基于中文机构名简称的检索方法研究

钟良伍

摘　要

机构名简称的使用是很普遍的，但是在计算机不知道简称和全称对应关系的时候，如何通过简称检索到机构就是一个必须解决的问题。

对于是否是中文机构名或机构名简称的自动判别，已经有广泛和深入的研究；但是对机构名简称和全称的匹配，目前鲜有研究成果。本文针对基于中文机构名简称的检索方法，做了深入的研究，设计出一个中文机构名简称的检索系统。本文的主要工作包括：

1. 研究机构名的结构特征，总结出机构名的组成结构，并形式化表示这些结构；
2. 研究如何描述人为简称的习惯，发现简称和全称的结构有关，基于此形式化描述人们的简称习惯，以便方便地生成一个机构名所有可能的简称；
3. 为了实现本文的检索系统，在对机构名结构的研究基础上，针对机构名的切分，定制了一个基于关键词类的分词工具；
4. 在研究字符串匹配算法和字段匹配算法基础上，提出一种简称和全称匹配算法；
5. 针对机构名的检索设计出多层索引结构，并使用扩展的 Boolean 模型，实现了多级索引技术，以期在不降低检索准确性的基础上提高检索效率。

在上述方法的基础上，设计并实现一个中文机构名简称的检索系统。实验结果表明，本文所提方法的准确性较好，首选准确率达到近 95%，在全称机构名总数达到 51 万的情况下，检索平均耗时约 0.21 秒，达到实用要求。

关键词：中文分词　　模糊匹配　　字段匹配　　多级索引

Abstract

Abbreviated organization names are widely used in daily life, but when the computer does not know the relation between an abbreviated name and a full name, how to retrieve the corresponding organization name from an abbreviated name becomes a problem that must be solved.

More research has been done on automatic recognition of organization names or abbreviated organization names; however there has been almost none on the matching between a full name and an abbreviated name. In this paper, study on approaches to abbreviation based Chinese organization name retrieval is carried out, and accordingly a retrieval system is implemented. Our work includes:

1. By studying the structure of organization name, the rules of organization name structure is proposed and formally described;

2. By studying the habits of organization name abbreviating, the relation between habits and organization name structures is found, and then is formally described so that all possible abbreviations can be generated from an organization name conveniently;

3. Based on the research of the structure of organization name, a keyword-based segmentation system is implemented to be integrated into the retrieve system;

4. Based on the research of fuzzy string matching and field matching, a novel algorithm of matching a full name with an abbreviated name is proposed;

5. A multi-level indexing structure is designed and adopted for Chinese organization name retrieval, and a multi-level indexing is implemented with extended Boolean model to improve the retrieval time efficiency.

Based on the above, a retrieval system is implemented which could achieve an accuracy of nearly 95% and a time efficiency of about 0.21 seconds per query when the total number of organization names is 510,000. The experimental results show that the foresaid system can meet the practical requirements.

Keywords: Chinese word segmentation fuzzy string matching field matching multi-level indexing

目 录

第 1 章　引言

1.1　机构名简称的检索方法研究的意义

1.1.1　机构名及其简称的使用非常普及

机构名主要是机关、团体、社会组织和企事业单位的名称。它的数目庞大，并且随着社会和经济的发展不断扩大并产生新的名词。

机构名覆盖范围较广，它包括很多类。国家机关，民间社团等，如中共中央办公厅，清华大学研究生会财富论坛；各类地点地址，如清华大学八号楼，北京建国门内大街甲一号等；与竞技、竞争、技艺等有联系的团队，如北京国安队，北京奥林匹克数学队；教育科研机构，如清华大学，北京外国语大学、中国科学院；金融机构，如中国银行，中国建设银行，交通银行，美国花旗银行等；一般公司，如联想集团有限公司，中国石油化工有限公司；其他企事业单位，如中国外汇管理局等。

在日常生活中，机构名的使用是很普遍的。国家机关的设置，政权组织结构等都要用到机构名。市场经济的任何一个环节都会和机构名挂钩。各类企事业单位社会团体的日常运作，对外对内宣传等，都需要内外部机构联系。我们的衣食住行等，也都离不开相关机构。机构名的使用已经是一种必要。

在机构名被人们广泛使用的同时，我们也意识到一个问题，人们在使用机构名的时候，很多情况下，使用其简称。比如说，某某是清华的校友，某某是北大的学生，联想收购了 IBM 的 PCD，外管局发布了关于外汇交易的新规定等。这其中，清华，北大分别指的是清华大学和北京大学；联想指的是联想集团有限公司；外管局就是中国外汇管理局。

1.1.2　机构名简称的使用带来的问题

机构名简称的使用，本文认为可以分为以下几类：

1．长期的习惯。大家约定俗成简称一个机构名，甚至有时候都淡忘了这个机构名的全称。这样的简称一般比较统一，比如说在大陆我们说清华，指的就是清华大学。很少人会简称清华大学为清大，这就是长期的习惯问题。这种简

称可以作为一个常识。

2．有些机构名称过于复杂，想要把它们完全记住，相当困难，这时候人们就会用简称。这种简称可能会根据个人的习惯不同而不同，会导致一个机构名的简称有很多，每个人简称都不尽相同。比如说中央人民政府驻香港联络办公室，官方的简称为“中联办”，但是很多人简称为中联办香港（还有一个中央人民政府驻澳门联络办公室，这样简称，可以区分这两个机构名），香港中联办，中央驻港办等等。

3．有些机构名并不复杂，简称纯粹为了方便。中国银行，有人简称中行，也有人简称中银，这些都是方便记忆。这种简称和 2 类似，会导致因人不同而得到不同的简称。

如果一种机构名简称出于第一种情况，那么可以经过统计，用常识库来解决，但是有三个问题：第一，如何判断这个简称已经作为一个常识了，标准很难定；第二，如何防止检索者别出心裁给这个机构名起个简称，我们光靠常识库是不够的；第三，机构名数量多，随时间的变化而变更的，那么配置一个常识库有难度。

如果一个机构名的简称不是出于第一种情况，那么要解决简称和全称之间的对应关系就更难了。

所以，不管是哪种原因导致人们简称一个机构名，都会给检索带来很大困难。因为机构名太多，而且很多简称也不统一，所以我们很难把这些简称一一列举，作为机构名的代称。

1.1.3　机构名简称检索的意义

正如 1.1.1 节所讨论的，机构名全称及其简称被普遍用于现实生活。在日常生活中，机构名的简称尤其普遍。

在中文信息处理中我们常常碰到以下问题：

1．在整篇文章中，如果第一次出现一个机构名全称，如“中国银行总行”，在文章后面都简称为“中总行”，其实我们知道这两个专有名称指的是同样的机构，但是计算机不知道。

2．现实生活中，很多机构名很长，很难记，人们平时使用的时候，都会简称这些机构名。比如说“中总行”指代“中国银行总行”，“农总行”指代“农业银行总行”，“东航”或者“东方航空”指代“东方航空有限公司”，“中

石化”指“中国石油化工有限公司”等等。如果不预先使用中文机构名的简称检索来确定它们的对应关系，则计算机很难处理这些简称输入。如果使用全文检索，则无从了解用户的真正需要。如果使用现在的通用领域的搜索引擎来检索的话，肯定会返回很多无用的信息。

3．在数据库中，人们经常会把地点名称，机构名称作为一个索引项，在查询的时候，由于用户的查询条件，尤其是针对机构名方面的查询条件跟数据库中所存贮的不一样，导致查询不到。在数据库中，存贮的一列机构名全称，而检索的时候，如果用全字匹配效果肯定很差。

综合以上三点，基于中文机构名简称的检索方法很重要。现在还没有非常相关的文献介绍该研究方向，但在信息检索领域，很多公司推出公司黄页查询，就是解决公司相关信息查询的问题。目前比较常见的有百度黄页、新浪黄页等，它们都有自己的搜索引擎，查询效率高，但基本上是全字匹配，没有简称匹配，所以有问题。例如查询“交行”，百度返回的是“合肥交行大厦物业管理处”，不是人们通常需要的“交通银行”；而新浪则没有查询到任何结果——且两家都查不到“交总行”。也就是说，这些黄页查询，在判断一个中文机构名的简称和中文机构名的全称的对应关系上，做得不够。

既然基于中文机构名简称的检索方法的研究工作很重要，目前又没有非常相关的文献介绍，同时现有的商业系统做得也不够，那么本文研究的这个工作就具有较强的现实意义。本文对机构名的结构做了深入的研究，结合信息检索领域的技术，重点解决用简称来检索全称的问题，可以用于对话系统，公司黄页查询，数据库查询等领域。

1.2　机构名简称的检索研究的关键技术

在机构名识别方面，目前很多学者在机构名的研究上做出了贡献。即是否是机构名或其简称的分类判别，在分词领域和信息抽取领域的命名实体识别上，已经有了广泛和深入的研究[1][2][3][4]，研究的主要内容是把机构名等未登录词识别出来并基于此实现一个较完美的分词系统。

但是在机构名简称的检索方面，比较相关的领域，如英文机构名[4][6][7]和中文机构名[8]，它们主要研究机构名简称的匹配，检索方法目前没有什么研究。根据研究，本文认为设计机构名简称的检索系统，需要解决两个关键问题：第一，

如何判断一个输入串最有可能是哪个机构名的简称，这需要解决机构名全称和简称之间的内在关系；第二，如何提高检索效率。

针对第一方面，对中文机构名的全称和简称的匹配，目前有些研究[4][6]。本文的研究重点之一就是解决机构名简称和全称的匹配问题。基本思路是：根据机构名全称和简称的匹配规则，离线得到每个全称所有可能的简称，在需要根据简称检索机构名全称时，让待查简称和每个全称的所有可能简称，使用字符串模糊匹配并计算其匹配得分，让待查简称和全称本身使用加权的字段匹配计算匹配得分，然后把这两个得分加权，作为待查简称和全称的最终匹配得分。这种匹配算法符合人们使用简称的习惯，应该能在机构名全称库中准确地找到最佳匹配结果。

针对第二方面，即检索效率，本文引入一种多级索引机制，以提高对大数据量的检索效率。

所以机构名简称的检索系统应该包括以下几个关键技术：

1．对机构名的结构要深入研究，对于中文专有名词的内部构成进行较为系统的分析，在此基础上，以求建立组织机构名的内部模式集合。

2．研究人们简称的习惯，建立机构名常用简称的模式，并且要形式化描述它们，便于计算机处理。

3．必须针对该系统定制一个分词系统：一方面用于提取出机构名的主要部分，如何提取机构名的主要部分；另一方面，用于信息检索。

4．设计一个简称和全称匹配算法，更好的判断一个简称输入的全称最有可能是哪个。

5．针对本领域，研究检索系统，使用目前搜索引擎的一些技术，用于检索效率的提高。

1.3　本文的主要工作和论文的组织结构

1.3.1　论文的主要工作

为了实现基于中文机构名简称的检索系统，本文做了广泛的调研，也做了大量的科学研究和工程设计。本文的主要工作可以归纳为以下几点：

1.　研究了中文机构名的组成结构并形式化描述了其结构；

2. 研究如何描述人为简称的习惯，发现简称和机构名全称的结构相关，并形式化描述人们简称习惯，便于生成一个机构名全称的可能简称;
3. 定制一个基于关键词类的带标注的中文分词系统，解决了该限定领域的分词问题，便于对机构名的成分进行划分；
4. 在匹配输入串和机构名的全称时，结合字符串匹配和字段匹配算法，研究了新的算法，提高了简称输入和机构名全称的匹配准确度；
5. 设计出针对本领域的多层索引结构，使用扩展的 Boolean 模型，实现了多级索引技术，在不降低检索准确性的基础上，提高了检索效率，同时利用单字和词，拼音的结合来检索，提高了搜索精度。

