第 36 卷 第 4 期 计 算 机 工 程 2010 年 2 月
Vol.36 No.4 Computer Engineering February 2010
·软件技术与数据库· 文章编号：1000—3428(2010)04—0036—03 文献标识码：A 中图分类号：TP301.6

基于集合覆盖的分布式信息检索资源选择

王秀红
(江苏大学科技信息研究所，镇江 212013)

摘 要：考虑到不同的数据资源(数据集)之间存在的覆盖问题，基于集合覆盖理论，针对提问 Q 的检索结果在融合排序后位置的不同，对其赋以不同的权值，用来计算该项检索结果对其所在的数据集的贡献。若检索结果在先选的数据集中出现过，则不再计入后选的数据集得分内。通过加权求和得到待选数据集的得分，从而确定资源选择的先后顺序。由此优选出的资源集合可用于检索与问题 Q 同类或类似的提问 Q'，缩短由于数据库之间的覆盖而重复检索的时间。

关键词：分布式信息检索；集合选择；资源选择；集合覆盖

Resource Selection in Distributed Information Retrieval Based on Set-covering

WANG Xiu-hong
(Institute of Science and Technology Information, Jiangsu University, Zhenjiang 212013)

【Abstract】 Considering overlapping extent between resources, a set-covering-based algorithm for resource selection in Distributed Information Retrieval(DIR) is proposed. Different document with different weight according to its position in merged results for question Q is given. Only results that have not appeared in some earlier selected resource are focused on in later selected resources. The score of each resource is decided by the total weights of those merged results included in, and only the resource with max score is selected in each selecting step. The selecting order is the actual rank of selected resources which are used to answer the question Q', which is similar to question Q. The approach makes time cost decreased in DIR.

【Key words】 Distributed Information Retrieval(DIR); collection selection; resource selection; set-covering

1 概述

分布式信息检索研究的主要内容包括资源选择、单数据集合检索、结果合并等几个部分。其中，资源选择又叫集合选择、数据集选择或数据库选择。利用集合选择算法找出最相关的数据集集合进行检索，从而实现通过查询部分资源集合而给出很好的检索结果的效果。集合选择的效果直接决定着最终检索结果的质量。

在集合选择方面，相关的代表性算法主要有以下几种：

(1)CORI(Collection Retrieval Inference Network)算法[1]。由原有的对文档进行相关性判断的贝叶斯推理网而来，本质上是把每个数据集表示成一个虚拟文献(看成是一篇巨大的文献)，此虚拟文献由一系列提问词在潜在数据集中的文献频率组成；虚拟文献由 IN-QUERY 检索系统标引；提问 Q 被应用于数据集选择索引来排序虚拟文献，虚拟文献的排序结果就是 CORI 对数据集的排序结果，与传统信息检索系统中的文献等级排列的方法相似。

(2)gGlOSS(generalized Glossary of Servers Server)算法[2]。根据数据集对输入提问式的友好性对数据集进行等级排列，这样做可以估计每个数据集中含有超过某一阈值的文献数量，然后根据文献数量确定数据集的得分。

(3)CVV(Cue-Validity-Variance)算法[3]。根据 Internet 的查询特点，在向量空间算法的基础上对算法做了改进。

(4)利用人工索引的方法提高资源选择和检索效果的算法。文献[4]比较了不同的集合选择方法之间的表现差异。文献[5]将集合选择问题转化为文档检索问题，尝试了多种文档检索方法来解决集合选择问题。然而，以往的研究缺点在于没有充分考虑数据集之间的覆盖，从而不可避免地使所选择的数据集合之间可能存在很大重叠而被忽略，增加了重复检索的时间，大大降低了检索的效率和效果。

本文提出了一种基于集合覆盖的分布式信息检索资源选择方法，即计算待选数据集对需要检索的数据的覆盖程度。根据覆盖程度的大小，考虑到数据库之间本身存在部分数据重叠的现象，确定选择数据库集合的先后选择顺序，从而优化数据集合选择，在保证查全率的同时，节约重复检索的时间开销，提高检索效率。

