中南大学

硕士学位论文

中文分词在聊天机器人中的应用研究

姓名：李鹏

申请学位级别：硕士

专业：计算机应用技术

指导教师：廖志芳

20090501

摘 要

随着社会的日益信息化，人们希望能用自然语言与计算机交流，聊天机器人就是这样一种通过自然语言同人进行交流的人机对话系统。ALICE 是一个基于经验的人工智能聊天机器人，由于它在问句查询推理过程中是以词为单位的，而中文句子中词与词之间没有明显的分隔符，所以它不能支持中文聊天。中文分词的功能就是把中文句子的汉字序列切分成有意义的词序列，因此分词技术是支持中文的智能聊天系统的一个关键技术。

本文紧紧抓住分词技术在聊天系统中的基础性地位进行研究，通过分析聊天系统中用户输入的特点选择适合支持中文的智能聊天系统的分词算法。在对现有的分词技术进行深入分析的基础上，结合中文字词的特点，提出了一种改进的分词词典结构。该结构的基本原理是以首字为索引建立首字 hash 表，将同一首字下不同长度的词分类组织在不同词表内，各词表内的词条按顺序排列，基于此词典结构，设计了相应的双向最大匹配分词算法。由于词典结构支持首字 hash 查找和二分查找，分词算法的时间复杂度是很低的。

通过对 ALICE 聊天机器人原型系统进行研究分析，针对其不支持中文聊天的缺陷，论文介绍了在系统整体框架不变的前提下，向其推理机制中加入中文分词功能，使其在处理中文句子时也像西方语言一样以词为单位进行查询推理，从而在和用户交互的过程中具有中文聊天的功能。

本文通过研究分词技术及其在智能聊天机器人中的应用，将中文分词模块集成到现有的聊天机器人中，初步实现了聊天机器人的中文聊天。

关键词： 聊天机器人；中文分词；分词词典；hash 查找；二分查找

ABSTRACT

Along with the rapid development of informationization, scientists dedicate themselves to communicating with computers in an exclusively natural language. A chat robot is an intellectual system which enables the function of communicating with human race via natural language. ALICE is an experience based artificial intellectual chatting robot. During the process of question searching, since it uses words as elementary bases, and according to the fact that there are no distinct boundaries between the words in Chinese sentences, ALICE is failed to chat with humans beings in Chinese language. The function of the Chinese word segmentation is to syncopate the characters used in Chinese sentences into several word sequences which carry different and certain meanings. Therefore, the principle technology which enables Chinese intellectual chatting system can be exclusively attributed to segmentation technology.

This paper adopts the word segmentation technology which achieves the dominant position in chatting system. Furthermore, by the means of analyzing the visible pattern of user's inputs, this paper chooses appropriate segmentation algorithm to enable Chinese chatting system. Based on existing segmentation technology and features of Chinese words, a newly developed word segmentation dictionary structure is proposed in this paper. The fundamental procedures of this structure is to build a hash table of the selected initial word and then, relocates the words with different length in corresponding tables and consecutive orders. Under the structure of the dictionary, a bidirectional matching segmentation algorithm is also put forward in this paper. Due to the capacity of this proposed dictionary structure, the time complexity of

segmentation algorithm is rather low.

In depth analysis of the prototype of ALICE chatting robot system and its drawbacks, on premise of integrality of the existing solution, this paper joins the Chinese segmentation technique to the reasoning mechanism so as to process the Chinese sentences in the way of western languages which adopt the words as basic unit to perform searching and reasoning. This method enables the chatting robot communicating with human users in Chinese.

This paper carefully examines the application of segmentation technology in the intellectual chat robot, integrates Chinese word segmentation module into the ongoing chatting robot and implements ALICE chatting in Chinese language.

KEY WORDS: chat robot, Chinese word segmentation, word segmentation dictionary, hashing operation, binary search

原创性声明

本人声明，所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了论文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得中南大学或其他单位的学位或证书而使用过的材料。与我共同工作的同志对本研究所作的贡献均已在论文中作了明确的说明。

作者签名：_____ 日期：09 年 5 月 25 日

学位论文版权使用授权书

本人了解中南大学有关保留、使用学位论文的规定，即：学校有权保留学位论文并根据国家或湖南省有关部门规定送交学位论文，允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，可以采用复印、缩印或其它手段保存学位论文。同时授权中国科学技术信息研究所将本学位论文收录到《中国学位论文全文数据库》，并通过网络向社会公众提供信息服务。

作者签名：_____ 导师签名_____ 日期：09 年 5 月 25 日

第一章 绪论

1.1 聊天机器人简介

1.1.1 聊天机器人定义

聊天机器人，是这样一段程序，通过自然语言，以聊天的方式与人交互，它模仿人的语言习惯，给出较为人性化的答案。聊天机器人系统是伴随着“图灵测试”诞生的，在过去的几十年里，许多学者在这个研究方向上做出了许多有意义的探索。

