# 档案管理系统主题词查询的一个优化算法

柳见成　（长沙水电师院计算中心）

**摘要**：本文分析了在档案管理系统主题词查询中普遍使用的倒排文件算法的不足，提出了一种改进算法。利用档案／主题词表的稀疏矩阵结构特点，将该稀疏矩阵的存储方式映射成两个索引文件，然后把对该矩阵的检索算法映射成对两个索引文件的顺序复合查找，从而可压缩存储空间，提高查询速度，且与数据库中档案的份数无关。

## 1.主题词查询与倒排文件算法

在档案管理系统的各种查询操作中，主题词作为档案内容特征描述的主要方面，其出现的频率一般来说是最高的，因此在数据库结构和查询程序的设计中应当尽量予以支持。

由于一份档案的主题词通常有许多个，按主题词查询在数据库中归结为多键值检索的问题。迄今为止，最普遍的方法是采用倒排文件算法，即由次关键字形成倒排文件并与主关键字索引相结合进行检索。次关键字和记录之间没有一一对应关系，因此次关键字不能唯一确定记录。由于该方法是在两个索引文件上进行顺序复合查找，因而与数据库记录多少无关，从而能提高查询速度。图 1 为按档案主题词表形成倒排文件。A1 至 An 为档案记录编号，B1 至 Bm 表示在该数据库中出现的不同主题词。

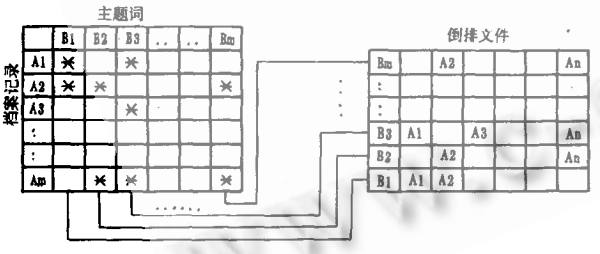


图　1

倒排文件虽然能提高多键检索的速度，但由于其增加了文件的辅助空间，当档案文件的数据量很大，次关键字的重复频率增高时，查询速度会受到一定影响，特别是在用户提出的查询表达式较复杂的情况下，查询速度会明显降低。

## 2.主题词稀疏矩阵算法的查询机制

对于各种档案，其内容特征都可以用若干主题词来加以描述。当一个档案数据库拥有数万条记录时，对应的档案也有几万份，而所有的主题词也可能成百上千。如图 1 左面所示，如某份档案包含某个主题词，则在对应位置上标上“＊”号。如档案 A1 用了主题词 B1，B2，Bm 等。由图中可以看出，对于一份档案，仅用了这一行主题词中的某几个。因此，如将图 1 左面看成是一个以主题词为列，档案记录为行的矩阵，则该矩阵显然为一稀疏矩阵，它刻画了档案数据库的特征，同时也指出了利用主题词查询数据库的新途径。

在图 1 所示的档案记录／主题词稀疏矩阵中，有如下两个基本特点：

(1)行数和列数是随着档案份数的增加和所用主题词的增加而动态扩大的。

(2)由于行太多而在内存中动态生成时占有大量空间，从而降低了查询速度。

为了加快检索速度，减少大量不必要空间的浪费，在存储过程中，首先将稀疏矩阵中的非零元素抽取出来，采用压缩存储方法，将其映射成一个关系，即档案记录—主题词关系。某份档案拥有几个主题词，在档案记录—主题词关系中就拥有对应的几个记录。然后分别对档案记录和主题词建立索引文件，从而将档案记录—主题词稀疏矩阵映射成图 2 所示的文件结构形式。其中 SY1 为对主题词进行索引的索引文件，SY2 为对档案记录进行索引的文件，通过间接索引对档案数据进行查询。

参考图 2，当用户通过主题词查询某方面的档案时，查询过程的一般机制如下：

(1)提出查询表达式；

(2)根据查询表达式的逻辑关系对索引文件 SY1 和

SY2 进行运算;

(3)用运算的结果去查询档案数据库。

现举例说明如下(以图 2 为例):

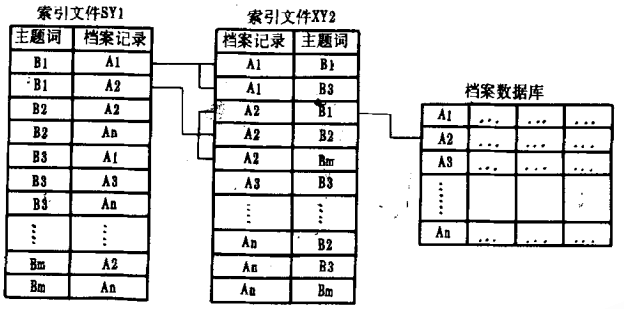


图 2

设用户提出的查询表达式为 B1 and B2 and Bm (and 是与逻辑运算符);

