迴溯查找算法的实现及应用

衡阳第四机械厂 刘 星
衡阳建设银行 罗家乐

一个完善的数据库管理系统，查询功能是必不可少的，而查询的速度和效益是衡量应用管理系统先进程度的一个重要指标。本文即是从应用程序设计的角度，提出一种旨在缩短整体查找时间的迴溯查找算法。

一、问题的提出

在一般查找过程中，无效时间花费过多，特别是在大数据库中进行模糊组合查找时由于符合条件的记录很多，情况就更突出。因此，有必要设计一种灵活而又功能齐全的算法，这种算法至少应具有下列功能：

1.已查到过的记录应能重新显示，勿须再次输入条件从头查找。

2.能查找任意指定区域的数据记录。

3.已找到的成批记录应能迅速上下翻阅和跳阅。

4.计算机能处理查找出所有符合条件的记录。

5.查询结果应能多次使用。

这种算法需要解决 LOCATE 查找命令只能向下找，不能向上找的现象，而解决问题的关键在于“记”住已找得数据的地址(记录号)。

二、算法的实现

算法采用顺序存储结构的线性表实现。用一组地址连续的存储单元依次存储线性表的各个数据元素。而各个数据元素就是查找到的符合条件的记录在数据库文件中的物理位置(记录号)。

这种线性表的顺序存储结构又被称之为向量。每一个数据元素的存储位置只取决于该元素在线性表中的序号，而与数据元素本身的值无关，只要确定向量的起始位置，向量中任一元素都可随机存取。我们确定向量的起始位置为 1,那么其对应的数据元素为我们查得的符合条件的第 1 个数据库文件记录的记录号。

下面是该算法数据结构的示意图：

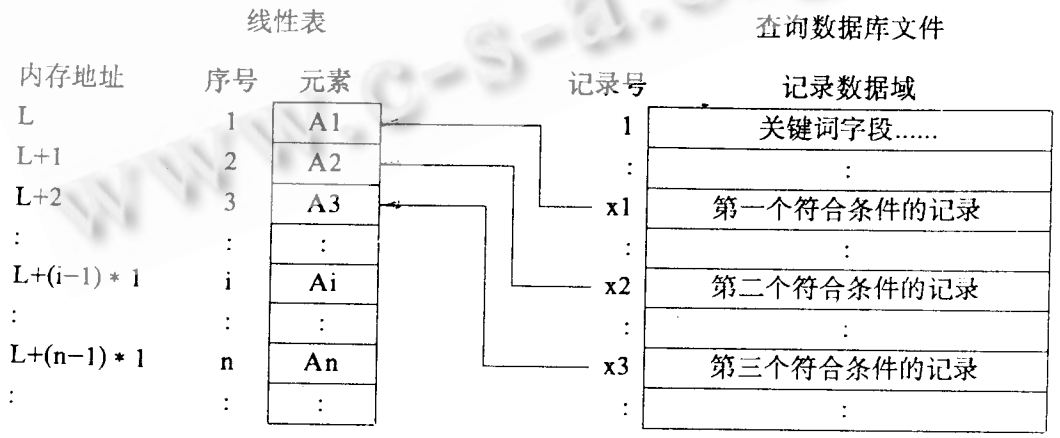

图 算法数据结构示意图

实现算法的简略步骤如下：

1.定义线性表并置空。

2.设N为符合条件的记录总数，I为显示指针，A(I)为符合条件记录的记录号，则 N＝I＝0。

3.设M为线性表的容量(最大序号)，当N＞M时，I＝1，构成循环线性表。

4.根据条件查找数据库文件，找到记录后比较N＞M，如果是则 I＝1，否则 I＝I+1，N＝N+1。A(I)＝记录号。

5.重复(4)直至数据库文件尾状态为真。

6.阅览时移动I，取出记录号，直接GO记录号。显示记录即可。

本文后附实现这种算法的示范程序。程序在LX286机器上通过，软件环境为DOS3.3，LXPCPLUS，C-FoxBASE PLUS2.0。

三、程序的说明

示范程序中的“线性表”使用 MFOXBASE PLUS2.0 中的数组来实现。第6行语句定义一个有200个元素的一维数组，可存放200个记录号，第145行语句定义“线性表”的终止位址接上起始位址，构成循环线性表，当查找到的记录数超过200个时，以后找到的记录号又从“线性表”的起始位址开始存放。

