 所包含的子串总数为 $\frac{1}{2}n(n+1)$,这种平方级的数量使得我们不可能对所有子串建立索引.Pat 数组模型从中挑选出 n 个子串进行索引,挑选方法是文本中任一位置的字符到文本结束形成的子串,称为半无限串(sistring).所有的半无限串经过排序形成串序列,因此全文检索的问题从检索串与文本串的关系转化为检索串与串序列的关系.

本节从数学的角度讨论一些串与串序列的关系和性质.这里给出的数学性质是本节后面的算法设计的基础.字符串简称串,我们认为它是公理性概念.一组按照字典序排列的串称为串序列,如果系列中没有两个串相等,则称为严格串序列.

在下面的叙述中,用小写字母表示串,用大写字母表示串序列.串在串序列上的上确界和下确界从数列中相应概念移植得到.

定义 **1**. 串的第 1 个字符称为串头,去掉串头的子串称为串尾.显然,串等于串头和串尾的连接结果.在文本串形成的半无限串序列中,文本中的后串是前串的串尾.

定义 **2**. 串序列的第 1 个串称为头串,最后一个串称为末串.在串序列中,相邻的两个串互称为前串和后串.由于串序列一般是按字典序升序排列的,因此后串比前串大.

定义 **3**. S 是一个串序列,从 S 中任取相邻的一组串得到的串序列称为 S 的子串序列.显然,子串序列的个数与串序列的串个数成指数关系.

定义 **4**. 存在串 α 和串序列 S.串 α 将 S 中的串分成 3 部分,第 1 部分是小于 α 的串,第 2 部分是等于 α 的串,第 3 部分是大于 α 的串.设 β 是第 1 部分的最大串,γ 是第 3 部分的最小串,则称 β 是 α 在 S 上的下确界,γ 是 α 在 S 上的上确界.

定义 **5**. 两个串 α 和 β 的比较有 4 种结果:(1) α 与 β 完全相同;(2) α 是 β 的前缀;(3) α 和 β 有共同的前缀;(4)其他.在 Pat 数组模型中,将前两种情况称为相等,其中第 1 种情况称为严格相等;第 3 种情况称为前缀相等;第 4 种情况称为不等.在 Pat 数组模型中,当创建索引时,相等的含义是指第 1 种情况;当全文检索时,相等的含义是指第 2 种情况.第 3 种情况常常用在近似检索和扩充检索中.

性质 **1**. 两个串进行比较的方法可以是,首先比较串头,然后比较串尾.因此,如果已知两个半无限串的串尾的比较结果,那么只需比较它们的串头就能得到最终比较结果,从而加快了比较速度.在 Pat 数组模型中,一个文本串往往是另一个文本串的串尾.因此,在比较一个文本串时,就意味着另一个文本串的串尾已有了比较结果.将来在比较另一个文本串时,可以借用这个结果.

性质 **2**. 一个串序列 S 和一个串 α,β 是 α 的前缀,如果 γ 和 δ 分别是 β 在 S 上的下确界和上确界,则 α 的下确界一定大于或等于 γ,α 的上确界一定小于或等于 δ.这个性质可用于缩小求取上、下确界时的搜索范围,即一个串的上、下确界是处于其前缀的上、下确界之间的.

性质 **3**. 设 α 和 β 是两个串,T 是一个串序列.如果 $\alpha<\beta$,则 α 在 T 上的上确界和下确界分别小于等于 β 在 T 上的上确界和下确界.如果 $\alpha>\beta$,则 α 在 T 上的上确界和下确界分别大于等于 β 在 T 上的上确界和下确界.这个性质能够缩小求取后续串在串序列上的上、下确界时的搜索范围.它也说明,如果求取一组串的上、下确界,那么依照它们的大小顺序这样求取,能够不断缩小搜索范围,达到优化检索的目的.

性质 **4**. 设有串序列 S 和串 α,则 α 在 S 上的上确界或下确界可能不属于 S,但是不可能都不属于 S.这个性质对于后面的算法对空间的使用很有帮助.

