模糊查找算法的设计实现

郑志宏[1],郑志高[1],王玉婷[2]

(1.吉林化工学院 理学院;吉林 吉林 132022;
2.中南财经政法大学 工商管理学院;湖北 武汉 430000)

摘 要:根据 KMP 算法的设计理念和基本模式,提出了分词、模糊相似度的概念和一个模糊匹配的函数 Fuzzysearch()。根据该算法按照模糊相似度的匹配实现分级别的模糊查找。

关键词:模糊查找;模糊相似度;分词;KMP 算法
中图分类号:TP317.4 **文献标识码**:A **文章编号**:1007-7634(2010)06-0915-04

Design and Implementation of Fuzzy Search Algorithm

ZHENG Zhi-hong[1], *ZHENG Zhi-gao*[1], *WANG Yu-ting*[2]

(1. *School of Science,Jilin Institute of Chemical Technology* ,*Jilin* 132022, *China*;2.*School of Business Administration,Zhongnan University of Economics and Law*, *Wuhan* 430000, *China)*

Abstract: According to the design concepts and the basic model of the KMP algorithm, this paper presented the concept of the sub-word method and the fuzzy similarity, then, we designed a fuzzy matching function named Fuzzysearch (). According to the fuzzy similarity and the matching algorithm we designed the function which achieved the sub-level of fuzzy searching.

Keywords:fuzzy search ;fuzzy similarity;sub-word; KMP algorithm

在情报系统中对信息的检索通常是利用把信息转换成字符串的方式来进行,但是在实际应用中往往对需要获得的信息的具体查询条件不够明确,因而模糊查找就显得特别重要。现有的模糊查找基本上仅是对字符串进行截断匹配,但实际情报检索中对模糊查找的信息并不局限于截断匹配。例如,输入查找信息时,经常会受汉语同音字的影响,误将"情报科学",输入为"情抱科学"。由此可见,模糊查找实现的意义极为重大,并且在这个领域里的研究基本上还没开始。本文创造性地设计一种分词算法并基于 KMP 算法改进出几个函数实现一种按模糊相似度进行模糊查找的算法,其主要原理是通过用户设定模糊比较级别来对字符串的模糊匹配进行数据筛选,从而获得用户所需信息。

1 设计实现的原理及相关定义的提出

1.1 相关概念

(1)模糊相似度:本文将用户查询关键字与数据库相应字段按照字符进行比较,相同字符所占比例称为查询字段的模糊相似度,实际应用时本文将模糊相似度扩大 10 倍后取整(这样做的目的是方便用户接口处精度的设置)。

(2)分词:将用户输入的一个完整句子按照词典有实体意义的词进行拆分的过程称之为分词,实现这一过程所用的方法称之为分词算法。

(3)模糊比较级别:把用户查询过程中对查询精

收稿日期:2009-11-30

作者简介:郑志宏(1959-),女,吉林省吉林市人,副教授,主要从事应用数学方面研究;郑志高(1988-),男,湖北麻城人,主要从事计算机软件与网络、人工智能方面研究;王玉婷(1989-),女,湖北麻城人,主要从事政府经济学、公共行政、新主义制度方向.

度的要求按照上面定义的模糊相似度进行区分，并留有系统设置接口，用户以系统管理员的身份通过设定的模糊相似度影响查询精度这一过程称之为模糊比较级别的设定。

1.2 设计实现的原理

首先，针对著名的 KMP 算法只能实现两个字符串的精确匹配[1]，当需要查找的信息与数据库中的信息并不完全相同时，无法实现查找，并且不能记录其相似度的弊端，借助 KMP 算法的设计理念重新定义数据结构并设计查找算法以实现其模糊查找功能，弥补 KMP 算法的不足满足一些实际需要。