2 算法的描述

通过给包含于待选数据集中的检索数据加权求和的方法，计算待选数据集的重要性分值。假设有一个提问 Q，将对该问题的检索结果融合排序后取前 n 个结果(n 为自然数)，依次分别记为 $d1,d2,\cdots,dk,\cdots,dn$；若 dk 在某个数据集 C_i 中出现，则对该数据集贡献分值记为 $1/k^{\beta}$，β 为正有理数；数据集的得分为其所包含的“特定”的数据的贡献分值之和。首先，对数据集 C_i 据其所包含的 $dk(k=1,2,\cdots,n)$的贡献分值求和后选择分值最大的数据集作为首选的数据集并记为 C'_1；接着，对余下的数据集 C_j，根据其所包含的未出现在已选数据集中

基金项目：江苏大学博士生创新基金资助项目(CX08B_18x)
作者简介：王秀红(1975－)，女，博士研究生，主研方向：信息检索，信息分析
收稿日期：2009-09-30 **E-mail：**lib510@ujs.edu.cn

的 $dk(k=1,2,\cdots,n)$的贡献分值求和后选择分值最大的数据集作为次选的数据集并记为 C'_2；依此方法，直到第 m 次选择 C'_m，当这 1~m 个被选择的数据集已覆盖所有的 $dk(k=1,2,\cdots,n)$时，结束数据集选择步骤。

举例如下：假设 n=10，即取融合排序后的结果中位于前 10 位的数据。且每个数据用自己的排序号标识，即这 10 个数据根据排序编号命名为 1,2,3,4,5,6,7,8,9,10。假设有以下各数据集：C_1 中包含了数据 1,2,3,4；C_2 中包含了数据 2,3,7,8；C_3 中包含了 1,5,6,7 文档；C_4 中包含了 4,5,6,9 文档；C_5 中包含了 9,10 文档。其他未包含所要求的检索结果的数据集不作选择对象考虑。第 k 个数据对其所在数据集的贡献分值为 $1/k$(此处假设 β=1)。

选择数据集的过程描述如下：

(1)第 1 个被选上的资源。对数据集 $C_i(i=1,2,\cdots,5)$据其所包含的数据贡献分值求和后选择得分最大的数据集，SC_1'= max{1+1/2+1/3+1/4, 1/2+1/3+1/7+1/8, 1+1/5+1/6+1/7, 1/4+1/5+1/6+1/9, 1/9+1/10}={1+1/2+1/3+1/4}。于是最重要的一个数据集也就是第 1 个被选择的数据集为包含 1,2,3,4 文档的那个数据集，记为 $C_1'=C_1$={1,2,3,4}。

(2)第 2 个被选上的资源。除了已被选出的 C_1 外，在剩下的 4 个数据集中，分别去掉第 1 次被选择出的数据集中已出现的数据后再将贡献分值求和找出最大值 SC_2'=max{1/2+1/3+1/7+1/8,1+1/5+1/6+1/7,1/4+1/5+1/6+1/9,1/9+1/10}={1/5+1/6+1/7}，表示第 2 次选出的数据集是包含 1,5,6,7 文档的数据库，记为 $C_2'=C_3$={1,5,6,7}。

(3)第 3 个被选上的资源。除了已被选出的 C_1 和 C_3 外，剩下的 3 个数据集中分别去掉第 1 次和第 2 次被选出的数据集中已出现的数据后，再分别进行打分找出得分最高的数据集。SC_3'=max{1/2+1/3+1/7+1/8, 1/4+1/5+1/6+1/9, 1/9+1/10}={1/9+1/10}。表示第 3 次选出的数据集是包含 9,10 文档的数据库，记为 $C_3'=C_5$={9,10}。