1950 年，著名的英国数学家图灵(A.M.Turing)发表的论文《Computing Machinery and Intelligence》中，他第一次提出了“机器智能”的概念，并提出判断计算机是否具备智能的实验方法——“图灵测试”，也就是通过自然语言问答的方式，判断计算机是否具备人的智能。其具体方案为：由测试人“甲”和另一间房中的两个对象“乙”和“丙”对话，“乙”和“丙”两个中一个是人，另一个是计算机。如果经过一段时间的对话之后，“甲”不能判断出“乙”和“丙”哪个是人，哪个是计算机，则可以认为计算机已经具备了人一样的智能[1]。

由于计算机能否具备和如何具备人的智能，是计算机科学中的一个根本问题，所以历来争论都非常激烈。John Searle 认为：非生物智能机的概念是不正确的[2]。但是他也否定不了图灵测试。图灵测试有着重要的意义，它的难度很大，要做好图灵测试，仅仅编出一个“有智能的”程序是不够的。现在有一种误解，认为只要计算机在某些方面做得很好，很有智能，甚至让人看不出这是计算机做的，就算是通过了图灵测试。这种看法是不正确的，实际上，计算机在计算方面早就做得比人类好了。图灵测试就其本质来说，测试的是一种特定的智能，称为“言语智能”，而言语智能要求计算机必须具备一般人的常识。

1.1.2 几个典型的聊天机器人

1) Eliza[3]

Jaseph Weizenbaum 在 1956 年创造的“Eliza”，是第一个聊天机器人。Eliza 扮演一个心理学者的角色，具有这个方面的一些特定领域知识，经常用反问的方式来对用户提问。Eliza 的原理是，根据用户输入语句中的关键词搜索保存在数据文件中的知识，找到与之匹配的回答。但是 Eliza 一个明显的弱点是当发现用户输入中的一个关键词便立刻寻求回答，而不继续寻找用户句子中的其它关键词，这样使得一些优先级别比较低的模式几乎永远不会被匹配到。

2) Claude

Brian Mclaughlin 所实现的 Claude 也是一个使用标准的模式匹配寻找合适回答的聊天机器人，通过将用户输入的语句进行拆分，分别找到匹配后再构造回答。然而 Claude 注重了回答语句语法以及语义的正确性，入通过分类规定哪些动词可以和哪类名词结合，使得内容很有限而且很多回答都毫无意义。

3) Cyber Ivar

Cyber Ivar 是 Jaczone 公司开发的一个聊天机器人。测试结果表明，Cyber Ivar 响应速度快，在回答 UML、WayPointer 和 Jaczone 之类的问题时，Cyber Iavr 给出的答案相当准确、全面；对于常识性问题，给出的答案也比较贴切。令人惊异的是，对于用户提出的问题“Who is Maozedong?”，Cyber Ivar 竟然回答：“He was a Chinese communist who reigned from 1949 until his death in 1983.”。在 Cyber Ivar 的知识库中，竟然有这么一条关于“Maozedong”的知识，虽然这条知识是错误的，但是同时说明了 Cyber Ivar 的知识库的强大。

4) ALICE

ALICE 是由 Richard S.Wallace 开发的。它使用 Java 作为引擎对用户的输入进行分析，在知识库中寻找最适合的回答并返回给用户。由于 ALICE 定义了丰富的标签，所以它具有各种拟人的智能。

ALICE 并没有采取复杂的算法，事实上，ALICE 有 40000 多个模板，也是用了模式匹配的方法来检索最合适的答案。但 ALICE 采用了一种很好的扩充机制，AIML 文件可以进行内联，许多包含特殊领域知识的 AIML 文件可以方便地合并成一个更大的知识库。

所有这些聊天机器人程序几乎全部采用模式匹配的方法，来寻找问题最合适的答案。它们有一个共同的特点，那就是在与用户的交谈过程中，都是基于谈话技巧和程序技巧，而不是根据常识。在它们的知识库中，可以存放多个句型、模板，但几乎都没有常识库。

1.1.3 聊天机器人的缺陷

由于 ALICE 具有良好的设计，在和用户聊天过程中效果逼真，曾在 2000 年和 2001 年两度获得著名的 Loebner 奖，具有很强的代表性，所以本文研究的聊天机器人系统以 ALICE 为基础，后续章节会对其模型进行阐述。因为 ALICE 在构架知识树和读取用户问句进行推理时，都是先把句子分成一个一个的词，再进行下一步处理的。在处理英语、法语、德语等西方语言方面取得了很大的成功[4]，但是在处理汉语这样的孤立语方面，却存在着天然的缺陷[5]：