本文综合应用上述方法以及其他实现技巧，设计并实现了一个基于中文机构名简称的检索系统。实验证明，本文系统取得较好的效果。

1.3.2　论文的组织结构

本文是根据作者的工作进行组织安排的，具体的结构如下：

第 1 章，首先说明了本论文工作的研究背景和意义，以及论文的主要工作和难点；

第 2 章，主要介绍了机构名的机构研究，并形式化描述了机构名的结构及其简称结构，提出两类规则；

第 3 章，介绍检索系统中的中文分词技术，并给出实验结果，以论证定制分词系统的必要性；

第 4 章，介绍了简称和全称匹配技术的研究，结合字符串匹配和字段匹配算法，提出了简称和全称的匹配算法，并给出了实验结果和结论；

第 5 章，在解决了匹配准确性问题后，本文引入多级索引技术，以提高检索的时间效率；

第 6 章，在前面几章介绍的基础上，给出检索系统总的框架，介绍检索系统的性能；

第 7 章，本文的工作总结与未来工作展望；

第 8 章，给出了本论文所引用的参考文献。

第 2 章　机构名的结构研究

2.1　综述中文机构名组成结构的研究成果

组织机构名是指机关、团体或其他企事业单位，包括：学校、公司、厂矿、银行、医院、研究所以及政府部门机关的厅、局、部、委等等。组织机构名的构成不稳定，随着社会的发展，新的组织机构不断涌现。

在中文机构名的组成结构方面，很多专家和学者都做了研究。[3]对中文机构名称尤其是中文高校名称的组成和特征进行了深人的分析，并采用基于规则的方法对高校名称进行识别，取得了很好的效果。[4]对金融新闻文本进行了深入的分析和研究，总结出了公司名的结构特征及其上下文信息。[9]对中文组织机构名的内部构成进行了较系统的分析，在此基础上，建立组织机构名的内部模式集合。[10]把机构名划分为 5 个大类，分别总结出其构成模式，包括团体、社团、民间组织机构名称模式，与竞技、竞争、技艺等有联系的团队，教育科研机构，金融机构名称和其他企事业单位名。[11]把机构名分为 9 类并对其进行特征分析，发现以地名作为开头的机构名占了 68.7%。

本文最主要做公司名称和学校名称结构的研究。公司名可以分为两种：全称，简称。所谓公司名全称，就是公司名的最正式的叫法，一般由地名、公司名关键字、公司类型名和公司名后缀组成。根据 1991 年 7 月 22 日国家工商行政管理局令第 7 号公布的《企业名称登记管理规定》和 1999 年 12 月 8 日国家工商行政管理局令第 93 号公布的《企业名称登记管理实施办法》以及 2000 年 1 月 13 日国家工商行政管理局令第 94 号公布的《个人独资企业登记管理办法》，占机构名绝大多数的企业名称，一般企业名称应当由行政区划、字号、行业、组织形式依次组成，如“ 青岛海尔空调有限公司 ”，包括了地名“ 青岛 ”，公司名关键字“ 海尔 ”，行业类型名“ 空调 ”、组织形式名“ 有限公司 ”。“ 中国石油化工有限公司 ”，“ 中国移动有限公司 ”，“ 微软中国有限公司 ”，“ 中国移动北京分公司 ”等分别可以类似得到。本文就以“ 所属地 ＋ 固有名称 ＋ 行业性质 ＋ 组织形式 ”作为公司名的结构形式。公司名的简称就是全称的缩写。

其他的机构名如大学，清华大学、北京大学、北京外国语大学等等，

也可以归为“所属地 ＋ 固有名称 ＋ 行业性质 ＋ 组织形式”结构。总的来说，机构名的构成是有一定的模式的。

2.2　机构名组成结构的形式化表示

本文为了形式化机构名的结构，便于计算机处理，把机构名的构成成分划分为如下几个关键词类：RegionName，Unknown，IndustryType，OrgType，用R，U，I，O来简称。RegionName是所属地名称，如“北京”，“上海”等；Unknown是固有名称，如“广州美的”中的“美的”，“四川长虹”中的“长虹”，IndustryType是行业性质，如“石化”，“银行”等；OrgType指的是组织形式，如“公司”，“股份公司”，“大学”等。关键词类可以扩展，以解决更复杂的组织结构名或者其他领域的问题。

本文认为机构名可以看成是关键词类名的有序排列。本文用规则描述这些有序排列，用<R>等表示属于 R 关键词类的一个关键词。规则可描述如下：

表2.1　机构名全称的规则描述

规则代号	规则	举例说明
Rule0	<R> <U> <I> <O>	北京斗牛士贸易公司
Rule1	<R> <I> <O>	中国移动有限公司
Rule2	<R> <I> <I> <O>	中国石油化工股份有限公司
Rule3	<U> <I> <O>	东风汽车股份有限公司
Rule4	<U> <O>	葛洲坝股份有限公司
Rule5	<U> <I> <R> <O>	海尔集团北京分公司
Rule6	<R> <I> <R> <U> <I> <O>	中国石化北京燕山石化有限公司
Rule7	<U> <I> <R> <I> <O>	中建国际北京装饰有限公司
Rule8	<R> <I> <R> <O>	中国银行北京分行
Rule9	<R> <I> <U> <I> <I> <O>	中国石化齐鲁石油化工有限公司
……	……	……

一个机构名的结构可以归为某一类规则。例如，“北京斗牛士贸易公司”分词后得到“北京/R 斗牛士/U 贸易/I 公司/O”，对应 Rule0，“斗

牛士”是固有名称。

如果机构名经过分词后，不能归结到现有的机构名规则中去，则系统生成新的规则，自动加入规则库中去，新规则和现有规则描述形式相同。

本文系统中目前有 25 条规则。本文在上海证券交易所上市的公司名称库上做了个实验，该公司名称库一共有 888 个公司名称，按照本节方法对它们进行分析后，得到每一个机构名的规则。经统计发现有 98%以上的公司名规则属于这 25 条规则，只有 15 个机构名经过分析需要产生新的规则。可见目前规则具有较大的覆盖率。

2.3　形式化描述中文机构名简称问题

人们简称机构名是有一定的习惯的，比如“建设银行”简称为“建行”，“联想公司”则简称为“联想”。机构名的简称和相应机构名的结构有关系。本文根据全称的结构和人们简称的习惯，得到一类机构名的简称规则。

[4]认为“公司名的简称是对公司名全称缩略的叫法，对于含有关键字的公司名全称，公司名的简称一般都包含公司名关键字，而地名、类型名和后缀都是可选的部分，可以有，也可以没有；对于含有多个类型名的公司名，简称中可以包含任意一个类型名，也可以含有多个类型名。例如“深圳市赛百诺基因技术有限公司”，既可以简称为“赛百诺公司”，也可以叫做“赛百诺基因公司”。

把中文机构名的简称分为四类，本文以一些上市公司的官方简称举例：

表2.2　简称的种类

简称种类	机构名全称	简称
1．缩写	中国石油化工有限公司	中石化
	中国银行	中行
2．连续简写	联想集团有限公司	联想
	南京水运实业股份有限公司	南京水运
	北京同仁堂股份有限公司	同仁堂
	浙江华立科技股份有限公司	华立科技

	林海股份有限公司	林海股份
3．不连续简写	上海国际机场股份有限公司	上海机场
	上海港集装箱股份有限公司	上港集箱
	金花企业(集团)股份有限公司	金花股份
4．简写缩写结合	武汉钢铁股份有限公司	武钢股份
	上海浦东发展银行股份有限公司	浦发银行
	中国国际贸易中心股份有限公司	中国国贸

如何描述这些简称规则，本文发现简称习惯和它们的组成结构有关系。比如说“中国银行”简称“中行”，“建设银行”简称“建行”；“安徽大学”简称“安大”，“北京大学”简称“北大”；“清华大学”简称“清华”，“复旦大学”简称“复旦”；“莱芜钢铁股份有限公司”简称“莱钢股份”，“武汉钢铁股份有限公司”简称“武钢股份”。

通过这些，本文认为一个机构名的简称和其组成结构有一定的关系。为了描述这种关系，本文把一个词的属性划分为以下 5 类，*Full*，*Null*，*LeftCC*(*n*)，*RightCC*(*n*)，*GetCC*(*i, j…*)，其中 *Full* 表示这个词的全部，*Null* 表示这个词的空，但 *Full* 和 *Null* 不在规则中显式出现，*LeftCC*(*n*)表示取出这个词的前 *n* 个汉字，*RightCC*(*n*)表示取出这个词的后 *n* 个汉字，*GetCC*(*i, j, …*)表示取出第 *i*、*j*、…个汉字等。简称规则中，<R>[*n*]表示，规则中第 *n*+1 个属于 R 关键词类的关键词。

简称规则的具体形式是：

表2.3　简称规则描述

符号	简称规则
AbbRule0	Rule0 -> <R> + <U>
AbbRule1	Rule0 -> <U> + <I>
AbbRule2	Rule1 -> <R>.*LeftCC*(1) + <I>
AbbRule3	Rule2 -> <R>.*LeftCC*(1)+ <I>[0].*LeftCC*(1) + <I>[1].*LeftCC*(1)

AbbRule2代表：属于全称规则Rule1的机构名，它们的简称可以表示为<R>的第1个汉字加上<I>的全部。

AbbRule3表示：输入全称规则Rule1的机构名，它们的简称可以表示为<R>的第一个汉字加上第1个<I>的第1个汉字和第2个<I>的第1个汉字。例如“ 中国移动有限责任公司->中 移动 ”就属于该类。由于简称规则较多，在这不一一列举。

简称规则由一些人们习惯得到，这些习惯以文本形式存于文件中，每一个习惯可以得到一类简称规则，人们简称的习惯表示，人为地以空格分开，这样便于计算机处理，举例如下：

表2.3　简称习惯举例

全称	简称表示
中国石油化工股份有限公司	中 石 化
联想集团有限公司	联想
联想集团有限公司	联想 集团
中国石化燕山石油化工有限公司	燕山 石 化
中国移动北京分公司	北京 移动
北京中经纬科技发展有限公司	中经纬
广东美的有限公司	美的
广东美的有限公司	广东 美的
济南钢铁股份有限公司	济 钢 股份
济南钢铁股份有限公司	济 钢
江苏新城房产股份有限公司	江苏 新城
江苏新城房产股份有限公司	新城 房产
上海华源股份有限公司	华源 股份
有研半导体材料股份有限公司	有研 半导体
有研半导体材料股份有限公司	有研 股份
北京保国食品有限公司	保国 食品
北京大学	北 大
清华大学	清华

简称习惯和简称规则之间的对应关系，举例如下：

表2.4　简称习惯及相应简称规则

简称习惯描述形式	简称规则
中国石油化工股份有限公司->中 石 化	Rule2 -> <R>.*LeftCC*(1) + <I>[0].*LeftCC*(1) + <I>[1].*LeftCC*(1)
联想集团有限公司->联想	Rule3 -> <U>
联想集团有限公司->联想 集团	Rule3 -> <U> + <I>
中国移动北京分公司->北京 移动	Rule7 -> <R>[1] + <I>
中国移动有限责任公司->中 移动	Rule7 -> <R>.*LeftCC*(1) + <U>

下面举例介绍简称规则的提取算法：

“ 中国移动有限责任公司->中 移动 ”, 对于全称“ 中国移动有限责任公司 ”, 分词的结果是“ 中国/R 移动/I 有限责任公司/O ”，属于规则 Rule1，而简称是“ 中 移动 ”，“ 中 ”对应“ 中国 ”，则简称时，把该分词成分的<R>部分的第 1 个字加到简称中去，“ 移动 ”对应“ 移动 ”，则把<I>全部加入，<O>部分在简称中没有，则<O>部分不加入，这样就得到了 Rule1 的一个简称规则。

本文把人们简称的习惯按照“ 中国移动有限责任公司->中 移动 ”这样的格式按每行一个存入一个文本文件。在程序运行时，可以首先分析这个文本文件，对每一行按照简称规则的提取算法，提取每一个简称规则，如有重复，则不算入。通过这样的简称规则提取，我们可以得到每一个机构名全称规则的所有的简称规则。

如何得到一个机构名的简称的过程，就是上面算法的一个逆过程。首先得到该机构名全称对应的规则，可以得到该全称规则对应的所有简称规则，根据每一个简称规则可以得到一个简称名，所以一个机构名可能会得到好几个不同的简称。本文系统，目前总共的简称规则有 63 个，平均每一个全称规则有 2.5 个简称规则，也即，根据本文算法每一个全称平均有 2.5 个简称。

本文针对 888 个上市公司全称按照以上方法生成简称，得到了 2，356 个简称。经过统计，这 2，356 个简称包含了 94%的上市公司股票名，剩下不到 6%的公司股票名虽然不在这些生成的简称中，但是本文生成的简称已经和股票名