性质 **5**. 设有串序列 S 和串 α 和串 β,β 属于 S.如果 $\beta=\alpha$,则 α 在 S 上的上确界大于等于 β,下确界小于等于 β;如果 $\beta<\alpha$,则 α 在 S 上的上确界和下确界都大于等于 β;如果 $\beta>\alpha$,则 α 在 S 上的上确界和下确界都小于等于 β.这个性质说明在搜索上、下确界的过程中,有许多重合的步骤,直到出现了等值以后,两者的搜索方向才变得不同.

性质 **6**. 存在串序列 S 和串 α,β 和 γ 分别是 α 在 S 上的上确界和下确界,则与 α 相等的串全部出现在 β 和 γ 之间.这个性质揭示了在 Pat 数组模型中,索引的创建算法和检索算法实质是相同的,即都是在串序列中寻找上、下确界.如果是检索串,则返回上、下确界之间的数值作为检索结果;如果是文本串,则将该文本串插入到上、下确界之间.

性质 **7**. 设有严格的串序列 S,任给串α.如果α在 S 上存在,则α在 S 上的上确界和下确界位置相差 2;如果α不在 S 上存在,则α在 S 上的上确界和下确界位置相差 1.这个性质是性质 6 的一个推论,它也预示了在严格的串序列上的运算要简单.

性质 **8**. 存在两个串序列 S 和 T,α是属于 S 的一个串.$b1,b2,...,bn$ 是属于 T 的一组串,且它们是 T 中所有在 S 上的上确界为α的串,则:(1) $b1,b2,...,bn$ 形成 T 的一个子串序列;(2) $b1,b2,...,bn$ 中的最大串是α在 T 上的下确界.

证明:(1) 用反证法.假设 $b1,b2,...,bn$ 不是 T 的一个子串序列,则至少存在一个属于 T 的串,不妨设为 $bi,b1<bi<bn$,且 bi 在 S 上的上确界不是α.则有两种情况,分别讨论如下:

· bi 在 S 上的上确界比α大,则 $bi>=\alpha$.根据 bn 在 S 上的上确界是α,则 $bn<\alpha$,得到 $bi>bn$,与前面的假设矛盾.

· bi 在 S 上的上确界比α小,则 $bi<\alpha$.根据 $b1$ 在 S 上的上确界是α,则 $b1<\alpha$,得到 $bi<b1$,与前面的假设矛盾.

即无论出现何种情况,由前面的假设都能导出与自身矛盾的结论.因此,假设不成立,命题得证.

(2) 用反证法.假设α在 T 上的下确界是γ,且γ不在$\beta1,\beta2,...,\beta n$(以后统称为$\beta i$)中,即$\gamma$在 S 上的上确界不是α.根据(1)可知,至多存在两种情况:

· γ比所有的βi 都小.由上面假设得到$\gamma<\alpha$,又由于βi 在 S 上的上确界是α,则$\beta i<\alpha$,因此 T 中的串$\gamma,\beta i$ 都小于α,由于γ是下确界,因此$\gamma>\beta i$.矛盾.

· γ比所有的βi 都大.设γ在 S 上的上确界是δ,则可以进一步分两种情况讨论.

① $\delta<\alpha$.由于$\gamma>\beta i,\gamma<\delta$,则$\beta i<\delta$,这与$\alpha$是$\beta i$ 在 S 上的上确界矛盾.

② $\delta>\alpha$.由于$\gamma<\alpha$,这与 δ 是γ在 S 上的上确界矛盾.

即无论出现何种情况,由前面的假设都能导出与自身矛盾的结论.因此,假设不成立,命题得证. □

性质 **9**. 存在两个串序列 S 和 T,α是属于 S 的一个串.$\beta1,\beta2,...,\beta n$ 是属于 T 的一组串,且它们是 T 中所有在 S 上、下确界为α的串,则:(1) $\beta1,\beta2,...,\beta n$ 形成 T 的一个子串序列;(2) $\beta1,\beta2,...,\beta n$ 中的最小串是α在 T 上的上确界.

性质 9 的证明类似于性质 8,它们沟通了两个序列之间上、下确界的关系,使得后面的算法能够通过一个串序列上的上确界求得另一个串序列上的下确界,反之亦然.另外,这个结论还可以推广到多个串序列.