1) ALICE 不能处理汉语问答：ALICE 以词作为处理的最小单位，英语等西方语言词与词之间有空格标记，因此在处理西方语言对话方面相对容易，但是在

汉语句子中是以单个汉字作为最小单位，词和词之间没有明显的分隔标记[6]。

2) ALICE 的推理过程是一个严格的匹配过程，但是汉语虚词运用较多，语序比较自由。

鉴于 ALICE 在处理时是以词为处理的基本单位，而汉语不像西方语言，词与词之间有空格作为标记，为了实现基于 ALICE 的聊天机器人，首要解决的问题就是对中文句子进行分词处理。

1.2 中文分词研究现状

中文分词是中文信息处理的基础[7,8]，在中文信息处理系统中具有广泛的应用前景。

1.2.1 问题的提出

计算机不能像人一样直接处理或识别自然语言，只能通过人们编写的应用程序来处理。这些应用程序的目标是能使计算机快速正确地处理自然语言。在当今信息化的社会，人们需要通过计算机来传递信息，而计算机在处理信息时还无法解决人机交互的困难。信息处理程序就是为了解决这些困难而产生的。

在西方国家，信息处理技术已经发展得比较成熟。而中文信息处理技术由于起步晚，再加上汉语句子中词与词之间不像西方语言那样用空格隔开，因此当计算机处理中文时，更加深了人机之间对自然语言理解的差距。基于此，西方语言信息处理技术无法直接应用到中文信息处理技术中去。

中文分词技术的发展直接影响着中文信息处理技术的发展[9]。我国自八十年代初期开始重视研究自动分词技术以来，分词技术得到了很大的发展，提出了很多分词算法和分词模型，开发出了多重分词软件。而随着中文信息处理技术的广泛使用，其对分词技术的要求也越来越高。

1.2.2 研究意义

中文分词能让计算机快速准确的处理中文信息，是进行中文信息处理的基础。在中文信息处理领域，对输入文本进行句法分析是一项必不可少的处理任务。计算机从事句法分析所凭借的语法信息来自机器词典和句法规则库。机器词典收录了每个词条的语法、句法和语义知识，句法规则一般是在词类知识的基础上构造的。因此，处理中文信息时，对汉语句子必须先进行分词处理，才能进行句法分析。如果对输入的源文件的句子没有进行分词，还是一些字符序列，这样就不能根据句子中出现的每个具体的词到机器词典中去查找相应的语言知识；如果不知道每个词的词性等词汇知识，就不可能直接调用句法规则来判断句子的句法结构。

中文信息处理的目标就是让计算机真正理解人类的语言，只有机器理解了人类的语言文字，才可能使人与机器的交流成为现实。而在语言文字中词是最小的能够独立活动的有意义的成份。从现阶段的实际情况来看，英文由于其语言自身的特性，不需要分词，在词的利用上已经领先我们，并且展现了良好的应用前景，无论是信息检索还是主题分析的研究都要强于中文，究其根本原因就是汉语要通过分词这道难关，只有攻破了这道难关，才有希望赶上英文在信息领域的发展，因此中文分词的研究意义重大，直接影响到使用汉语的每个人的很多方面。

1.2.3 发展概况

国内外对中文分词的研究取得了很大的进展，很多科研院校都有自己的中文分词研究队伍，以下简单介绍一下几个比较典型的中文分词系统：

CDWS(The Modern Written Chinese Distinguishing Word System)是我国第一个实用性的分词系统，由北京航空航天大学设计与实现的。它采用的分词方法是MM 方法，辅助以词尾字构词检错技术，使用知识库进行纠错。CDWS 的分词精度约为 1/625（人工干预，不考虑多音字构词所引起的分词错误），基本上满足了词频统计和其它一些领域的应用要求。

清华大学先后研制开发了 SEG 分词系统和 SEGTAG 系统。前者提供了带回溯的正向、反向、双向最大匹配法和全切分法，可以由用户选择合适的切分算法。系统首次提出了全切分的概念，即找出输入字串的所有可能的词串，再从所有可能的词串中选出最佳词串序列作为分词结果。后者着眼于将各种各样的信息进行综合，以便最大限度地利用这些信息提高切分精度。系统使用有向图来集成各种各样的信息，这些信息包括切分标志、预切分模式、其它切分单位，该系统的切分精度基本上可达到 99%左右，能够处理未登录词较多的文本。

复旦大学研制的分词系统由四个模块组成：预处理模块（利用隐式标记将文本分割成较短的汉字串）；歧义识别模块（正向最小匹配和逆向最大匹配进行双向扫描）；歧义字段处理模块（利用构词规则和词频统计信息来消除歧义）；未登录词识别模块（解决未登录词造成的分词错误）。该系统对中文姓氏的自动识别达到了 70%的准确率，对文本中的地名和一些领域专有词汇也能进行一定的识别。