称相似，而且同样可以作为这些公司名的简称。每个上市公司平均拥有 2.7 个简称，根据分析，这些简称在日常生活中都有可能作为公司名的简称。所以本文这个简称生成算法不仅可以考虑官方简称，也兼顾了人们简称的习惯，有很大的实用性和广泛性。故本文的简称生成算法是有效的。

本章方法存在这样的两个问题：第一，多个机构名对应同一个简称；第二，比如说“ 中国石油天然气有限公司 ”可能被简称为“ 中石天 ”。第一个问题是很正常的，因为现实生活中就存在很多机构名对应统一简称问题；至于第二个问题，如果解决它，就需要依靠常识库。

2.4　本章小结

本章主要对机构名的组成结构做了深入研究，并用规则形式化描述其结构；然后本文研究人们简称的习惯，总结出简称规则，为后续的处理做准备。

第 3 章　检索系统中的中文分词技术

3.1 中文分词系统综述

3.1.1　分词意义

中文自动分词是对汉语文本进行自动分析的第一个步骤。中文信息检索、机器翻译、自动文摘、文本校对、繁简转化和语音合成等领域，都需要中文分词[12]。

本文着重讨论中文分词在文本检索中的意义，它主要表现在下面两个方面：

1．准确性的提高。对中文文本信息检索来说，由于中文本身的特殊性，词与词之间没有明显的切分标记，如果不采用分词技术，简单的以字为单位建立索引，那么信息检索的结果就过于粗糙而导致检索资源不正确或不可用。如果不切词(按字检索)，可能检索的结果与用户的查询要求会大相径庭，例如当检索德国货币单位“ 马克 ”时，就会把“ 马克思 ”检索出来，而检索“ 华人 ”时会把“ 中华人民共和国 ”检索出来。因而，分词能有效地提高文本检索的准确率。

2．时间效率的改进。在文本信息检索中，如果按照单字来建索引和检索，时间效率不高，因为如果按照词检索，则在索引中查询的次数要比按照单字检索要少。比如查询“ 清华大学 ”，经过分词，只需要查询“ 清华 ”和“ 大学 ”两个词即可，如不用分词，则需要查询四个字，其检索效率肯定不如分词后的文本检索系统。

因此，通常的检索引擎都是以每一个独立的词为单位建立索引，在查询时按照检索词出现的位置和频率对文档进行输出。本文所设计的是一个检索系统，所以应该使用中文分词系统，同时本文使用分词系统还有其他作用，在本章的其他的节将详细描述。

3.1.2　分词综述

20 多年来，国内外学者们对中文文档的自动分词工作开展了大量研究，提出了许多自动分词方法。很多研究者都对中文分词研究做了综述[13][14]。

分词系统按照其使用方法可以分为 3 大类：

第一种是字典法[15][16]，也叫机械匹配算法，正向最大匹配(FMM)、逆向最

大匹配(RMM)、双向最大匹配(BDMM)。

字典法易于实现，但其缺点在于：由于词典是在分词之前准备的，其规模和内容受到了限制，所以没有哪个词典是完备的，语言中常出现新的词语，基于词典的这一类算法无法解决文本中大量出现的未登录词的问题。

第二种基于概率统计的分词方法[17][18][19][20][21]。例如 N-Gram 算法，HMM 算法，最大熵算法，基于 EM 的算法等等。

统计方法的优点在于：它可以从已有的大量实例中进行归纳总结，分析语言内在的关联信息，将其加入到统计模型中去。统计的方法的缺点是：训练语料库的规模严重影响着分词的效果，不同领域的语料对于统计模型起着决定性的作用。

第三种统计与词典的结合方法进行[22][23][24]，这些研究逐渐成为目前分词系统的主流，并且取得较好的效果。

但这种分词算法也存在着不足：其上下文信息都是从训练语料库中获取，忽略了切分文本的上下文反馈信息。

3.2　带标注的基于关键词类的中文分词系统

3．2．1　设计目的

本文需要对中文机构名全称进行分词，尤其是公司名的结构。经过分析，机构名称中包含很多未登陆词。所以，本文必须在识别这些未登陆词方面要下功夫。

目前反映比较好的分词系统，如清华大学的 CSeg&Tag，北京语言文化大学的“ 现代汉语通用分词系统 ”，中科院的 ICTCLAS[25]，北京大学[26]分词系统等。这些通用的分词系统，在通用文本上的分词性能的确很好，但是未必对本文这个特殊领域有用。针对要不要开发一个针对该领域的分词系统，本文做了一些研究，测试了两个分词系统，其一是中科院的 ICTCLAS，另一个是北京大学分词系统。

本文用中科院 ICTCLAS2.0 系统来分析一些待处理的机构名，结果如下：

表3.1　ICTCLAS2.0的部分分词结果举例

输入	分词结果
北京中经纬科技发展有限公司	北京 中 经纬 科技 发展 有限公司
北京众帮纸制品加工厂	北京 众 帮 纸制品 加工厂
北京红石坊广告设计有限公司	北京 红石 坊 广告 设计 有限公司
北京盛强精卡贸易中心	北京 盛 强 精 卡 贸易 中心

本文使用分词系统对上面的机构名进行切分，结果如下：

表3.2　北大分词系统的部分分词结果举例

输入	分词结果
北京中经纬科技发展有限公司	北京 中 经纬 科技 发展 有限公司
北京众帮纸制品加工厂	北京 众 帮 纸 制品 加工厂
北京红石坊广告设计有限公司	北京 红石 坊 广告 设计 有限公司
北京盛强精卡贸易中心	北京 盛 强精 卡 贸易 中心

根据表 3.1 和表 3.2，在机构名的分词方面，这两个系统都不太令人满意。虽然它们在通用文本的分词已经取得了很好的效果，但是在分析机构名时，存在一些问题。所以针对机构名的处理，本文必须定制自己的分词系统。

本文分词的目的可以概括为：

1. 把机构名的各个部分划分出来，尤其是其中的固有名称，固有名称一般是未登陆词；
2. 对分词的各个部分进行标注，标注的作用是为了更清楚的得到机构名的结构，明白各个成分，得到第 2 章的两类规则；
3. 分词便于多级索引系统的索引设计和检索，详见第 5 章。

3．2．2　具体实现

根据分析，机构名的分词有以下几个特点：

1．机构名中有很多未登陆词，如“ 广州美的公司 ”中的“ 美的 ”；

2．未登陆词的前后词类，往往比较固定，比如“ 通威 股份 有限公司 ”，“ 华纺 股份 有限公司 ”，“ 通威 ”和“ 华纺 ”都是未登陆词，它们后面都出现了公司类型词；

3．机构名的组成结构相对比较简单，所涉及的词类有限；

4．机构名所用的词和平时的意义可能不同，“ 江苏阳光股份有限公司 ”中的“ 阳光 ”，“ 北京联想集团有限公司 ”中的“ 联想 ”，虽然都是一个通用词，但是本文认为其是未登陆词，它们代表公司的字号。

于是，采用带标注的基于关键词类的中文分词系统是一个好的选择。通过这个分词系统，本文不仅仅能够切分出机构名的各个组成部分，而且能够标注各部分，把机构名中的未登陆词划分出来。

具体做法为了以下三个部分：

1．关键词类的生成

根据机构名的结构，机构名中的关键词分为 4 类。本文从 5 万机构名称库中，根据词频，得到一些常用的词，然后人工的把它们划分为 4 类关键词，分别为区域类、行业类、组织结构类，和一个未知类，分别用 R，I，O，U 来表示。这四类对于公司名称已经可以覆盖了。但是如果需要处理其他类型的机构名等，需增加词类或者相应词类的词库。

如“ 中国 ”，“ 北京 ”等关键词都属于区域类；“ 石油 ”，“ 化工 ”等都属于行业类；“ 有限公司 ”，“ 分公司 ”，“ 大学 ”等都属于组织结构类。而不属于以上各类的关键词，叫固有词，归为未知类，离线分析时未知词类始终是空的。本文不把固有词作为一个常识性的词，因为一个词在一个机构名中是固有词，则在其他机构名中它或许就不能作为一个固有词，如“ 广州宏达贸易公司 ”中的“ 宏达 ”是固有词，但是在“ 广州宏达昌食品公司 ”中“ 宏达昌 ”是固有词，而不是“ 宏达 ”，所以未知类只作为标识来用。

在机构名中，往往简称的主要部分来源于固有词，它是标识一个公司区别于其他公司的最重要部分。把固有词划分出来，是本文分词的一个主要目的。

2．双向最大匹配算法

本文所设计的分词系统基于关键词类，没有用到任何概率和统计信息，本文采用最大匹配算法。

Sun M. S.和 Benjamin K.T.[27]注意到：汉语文本中 90.0%左右的句子，FMM 和 RMM 的切分完全重合且正确，9.0%左右的句子 FMM 和 RMM 切分不同，但其中必有一个是正确的(歧义检测成功)，只有不到 1.0%的句子，或者 FMM 和 RMM 的切分虽重合却是错的，或者 FMM 和 RMM 切分不同但两个都不对(歧义检测失败)。这就是双向最大匹配算法被广泛应用的重要原因。本文使用双向最

大匹配算法，也是因为它可以解决绝大多数匹配歧义的问题。

3．未登陆词识别策略

在汉语自动分词处理中，未登录词的识别是一个难点。在未登陆词识别方面，已有的工作涉及了四种常见的专有名词，中国人名的识别[28][29][30]、外国译名的识别[31]、中国地名的识别[32]及机构名的识别[3][4][33][34]。

虽然最大匹配算法解决不了未登陆词识别的问题，但是本文通过词类和规则的约束把这些未登陆词给识别出来。基本思路就是：首先划分出那些确定的部分，即区域类、行业类和组织结构类，再把没有归为这些确定类的词归为未知类。这种方法比较简单易于实现，但是很有效。

经过以上三步，针对上面的机构名，使用本文定制的分词系统得到的部分结果举例如下：

表3.3　本文系统的部分分词结果举例

输入	分词结果
北京中经纬科技发展有限公司	北京/R 中经纬/U 科技/I 发展/I 有限公司/O
北京众帮纸制品加工厂	北京/R 众帮/U 纸制品/I 加工厂/O
北京红石坊广告设计有限公司	北京/R 红石坊/U 广告/I 设计/I 有限公司/O
北京盛强精卡贸易中心	北京/R 盛强精卡/U 贸易/I 中心/O

3.3　实验结果

根据本章 3.2 小节介绍，本文设计出针对机构名的分词系统。本文采集了上海证券交易所上市的公司名称库，该库包括 888 个上市公司名。在该库上，本文进行了几组对比实验，分别和 ICTCLAS 和北大分词系统比较。切分正确的标准是：只有把机构名的各成分切分开来，本文才认为是正确的。比如把机构名的固有名称分开的，就认为是切分错误。

表3.1　三个分词系统的分词结果比较

方法	切分正确率
ICTCLAS	57.5%

北大系统	42.9%
本文系统	96.6%

本文分析了 ICTCLAS 的分词错误，发现绝大多数都是未登陆词的切分错误。北大系统在切分机构名的未登陆词方面表现更差一些。而本文系统在机构名分词方面取得了不错的效果。

本文系统分词中也出现了一些问题，主要分为两类：

1).切分歧义问题

成都旭光电子股份有限公司 -> 成都 旭 光电子 股份有限公司

吉林华微电子股份有限公司 -> 吉林 华 微电子 股份有限公司

2).词库不足的问题

上海港集装箱股份有限公司 -> 上海 港 集装箱 股份有限公司

上海陆家嘴金融贸易区开发股份有限公司 -> 上海 陆家嘴 金融 贸易 区 开发 股份有限公司

以上两类问题，有一个共同点就是出现单字成词现象，根据机构名中一般不会出现单个字的词，如果出现单个字的词则约束处理，要么放在前面，要么放在后面。所以本文可以调整算法把单字归为左边或者右边，以解决这两类问题。

本文系统也是有缺陷的。本文分词系统只适合机构名的分词，对于通用领域的分词不如现在各研究单位的分词系统。另外，本文分词系统如果要进行其他领域的分词，必须增加关键词库。

3.4　本章小结

本章介绍了中文分词系统的重要性，并且综述了目前的中文分词研究。由于本文涉及领域的特殊性，目前的分词系统针对该领域分词效果不理想，所以本文定制了一个基于关键此类的带标注的分词方法，并和两个通用的分词系统进行实验对比，实验表明，在针对机构名这个特定领域，本文分词系统有更好的切分性能。