(4)第 4 个被选上的资源。除了已被选出的 C_1，C_3 和 C_5 外，对于剩下的 2 个数据集中，分别去掉第 1 次、第 2 次和第 3 次被选择出的数据集中已出现的文档后，再分别进行打分，找出得分最高的数据集。SC_4'=max{1/2+1/3+1/7+1/8, 1/4+1/5+1/6+1/9}={1/2+1/3+1/7+1/8}，于是第 4 个被选择的数据集为包含数据 2,3,7,8 的数据集，记为 $C_4'=C_2$={2,3,7,8}。

(5)第 5 个被选上的数据集。除了已被选出的 C_1,C_2,C_3 和 C_5 外，对于剩下的 1 个数据集中，分别去掉第 1 次、第 2 次、第 3 次和第 4 次被选择出的数据集中已出现的数据后，再打分并找出得分最高的数据集。SC_5'=max{1/4+1/5+1/6+1/9}=0，说明剩下的数据集选择已经没意义，集合选择到此结束。于是被选择出来的数据集共为 4 个，m=4。整个算法过程如图 1 所示。

假设对于一个检索提问可供检索的数据资源集合，有包含提问答案的 M 个数据集的集合，$C=(C_1,C_2,\cdots,C_M)$，C_i 为第 i 个可供选择的数据集，$i=1,2,\cdots,M$；n 为经过合并并经过相关度大小排序后的检索结果中前 n 个数据。C'为根据相关度大小，所有的被选择上的 m 个数据集的集合 $C'=(C_1',C_2',\cdots,C_m')$，$C_j'$为第 j 个被选择上的数据集，$j=1,2,\cdots,m$，SC_j'为第 j 个被选择上的数据集的得分，$j=1,2,\cdots,m$，β 为权函数中的常参数，为正有理数；第 k 个数据出现在数据集 C_i 中，在形式上标记为 kC_i，其贡献分值形式上记为 $1/(kC_i)^\beta$，大小实为 $1/k^\beta$。第 k 个数据出现在数据库 C'_j 中，在形式上标记为 kC'_j，其贡献分值形式上记为 $1/(kC'_j)^\beta$，大小实为 $1/k^\beta$。$KC_i-\sum_{j=1}^{l-1}Kc'_j$ 表示只有第 k 个数据未出现在已选择的前 l-1 个数据集中时，计算其后的数据集得分时才能将其算入，其贡献分值形式上记为 $1/(KC_i-\sum_{j=1}^{l-1}Kc'_j)^\beta$，大小实为 $1/k^\beta$。从而数据集的得分计算公式为

$$SC'_1=\max_{i=1}^{M}\sum_{k=1}^{n}\frac{1}{(kC_i)^\beta} \tag{1}$$

$$SC'_l=\max_{i=1}^{M}\sum_{k=1}^{n}\frac{1}{(kC_i-\sum_{j=1}^{l-1}kC'_j)^\beta},\ 2\le l\le m \tag{2}$$