程序经过一定优化，如果输入查询条件中存在有主关键词，程序将自动判别并自动缩小查找范围以缩短查找时间；回溯查找的实现过程是“透明的”，勿须人工干预。

程序中用→，←，↑，↓，PageUp，PageDown，Home，End 键及 Ctrl 键与以上键的组合来实现翻阅跳阅及查询。如果查询时按下 End 键，这时用户可以去做些别的事情，程序将把整个库中符合条件的记录找出并“装订成一本书”，然后唱着歌等待阅览。以后再按 End 键，程序将简单的直接“GO 记录号”，显示出库中最后一条符合条件的记录。这就是批处理查找和一次找出多次使用！此后，以上键仅仅用来翻阅和跳阅已找出的符合条件记录的“一本书”。第21-31行语句是根据数据库的大小自动计算并设定跳阅的步长。

查询时按下 End 键，机器在查找期间将一心一意的查找整个库中符合条件的记录而不理睬用户命令。如果既想有批处理的功能又可以随时阅览已找到的记录怎么办？去参看下一部分。

第127行语句是判断查到的记录数是否超过200条，如果是则提醒用户仔细察看比较起初找到的记录是否是理想的数据记录，再继续查找到的记录号将覆盖以数组为线性表的起始位置。有兴趣的同行在实际应用中可使用宏代换函数或者数据库定义出可变长线性表，就不会出现这种装不下而导致覆盖的情形，但稍稍会付出减慢程序运行速度的代价。当然，在内存足够的机器上，可以预先定义大容量的线性表以解决这个矛盾。FoxBASE PLUS2.0 的内存变量数组最多可定义3600个元素，相信已足够使用。

程序定义 F2 键为查找锁停＼开放开关，当查到的记录数超过200条或者在查找任意时刻，可按 F2 键锁停查找以便仔细翻阅已查到的记录，再按 F2 键则查找继续。

如果查找过程伴随着修改或删除操作时，“迴溯查找算法”的效果将更加明显，由于直接“GO 记录号”操作，速度快且可多次前后比较。

特别是当索引文件打开时，可以放心大胆的成批修改主关键词，而不用担心逻辑指针的移动使屏幕内容莫明其妙。

由于“迴溯查找”能记住多次查询结果，因此对于复杂的逻辑组合条件查找，该程序可简化条件的输入以适应非专业操作人员的需要，如仅考虑“与”条件的查找，“或”条件的查找可分为两次进行。这样也简化了编程。

四、“分时”查找与“零等待”

原理如下：首先设计一固定长的记录块，指定程序每次查找固定长的记录块后扫描一次键盘，看是否有用户命令键入，如果有，则转而执行命令后再继续查找下一个固定长的记录块。固定长记录块的大小可根据机器及系统软件的运行速度设定，以查找一次所需时间不超过3秒为好。本人在 LX286 机器和 FoxBASE PLUS2.0 环境中用 LOCATE 命令进行复杂条件的模糊组合查询，使用&函数查找1000个记录费时3秒；因此在示范程序中设计固定长记录块的长度为1000个记录。在查找一

个固定长记录块时，可能有符合条件的记录，也可能没有符合条件的记录。程序设计为每当找到一个记录时即暂停查找，去扫描键盘，然后再继续找。这样可保证程序对键盘的响应时间小于3秒。

在分时操作系统中，系统是轮流给一个终端分配相等的“时间片”，使每一个终端用户感觉上象独占机器资源。而在这里，程序是轮流给“查找一个固定长记录块”分配一小段时间(约3秒)，给键盘扫描操作分配一点时间(约4微秒)。当符合条件的记录在数据库中的位置分布不是太畸形时，用户感觉上象屏幕中早已准备好符合条件记录的“一本书”，等待着“浏览”。从而实现所谓“零等待”查找。