3.2 模型实现

后继数组模型是在改进 Pat 数组模型的基础上提出的,因此它的创建算法也可以将 Pat 数组创建算法作为改进的基础.Pat 数组模型的创建算法分为两大步骤:(1) 单个文本的 Pat 数组的创建.由于空间小,整个过程可以在内存中进行.我们常常把它称为小 Pat 数组、内存 Pat 数组等等.(2) 单个文本的 Pat 数组与已有的 Pat 数组的合并.已有的 Pat 数组是已处理的文本的 Pat 数组合并的结果,随着合并数目的增加,合并的结果也越来越大.已有 Pat 数组常常称为大 Pat 数组,又由于它总是保存在磁盘中,而且处理时也一般使用磁盘操作手段,因此也常常被称为文件 Pat 数组或 Pat 数组文件.

3.2.1 单个文本的后继数组模型的创建

算法的前提是 Pat 数组已创建完毕,即文本包含的所有半无限串已按字典排序完毕.根据当前半无限串,找到它在文本中的下一个半无限串中,在 Pat 数组上确定找到的半无限串的位置,将此位置记录在当前 Pat 数组的元素中.这个方法要求所有的半无限串都在数组中(包括以禁用字符开头的),否则,找到的半无限串可能是等值而不同位置的,它除了带来空间的额外开销,还在两方面带来额外的时间开销,一方面是做了大量无用半无限串的排序,另一方面是半无限串在 Pat 数组上的查询.

设文本长度为 m,共有 $m/2$ 个半无限串(也是 Pat 数组的长度),查找每个半无限串要做 $\log(m/2)$次半无限串比较,每次半无限串比较要 m 次字符比较,则总的字符比较次数是 $O(m^2\log(m/2))$.

3.2.2 归并计数数组的计算

小 Pat 数组与大 Pat 数组的合并实质分为两步,首先填写归并记数数组,然后在归并记数数组的指导下完成归并.总的来说,后继数组模型完全避免了源文本的打开操作,但是在数组文件上的移动次数大大增加,实验结论是,在 Pat 数组文件上的移动次数为 cm(m 是文件长度),打开(关闭)文件次数也为 cm(原 Pat 数组创建方法的

特点就是每在Pat数组文件上移动一次就伴随着一次指向文本的打开).实验表明,99%的半无限串在6个字符内比较出结果,因此,如果是在带字符的Pat数组上进行比较,则每比较一次半无限串平均约要在Pat数组文件上移动7次,半无限串的内存比较次数共有$O(k\log^2 m)$,由于常常是多次文件移动才有1次读盘操作,因此,我们相信后者效率高于前者.实验结果证实了我们的猜想,归并计数数组计算的时间开销大约只有原算法的1/6.

归并计数数组实际上记录了增加文本的每个后继数组(小数组)索引值在原后继数组(大数组)索引上的下确界.根据性质8和性质9容易反向求得大数组值在小数组上的上确界.

3.2.3 归并时后继数组的后继信息的调整

归并导致两个数组(内存中的小后继数组和外存中的大后继数组)的后继信息的重定位,但是小数组的重定位由于数组长度有限且在内存进行,算法效率很高.因此,我们只考虑大数组的重定位问题.

大数组的重定位根据两个数据得到:(1) 原来的后继字符指针;(2) 归并时生成的归并计数数组.根据性质8和性质9,需要反向求出大数组每个元素在小数组上的下确界.简单的方法是遍历计数数组,找到相同值的最大下标作为增加值.由于大数组长度很大,如果每个元素都要在计数数组上遍历,时间开销是很大的,为了提高效率,程序先对计数数组做一个处理,使得计数数组的所有值保持严格有序排列,随后的增加值则能以二分方式在计数数组上查找到,将线性的时间开销缩减到对数级.实验效果也很明显.

尽管如此,归并时的时间开销仍然要高于Pat数组创建算法的相应时间开销,因为原算法没有位置指针的重置.

4 实验结果

实验文档来自复旦大学BBS武侠版,主要是金庸的武侠小说(这样,文本间就有大量的共同词汇,以增加测试的难度),共有50个文本.

表7是测试文本属性一览表.其中TextId是文本号(the number of the text),TxtSize是文本字节数(the size of the text).