然后，根据数据结构和数学概念定义模糊查找的原则和匹配依据，查找的原则是满足用户查找需求（非严格精确匹配例如我们在搜索 A 时会出现 a，ab，Ab 等模糊匹配结果），匹配依据是按照模糊相似度进行分步查询[2]。首先改进原来的利用单个字符（中文状态下是单个字）查找的模式[3]，在改进程序中引进数据库分词程序对用户查询关键字按照有实体意义的词进行分词，按照分词的结果进行查询，例如在查询“北京欢迎你”时，将查询关键字“北京欢迎你”按照“北京”、“欢迎”、“你”这三个关键字进行 6 次比较匹配即可完成查询，这样该字段的查询次数将由原来的 15 次（按照单个汉字 5 次，两个汉字顺序组合 4 次，三个汉字顺序组合 3 次，四个汉字顺序组合 2 次，五个汉字顺序组合 1 次）比较匹配减少为 6 次，查询效率约提高了 3 倍。但是在这种查询状态下必须由系统预先设定模糊相似度[4]，在查找过程中按照模糊相似度依次递减搜索，搜索结果按照匹配精度递减依次输出。本文中提出的模糊相似度是按照正常匹配得到的模糊相似度扩大 10 倍或 100 倍后取整所得数值，算法中是按扩大 10 倍计算的。

2 算法的整体设计

算法的整体设计过程如下：

（1）将用户输入的关键字进行分词处理；

（2）原串（用户输入的关键字）与数据库中的串（数据库中有实体意义的字段）进行模糊比较；

（3）根据步骤（2）的比较结果得出模糊相似度；

（4）将模糊匹配有关联的串入队列；

（5）判断模糊相似度，如果模糊相似度符合查找要求则执行步骤（4）；如果模糊相似度达不到要求则将要求降低一级转向执行（2），相似度值加 1；

（6）相似度值达到限制最大值如有匹配项则按照队列的元素输出否则提示模糊匹配不成功，转向步骤（7）；

（7）退出程序。

3 数据结构的设计

首先在用户输入查询关键字时先把查询关键字转入系统预先设置的词典程序[5-6]，按照词典规则对查询关键字进行拆分（本文将此过程定义为分词）。例如：在查询“情报科学”时就可按照汉语词典把查询关键拆分为“情报”、“科学”两个关键词进行查找，按照查询结果（查询关键字与数据比较得到的相似度）排序，再在查询结果里面进行匹配，这样将大大减少了查询过程中与数据库字段的比较次数，当然这样也额外增加了拆分的运行时间（在词典数据库中搜索关键字）和拆分方法的选择过程，这就需要我们在查询效率和查询速度上做出折衷。考虑到按照分词法查询效率将成倍提高，同时现代大型计算机运算速度之快，选择拆分算法并将查询关键字进行拆分的整个运行时间将在秒微级以下，对比这两者对查询的整体影响，这样的折衷还是有利于查询的实现的，并且能够减少系统总的时间开销。

在拆分过程中参考北京大学计算机语言研究所[ICL]开发的分词软件的工作模式结合搜索引擎工作原理[7]设计出中文分词算法的数据结构如下：

```
@param   int  char //参数的汉字说明，用于接收查询界面 get 函数传递的参数（即搜索的关键字）
int main (int argc, char* argv[])
{
    程序开始执行（当前查询和结果表号）iQuery.GetInputs();
    开始具体搜索程序 iQuery.SetStart();
    调用 gettimeofday 函数开始计时获取程序运行时间差
  利用 iHzSeg.SegmentSentenceMM 函数将查询变量分词分成多个字段的形式
    将以"/" 划分开的关键字一一顺序放入一个向量容器 iQuery.ParseQuery 中；
    调用 iQuery.GetRelevantRst 检查参数是否结束；
    调用 gettimeofday 结束搜索。
    显示查询结果 CDisplayRst iDisplayRst
   return 0;
```

}

数据结构的相关说明：

⑴GetInputs：这个方法的功能是将用户通过前台查询界面输入的查询关键字通过 get 函数传递过来的变量转换到搜索引擎的内部函数 HtmlInputs 结构体数组中；