哈尔滨工业大学统计分词系统是一种典型的运用统计方法的分词系统，它试图将串频统计和词匹配结合起来。系统由预处理模块、串频统计模块、切分模块三部分构成，能够利用上下文识别大部分生词，解决一部分切分歧义，但是统计分词方法对常用词识别精度差的固有缺点仍然存在。此系统的分词准确率为 98.5%。

北京大学计算语言学研究所开发的分词系统，具有分词和词性标注功能。由

于将分词和词性标注结合起来，系统可利用词性信息对分词决策提供帮助，并且在标志过程中又反过来对分词结果进行检验。系统的处理包括自动切分和初始词性标记、切分歧义字段识别、组词和标志预处理、词性标记排歧、切分和词性标注后处理等过程。算法综合了多重数据结构和搜索算法，实现了快速匹配和查找。系统强调了通用性，将最稳定、最常用的现代汉语基本词汇及其有关属性组织成为基本词典，可识别出大部分的常用词。

1.3 分词的重点和难点

1.3.1 分词规范的问题

1) 中文词的概念

中文自动分词的首要困难是词的概念不清楚。书面汉语是字的序列，词之间没有间隔标记，使得词的界定缺乏自然标准。词是什么：词的抽象概念；什么是词：词的具体界定。这两个基本问题没有弄清楚，迄今没有一个公认的、具有权威性的词库，主要困难表现在两个方面：一方面是单字词与语素之间的划分；另一方面是词与短语（词组）的划分。另外，对“词”的认识，群众的语感与语言学家的标准有较大的差异，对中文词认识上的差异，必然会给自动分词带来一定的困难。中文的词汇平面构成了现阶段中文信息处理应用领域的主要支撑平台，摆在人们面前的不是单纯的学术问题，也是一项颇具规模的语言工程，到目前为止没有得出合理的可操作的理论[10]。

2) 应用的不同对切分规范的影响

分词系统的通用性、适应性不足。一般研究自动分词软件一定要满足某种特定的需要，由于应用的不同，可能会有不同类型的分词系统。其分词结果很难采用统一的、通用的分词标准来评价。许多中文处理系统，根据应用目的开发适合自己需要的分词系统。分词单位界定的大小不同，必然造成统一评价分词系统的困难[11-14]。

1.3.2 分词算法的困难

要将中文文本的字序列切分成词的序列，即使确定了一个合适的分词标准，要实现这个标准也还存在算法方面的困难。

1) 歧义切分字段处理

一个中文句子是以连续字串的形式书写的。由于可能存在歧义，分词并不是一个简单的从输入串中发现合法词的过程。一个句子经常对应几个合法词序列，因此，中文分词中的一个重要问题就是在所有这些可能的序列中选出一个正确的结果。歧义切分是自动分词中不可避免的现象，是自动分词中一个比较棘手的问

题。对歧义切分字段的处理能力，严重影响到中文分词系统的精度。实践表明，只用机械匹配进行分词，其精度不高，虽然有时也能满足一些标准不高的需要，但不能满足中文信息处理高标准的要求。

2) 未登录词识别

未登录词包括中外人们、中国地名、机构组织名、事件名、货币名、各种专业术语等以及在不断发展和约定俗成的一些新词语，是种类繁多，形态组合各异，规模宏大的一个领域，对这些词语的自动识别，是一件非常困难的事[15]。但是未登录词的识别对于各种中文信息处理系统不仅有直接的实用意义，而且起到基础性的作用。

3) 分词与理解的先后

计算机无法像人在阅读汉语文章时那样边理解边分词，而只能先分词后理解，因为计算机理解文本的前提是识别出词、获得词的各项信息。这就是逻辑上的两难：分词要以理解为前提，而理解又是以分词为前提的。由于计算机只能在对输入文本还没有理解的条件下进行分词，则任何分词系统都不可能达到百分之百的切分正确率。

中文分词这个课题已经研究了很长时间，主要存在的问题一直都难以圆满解决。中国及国外一些研究中文的机构在对这些问题的研究上每取得一点进步都是非常费时费力的，而其中最为困难的则是被称为中文分词技术两大难题的歧义切分和未登录词的识别[16]。由于现有的计算机水平还不足以像人一样理解中文，所以对于一些能基本满足实际需要的分词技术就可以认为是很成功的了。

1.4 论文研究内容和组织结构

本文针对聊天机器人系统 ALICE 不能支持中文聊天的缺陷，提出了在聊天系统中加入中文分词的解决方法。在对现有分词技术进行研究的基础上，提出了一种新的分词词典结构，并且基于该结构设计了相应的分词算法。