我们已测试了大量文档,有些问题也只有在较大数量的文档情况下才能看得比较清楚,但限于文章篇幅,我们只选取了50个文档的实验结果,选取方法是每隔5个文档选取一个.用于实验的机器配置是Intel Pentuem 166,内存64M.程序用Microsoft Visual C++ 5.0编写.

Table 7 The list of the texts in the experiment

表7 实验文档情况一览表

TextId	TxtSize	TextId	TxtSize	TextId	TxtSize	TextId	TxtSize
1	43 519	14	44 664	26	50 870	39	50 672
2	48 166	15	46 367	27	57 351	40	46 829
3	43 992	16	45 304	28	46 044	41	45 982
4	41 666	17	43 137	29	49 902	42	41 266
5	43 387	18	50 228	30	57 312	43	41 053
6	48 403	19	41 475	31	50 452	44	46 320
7	46 036	20	41 180	32	47 916	45	43 998
8	44 743	21	53 439	33	47 981	46	52 105
9	47 726	22	51 352	34	47 864	47	41 401
10	47 779	23	42 557	35	50 029	48	53 107
11	45 347	24	52 185	36	46 026	49	47 106
12	46 924	25	48 601	37	44 300	50	52 197
13	47 474			38	49 795		

根据汉语的特殊性,在实际创建Pat表时,常应用前缀文件.前缀文件保存了所有半无限串的一定长度的前缀,这些前缀的保存顺序与对应的半无限串在Pat数组文件的顺序是一致的.因此,当算法在Pat数组文件上移动时,能够很快得到该数组元素所指向的半无限串的前缀.根据文献[2]的汉语词频表,也可以得出类似的结论.

后继数组空间开销相当于带有长度为2的前缀的文件的原Pat数组创建算法,因此,比较两者的效率很有意义.另外,根据实验结果,当前缀长度为5时,原Pat数组创建算法在时间和空间两方面的整体性能最优.我们也给出带字符的Pat数组与前缀长度为5的原Pat数组创建算法的比较.它证明带字符的Pat数组的创建算法在节

省空间的同时,还显著地提高了时间效率.

以下的 Pat 数组创建算法都选用比较快的 DMB(基于内存的双边二分归并算法(the merging algorithm in double sides based on memory))[3]算法.为了说明方便,我们记后继数组模型的创建算法为 SSA(the algorithm of structuring subsequence array),记前缀长度为 n 的原 Pat 数组创建算法为 Pat-n,比如前缀长度为 2 的原 Pat 数组创建算法记为 Pat-2.有时我们将用到的具体算法写出来,比如,DMB-2 表示使用前缀长度为 2 的 DMB 算法.实验结果如图 2~图 5 所示.

从总的时间效率来看,SSA 比 DMB-5 提高 1 倍,而 DMB-5 又比 DMB-2 高出 1 倍多,因此,SSA 的效率大约为与它空间消耗相当的 DMB-2 的 4 倍.

5 结 论

本文提出了一种新的全文索引模型——后继数组模型,它结合了目前多个主流全文检索模型(倒排表模型、Pat 数组模型等)的优点,提高了空间效率和时间效率.它的创建速度优于 Pat 数组,查询速度优于倒排表,其空间效率与 Pat 数组和倒排表相当.理论分析和实验结果都证明了该结论.

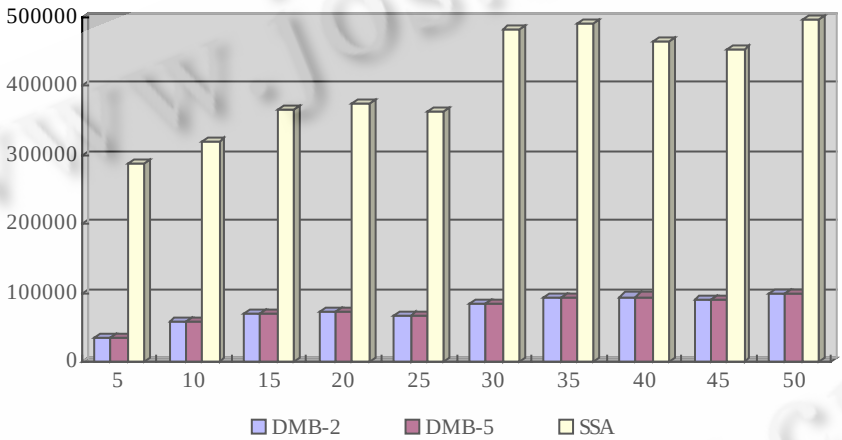

Fig.2 Comparison of times of moving in Pat array file (MIdxNum)