⑵iQuery.ParseQuery：查询数据结构的关键字段标号，并将关键字入队列。

在完成了中文分词之后，我们只需在查询数据库中加入英文词典，并使用同样的程序即可对英文关键字按照有实体意义的词为单位进行分词处理。

在对查询关键字做完分词处理之后我们需要定义一个结构体用来存储匹配的结果和过程值，结构体名 indextype。该记录包括两项：整型变量 fmdegree 和布尔型变量 find，其中 fmdegree 用来存储 KMP 算法过程中的相似度，find 用来区别是否传统匹配成功，我们对 find 赋值 TRUE 或 FALSE，其中 TRUE 表示匹配成功，FALSE 表示匹配失败。如下：

```
struct indextype
{
    int fmdegree;
 Boolean find;
}m;
```

在定义好了算法的数据结构以后，必须设计查找方法才能实现比较广义的模糊匹配。用一个函数实现 KMP 算法，在函数头定义了三个参数 S,T,M，其中 S,T 是字符串类型，M 用来判断是否完全匹配和匹配相似度。在算法的过程中需要设定两个指针 i 和 j 分别指示主串（数据库中的关键字字段）和模式（对用户关键字通过分词技术处理得到的分词）中正待比较的字符，按照 KMP 算法的模式对指针进行移位操作，直至比较到主串结尾或者因匹配成功而结束。并且这样比较的结果是可以通过当前指针的值作出判断的，如果没有匹配到，将通过当前关键字的模糊相似度来记录匹配的最大相似度。

数据结构设计如下：

```
引入 indextype 结构体；
indexf (string s,string t, indextype m)
{
    变量初始化；
    获取主串和模式的字符长度并做记录；
    while(分词长度小于数据库关键词字段)
    {
        利用匹配程序进行匹配；
    } //end while
    如果匹配到的字符个数大于长度较短字符串的个数则返回
        否则返回假
} //end indexf 进行循环比较
```

在本算法中主要实现对已经分词的查询关键字进行模糊匹配，当查询接口接收到分词程序传递过来的拆分关键字时就进行相关的匹配搜索，搜索过程的实现可以借助多种方法实现，比如 KMP 串的滑动匹配方式进行，也可以通过快速匹配模式实现。

4　模糊查找的实现

上面基于 KMP 算法的设计理念设计了一个函数实现了匹配过程和匹配相似度的记录，下面设计一个过程 Fuzzysearch 调用该函数以实现模糊查找，其中数据库关键词字段和分词是需要判断的两个字符串；对于汉字和字符的比较不同只需根据预先设定的词典进行分词即可，在此不作区分；fmdegree 用来标识模糊的精确度，该函数的原理是：首先调用分词程序对查询关键字按照有实体意义的词为单位进行分词，在分词成功后调用上面定义的 indexf()函数对两个字符串进行匹配计算（关键字的查询），得出是否匹配及匹配的相似度，如果匹配（查询成功）则返回 Ture，否则提示系统重新设定更低级别（模糊相似度增加）继续查询，以此类推，直到模糊相似度增加到系统能够忍耐的最大值[8]。最后通过回送显示程序显示已经查得的结果[9]。

过程设计如下：

```
引入 indextype 结构体；
    函数头 Fuzzysearch(数据库关键字字段，分词，模糊相似度)
    {
        调用分词程序对用户关键字进行分词并计算字段长度；
        判断是否为词典中有实体意义的最小单位的词，是则继续，否则返回分词；
            if 调用 indexf 函数的返回值为真则说明首次查询成功，直接返回真，结束程序；
                else 相似度为 1，但是 indexf 返回的 find 为假则返回假
                    {
        按照相似度循环，直到得到和理解过，结束程序；
```

调用回送程序,显示查询结果;

}

}//end

本次设计中模糊相似度可以有多种级别,在实际实现时可以根据对匹配精度和查询度的要求以及服务器数据库的容量和配置等因素设定模糊相似度的值,使查询结果达到最合理的状态。