本文将按以下结构对中文分词算法的研究及其应用展开讨论，具体的章节内容安排如下：

第一章简单介绍了聊天机器人的基本概念和研究现状，阐述了中文分词的研究意义和发展概况，讨论了分词的重点和难点；

第二章主要介绍了目前经常采用的各种中文自动分词方法，以及基于词典的机械分词方法的词典机制的研究；

第三章在对现有中文分词词典机制分析的基础上，结合中文字词的特点，提出了一种新的词典数据结构，提高了词典的查询匹配速度；基于本章提出的数据结构，详细设计了与之对应的中文自动分词算法。

第四章讲述了分词技术在智能聊天系统中的应用：当 ALICE 加载知识库时，首先对模式部分进行分词处理，再把分词结果交给 ALICE 的下一个模块继续处理；用户输入的问句采取同样的分词算法对问句进行分词后，再到内存知识树中查找对应的回复模板。

第五章对全文进行了总结并指出了论文存在的不足以及下一步的研究方向。

第二章 中文分词理论基础

中文句子的基本单位是单个的字而不是词，但是理解一个句子的单位却是一个一个的词。字成词，由词组成了句子才使得一个句子有意义。中文分词的目的就是把组成句子的字序列通过一定的算法加工成词序列。本章将对分词模型、现有的分词基本方法、分词词典等基础知识进行研究。

2.1 中文分词模型

2.1.1 分词系统理论模型

设 W 为分词过程中依据的分词词典，$W_1,W_2,W_3,...,W_n$ 是 W 中的元素（词条），n 为自然数。设 T 为要进行分词的中文文本，A 是非汉字字符集合，即外文字母、阿拉伯数字、标点符号和空格的集合；C 为汉字的集合，则 T 是由 C 的元素和 A 的元素组成的序列。设 D 是 T 中短句的集合，则 T 是 D 和 A 的元素组成的序列[17-20]。

对于任意的歧义字段 P，在确定的语言环境中都存在唯一的一种正确切分。即对任意的一个歧义字段 $P=C_1C_2...C_i$ 都有唯一的映射 $K:C_1C_2...C_i \rightarrow W_1,W_2,...,W_s$，其中 $W_j \in W$，$j=1,2,...,s$，使得 P 在 K 的作用下得到正确切分。所有的 K 的集合称为知识库 K。因此，知识库 K 可用于处理各类歧义字段。又因为汉语的构词法特别灵活，W 在实际中不可能收录所有的词。为了提高分词精度，K 中也应该包括构词知识。它们可构成 W 中没有的词条（假定 W 是完备的，即组成 T 的任意词条都包含在 W 中）[21-25]。

衡量一个自动分词系统的优劣主要在于分词的速度及分词的精度（即分词的正确率），其中分词的精度尤为重要，而影响分词精度的主要因素是歧义处理问题和专有名词的识别问题。

因此选择一个恰当的分词方法，并辅以必要的手段来解决歧义切分问题和专有名词的识别问题是提高分词精度的关键。由于基本的分词方法的分词精度都不是很高，因此提高分词精度的手段只有使用分词知识，以及进行语法分析和语义理解。

图 2-1 是中文分词系统的基本框架图。在后续章节中，将主要探讨有关基于词典的机械分词方法。

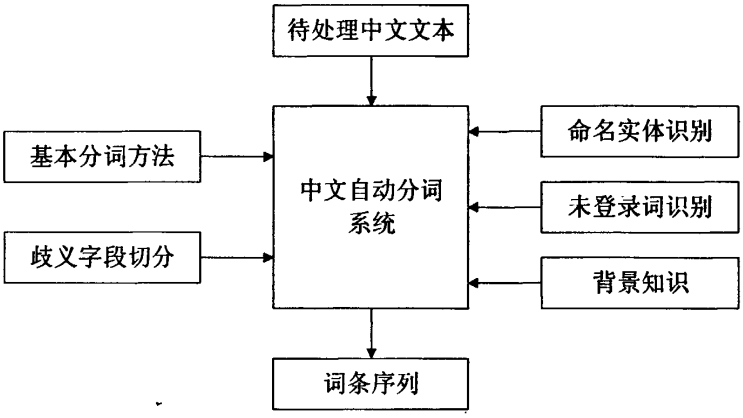

图 2-1 中文分词系统框架图

综上所述，可得如下分词模型 CWSM(Chinese Word Segmentation Model):M(F,W,T,K)。其中 F 是基本的分词方法，W 是分词词典，T 是中文文本，K 是知识库，对任意的一个短句（或字段）d∈T，有 $f(d/k)=W_1W_2...W_n$，其中 f∈F；$W_1W_2...W_n$∈W，k∈K。上式表示 f 在 k 的约束下，将 d 切分成 $W_1W_2...W_n$，有时 k 还包括 d 的上下文知识。