如果符合条件的记录均在数据库底部，即使在这种最坏的情况下，程序也不会只顾查找而对用户的命令无动于衷，而是会殷勤的回答用户的键盘命令，从而创造出更友好的用户界面。

考虑到这种最坏的情况，程序还可作进一步的改进，把数据库以某种合乎经验的算法分割成非连续的一些“固定长记录块”，将能提高“尽快查到符合条件记录”的概率。

五、查询“中断”

当按下F1功能键时，程序中断查询转为浏览源数据库，再按F1键则返回查询状态。

F1键相当于一个开关，“开”时线性表中记录号起作用，“关”时将线性表“挂起”而使数据库文件中的记录指针起作用。实现这种转换的关键在于“记住”已找得记录的记录号和能立即恢复转换前状态。用中断的术语来说就是现场保护和恢复现场。

这里也用到了一点小技巧，F1键是系统固定设置的HELP键，不能重新定义；但演示程序却可将其定义为查询或浏览的开关。我们只要提醒一下INKEY( )函数返回F1键键值为28，有心的同行就一定能悟出原因何在了。

六、分析与总结

在迴溯查找算法的基础上，可以将现代计算机系统设计中一些重要思想(如中断，分时，批处理)在一个短短查询程序中得到应用，从而大大提高查询的效率。该算法并不尽适用于所有全部的查询场合。实事上，该算法也不能缩短单个命令的查找时间，它只是从程序设计的角度来缩短查询所需的整体时间。

在下列情况中，使用该算法进行查询堪称最快；(1)需要多次显示查找到的成批记录；(2)对大数据库的模糊组合查询；(3)符合条件的记录可能较多；(4)非关键词查询；(5)查找伴随着修改；(6)需要多次存储查询结果。