第 4 章　简称和全称匹配技术研究

4.1　匹配算法概述

在文本处理领域中字符匹配算法是非常重要的。它可用于数据处理、数据压缩、文本编辑、信息检索等多种应用领域中。在实际应用中字符串匹配技术不仅适用于以上领域，在语义学、分子生物学等领域也具有相当重要的应用，在以模式匹配为特征的网络安全应用中也发挥了举足轻重的作用。本文把匹配算法分为两类：第一类是字符串匹配，就是基于整个字符串来说的匹配；另一类，是字段匹配，首先把字符串分割成字段，再匹配。

4.1.1　字符串匹配

字符串匹配包括精确匹配和模糊匹配。目前关于字符串匹配算法，有不少综述文章[35][36]。

精确匹配算法，也即模式匹配。字符串的模式匹配解决的问题是要寻找一个字符串在另一个字符串中是否出现，以及出现的位置，如 Knuth-Morris-Pratt (KMP)[37]、Boyer Moore (BM)[38]、Quick Search (QS)[39]。以上所举都是字符串的单模式匹配，在字符串的模式匹配领域，多模式的匹配是另一个研究热点，如 Aho-Corasick 算法[40]。本文不采用精确匹配的路线，故不对每一个算法详细介绍。

字符串的模糊匹配解决的问题是：在检索时找出与模式串相似的字串，也就是允许找到的字串与模式串有一定的偏差。比如说，模式串“ 中石化 ”，文本中包含“ 中国石化 ”，如果这种差别在允许的范围类，则可以认为文本中出现了模式“ 中石化 ”。相对于精确匹配，模糊匹配在很多领域更有实用价值。

在字符串模糊匹配这个领域，有四条主要的技术路线：

1．动态规划算法。动态规划算法最早于 1968 年由 Vintsyuk 提出[41]，以后又经过了不同程度的改进。动态规划算法灵活并且可扩展，是其它 3 种算法的理论基础。

2．自动机算法。自动机算法的思路是建立一个非确定状态自动机(NFA)。目前有两种思路，第一个是把这种 NFA 思路转化为确定状态的自动机(DFA)，这是一个自动机类算法[42]。另一个是直接模拟该 NFA[43]。

3．过滤算法。过滤算法的基本思路是基于过滤文本丢弃不可能匹配的大部

分文本。很多时候，我们能很容易地确定两个字符串不相似，比如两个字符串所包含的字符完全不同。过滤算法借鉴了这种想法，采用简单快速的方法过滤掉大部分不匹配的串，然后再采用其他的模糊匹配算法与剩下的小部分串匹配。过滤算法必须配合其他模糊匹配算法使用。

4．位并行算法。位平行算法的基本思路是用“位平行”的方法来改造已有的其他算法，在模式串较短时特别有效，因此在文本检索领域应用广泛。位平行化算法的缺点是算法依赖于计算机字的长度（一般为 32 或者 64 位）。

本文只用到模糊匹配算法，在衡量两个字符串的匹配度方面，采用 Levenshtein edit distance 算法[44]。

4.1.2　字段匹配

在匹配算法中，字段匹配也可用以确认两字符串的匹配度。字段匹配技术分为两种，简单的基于字符的字段匹配算法和处理单词缩写的回归字段匹配算法。

1．简单的基于字符的字段匹配算法

英文字段匹配方面有一些研究[5][6][7]。1 个字符串是 1 个字符序列，1 个简单的匹配度定义是 2 个要进行匹配的字符串中可以互相匹配的单词个数除以 2 个字符串平均所有的单词个数。如 2 个单词相同，或 1 个是另 1 个的前缀，就认为它们匹配。例如：A=“Comput. Sci. & Eng. Dept., University of California, San Diego”和 B=“Department of Computer Science, Univ. Calif., San Diego”。去掉分隔符、无意义字符等。在西文下，一个单词用它的前缀表示时，都在最后用一个“.”作标记。故在 A 中有 k=6 个单词（Comput., Sci., University, California, San, Diego）和 B 中的单词匹配，则匹配度为 k/(|A|+|B|)/2=0.8。|A|表示 A 中的单词个数，为 8 个，of 不算。|B|=7。算法很简单：首先，将每个字符串的单词抽取出来排序；其次，是用 1 个字符串中的每一个单词到另一个字符串中的单词中搜寻。匹配的单词个数被记录下来。算法的复杂性取决于单词的排序。

中文字段匹配，和英文有一些区别。第一是：中文字段匹配，必须使用分词系统对中文字符串进行分词，每一个分词成分作为一个字段，或者是几个分词的组合作为一个字段；第二是：在中文中没有前缀词的概念，只有简称，所以在只有当两个字段相同时，才认为它们匹配。[8]在这方面做了研究。首先对

中文字符串进行自动分词处理，使得中文字符串变成为加有分词标记的中文字符串，每一个分词或者几个分词的组合作为一个字段。算法的匹配度定义就是两个分词串中匹配的分词个数除以它们平均的分词个数。如果两个分词相同，就认为它们匹配。例如：A=“清华大学计算机系”，B=“清华大学计算机科学与技术系”，对 A 分词后为 A=“清华 大学 计算机 系”；对 B 分词后为 B=“清华 大学 计算机 科学 与 技术 系”。用空格分割两个切分成分。在 A 中有 k=4 个分词（清华，大学，计算机，系）和 B 中的分词匹配，则匹配度为 4/(|A|+|B|)/2=0.727。|A|表示 A 中的分词个数，为 4。|B|=7。

简单的基于字符的字段匹配算法在英文字符匹配取得不错的效果，因为英文有明显的前缀标识，但是对中文字段匹配效果不好，中文中没有前缀的问题，在中文字段处理中，这种算法很有可能退化为两个字符串中共同的字符数的比较。

另外，简单的基于字符的字段匹配算法在中文和英文匹配方面有一些共同的缺点，如不解决简称问题，不考虑字符串顺序，不考虑字段权重等。

所以，本文系统不采用简单的基于字符的字段匹配算法。

2．处理单词缩写的回归字段匹配算法

该算法使用了典型文本值字段的回归结构。在英文字段匹配研究中，[7]做了研究。基本思路是：如果 A 和 B 是相同的字符串或一个是另一个的缩写，匹配度是 1.0，否则是 0.0。每一个 A 的子字符串和 B 的子字符串进行匹配，记录下子字符串匹配得分最高的情形。每一个 A 的子字符串和 B 的子字符串进行匹配，记录下子字符串匹配得分最高的情形。A 和 B 的匹配度公式缩写的情形有下面 4 个模式：

1) 缩写是单词的前缀,例如 Univ 是 University 的缩写；

2) 缩写是单词的前缀和后缀的组合，例如 Dept 是 Department 的缩写；

3) 缩写是单词的首字母组合，例如 UCSD 是 University of California, San Diego 的缩写；

4) 缩写是单词前缀的串联，例如 Caltech 是 California Institute of Technology 的缩写。

给定 A 和 B，A 中的每个子字符串都必须和 B 中的每个子字符串匹配。在最坏情况下，A 的每个子字符串都的和 B 的每个子字符串比较。这里的子字符串也就是单词，但也可能是单词组成的句子，如上面的“University of California,

San Diego ”。

在中文字段匹配方面，[8]做了研究，基本思路是如果 A 和 B 是相同的字符串或一个是另一个的缩写，匹配度是 1.0，否则是 0.0。每一个 A 的子字符串和 B 的子字符串进行匹配，记录下子字符串匹配得分最高的情形，这个得分作为 A 中该字段的得分，把 A 中所有的字段得分求和，求和的结果对 A 中字段个数求平均，即得到 A 和 B 的匹配度。

处理单词缩写的中文回归字段匹配算法，需要两个条件：第一，要对中文分词；第二，通过表示领域知识的外部文件解决中文缩写和简称问题，必须把相关的中文缩写和简写知识整理好，按照一定的格式写在一个外部文件中，否则计算机无法知道简写和全称之间的对应关系。这些简写知识，如“ 高级人民检察院 ”写成“ 高检 ”，“ 中华人民共和国 ”和“ 中国 ”，“ 交通大学 ”写成“ 交大 ”等。

处理单词缩写的回归字段匹配算法可以解决中英文中的简称问题，比简单的基于字符的字段匹配算法具有更好的适应性。

它有几个问题：

1．字段匹配得分要么是 1，要么是 0，太笼统。很多时候，确定一个字段是不是另一个字段的简称是很难的，如果仅仅用 1 和 0 来衡量，给匹配准确度带来很大的随意性；

2．字符串各部分在标识该字符串的作用可能是不同，这个算法没有考虑到这点，比如机构名中的各成分的作用就是不同的；

3．对于中文的字段匹配，需要配置一个简称库，但是这个简称库是不容易得到的。

但是，这个算法无疑给出了一种匹配思路。本文主要借鉴处理单词缩写的回归字段匹配算法，而不使用简单的基于字符的字段匹配算法，因为后者不能解决中文缩写等问题。本文为了避免回归字段匹配算法的三个问题，打算采用相应的解决办法：

1．针对第一个问题，我们计算匹配得分的时候，采用匹配度来计算，得分介于 0 和 1；

2．针对第二个问题，对字符串分词后，标注各个字段的所属词类，按照每个词类的重要性，采用加权的方法，计算匹配得分；

3．基于 1 和 2 两点，我们就不需要配置简称库了。

4.2　简称和全称的匹配算法

4.2.1　设计思路

本文需要解决的是机构名简称和机构名全称之间的匹配问题，它和传统的字符串匹配是不同的。

1. 如果使用字符串相同的精确匹配，但是很少有一个简称和全称完全相同，所以，字符串精确匹配不适应于本文系统。

2．使用字符串精确匹配算法，用以判断一个字符串是否是另外一个字符串的子串，也会碰到一些问题。比如说，“ 中石化 ”是“ 中石化齐鲁股份有限公司 ”的子串。但是“ 中国石油化工股份有限公司 ”中不包含“ 中石化 ”三个字，如果使用字符串匹配，“ 中石化齐鲁股份有限公司 ”的匹配得分比“ 中国石油化工股份有限公司 ”要高，而实际是，中石化应该指的是“ 中国石油化工股份有限公司 ”，“ 齐鲁石化 ”才是“ 中石化齐鲁股份有限公司 ”。又如“ 中国国贸 ”检索不到“ 中国国际贸易有限公司 ”，因为后者中不包含“ 中国国贸 ”整个词。所以，在本文系统中，我们也不能用该匹配算法。

综合 1，2 两点，在本文系统中，不能使用精确匹配算法，至少是不能单独使用。

3．字符串模糊匹配，使用字符串模糊匹配算法计算两个字符串之间的匹配得分。本文发现，一般情况下，使用简称来检索的时候，简称串长度为 2~6 个字符，通常为 4 个中文字符，而机构名名称则通常是 10 个中文字符以上，如果用 4 个字符和 10 个以上字符来匹配的话，通常得分很低，而且区分度很低。

4．字符串模糊匹配，使用字符串模糊匹配算法，计算一个字符串是另一个字符串的子串的得分。比如计算“ 南航 ”和“ 中国南方航空股份有限公司 ”之间的匹配得分，使用该类方法即可。但是碰到一些问题，如果计算“ 南航 ”和“ 中国南方航空股份有限公司 ”和“ 南京水运有限公司 ”，“ 中国东方航空股份有限公司 ”它们的匹配得分，都比较相近，所以该方法有一定的缺陷。

综合以上 4 点，字符串匹配算法针对本文特定的背景下很难发挥有效的性能。

如果单独使用字段匹配算法，也不能完全解决问题：

1. 如果使用简单的基于字符的字段匹配算法，那么使用“ 清华 ”来匹配“ 清华大学 ”和“ 华清大学 ”得分都相同。同时，它也没法解决中文简称的问题。

2．使用单词缩写的回归字段匹配算法时，它有很大的好处，能够处理中文缩写的问题，但是虽然人们使用缩写，但是也有时候简称本身就是连续的，比如“林海股份”和“林海股份有限公司”之间的匹配，使用该类算法就不正确。同时，该算法不考虑机构名的各成分的作用不同；最后，这个算法需要一个简称常识库，本文很难得到。

故单纯使用字段匹配算法，也不能完全解决问题。本文的做法就是把两类匹配算法结合到一起。

首先是把机构名全称的所有可能简称都生成，用字符串模糊匹配算法来计算用户简称输入和这些生成的简称之间的得分。这样的话，可以顺利解决“中石化”和“中国石油化工有限公司”的匹配问题。因为输入就是后面全称的一个简称。