2.1.2 分词系统评价准则

自动分词系统的最主要的工作就是进行分词。对于分词而言，不仅要求所研制的软件在分词的正确率和速度方面满足一定的要求，而且要像开发大型传统软件那样，在各个阶段不断地进行评价，其目的主要是检查它的准确性和实用性，分词的评价主要有以下几个方面[26]：

1）分词正确率 书面的中文文本可以看出是字符序列，分词的正确率直接影响更高一级的处理。现有的分词系统切分错误主要集中在歧义字段和专有名词（如人名、地名、机构名和未登录词等）。为了获得分词系统切分正确率，应该进行整体测试、歧义测试和专业词测试。因此，自动分词系统切分正确率的基本公式为：

$$S=\sum_{i=1}^{3}\beta_i S_i \qquad (2\text{-}1)$$

其中，S_1,S_2,S_3 分别为总体测试、歧义测试和专业词测试的正确率；β_i(i=1,2,3)为三种测试相加的权值。

2）切分速度 切分速度是指单位时间内所处理的汉字个数。在分词正确率基本满足要求的情况下，切分速度是另一个很重要的指标，特别是对于算法不单一，使用了辅助手段（比如联想、基于规则、神经网络、专家系统等）的算法，更应

该注意分词的切分速度。通常中文信息处理的文本数量是相当大的，因此必须考虑分词方法是否能使系统总开销合理。在人机交互方式下处理歧义问题的策略和人机接口的设计，有时会严重影响切分速度，这也是一个应该考虑的因素。

3）功能完备性 自动分词方法除了完成分词功能外，还应具备词库的增、删、改、查等功能。

4）可移植性 可移植性是指分词方法能从一个计算机系统或环境转移到另一个系统或环境的容易程度。一个好的分词方法不应该只能在一个环境下运行，而应该稍作修改便可在另一种环境下运行，使得它便于推广。

5）适应性 自动分词只是一种手段而不是目的，任何分词系统产生的结果都是为某个具体的应用服务的。好的分词系统具有良好的适应性，可以方便地集成到各种各样的汉语信息处理系统中去。

2.2 中文分词基本方法

中文分词技术属于自然语言处理技术范畴，对于一句话，人们可以通过自己的知识知道该句子由哪些词组成，但怎么让计算机理解？这个处理过程就是分词算法。现有的分词算法可以分为三大类：基于字符串匹配的分词方法、基于统计的分词方法和基于理解的分词方法。

2.2.1 基于字符串匹配的分词方法

这种方法又叫做机械分词方法，它是按照一定的策略将待分析的字串与一个“充分大的”机器词典中的词条进行匹配，若在词典中找到某个字符串，则匹配成功[37]（识别出一个词）。机械分词可以按照以下几种方法进行分类：

按照扫描方向的不同，机械分词法可以分为正向匹配和逆向匹配。逆向匹配的切分正确率要高于正向匹配法，为了便于发现歧义切分，有时候将两者结合起来形成双向匹配法。由于正向匹配法和逆向匹配法对词库的组织要求不同，所以将它们结合时，要重新考虑词库的组织以便两者都能快速执行。

按照不同长度优先匹配的情况，可以分为最大（最长）匹配和最小（最短）匹配，由于大多数汉字可以构成单字词，所以按最小匹配法分词的结果通常因为分得太细而不合要求。

按照匹配不成功时重新切取的策略，机械分词法可以分为增字法和减字法。增字法一般和最小匹配法结合使用，减字法则一般和最大匹配法相结合。

根据以上分类，基本的机械分词方法可以分为以下三种：最大匹配法、最小匹配法和全切分法。

2.2.1.1 最大匹配法(Maximum Matching Method)

最大匹配法可以分为正向最大匹配法(MM)和逆向最大匹配法(RMM)。最大正向匹配法的基本思想是：设 D 为词典，M 表示词典 D 中的最大词长，str 为待切分字符串。每次按正向顺序取长度为 M 长的字符串与词典中的词进行匹配。若匹配成功，则得出该字符串为一个词，后移 M 个字继续进行匹配。否则字符串减一个字（减去串末尾的字）继续进行匹配，直到匹配成功为止。这样就完成了一次匹配过程，即切分出一个词。然后再按照上面的步骤进行，直到切分出文本中的所有词为止。这也是一种减字的匹配法，其流程如图 2-2 所示：