例如在通过本算法查询"北京欢迎你"时,系统首先调用分词程序将输入的短句 "北京欢迎你",按照有实体意义的词分解为 "北京"、"欢迎"、"你",然后分别以这三个词作为独立的原子实体查询条件进行查询。在第一次循环时先以"北京"为一个原子实体进行查询,并显示出所有符合条件的查询结果。程序转入第二次循环分别以"北京欢迎"和"欢迎北京"作为一个原子实体进行进一步的匹配查询,并通过回送终端实时显示查询结果,在本次查询结束,程序转入第三次循环,此时程序分别以 "北京欢迎你"、"你欢迎北京"、"欢迎你北京" 作为一个原子实体进行精度更高的模糊匹配,查询结果通过回送终端实时显示"北京欢迎你"的查询结果。通过该方法进行模糊查询可以极大地缩短查询过程中的时间开销、方便地调整查询精度(设置模糊度)从而满足不同状态下的查询需要。并且还可以根据用户需要在此程序中添加相应的控制模块对查询结果做进一步的控制,进一步提高查询精度,得到理想的查询结果。

5 结 语

本文提出了分词、模糊相似度的概念和一个模糊匹配的函数,在算法设计时通过对用户输入的查询关键字进行分词实现了分级别的模糊查找,在此基础上从理论上说明了不完全相同的字符可以进行模糊匹配,在模糊查询中系统首先按照有实体意义的词为单位对查询关键字进行分词,再根据拆分关键字以及对查询精度的要求设置模糊相似度来进行查找。该方法的设计,可以极大地缩短查询过程中的时间开销、方便地调整查询精度(设置模糊度)从而满足不同状态下的查询需要。并且还可以根据用户需要在此程序中添加相应的控制模块对查询结果做进一步的控制,例如添加自动纠错模块可按照语义和语法习惯对程序中的错误进行自动纠正,进一步提高查询精度,得到理想的查询结果。

参考文献

1 鲁宏伟,魏 凯,孔华锋.一种改进 KMP 高效模式匹配算法[J].华中科技大学学报,2006,(10):41-43.

2 Navarro G, Fredriksson K. Average complexity of exact and approximate multiple string matching [J]. Theoretical Computer Science,2004,321 (2-3):283-290.

3 Shinaya OBARA.Energy Cost Of an Independent Micro-grid with Control of Power Output Sharing of a Distributed Engine Generator[J].Ichinoseki National College of Technology,2007,(2):66-78.

4 Hiroaki Kikuchi, Kazuyoshi Matsuoka.Nekogole -Interactive Web Search Engine Using Data -Mining Technology [J]. Tokai University,2006,(4):1396-1401.

5 吕学强,苏 祺,孙 斌,等.搜索引擎用短语词典建设[J].清华大学学报,2005,(5):1892-1895.

6 AndrewS.TAnunBAum. COMPUTER NETWORKS [M]. Beijing:Tsinghua University Press,2005:136-165.

7 刘务华,罗铁坚,王文杰.一个 Web 社区搜索引擎系统[J].中国科学院,2007,(9):275-278.

8 Lotfi A. Zadeh. From Search Engines to Question-answering Systems-The Role of Fuzzy Logic[J]. Progress in Informatics, 2005,(1):1-3.

9 Yasuhiro Sudo, Masahito Kurihara. Combining Systematic and Local Search for Approximately Solving Fuzzy Constraint Satisfaction Problems [J]. Transactions of the Japanese Society for Artificial Intelligence,2006,(1):20-21.

(责任编辑:徐 波)

(上接第 902 页)

Communications of the ACM, 2005, 48(6): 82 - 87.

12 高 丹. B2C 电子商务顾客满意度的评价指标浅析[J]. 中国电子商务, 2004, (6):43-45.

13 KIM H R. Developing an index of online customer satisfaction[J]. Journal of Financial Services Marketing, 2005, 10(1): 49 - 64.

14 SteinerMarketing (2003) Customer Satisfaction Research to Increase Repeat Business [EB/OL]. http://www.steinermarketing.com/customer_satisfaction.htm,2004-12-08.

(责任编辑:徐 波)