图 2 算法在 Pat 数组文件上的移动次数(MIdxNum)的比较

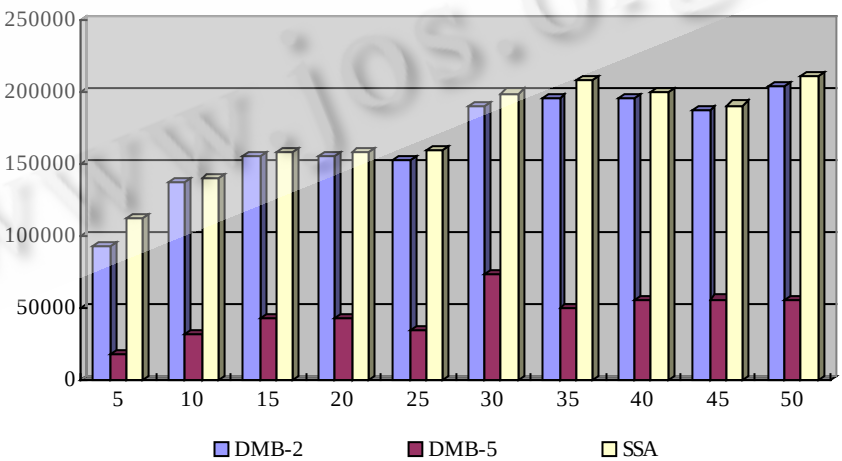

Fig.3 Comparison of times of moving in memory Pat-array (MArrNum)

图 3 算法在内存 Pat 数组(MArrNum)上移动次数的比较

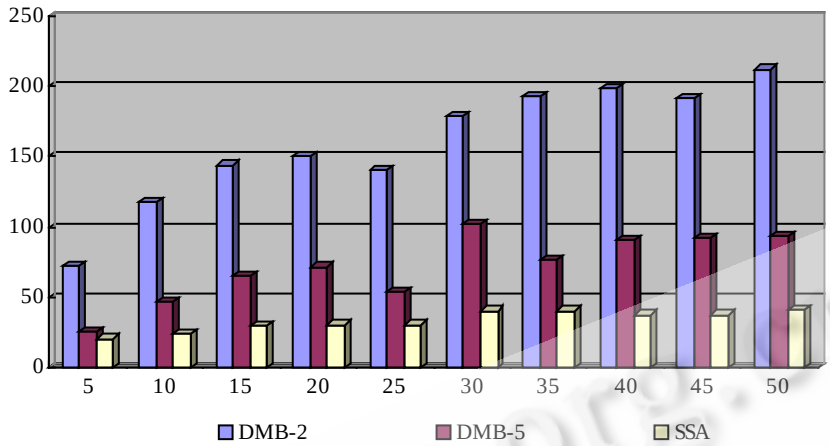

Fig.4 Comparison of the time of computing the array for counting (CCATime)

图 4 算法计算归并计数数组的时间开销(CCATime)的比较

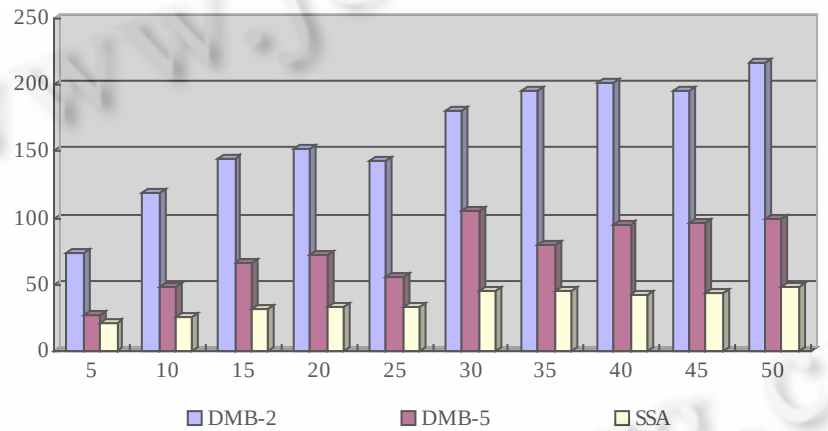

Fig.5 Comparison of algorithms in the cost of time

图 5 算法总的时间开销比较

References:

[1] Gaston, Gonnet, Ricardo, Baeza-Yates, Snider, T. New indices for text: Pat trees and Pat arrays. In: Frakes, W.B., Ricardo Baeza-Yates, eds. Information Retrieval Data Structure and Algorithms. Englewood Cliffs, NJ: Prentice Hall, 1992. 66~81.