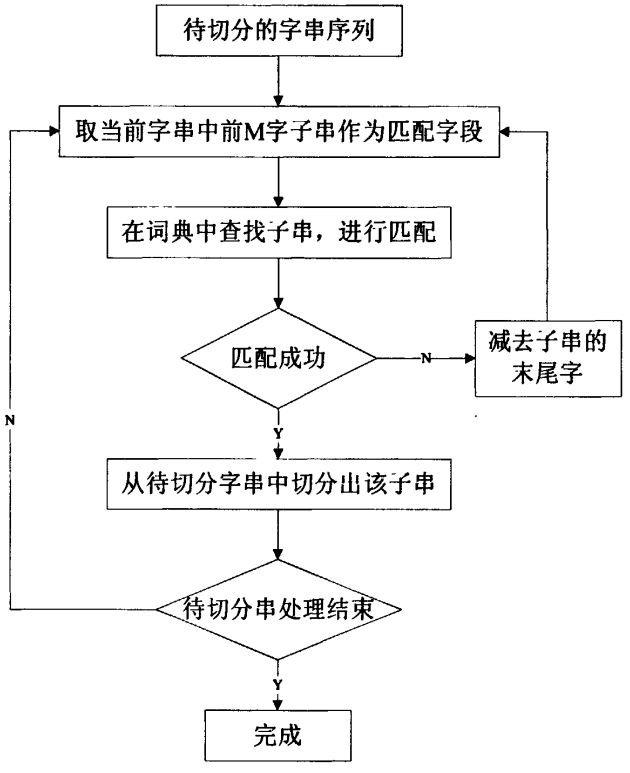

图 2-2 最大正向匹配法

MM 法的原理简单，易于在计算机中实现，时间复杂度也比较低。但是，最大词长的长度难于确定，如果定得太长，则匹配时花费的时间就比较多，算法的时间复杂度将明显提高，如果定得太短，则不能切分长度超过它的词，导致切分正确率降低。统计表明 MM 方法的错误切分率为 1/169。所以，该方法一般不单独使用，而是作为一种基本的方法和其它方法配合使用。

逆向最大匹配法(The Reverse Directional Maximum Matching Method)，也称为 RMM 方法。它的分词过程与 MM 方法相同，不同的是每次从待切分文本的末尾开始处理，每次匹配不成功时从匹配串的前面开始逐个减字。RMM 方法的

精度比 MM 法要高些，它的错误切分率为 1/245[27]。

2.2.1.2 最小匹配法

最小匹配法也分为正向匹配和逆向匹配两种方式。它是一种增字的匹配方法，其基本原理和最大匹配法差不多。

例如待切分串“中华人民共和国”，首先取“中”作为字段来匹配，由于词库中没有逐个词，所以匹配失败。然后增加一个字，取“中华”作为字段进行匹配，在词库中查找到这个词，这时匹配成功，把“中华”作为词切分出来。再对剩余字串“人民共和国”重复上述过程，直到所有的词都被切分出来为止。

2.2.1.3 全切分法

这种方法是将词库中的词由长到短递减的顺序，逐个在待切分串中搜索，直到切分出该串中所有的词为止[28,29]。

该分词方法在切分任意字符串的过程中，不论分词词库有多大，都得把整个分词词库匹配一遍。所以这种方法的时间复杂度比较高，切分速度慢，切词的效率不高。

通过以上分析，可以得出：对于机械分词方法，可以建立起一个一般的模型。其函数形式为 ASM(d,a,m)[30]。其中：

ASM 代表自动分词方法；

d 代表分词方法的切分方向；

d∈D={+1,-1}+1 代表正向匹配，-1 代表逆向匹配；

a 代表分词方法是增字，还是减字；

a∈A={+1,-1}+1 代表增字，-1 代表减字；

m 代表分词方法是最小匹配，还是最大匹配；

m∈M={+1,-1}+1 代表最大匹配，-1 代表最小匹配。

2.2.2 基于统计的分词方法

从形式上看，词是稳定的字的组合，因此在上下文中，相邻的字同时出现的次数越多，就越有可能称为一个词。因此字与字的相邻共现概率[39]能较好地表达成词的可信度。这就是词频统计的基本原理，这种技术发展至今已经有许多不同的统计原理：互信息原理、N 元统计模型原理、t-测试原理等。

2.2.2.1 互信息原理

有序汉字串 AB 中汉字 AB 之间的互信息定义如公式（2-2）所示：

$$I(A,B)=\log_2\frac{P(A,B)}{P(A)P(B)} \qquad (2\text{-}2)$$

互信息体现了汉字之间结合关系的紧密程度，只有当紧密程度高与某一个阈值时，才可认为这个字组可能构成了一个词。其中$P(A,B)$是汉字串 AB 一起出现的概率，$P(A)$是汉字串 A 出现的概率，$P(B)$是汉字串 B 出现的概率，它们在汉字串中出现的次数分别记作$n(A)$、$n(B)$、$n(AB)$，n是词频总数，则有公式（2-3）：

$$P(A,B)=\frac{n(AB)}{n},P(A)=\frac{n(A)}{n},P(B)=\frac{n(B)}{n} \tag{2-3}$$

互信息反映了汉字串 AB 间相关的程度：

1）如果$I(A,B)\geq 0$，即$P(A,B)\geq P(A)P(B)$，则 AB 间是正相关的，随着$I(A,B)$增加，相关度增加，如果$I(A,B)$大于给定的阈值，则可以认为 AB 是一个词；