其次，当用户的输入和机构名能得到的简称有很大的差别的时候，光用字符串匹配是不够的。本文就使用字段匹配，根据机构名的各个部分的不同的重要性，使用加权的字段匹配算法。

最后，把这两个得分再加权，可以解决很多复杂问题。它可以解决“中国石化”匹配“中国石油化工股份有限公司”和“中国石化齐鲁石化有限公司”的问题。

4.2.2　具体算法实现

在计算待查简称和全称的匹配得分时，本文不仅考虑它和全称本身的匹配，也考虑其与全称的简称匹配。下面具体介绍简称和全称的匹配算法的步骤：

1．对机构名的全称分词，得到与之对应的全称的规则。

这个步骤需要用到第 3 章的分词系统，分词后对每个词进行标注，然后根据标注词类的顺序，可以得到一个机构名全称的规则，这个规则可以用于后面的简称生成。

2．得到所有的简称。

得到这个机构名称的全称规则后，可以得到该全称规则对应的所有简称规则，全称规则和简称规则是一对多的关系，也就是说，一个全称规则可以得到几个不同的简称规则，通常一个全称规则对应 1~3 个简称规则产生式。

每一个简称规则，也就是一个简称产生式，通过简称产生式，可以得到相应的简称，可参考 2.3 节。

3．使用字符串模糊匹配，用待查简称与全称的所有简称分别匹配，待查简称和全称的一个简称匹配得分为：

$$f\left(strInput,strAbb\right)=\frac{len_{strInput}+len_{strAbb}-dis_{IA}}{len_{strInput}+len_{strAbb}} \tag{4-1}$$

其中 $len_{strInput}$ 表示 *strInput* 的长度，len_{strAbb} 表示 *strAbb* 的长度，dis_{IA} 表示 *strInput* 和 *strAbb* 之间的差异，用 Levenshtein edit distance 度量。

这就是字符串匹配算法。它避免了 *strInput* 和 *strFull* 的直接匹配。如果用 *strInput* 和 *strFull* 直接匹配，则区分度很小。它同时尊重了习惯简称，可以顺利处理很多常识性的简称和全称匹配的问题。

4．待查简称和通过规则得到的简称的匹配得分为：

$$score_{abb}=\max\bigcup_{strAbb}\left\{f\left(strInput,strAbb\right)\right\} \tag{4-2}$$

把这些简称和待查简称都用公式 4-1 计算一下，取最高的得分为最后得分，这是为了适应不同的用户输入，适合更多的情况，具有普遍性。比如简称“中国建设银行股份有限公司”，可能被不同的人简称为“建设银行”，“建行股份”等。本文取得分最大值，应该来说，更具有广泛性。

5．再用字段匹配，实现待查简称与全称匹配，根据公式 4-3 和公式 4-4 可以算出待查简称和全称的字段匹配得分。

$$g\left(Ti,strFull\right)=g\left(Ti,Pj\in strFull\right)=\frac{num_{TiPj}}{num_{Pj}} \tag{4-3}$$

公式 4-3 计算 *strInput* 的一部分和 *strFull* 的得分，T_i 表示 *strInput* 经过分词后的一部分，P_j 表示 *strFull* 经过分词后的含有 T_i 的那部分，num_{TiPj} 表示 P_j 中含有 T_i 的字符数，num_{Pj} 表示 P_j 的字符数。

回归字段匹配算法用 1 和 0 来区分字段匹配与否，但是本文不使用这种方法。因为 1 和 0 不能定义区分度，同时判断一个字段是否和另一个字段是否匹配是很难的。本文计算字段的匹配得分，这个方法比较简单可行，并且更有区分性和容错性。

$$score_{full}=\sum_{Ti\in strInput}\left\{\theta_i\cdot g(Ti,strFull)\right\} \tag{4-4}$$

公式 4-4 计算输入串和全称匹配得分，其中 θ_i 是加权系数，θ_i 代表 T_i 的重要程度。根据研究，固有名称是一个机构名区别于其他机构名的最重要的成分，所以固有名称的权重应该最高。行业类型名在区分机构名时也是比较重要的，尤其是区别具有相同固有名称的机构名。所属地名称和组织结构在区分机构名时候的作用不如前两个，但是它们也是有作用的，例如“北京移动有限公司”和“广州移动有限公司”就依靠所属地名称来区分；“中国移动研究院”和“中国移动股份有限公司”就得依靠组织结构来区分。本文系统中的加权系数的值是通过实验，取出最好的组合得来的。根据实验，当 Unknown 的权重为 0.5，IndustryType 的权重为 0.3，RegionName 的权重为 0.1，OrgType 的权重为 0.1 时，该算法取得最好效果。

综上，公式 4-3 和公式 4-4 就具体实现了加权的字段匹配算法。

6．公式 4-5 计算得到待查简称和全称的最终匹配得分

$$score_{end} = \partial \cdot score_{abb} + (1-\partial) score_{full} \tag{4-5}$$

∂ 为加权系数，∂ 为 0 时，公式 4-5 就退化为加权的字段匹配算法，∂ 为 1 的时候，公式 4-5 退化为针对公司名简称的模糊匹配算法。∂ 的作用，就是约束两个算法的影响，经过本文实验证明 ∂ 为 0.5 时候，效果比较好，所以本文的匹配算法中 ∂ 取 0.5。

4.3　实验结果

4.3.1　实验条件

为了验证本文所提的简称全称匹配算法的性能。本文做了几组对比试验。实验在一台 AMD 1.75GHz 主频，1GB 内存，Windows XP 的机子上运行，程序由 VS.net 2003 设计。

实验数据是在上海证券交易所上市公司的公司名全称和股票名来组成，我们从上海证券交易所的官方网站上采集了 888 对。我们用每一个股票名分别和这 888 个上市公司名进行匹配，得分最高的上市公司名就被认为是该股票对应的上市公司。

本章在 888 个上市公司库中，使用 888 个上市公司官方简称名来测试，

使用首准确率来判断实验结果。

首选错误率和首选正确率分别用公式 4-6 和公式 4-7 计算：

$$FSE = \frac{difNum}{totalNum} \tag{4-6}$$

$$FSC = 1 - \frac{difNum}{totalNum} \tag{4-7}$$

其中，FSE 表示首选错误率，dif_{Num} 表示结果文件和全称文件不相同的行数，其中一行一个公司名，$total_{Num}$ 是测试串的数目（本实验中为 888）；而 FSC 表示首选正确率。

本在同样的测试环境下，做了一个基准系统，同时设计了 5 个算法，其中包括 4.2.2 节提出的算法，即用字符串和字段匹配加权的得分的算法。

基准系统，根据包含检索串的所有字符数和字符排列顺序来计算得分，得分最高的为该股票对应的公司名。

算法 1，使用 Levenshtein edit distance 算法来做实验，直接使用股票名输入和公司全名进行模糊匹配，得分最高的作为该股票名对应的公司名。

算法 2，使用 Levenshtein edit distance 算法来做实验，使用股票名输入和公司全名得到的所有简称和股票名进行模糊匹配，取得分最高的作为该公司全名和股票名输入的匹配得分，取得分最高的公司全名作为股票名对应的公司名。

算法 3，把算法 1 和算法 2 结合在一起，用股票名输入和公司全名及其简称分别进行模糊匹配，得分最高的公司名作为该股票名对应的公司名。

算法 4，使用字段匹配算法。采用加权的字段匹配算法，权重依次为 0.5，0.3，0.1，0.1。首先对股票名输入进行匹配，得分最高的公司名为股票名对应的公司名。

算法 5，使用算法 2 和算法 4，最后的得分按 0.5 和 0.5 加权，取得分最高的公司作为股票名对应的公司名。

4.3.2　实验结果

表4.1　6个匹配算法的实验结果

方法	***FSC***

基准	80.4%
算法 1	75.0%
算法 2	93.5%
算法 3	93.5%
算法 4	94.9%
算法 5	96.5%

4.3.3　实验结果分析

从表 4.1 的实验结果可以发现：

1．算法 1 最差，并且低于基准系统，因为该算法计算的得分不具有区分性；

2．算法 2 和算法 3 效果基本相同，因为用股票名输入和公司全名及其简称分别进行模糊匹配，取最大值，一般就是用股票名输入和公司全名的简称匹配的最大值。从结果来看，这两个算法的准确率都在 90%以上，可见本文使用的得到简称后再模糊匹配的算法，具有较好的性能；

3．算法 4 的准确性在接近 95%，表示本文提出的加权字段匹配算法，性能比较好；

4．算法 5 得到了最好的性能，表示把模糊匹配和加权字段匹配算法有效的结合在一起，能够取得更好的效果。

实验表明，算法 5 具有最好的性能，也即本文所设计的系统能够更好得解决简称和全称得匹配问题。

但是，算法 5 也有缺陷：

1．算法 5 是算法 2 和 4 的结合，所以计算时间来说肯定比算法 2 和算法 4 都高，也即时间性能不如算法 2 和 4。

2．实际数据库是很大的，如果对每一个数据库中机构名和输入按照算法 5 计算，则耗时很大，不能实用。

所以，针对大数据库集的检索，时间效率是不得不考虑的事情。本文采用多级索引算法来解决时间效率的问题，在下一章节将做详细的介绍。

4.4　本章小结

本章首先介绍字符串匹配和字段匹配算法，然后分析这两种算法在本文所设计的系统中的可用性，最后介绍了本文设计的简称全称匹配算法，它结合了字符串匹配算法和加权的字段匹配算法，根据对比实验，本文所提的匹配算法具有较好的准确性。

第 5 章　多级索引系统与信息检索

5.1　索引技术的作用

索引的指示功能，并非直接提供所需的信息，取而代之地，透过一组字段或是叙述语标记信息的特性与出处。索引提供的可能是信息所在的位置，也可能是表达信息内容的叙述语，所以索引是信息需求与信息之间的桥梁。

索引是传统图书馆学的核心之一，是图书馆组织信息的主要方法。

在信息检索领域，尤其是搜索引擎系统中，索引技术也有广泛的应用。本文就是利用了索引技术来提高检索效率的。

5.2　索引结构的设计

5.2.1　传统的倒排索引

先了解一下传统的倒排文件结构实例，如图所示。

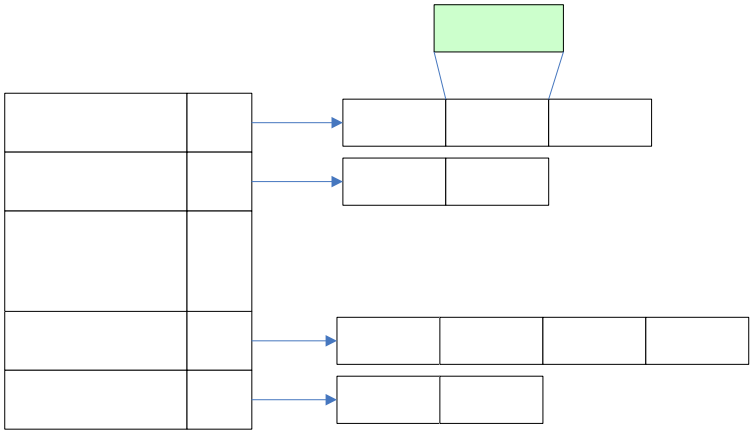

图 5.1　传统的倒排文件结构实例

该索引方式已经在大量的商业检索系统（如图书馆馆藏系统）中得到广泛的应用，倒排表的使用大大提高了检索的效率和速度，尤其面对大型的数据集合的时候，通常表现会非常好。

5.2.2　本文的索引结构设计

本文在传统的倒排索引基础上，把每一个机构名作为一个文档，实现拼音索引项，汉字词索引项，索引字词号，文档号之间的多层索引，如图 5.2 所示。

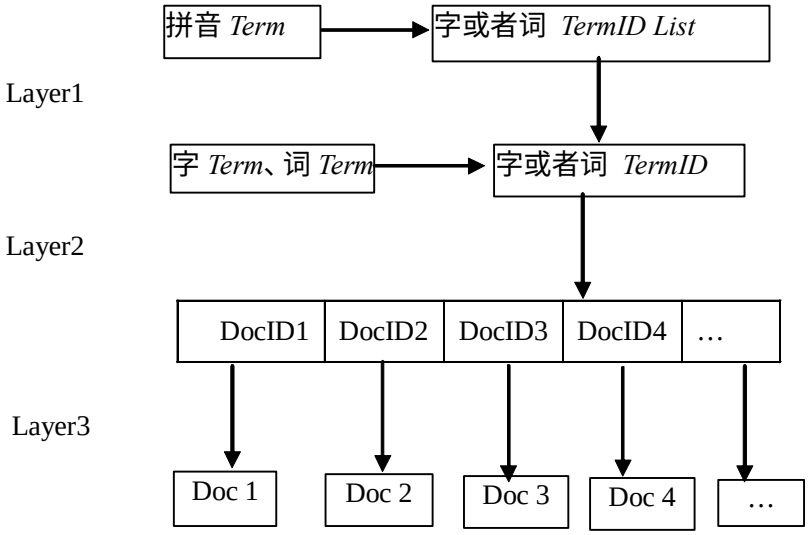

图 5.2　三层索引结构示意图

图 5.2，Layer1 表示的是 *TermID List* 和 *TermID* 之间的对应关系，首先根据一个字词 *Term*，产生该字词 *Term* 的 *TermID*，同时根据每一个字词 *Term* 的拼音 *Term* 得到和这个拼音相同的所有词 *Term*，保存这些词 *Term* 的 *TermID List*，这样做可以同时解决汉字检索和同音词检索问题，或者在错误处理的时候，可以增加相应的处理办法，防止人为利用拼音打字产生的错误，同时也便于以后扩展到拼音检索；Layer2 表示的是 *TermID* 和 *DocID List* 之间的对应关系；Layer3 表示的是 *DocID List* 和 *DocID* 之间的对应关系。