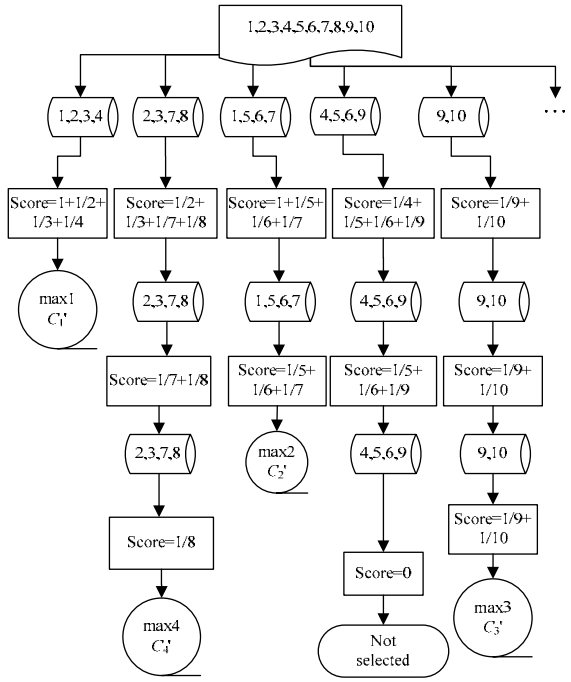

图 1　算法过程例图

整个数据集集合选择过程如图 2 所示。具体参数详见表 1。

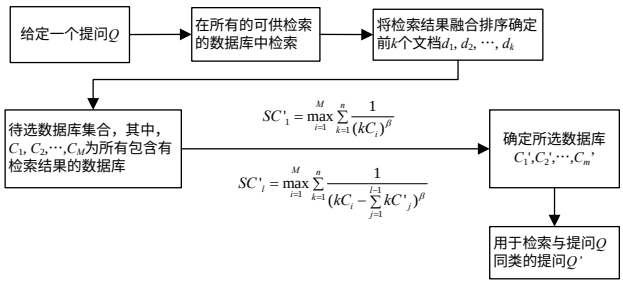

图 2　集合选择过程

表 1　表示符号及其意义

符号	含义
M	包含提问答案的所有可供选择的数据集个数
C_i	第 i 个可供选择的数据集，$i=1,2,\cdots,M$
C	对于一个检索提问可供检索的数据集集合 $C=(C_1,C_2,\cdots,C_M)$
m	最终被选择的数据集个数
C_j'	第 j 个被选择上的数据集，$j=1,2,\cdots,m$
SC_j'	第 j 个被选择上的数据集的得分，$j=1,2,\cdots,m$
C'	所有的被选择上的数据集集合，$C'=(C_1',C_2',\cdots,C_m')$
n	经过融合排序后的检索结果中，前 n 个数据(可以表现为文档)
kC_i	检索结果中排在第 k 个位置的数据出现在第 i 个可供选择的数据集 C_i 中
kC_j'	检索结果中排在第 k 个位置的数据出现在第 j 个被选择上的集合 C_j' 中
β	β 为权函数中的常参数，为正有理数，本文实例中取 β=1

3 算法的实现

将对问题 Q 的检索结果经过融合后取前 n 个文档按行矩阵排列，则矩阵的每个元素就代表一个文档(检索结果)，实验中用不同的颜色代表不同的数据集。如果给定前 n 个文档，就能从 M 个可供选择的数据集($C_1,C_2,\cdots,C_M$)中找到最优的 m 个数据集($C_1',C_2',\cdots,C_m'$)，这 m 个被选择的数据集之间的彼此覆盖程度达到最小，节省了因数据库之间的覆盖而重复检索的时间，且能包括所有的检索结果，保证了查全率。本算法的时间复杂度为 $O(m\times n)$，由于 m 为被选择的数据集的总数，其数值不会太大，因此该算法的时间复杂度相当于 $O(n)$。实验中，对于提问 Q 的检索结果融合排序后选择前 100 个文档(检索结果)，按对问题的相关程度从大到小标记为 1,2,⋯,100，取参数 β=1，则各文档对应的重要性分值分别为 1,1/2,⋯,1/100，排序为第 k 位的文档出现在某个数据中，则该文档对该数据集的贡献分值为 1/k。实验中假设有 60 个数据资源可供选择用于检索提问，即取 M=60。实验结果如图 3 所示。