附带提一下，如果数据库系统软件的设计者在设计LOCATE、SEEK等命令时，使系统具有暂存查找结果(记录号)的功能，将给应用人员带来更多的方便。

本文提供的示范程序稍加修改可作为一个通用子程序供调用。

```
1  *迴溯查找算法示范程序
2  SET TALK OFF
3  SET ESCA OFF
4  CLEAR
5  LDATA=200
6  DIMENSION FORM (LDATA)
7  PNT=0
8  LPNT=0
9  MPNT=0
10 SPNT=0
11 STORE PNT TO FORM
12 F1=.F.
13 END=.F.
14 OTH=.T.
15 CNTN=.T.
16 CONDI='F'
17 PIECE=1000
18 SELECT A
19 USE CLWJ INDEX CLWJSY
20 END_REC=RECNO( )
21  STEP=SQRT(SQRT(SQRT(RECCOUNT( )/3)))
22 STEP24=1
23 STEP5=-1
24 STEP4=VAL(STR(INT(STEP)))
25 STEP19 =-1*VAL (SUBSTR (LTRIM (STR
   (STEP^2)),1,1)+SUBSTR ('0000000',1, LEN (LTRIM (STR
   (STEP^2)))-1))
```

```
STEP3 =VAL (SUBSTR (LTRIM (STR (STEP3)),1,1)+ SUBSTR ('0000000',1, LEN (LTRIM (STR (STEP3)))-1))
STEP18= -1 * VAL (SUBSTR (LTRIM (STR (STEP4)),1,1)+ SUBSTR ('0000000',1, LEN (LTRIM (STR (STEP4)))-1))
STEP2= VAL (SUBSTR (LTRIM (STR (STEP5)),1,1)+ SUBSTR ('0000000',1, LEN (LTRIM (STR (STEP5)))-1))
STEP26= -1 * VAL (SUBSTR (LTRIM (STR (STEP6)),1,1)+ SUBSTR ('0000000',1, LEN (LTRIM (STR (STEP6)))-1))
STEP30= VAL (SUBSTR (LTRIM (STR (STEP7)),1,1)+ SUBSTR ('0000000',1, LEN (LTRIM (STR (STEP7)))-1))
STEP31= -1 * VAL (SUBSTR (LTRIM (STR (STEP8)),1,1)+ SUBSTR ('0000000',1, LEN (LTRIM (STR (STEP8)))-1))
XX24="退出 Esc 打印 P 条件 F 查询/浏览 F1 锁停/开放 F2"
   24,0 SAY XX24
DO WHILE .T.
IF F1
   0,0 say '查询'
ELSE
   0,0 say '浏览'
ENDIF
IF CNTN .OR. .NOT. F1
   0,col( ) SAY'开放'
ELSE
   0,col( )SAY'锁停'
ENDIF
IF OTH
CLEAR
IF PNT>0 .OR. .NOT. F1
显示记录内容
ENDIF
OTH = .F.
ENDIF
KEY = INKEY( )
DO CASE
CASE DEY = 27
CLEAR
CLEAR ALL
SET ESCA ON
RETURN
CASE KEY = 28
F1 = IIF(F1, .F., .T.)
IF PNT > 0.AND. PNT > = SPNT .AND. PNT < = LPNT
GO FORM (PNT)
ENDIF
OTH = .T.
LOOP
CASE KEY = -1 .AND. F1
CNTN = Ⅱ F(CNTN, .F., .T.)68 CASE KEY = 24 .OR. KEY = 5 .OR. KEY = 19 .OR. KEY = 4 .OR. KEY = 3 .OR. KEY = 18 .OR. KEY = 26 .OR. KEY = 30 .OR. KEY = 31
STEP = 'STEP'+LTRIM(STR(KEY))
STEP = &STEP
DO MOVE_PNT
OTH = .T.
LOOP
CASE KEY = 1 .OR. KEY = 6
IF F1
PNT = IIF(KEY = 1,SPNT,LPNT)
IF PNT > 0
GO FORM(PNT)
ENDIF
ELSE
IF DEY = 1
GO TOP
ELSE
GO BOTTOM
ENDIF
ENDIF
OTH = .T.
LOOP
CASE KEY = 70 .OR. KEY = 102
CONDI1 = (10)
COND12 = (70)
调用查询条件输入子程序
IF CONDI2 = ' '
LOOP
ELSE
CONDI = LTRIM(TRIM(CONDI2))
ENDIF
```

```
98 CONDI1 = LTRIM(TRIM(CONDI1))
99 OTH = .T.
100 F1 = .T.
101 END = .F.
102 IF ASC(CONDI1) = 0
103 GO TOP
104 ELSE
105 SEEK CONDI1
106 IF EOF( )
107 GO BOTTOM
108 ENDIF
109 ENDIF
110 END REC = RECNO( )
111 IF LPNT > 0
112 YN = ' '
113 @ 23,0 SAY '按Y键保存上次查得记录,其它键覆盖...
    GET YN
114 READ
115 IF UPPE(YN) < > 'Y'
116 PNT = 0
117 LPNT = 0
118 MPNT = 0
119 SPNT = 0
120 STORE PNT TO FORM
121 ENDIF
122 ENDIF
123 ENDCASE
124 IF F1 .AND. .NOT. END .AND. CNTN .AND.
    CONDI < > 'F'
125 GO END REC
126 LOCA NEXT PIECE FIR &CONDI
127 IF FOUND( )
128 MPNT = MPNT+1
129 IF MPNT > LDATA
130 @ 23,0 SAY '查得记录数超储界限,按F2键仔细阅览,Y键
    覆盖 ...'
131 YN = 'Y'
132 DO WHILE YN = 'Y'
133 KEY = INKEY( )
134 DO CASE
135 CASE KEY = -1
136 CNTN = F.
137 YN = 'N'
138 CASE KEY = 121 .OR. KEY = 89
139 YN = 'N'
140 ENDCASE
141 ENDDO
142 IF .NOT. CNTN
143 LOOP
144 ENDIF
145 MPNT = 1
146 ELSE
147 IF LPNT > = MPNT
148 LPNT = MPNT
149 ENDIF
150 ENDIF
151 FORM(MPNT) = RECNO( )
152 ENDIF
153 IF .NOT. EOF( )
154 SKIP
155 ELSE
156 GO BOTTOM
157 END = .T.
158 ENDIF
159 END_REC = RECNO( )
160 ENDIF
161 IF SPNT < = 0 .AND. LPNT > 0
162 SPNT = 1
163 PNT = SPNT
164 GO FORM(SPNT)
165 OTH = .T.
166 ENDIF
167 ENDDO
```

注:限于篇幅,程序有删节。

▲