为了设计这种格式，本文采用如下的数据结构，见表 5.1。

索引项列表，通常采用 Hash 表[45][46][47]的方法保存，好处是能够为检索实现提供高的查找效率。所以本文使用 Hash-map 来解决索引项保存，即字符串和整形变量之间的对应关系，比如字词 *Term* 和字词 *TermID* 之间的对应关系就是使用 Hash-map，拼音 *Term* 和 *TermID List* 之间的对应关系也是用 Hash-map。本文使用 Vector 来表示整形变量和字符串之间的对应关系，比如 *DocID* 和 *Doc* 内容就是使用 Vector 数组，*TermID* 和 *Term* 的内容之间就也是使用 Vector 数组。使用 Vector<int*>来表示整形和整形数组之间的对应关系，比如 *DocID* 和 *TermID*

List，*TermID* 和 *DocID List* 之间的对应关系。这样设计的目的是，提高存取时间效率和降低内存消耗率。

表5.1　三层索引结构的数据结构

数据结构	意义及说明
Vector<string>	*DocID -> Doc context*
Hash-map<string, int>	*Term context* (word or character) ->*TermID*
Vector<string>	*TermID -> Term context*
Hash-map<string, int*>	*Pinyin -> TermID*
Vector<int*>	*DocID -> TermID List*
Vector<int*>	*TermID -> DocID List*

5.2.3　三层索引结构的建立

1．词表切分

在把语句按“ 词 ”进行索引之前，首先要解决词的切分问题。在检索系统中，词切分分为三种：

第一种，单个字符为索引单元[48]。它有缺陷，查“ 上海 ”时，会让“ 海上 ”也能匹配。当然，如果在建立索引的时候，考虑到字与字间顺序位置的前后关系，这种切分也是可以考虑的，但是这种切分方式对于检索的工作效率势必会有大的影响。

第二种，自动切分算法，将单词按照二元语法方式切分出来，比如将“ 北京天安门 ”切分成“ 北京 京天 天安 安门 ”。

这样，在查询的时候，无论是查询“ 北京 ”还是查询“ 天安门 ”，将查询词组按同样的规则进行切分：“ 北京 ”、“ 天安 ”和“ 安门 ”，多个关键词之间用“ and ”的关系组合，同样能够正确地映射到相应的索引中。

基于自动切分的最大优点是没有词表维护成本，实现简单。缺点是索引效率低，利用向量空间模型的相似度计算方法，所得到的索引结果的文档排序权重跟实际的情况有偏差。本文系统不考虑这个切分方法。

第三种，基于词表的切分算法。现在比较常用，一般的中文搜索引擎和全文检索系统都会用到基于词表的切分算法。

本文采用了单字和词表切分相结合的方法。使用词表切分，能够更准确地表达检索意义，提高检索效率，其中词表切分采用的是本文第 3 章所说的分词方法。为了解决简称方面的搜索，考虑到用简称检索时候的跳字和跳词现象，可参考表 2.2，本文必须辅助用到单字检索以保证查询的召回率，所以本文采用单字和词表结合的方法来建立索引。同时，为了解决多音字问题，本文加入了拼音检索的功能。

2．索引建立的步骤

当语言分析的切分词问题解决之后，索引文件的建立就相对比较简单了，算法具体描述如图 5.3。

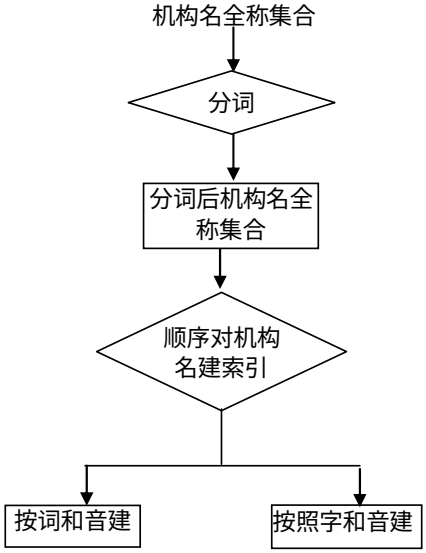

图 5.3　索引建立算法流程图

5.3　检索模型

在信息检索领域，具有代表性的文本检索模型主要有布尔模型(Boolean Model)、向量空间模型(Vector Space Model，简称 VSM)、概率模型（Probabilistic Model）等。由于本文没有使用到任何概率信息，本文只着重介绍 Boolean 模型和 VSM 模型。

5.3.1　Boolean模型

布尔检索模型[49]是最早也是最简单的一种检索模型，其理论已基本成熟。

除了传统的书目型检索系统外，目前有很多成功的全文检索系统也是采用布尔逻辑模型为其基本的检索技术的。

到目前为止，布尔模型是最常用的检索模型。由于查询简单，因此容易理解；通过使用复杂的布尔表达式，可以很方便地控制查询结果；相当有效的实现方法，相当于识别包含了一个某个特定索引项的文档；同时也很容易把用户查询传化为布尔查询式。

但是，布尔模型被认为是功能最弱的方式，其主要问题在于：不支持部分匹配，而完全匹配会导致太多或者太少的结果文档被返回；同时不对输出进行排序，很难控制检索的文档数量，很难表示用户复杂的需求；也不考虑索引词的权重，所有文档都以相同的方式和查询相匹配，对用户的满意度也不加区分。

5.3.2　VSM模型

Gerard Salton 等人[50][51][52]在上世纪 60 年代提出的向量空间模型（Vector Space Mode，VSM）将文档和用户查询式转化为向量形式，用 TFIDF[53]（Term-Frequency Inverse-Document-Frequency）将文档转化为向量形式，从而将信息检索问题转化为向量空间的向量匹配问题，再通过相关度的计算，利用倒排文档结构进行索引，从而使用户得到一个清晰的检索结果。该模型成功应用于 SMART[54]文本检索系统，这一系统理论框架到现在仍然是信息检索技术研究的基础。

向量模型的优点在于：术语权重的算法提高了检索的性能；采用部分匹配的策略使得检索的结果文档集更能接近用户的检索需求；可以根据结果文档对于查询串的相关度对结果文档进行排序；有利于用户用非结构化的形式表达其检索需求，有利于对模糊需求的表达；用户可按照自己的需求对检索词加权并设定阈值，具有对检索的文献数量和质量的控制权。

缺点是：返回大量无关信息，大大影响查准率；对用户的需求表达不是很准确，也没有清晰的逻辑层次；而且计算量都非常大，算法复杂，对计算机的存储量、运算速度及软件水平都有较高要求。

5.3.3　改进的Boolean检索模型

根据 5.3.1 和 5.3.2 两节所描述，本文采用的是 Boolean 模型和 VSM 模型的结合，就是改进的 Boolean 检索模型[55]。它能够把这两个模型的优

点相结合。Boolean 模型可以和 VSM 模型相结合，先做布尔过滤，然后进行排序。首先进行布尔查询，将全部满足布尔查询的文档汇集成一个文档，用向量空间法对布尔检索结果进行排序，即用 Boolean 模型检索，用 VSM 模型打分。

5.3.4　具体检索模型算法

考虑到领域的独特性，本文在使用 VSM 模型时，不使用文档的相似度算法，因为相似度算法计算量相当复杂，而且在本文中也不需要，本文只采用索引项权重模式。

由于不采用向量的相似度算法，算法中会不考虑查询项的顺序问题。但是如果不加入顺序信息，则查询“ 清华 ”时，“ 清华 ”和“ 华清 ”得分相同。故本文加入顺序信息，见公式 5-1。

1．顺序信息

$$order_{IF} = \frac{Num_{order}}{Num_{IP}} \tag{5-1}$$

公式 5-1，计算 *strInput* 经过分词后各部分在 *strFull* 中出现的顺序得分。*strInput* 表示待查简称，*strFull* 表示待匹配的全称串。Num_{IP} 为 *strInput* 经过分词后词数，Num_{order} 表示顺序因子，初始时等于 Num_{IP}，出现一次逆序，则 Num_{order} 减 1。$order_{IF}$ 介于 0 和 1 之间，它用来降低倒序得分，查询“ 清华 ”时，“ 清华 ”和“ 华清 ”得分不同就是通过该公式实现的。

2．索引项频率（Term Frequency）

在逻辑文档中，索引项的重要程度和它在逻辑文档中的出现次数是成正比的，用 tf_{ik} 表示索引项的出现频率，它是衡量一个索引项频率的指示器，定义如下：

tf_{ik} = 文档 k 中索引项 i 的出现次数

在本文的机构名的检索系统中，一般情况下索引项频率为 1，因为一个机构名中，一般只会含有一个这样的索引项。但是也不排除多次出现索引项的情况，同时也为了便于扩充到其他领域的检索系统，所以本文仍然使用索引项频率。

由于本文同时利用词和字建索引，在检索的时候也是先利用词再利用字检索，就会碰到一个问题：比如用“ 东电股份 ”来检索，在数据库中有“ 广西桂

东电力股份有限公司 ” 和 “ 浙江东南发电股份有限公司”，如果利用单字检索，把 “ 东电”分为 “ 东”和 “ 电 ”，如果利用上面的公司会导致用 “ 东”和 “ 电 ” 检索时，比用 “ 东电”检索得分高。所以为了使检索算法能在同一纬度下比较，本文改用下面公式：

$$TF_{ik} = tf_{ik} \cdot N_i \tag{5-2}$$

N_i 表示索引项 i 中的字符数。这样能够把索引项字数也考虑在内，不会出现上文所述的问题。

3．逆向文档频率（Inverse Document Frequency）

包含该索引项的文档数越多，则该索引项用于区别文档的能力就越低。文档频率（Document Frequency）的定义如下：

df_i = 索引项 i 的文档频率

= 包含索引项 i 的文档数目

idf_i = 索引项 i 的逆向文档频率

= $\log(N_{Doc}/df_i)$　（N_{Doc}：文档总数）

这是一个表示索引项 i 区分能力高低的指标，表明不同索引项 i 对文档的区分能力大小，和索引项频率（Term Frequency）共同决定索引项的区分能力。

df_i 越高，意味着索引项 T_i 在衡量文档之间相似度方面的作用越低；idf_i 越高，意味着索引项 T_i 对于文档的区别作用越大。如果一个索引项 T_i 仅出现在一个文档中，则 $idf_i=\log(N_{Doc})$；如果一个索引项 T_i 出现在所有文档中，则 $idf_i=\log(1)=0$。

4．索引项权重(TF-IDF Weighting)