指定 SY1 为主索引文件,检索找到 B1,对应档案记录有 A1;

指定 SY2 为主索引文件,检索找到 A1,对应主题词有 B1,但无 B2,则档案 A1 未被选中;

继续对 SY1 检索,找到 B1,对应档案记录有 A2;

对索引文件 SY2,检索找到 A2,对应主题词有 B2,BM;记录 A2 被选中。

对索引文件 SY1 继续检索,已无主题词 B1,检索结束,指针指向档案数据库中记录 A2。

从上面的例子及算法中可以得出结论:在档案数据库中要查询满足一定逻辑组合条件的档案记录时,查询速度依赖于索引文件 SY1 中相同主题词的个数和索引文件 SY2 中各对应档案记录的个数。

**3.对稀疏矩阵检索与倒排文件检索速度的比较**

在以上检索过程中,由于是在两个索引文件上进行,其速度主要取决于索引文件 SY1 中对应的相同主题词个数和文件 SY2 中对应的相同档案记录的最多个数,同时也与主题词及档案记录在 SY1 及 SY2 中的分布规律有关。对于 SY2 中对应的相同档案记录,通常情况仅有几个,最多也不过十几个(因为每一档案的主题词一般为几个,很少有十几个以上的),因此,影响检索速度的主要原因是索引文件 SY1 中重复出现的主题词的个数。由此可见,在查询的逻辑表达式中,主题词的先后位置与查询速度有关,通常应尽量将重复率较低的主题词放在第一个位置上。但就其检索的整个过程来说,与文件 SY1 和 SY2 本身的长度无关,所以应用该方法可显著提高查询的速度。作为算法的两个极端,设:

L1=索引文件 SY1 中相同主题词"*"的个数;

L2=索引文件 SY2 中对应的相同档案记录的个数;

M1=比较次数;

则最坏情况为:M1=max(L1)*max(L2);最好情况为:M1=1

同样,用以上档案记录—主题词关系按主题词排序,形成倒排文件,用相同的查询表达式:B1 and B2 and Bm 进行检索。

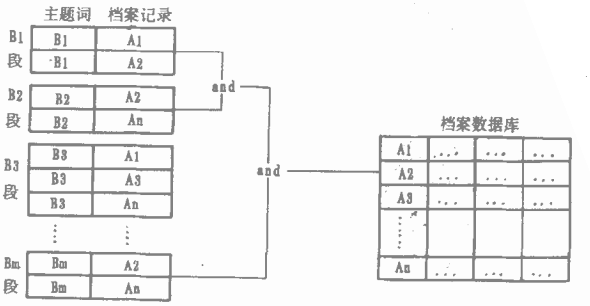


图 3

如图 3 所示,将倒排文件中包含主题词 B1 的记录称为 B1 段,包含主题词 B2 的记录称为 B2 段,……包含主题词 Bm 的称为 Bm 段。

查询过程如下:

(1)从 B1 的第一个记录查起;

(2)B1 段记录结束否?如结束转(5),否则转(3);

(3)将 B1 段的档案记录号与 B2 段逐一比较,若 B1 段找到相同记录号则转(4),否则下移一个记录后转(2);

(4)逐步检索 Bm 段的档案记录号,若无 B1 和 B2 段当前所指的记录号,则 B1 段的记录指针下移一个记录后转,否则选中所需查询的记录号,对该记录作处理后 B1 段记录指针向下移一步,然后转(2);

(5)检索结束。

上述算法与倒排文件的长度无关,而与倒排文件中档案的分布规律及 B1,B2,Bm 段的长度有关,因此也可提高检索速度。作为该算法的两个极端如下:

(下转第 33 页)

(上接第 29 页)

设：L1＝B1 段的长度　　L2＝B2 段的长度

L3＝Bm 段得长度　　M2＝比较次数

则最坏情况为：M2＝max (L1) * max (L2) * max (L3)；最好情况为：M2＝1

从以上分析可看出，倒排文件检索比稀疏矩阵法在算法上要复杂一些，比较次数也多一些，如果查询的逻辑表达式更复杂的话，其比较次数还要更高。而稀疏矩阵法即使在查询逻辑表达式更复杂的情况下也很少增加比较次数。因此，尽管两种算法都与本身数据库的大小无关，但对于满足稀疏矩阵条件的数据库，利用本文提出的算法可获得比倒排文件算法更快的查询速度，在查询组合条件复杂的情况下，更能体现该算法的优越性。