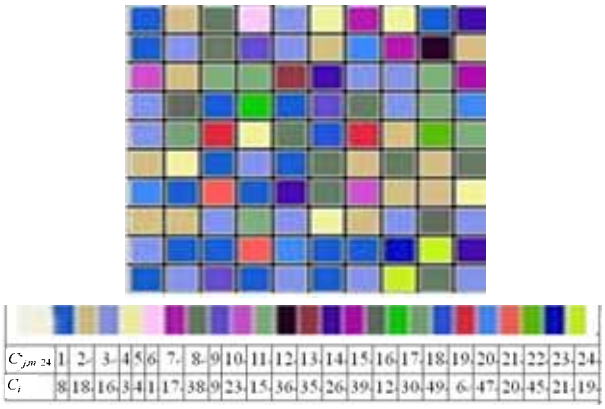

图 3　基于集合覆盖的资源选择结果

4 实验分析

从这 60 个数据集中选择出 24 个优化组合的数据集集合，可供查全这 100 个文档，在保证查全率的同时，被选择的数据集之间彼此覆盖达到最小。实验结果再现了被选中的 24 个数据集相互补充覆盖这 100 个文档的具体情况。实验结果显示，第 1 个被选出的数据集也是最重要的一个数据集，对应于图 3 中颜色条第 1 位 j=1 对应的颜色(蓝色)，覆盖了第 1、第 9、第 11、第 33、第 46、第 53、第 55、第 62、第 64、第 82、第 83、第 86、第 87、第 91、第 94 和第 96 个检索结果。又如第 11 个被选择的数据库(j=11，黑色)，对提问的重要贡献在于它包含了检索结果 19。实验显示在考虑数据集之间的重叠情况后，通过本算法在保证查全率的前题下，检索时间仅为原先的 40%，大大提高了检索效率。

5 结束语

针对分布式信息检索中资源选择方法的不同会影响分布式信息检索的查全率和检索效率问题，本文提出了一种基于集合覆盖的资源选择方法。考虑数据集之间存在的重叠情况，运用贪心算法通过给数据集中包含检索结果的重要程度不同而加权求和给数据集打分，从而确定数据资源选择的先后顺序。该方法不仅保证了查全率，同时大大节约了重复检索的时间，提高了检索效率，可适用于分布式异构数据资源之间跨平台的信息检索。

参考文献

[1] Callan J P, Lu Zhihong, Croft W B. Searching Distributed Collections with Inference Networks[C]//Proc. of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Seattle, Washington, USA: [s. n.], 1995: 21-28.

[2] Gravano L, Garcia-Molina H, Tomasic A. GIOSS: Text-source Discovery over the Internet[J]. ACM Transactions on Database Systems, 1999, 24(2): 229-264.

[3] Yuwono B, Lee D L. Server Ranking for Distributed Text Retrieval Systems on Internet[C]//Proc. of the 5th International Conference on Database System for Advanced Applications. Melbourne, Australia: [s. n.], 1997: 41-49.

[4] Powell A L, French J C. Comparing the Performance of Collection Selection Algorithms[J]. ACM Transactions on Information Systems, 2003, 21(4): 412-456.

[5] 张　刚, 郭　岩, 张　凯. 分布式信息检索的集合选择研究[J]. 计算机工程, 2007, 33(2): 158-166.

编辑　顾逸斐

(上接第 35 页)

5 结束语

本文通过基于信息熵差的灰色关联方法，不仅可以定量分析网络存储系统的可生存性，描述系统的可生存态势，而且可以为不同系统之间或系统的不同时刻的可生存性比较提供量化值。当前网络存储系统可生存性研究工作尚处于起步阶段，其中的许多关键问题，如可生存性态势的实时预测和关键服务间的交互关联对生存性的影响等还需要进一步研究，尤其对具体的存储系统需要详细分析，建立比较完善的设计方案。

参考文献

[1] 王　超, 马建峰, 朱建明. 网络系统的可生存性研究综述[J]. 网络安全技术与应用, 2006, (6): 15-17.

[2] 杨　超, 马建峰. 可生存网络系统的形式化定义[J]. 电子科技, 2004, (4): 1-4.

[3] 邓聚龙. 灰色系统理论教程[M]. 武汉: 华中理工大学出版社, 1990.

[4] 张义荣, 鲜　明, 赵志超. 计算机网络攻击效果评估技术研究[J]. 国防科技大学学报, 2002, 24(5): 24-28.

[5] 王清印, 崔援民, 赵秀恒. 预测与决策的不确定型数学模型[M]. 北京: 冶金工业出版社, 2001.

[6] 赵国生, 王慧强, 王　健. 基于灰色关联分析的网络可生存性态势评估研究[J]. 小型微型计算机系统, 2006, 27(10): 1861-1864.

编辑　顾逸斐