[2] Language-Teaching Research Group. The Dictionary of Modern Chinese Frequency. Beijing: Publishing Company of Beijing Institute of Language, 1986 (in Chinese).

[3] Tao, Xiao-peng. Full-Text index research based on full-text database (Chinese) [Ph.D. Thesis]. Shanghai: Department of Computer Science, Fudan University, 1999 (in Chinese).

附中文参考文献:

[2] 北京语言学院语言教学研究所.现代汉语词频词典.北京:北京语言学院出版社,1986.

[3] 陶晓鹏.面向(中文)全文数据库的全文索引的研究[博士学位论文].上海:复旦大学计算机科学系,1999.

A New Full-Text Index Model-Subsequence Array Model*

LIU Xue-wen, TAO Xiao-peng, YU Yu, HU Yun-fa

(*Fudan University*, *Shanghai* 200433, *China*)

E-mail: liuxuewen2001@yahoo.com

http://www.fudan.edu.cn

Abstract: In this paper, a new full-text index model, subsequence array model, is put forward. It has the advantages of many popular full-text index model, such as inverted-list model and Pat array model, and improves the efficiency of the space and time, which is proved by theory and experiment.

Key words: full-text index; inverted list; Pat array; subsequence array

* Received March 15, 2000; accepted July 26, 2000

第 1 届全国几何设计与计算学术会议
征 文 通 知

由中国工业与应用数学学会几何设计与计算专业委员会主办、山东大学承办、青岛大学协办的第 1 届全国几何设计与计算学术会议(CSIAM Geometric Design & Computing 2002)定于 2002 年 6 月 1 日~3 日在青岛召开。本次会议将以“几何设计和计算理论及其在工业中的应用”为主题,结合工业设计中急需解决的关键问题和难点问题,开展广泛的学术交流和讨论。会议将特邀国内外著名专家学者就几何设计和计算的最新动态和热点问题做专题讲演和报告。会议期间将举办“计算机辅助几何设计”高级研讨班,还有 CAD 系统、图形软件和硬件、计算机新产品展示会和专题报告会。欢迎有关公司、厂商和研究单位报名参展。有关问题请与 CSIAM Geometric Design & Computing 2002 秘书处联系。

一．征文范围

自由曲线和曲面设计	科学计算可视化	代数曲面及其在 CAGD 中的应用
计算几何	网格的生成与处理	几何计算的稳定性
CAD/CAM 技术及应用	细分曲面造型方法	图形动画设计技术
基于图像的建模和绘制	反向工程	三维造型系统介绍
计算机图形学	散乱数据点造型	

二．征文要求及重要日期

1．稿件应反映在几何设计与计算及相关领域有创见的理论或应用研究成果,并未在国内外公开刊物及其他会议上发表过。可先提交论文摘要。

2．大会将正式出版会议论文集。会议的优秀论文将推荐到《计算机辅助设计和图形学学报》作为专辑在 2002 年正式发表,部分优秀论文将推荐到《计算机学报》、《软件学报》。

3．鼓励电子投稿(Word 格式),联系方式: yxq@cs.sdu.edu.cn

邮寄稿件需打印稿一式三份,来稿请寄: 山东大学计算机科学与技术学院 杨兴强

三．重要日期

截稿日期: 2002 年 2 月 25 日 录取通知日期: 2002 年 3 月 15 日 论文付印截止时间: 2002 年 5 月 1 日

四．联系地址**(CSIAM Geometric Design & Computing 2002 秘书处)**

250100 济南市山大南路 27 号 山东大学计算机科学与技术学院 张彩明 杨兴强

Tel: 0531–8565415