用 w_{ik} 表示索引项 T_i 对文档 D_k 的重要程度。如下：

$$w_{ik} = tf_{ik} \cdot N_i \cdot \log(N_{doc} / df_i) \tag{5-3}$$

5．所以文档 D_k 和检索串的之间的匹配得分是：

$$score_k = order_{IF} \cdot \sum_{Ti \in k} \{w_{ik}\} \tag{5-4}$$

公式 5-4 计算 *strInput* 匹配文档 D_k 时的索引得分，它既考虑到检索串的顺序信息，又兼顾了各检索词的权重。$order_{IF}$ 表示 *strInput* 在文档 D_k 中的顺序得分，

T_i是 *strInput* 分词后的第 i 个词，也就是第 i 个索引项，w_{ik}表示索引项 T_i对文档 D_k的重要程度。

5.4　实时检索算法

如何在检索系统中综合使用单字检索和词检索，本文做了很多试验。

如果先用词检索，只有当词索引项检索不到的时候，才使用字检索，则会出现如下问题：例如，“ 中国国贸 ”检索不到“ 中国国际贸易有限公司 ”，“ 中国 ”和“ 国贸 ”两个词索引项都能查到，用“ 中国石化 ”希望检索“ 中国石油化工有限公司 ”，却只能检索到“ 中国石化齐鲁石油化工有限公司 ”，因为“ 中国 ”和“ 石化 ”都在后者出现。这种实时检索算法会导致检索的召回率和准确性不高。

所以本文先对检索串分词，用词来检索；然后用检索串中未登录词切分为单字，再检索，两个集合取并集。如果用字或词都没有检索到的话，就采用拼音检索，如果拼音检索不到，就采用错误处理，减少检索串。算法流程图可以参考图 5.4。

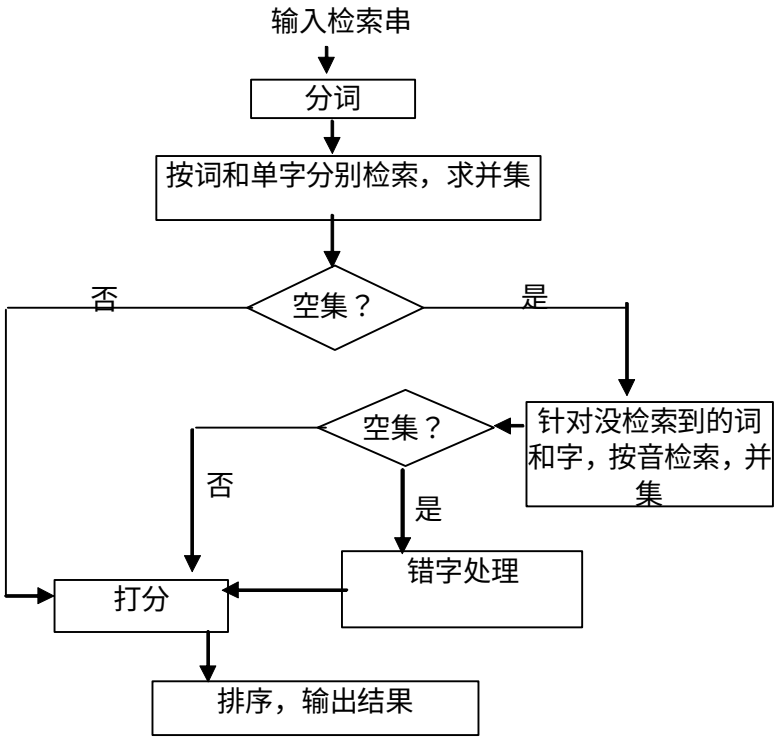

图 5.4　实时检索流程图

5.5　实验结果与分析

5.5.1　实验条件

为了验证本文所提的多级索引算法的性能。本文做了几组对比实验。实验在一台 AMD 1.75GHz 主频，1GB 内存，Windows XP 的机子上运行，程序由 VS.net 2003 设计。

总的测试数据库是 510，000 个公司名全称库。测试输入数据分为三种：

Input1:它包含 1，065 个检索输入，主要是简写名称，这里面不包含字错信息，也就是说通过字词检索肯定能够查到结果的。这个测试为了测试四个算法在检索输入全部正确的情况下的性能。

Input2:它包含 1，065 个检索输入，其中 7%含有错字信息，也就是说按照字词检索是检索不到的，必须容许错字检索。这个测试为了模拟通常情况，输入有时候有错，但是绝大多数情况时正确的汉字输入。

Input3:它包含 1，065 个检索输入，其中 100%含有错字信息。这个测试，是举一个极端的例子，测试在输入串有错的情况下，四个算法的容错能力。

5.5.2　实验结果

本文分别用了四种算法，做了多组对比实验，以说明本文采用方法的效率：

算法 1：使用单个字建索引，检索时候也把输入串切分为单字

算法 2：使用切分词建索引，检索时候，也按照对输入串的词切分来检索

算法 3：使用字和词共同建索引，不包含拼音，不包含错字处理，检索的时候，先按照对输入的词切分检索，如果找不到这个词，则把词切分为单字，再检索。

算法 4：使用字和词，以及拼音同时建索引，包含错字处理，也就是本系统所采用的方法。

本文采用通用的标准来评价信息检索性能，使用召回率和准确率，附加平均检索耗时，如下：

$$召回率 = \frac{检索到的并且相关的文档数}{所有相关的文档数} \times 100\% \qquad (5\text{-}1)$$

$$\text{精确度} = \frac{\text{检索到的并且相关的文档数}}{\text{所有检索到的文档数}} \times 100\% \tag{5-2}$$

$$\text{平均耗时} = \frac{\text{所耗的总时间}}{\text{总的检索次数}} \tag{5-3}$$

表5.2　针对input1的四组对比实验

方法	召回率	精确度	平均耗时
算法 1	100%	30.2%	0.19s
算法 2	9.3%	100.0%	0.07s
算法 3	100%	43.1%	0.12s
算法 4	100%	43.1%	0.12s

表5.3　针对input2的四组对比实验

方法	召回率	精确度	平均耗时
算法 1	80.4%	28.9%	0.17s
算法 2	8.8%	100.0%	0.07s
算法 3	80.5%	32.7%	0.11s
算法 4	100%	39.0%	0.18s

表5.4　针对input3的四组对比实验

方法	召回率	精确度	平均耗时
算法 1，2，3	0	0	-
算法 4	100.0%	15.7%	0.46s

5.5.3　实验结果分析

从表 5.2 和表 5.3，可以看出完全使用单词来检索的时候，虽然时间效率挺高，准确性将近 100%，但是召回率很低，这是因为检索串，一般就 2～4 个字，

不容易被切分，通常情况下，要么就被切分为一个词，但是当找不到这个词的时候，就返回空。同时，很多简称输入中简称属于不连续速写，单独用词检索有问题。

从表 5.2 和表 5.3，可以看出，单纯使用单字检索比使用单词检索具有更高的召回率，但是准确率比较低，检索的时间效率也不高。

从表 5.2 和表 5.3，使用字词检索，不包括错字拼音处理的算法，相对于单字检索来说，都具有更好的准确率和召回率。

使用字词错检索，从表 5.2 结果中看，其和字词检索相同，因为本文只有在查询不到的情况下，才使用字词错检索，而针对 Input1 测试库，肯定能够查询到所需内容，所以错误处理就不被使用。从表 5.3 结果看，虽然时间效率不如字词处理，但是它比字词检索具有更好的召回率和准确率。从表 5.4 结果看，在有错字的情况下，只有使用字词错检索才能解决问题。

综合以上分析，本文发现采用字词检索算法具有更好的性能，如果附带错误处理，可以解决错误输入问题。

5.6　本章小结

本章介绍了索引技术在信息检索中的作用，然后介绍本文采用的多级索引结构，以及如何建立这种索引，接着介绍了本文所采用的检索模型，它解决了给检索结果排序的问题，随后介绍具体的检索算法，给出实时检索的流程图。为了验证本文的多级索引算法的有效性，本文做了几组对比实验，根据实验结果，本文采用的字词检索和错误处理，具有较好的效果。

第六章　中文机构名检索系统的设计及性能

6.1　总体架构

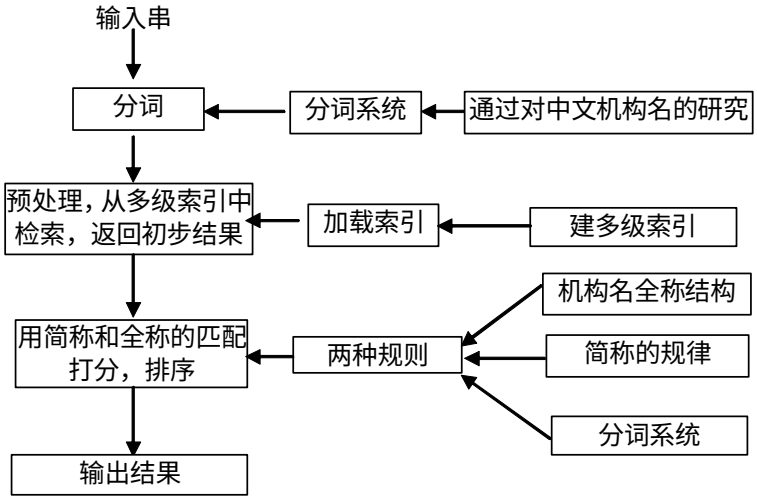

图 6.1　检索系统的总框架图

检索算法分为两个主要部分：（1）离线的准备；（2）实时的检索。基于此，共有三个模块，即规则生成，多级索引的建立以及检索，以及简称全称的匹配算法，见图一。

规则生成，是离线准备的一部分，它定制一个分词系统，为其他模块提供分词结果，并生成机构名全称规则和简称规则，为全称和简称的匹配做准备。多级索引建立及检索模块有两个部分，建立多级索引也属于离线准备，它对中文机构名全称按照词、字、音的顺序分别建索引；第二部分是多级索引中进行检索，是实时检索的重要步骤，它对输入串分词，分别通过词、字、音来检索，并打分排序。简称和全称匹配模块则对多级索引检索后返回的结果，进行打分排序，并输出最终结果。

6.2　实验结果及分析

6.2.1　实验条件

实验在 CPU 为 AMD 1.75GHz 主频，1GB 内存的计算机上进行，测试数据库含有 51 万个公司名称。

6.2.2　实验结果

基准系统是用 BM 模糊匹配算法[56]实现的一个检索系统，该系统扩充了拼音检索方法，以使其具有和本文所提系统相同的功能。它计算的是一个字符串包含有一个字串的得分，得分最高的是检索结果。

本文分别在人为随机简称输入和官方简称输入两类测试集上做了对比试验：

第一组，人为随机简称输入测试集，这个主要测试系统的通用性和准确性。测试时由 5 个测试者分别随机产生约 200 个公司名的简称，共计 1，065 个简称，按每行一个存在一个简称文件中；把它们实际对应的全称存于全称文件。简称文件作为系统输入，分别送入基准系统和用本文方法构筑的系统（简称本文系统），分别得到检索结果文件，将之于全称文件比较，可分别得到基准系统和本文系统的性能。首选错误率和首选正确率分别用公式 4-6 和 4-7 计算。批处理的平均耗时用以衡量系统的效率，用公式 5-3 来计算。

实验结果是：基准系统的首选错误率为 8.07%，本文系统的首选错误率为 5.35%，错误率相对于基准系统下降了 33.7%；基准系统的平均耗时是 0.68 秒，本文系统的平均耗时是 0.21 秒，优于基准系统。

第二组，官方简称输入测试集，主要测试系统的准确性。本文用上海证券交易所的上市公司的股票名称库做实验。股票名称被认为是公司名的官方简称，一共 888 个。同时把对应的上市公司名称全部加入总数据库中，测试方法同第一组。

实验结果是：基准系统的首选错误率为 10.90%，本文系统的首先错误率在 4.73%，错误率相对于基准系统下降了 56.6%。在时间效率上，两个系统分别与第一组相当。

6.2.3　结果分析

从上面两组实验我们可以看出，无论是人为随机简称输入测试集还是官方简称输入测试集，本文系统总体性能均好于基准系统。

从准确性上看，BM 模糊匹配方法是计算一个字符串包含输入串的得分，当输入串刚好是机构名中的子字符串时，查询的准确性很好。但在判断输入串最可能是哪个机构名的简称时，BM 模糊匹配算法没有有效的措施。比如，输入“ 中石化 ”会查到“ 中国石化齐鲁石化有限公司 ”，而不是“ 中国石油化工有限公司 ”。所以，BM 模糊匹配方法在判断一个简称和全称是否匹配上，性能不是很好。而本文的方法弥补了这一点。

从第二组实验上，可以看出本文系统准确性显著优于基准系统。我们发现，相对于人为随机简称输入，官方简称输入有更多的跳字、跳词、缩略现象。本文系统能更有效得解决跳字、跳词、缩略问题，故本文所提的简称和全称匹配算法能更好得解决简称检索全称问题。

从搜索效率上看，BM 模糊匹配，需要全字串匹配，和本文系统有较大差距。本文系统所用的是字词音相结合的多层索引算法，在检索时间效率上要大大优于不使用索引算法的匹配系统。

但是，本文系统也存在一些问题：

1. 本文所用的测试数据库集只有 51 万，如果数据库更大，那么建索引和实时检索时，对内存的需要不是一台计算机可以完成的，这个时候，需要解决几个问题：第一个，是多台计算机并发操作，以提高建索引和实时检索的效率；第二个，索引结构要是增量式的；第三个，对于倒排索引文件必须压缩，以降低对硬盘和内存的需要。
2. 字词音相结合的多层索引结构，其检索时间效率不如单词的检索时间效率和精确度。同时，这种索引结构肯定比单字和单词索引结构耗费更大的存贮空间。
3. 本文所针对的领域是机构名的简称问题，如果涉及其他领域，比如全文检索系统，可能就不需要字词音相结合的索引结构。

但是，总的来说，在本文所涉及的特定研究领域，本文系统能够提高查询的准确性，并且检索效率也达到了实用的标准。

6.3　本章小结

本章给出了中文机构名检索系统的具体框架，并做了详细的介绍。为了验证本文系统的重要性，本文设置了一个基准系统，分别针对人为随机简称输入测试集和官方简称输入测试集，做了对比实验。根据实验结果，在这两个测试集上本文的检索系统总的性能均好于基准系统。

第 7 章　结论

7.1　研究总结

本文是在广泛分析国内外相关研究的基础上，对如何有效地实现中文机构名简称检索系统进行了深入的分析和研究，再设计实现。本文针对基于中文机构名简称的检索方法，研究了机构名的结构特征，总结出两种规则，定制了一个基于关键词类的分词工具，提出简称和全称匹配的一种算法，并且结合多级索引技术，实现了基于中文机构名简称的检索系统。主要研究工作包括：

1．研究了中文机构名的组成结构并形式化描述了其结构；
2．研究了如何描述人为简称的习惯，并可以通过计算机学习，生成一个机构名全称的可能简称；
3．定制了一个基于关键词类的带标注的中文分词系统，解决了该限定领域的分词问题，便于对机构名的成分进行划分；
4．在匹配输入串和机构名的全称时，结合字符串匹配和字段匹配算法，研究了新的算法，提高了简称输入和结构名全称的匹配准确度；
5．引入了多级索引结构，在不降低准确性的前提下，提高了检索效率，同时利用单字和词，拼音的结合来检索，提高了搜索精度。

实验结果表明，本文系统检索准确性较好，查询效率也已经达到实用系统的需要。

7.2　需进一步开展的工作

下一步的研究重点将是改善检索算法，把本文的方法扩展到其他类似的专名查询中去，实现一个的特殊领域搜索引擎，比如黄页查询系统。

在索引算法上，提供增量索引的支持，快速索引的支持，解决倒排文件的压缩问题，以及采用并行技术提高检索效率。

参考文献

[1] Sun J., Gao J. F., Zhang L., Zhou M Huang. Chinese Named Entity Identification Using Class-based Language Model. Proc of the 19th International Conference on Computational Linguistics, Taipei, 2002. 967-973

[2] 张艳丽，黄德根，等．统计和规则相结合的中文机构名称识别．自然语言理解与机器翻译，清华大学出版社，2001，233-239

[3] 张小衡，王玲玲．中文机构名称的识别与分析．中文信息学报，1997，1(4)：21-32

[4] 王宁，葛瑞芳，等．中文金融新闻中公司名的识别．中文信息学报，2002，16(2)：1-6

[5] Smith T F, Waterman M S. Identification of Common Molecular Subsequences. Journal of Molecular Biology, 1981. 147:195-197

[6] Hernandez M, Stolfo S. The Merge/purge Problem for Large Databases. Proc. of ACM SIGMOD Int. Conference on Management of Data, 1995. 127-138

[7] Monge A E, Elkan C P. The Field Matching Problem: Algorithms and Applications. Proc. of the 2nd Int. Conference on Knowledge Discovery and Data Mining, 1996. 267-270

[8] 陈挺，郭颖，等．中文字段匹配算法．计算机工程，2003，29（13）：118-124

[9] 王兴义．基于模式匹配的中文专有名词识别：[硕士学位论文]．山西：山西大学计算机与信息技术学院，2005

[10] 雷静．汉语机构名的构成模式．第七届全国计算语言学会议论文，2003．91-96

[11] 吴雪军．面向信息抽取的命名实体识别与模板获取技术研究：[硕士学位论文]．辽宁：东北大学信息科学与工程学院，2005

[12] 吴栋，滕育平．中文信息检索引擎中的分词与检索技术．计算机应用，2004，24(7)：128-131

[13] 杨超．基于最大匹配的书面汉语自动分词研究：[硕士学位论文]．湖南：湖南大学计算机与通信学院，2005

[14] 孙茂松，邹嘉彦．汉语自动分词研究评述．当代语言学，2001，3(1)：22-32

[15] 黄德根，朱和合，等．基于最长次长匹配的汉语自动分词．大连理工大学学报，1999，39(6)：831-835

[16] 李振星，徐泽平，等．全二分最大匹配快速分词算法．计算机工程与应用，2002，38(11)：106-109

[17] 王开铸，李俊杰，等．无词典自动分词的研究．计算机语言学进展与应用，北京：清华大学出版社，1995

[18] 刘挺，吴岩，等．最大概率分词问题及其解法．哈尔滨工业大学学报，1998，30(6)：37-41

[19] 韩客松，王永成，等．汉语语言的无词典分词模型系统．计算机应用研究，1999，16(10)：8-9

[20] 李家福，张亚非．基于 EM 算法的汉语自动分词方法．情报学报，2002，21(3)：269-272

[21] 孙茂松，肖明，等．基于无指导学习策略的无词表条件下的汉语自动分词．计算机学报，2004，27(6)：736-742

[22] Lai, B.Y., Sun M. S., et al. Chinese word segmentation and part-of-speech tagging in one step. Proceedings of International Conference: Research on Computational Linguistics, Taipei, l997. 229-236

[23] Zhang H. P, Yu H.K., Xiong D.Y, et al. HHMM-based Chinese lexical analyzer ICTCLAS. 41st Annual Meeting of the Association for Computational Linguistics, Sapporo, 2003. 184-187

[24] 刘挺，吴岩，等．串频统计和词匹配相结合的汉语自动分词系统．中文信息学报，1998，12(1)：17-25

[25] 张华平，刘群，等．中文词语一体化分析 ICTCLAS．中科院计算所，中文自然语言处理开放平台 http://www.nlp.org.cn，2002

[26] 北大分词系统，http://www.icl.pku.edu.cn/icl_res/segtag98/

[27] Sun, M.S., Benjamin K. T. Ambiguity resolution in Chinese word segmentation. Proceedings of the 10th Asia Conference on Language Information and Computation, Hong Kong, 1995. 121-126

[28] 张俊盛，陈舜德，等．多语料库作法之中文姓名辨识．中文信息学报，1992，6(3)：7-15

[29] 宋柔，朱宏，等．基于语料库和规则库的人名识别法．计算语言学研究与应用．北京：北京语言学院出版社，1993．150-154

[30] 孙茂松，黄昌宁，等．中文姓名的自动辨识．中文信息学报，1995，9(2)：16-27

[31] 孙茂松，张维杰．英语姓名译名的自动识别．计算语言学研究与应用．北京：北京语言学院出版社，1993．144-149

[32] 沈达阳，孙茂松．中国地名的自动辨识．计算语言学进展与应用，1995．68-74

[33] Chen H.H., Lee J.C. The identification of organization names in Chinese texts. Communications of COLIPS, 1994. 131-142

[34] Chen K.J., Chen C.J. Knowledge Extraction for identification of Chinese Organization Names. ACL workshop on Chinese Language Processing, 2000. 15-21

[35] 陈儒 面向短信过滤的中文信息模糊匹配技术：[本科学位论文]. 辽宁：哈尔滨工业大学计算机科学与技术学院，2003

[36] 王静帆，中文模糊信息检索系统的模糊匹配算法研究：[本科学位论文]. 北京：清华大学计算机科学与技术系，2005

[37] Knuth D E, Morris J H, Pratt V R. Fast pattern matching in strings. SIAM Journal on Computing, 1977. 6(2):323-350

[38] BOYER R.S, MOORE J.S. A fast string searching algorithm. Communications of the ACM, 1977. 20(10):762-772.

[39] Sunday D.M. A very fast substring search algorithm. Communications of the ACM, 1990. 33(8):132-142

[40] Aho A, Corasick M. Efficient String Matching: An Aid to Bibliographic Search. Communications of the ACM, 1975. 18(6):333-343

[41] Vintsyuk, TK. Speech discrimination by dynamic programming. Kibernetika, 1968, 4(1):81-88

[42] UKKONEN, E. Finding approximate patterns in strings. J. Algorithms, 1985, 6(1): 132-137

[43] Wu, S., Manber, U. Fast text searching: allowing errors, Communications of the ACM, 1992, 35(10): 83-91

[44] Levenshtein V.I. Binary codes capable of correcting deletions, insertions and reversals. Sov. Phys. Dokl, 1966. 707~710

[45] Knott, G. D. Hasing Functions. Computer Journal, 1975, 18(3):265-278

[46] Knuth, D. The Art of Computer Programming: Sorting and Searching,. Reading, Mass.: Addison-Wesley, 1973, 3

[47] Larson, P.A. Linear Hashing with Partial Expansions. VLDB, Montreal, 1980. 6: 224-32

[48] 余海燕，张仲义. 基于单汉字索引的全文检索系统的优化研究. 中文信息学报，1996，15(3)：98-106

[49] Yan, T. W., Garca-Molina, H. Index Structures for Selective Dissemination of Information Under the Boolean Model, ACM Transactions on Database Systems (TODS), 1994, 19(2):332--364

[50] Salton G., Wong A. On the Specification of Team Value in Automatic Indexing. Journal of Documentation, 1973, 29(4): 351~372

[51] Salton G. Automatic Text Processing. Addision-Wesley, Reading, Mass, 1989

[52] Frakes, W.B., Baeza-Yates, R., eds. Information Retrieval: Data Structures and Algorithms, Prentice Hall, Englewood Cliffs, New Jersey, 1992

[53] Aho, A., Hopcroft, J., and Ullman, J. The Design and Analysis of Computer Algorithms. Addison-Wesley, Reading, Mass, 1974

[54] Salton, G., The SMART Retrieval System – Experiments in Automatic Document Processing. Prentice-Hall, Englewook Cliffs, New Jersey, 1971

[55] Salton, G., Fox, EA, Wu, H. Extended Boolean information retrieval. Communications of the ACM, 1983, 26(11):1022-1036

[56] Tarhio, J., Ukkonen, E. Approximate Boyer-Moore string matching. SIAM Journal on Computing, April 1993,22(2):243-260

致　谢

衷心感谢导师郑方研究员对本人的精心指导。他的渊博的知识、严谨的治学态度、宽厚的学者风范，以及言传身教将使我终生受益。

感谢实验室的方棣棠教授、吴文虎教授和徐明星副教授、邬晓钧老师的良好建议，感谢实验室的各位同学在工作中的帮助和支持。

感谢所有关心和支持我的朋友。

声　明

本人郑重声明：所呈交的学位论文，是本人在导师指导下，独立进行研究工作所取得的成果。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含任何他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。

签　名：_____日　期：_____

个人简历、在学期间发表的学术论文与研究成果

个人简历

1982 年 10 月 8 日出生于安徽省含山县。

1999 年 9 月考入北京信息工程学院计算机科学与技术系计算机科学与工程专业，2003 年 7 月本科毕业并获得工学学士学位。

2003 年 9 月考入清华大学计算机科学与技术系攻读计算机应用硕士至今。

发表的学术论文

[1] 钟良伍，郑方，基于中文机构名简称的检索方法研究，中文信息学报，已录用