2）如果$I(A,B)\approx 0$，即$P(A,B)\approx P(A)P(B)$，则 AB 间是不相关的；

3）如果$I(A,B)<0$，$P(A,B)<P(A)P(B)$，则 AB 间是互斥的，这时 AB 间基本不可能成词。

2.2.2.2 N 元统计模型原理

N-Gram 统计计算语言模型的思想是：一个单词的出现与其上下文环境中出现的单词序列密切相关，第 n 个词的出现只与前面 n-1 个词相关，而与其它任何词都无关，设 $W_1W_2...W_n$ 是长度为 n 的字符串，则字符串 W 的似然度用方程表示如公式（2-4）所示：

$$P(W)=\prod_{i=1}^{n}P(W_i\mid W_{i-n+1}W_{i-n+2}\cdots W_{i-1}) \tag{2-4}$$

不难看出，为了预测 W_n 的出现概率，必须知道它前面所有词的出现概率。从计算上来看，这种方法太复杂了。如果任意一个词 W_i 的出现概率只同它前面的两个词有关，问题就可以得到极大的简化[30]。这时的语言模型叫作三元模型(tri-gram)，如公式（2-5）所示：

$$P(W)\approx P(W_1)P(W_2\mid W_1)\Pi_{i=3,...,n}P(W_i\mid W_{i-2}W_{i-1}) \tag{2-5}$$

符号$\Pi_{i=3,...,n}P(W_i\mid W_{i-2}W_{i-1})$表示概率的连乘。一般来说，N 元模型就是假设当前词的出现概率只同它前面的 N-1 个词有关。重要的是这些概率参数都是可以通过大规模语料库来计算的，比如三元概率如公式（2-6）所示：

$$P(W_i\mid W_{i-2}W_{i-1})\approx\frac{count(W_{i-2}W_{i-1}W_i)}{count(W_{i-2}W_{i-1})} \tag{2-6}$$

式中$count(L)$，L 为字符串，表示一个特定词序列在整个语料库中出现的累

计次数。

2.2.2.3 t 测试原理

对有序汉字串 xyz，汉字 y 相对于 x 及 z 的 t-测试定义公式如公式(2-7)所示：

$$t_{x,z}(y)=\frac{p(z|y)-p(y|x)}{\sqrt{\delta^2(p(z|y))+\delta^2(p(y|x))}} \tag{2-7}$$

其中 $p(z|y)$，$p(y|x)$ 分别是 z 关于 y，y 关于 x 的条件概率，$\delta^2(p(z|y))$，$\delta^2(p(y|x))$ 则代表各自的方差，公式（2-7）中的各个量可以用公式（2-8）、（2-9）估计：

$$p(y|x)=\frac{p(x,y)}{p(x)}=\frac{r(x,y)}{r(x)},p(z|y)=\frac{p(y,z)}{p(y)}=\frac{r(y,z)}{r(y)} \tag{2-8}$$

$$\delta^2(p(y|x))=\frac{r(x,y)}{r^2(x)},\delta^2(p(z|y))=\frac{r(y,z)}{r^2(y)} \tag{2-9}$$

从 t-测试的定义可知：

1）当 $t_{x,z}(y)>0$ 时，字 y 有与后继字 z 相连的趋势，值越大，相连的趋势就越强；

2）当 $t_{x,z}(y)=0$ 时，不反映任何趋势；

3）当 $t_{x,z}(y)<0$ 时，字 y 有与前趋字 x 相连的趋势，值越小，相连的趋势就越强。

互信息反映的是字与字之间的静态结合，因为它计算的就是相邻字出现的频率，根据这个频率与字单独出现的频率进行比较，计算出互信息来判断相邻字能否组成词语。t 测试能较好地反映字与字之间的动态结合，它是通过公式计算比较当前字与前面字的结合能力以及与后一个字的结合能力，来判断它到底与哪个字结合得更紧密，从而更能组成词[32]。由于互信息反映的是字与字之间的静态结合能力，t 测试反映的是字与字之间的动态结合能力，因此如果能将这两个统计原理相结合，一定能起到互补的效果。

2.2.3 基于理解的分词方法

除了上述机械分词方法和基于统计的分词方法，分词方法还包括基于理解的分词方法。

通常的分词系统，都力图在分词阶段消除所有歧义切分现象。而理解性分词系统则在后续过程中处理歧义切分问题，它的分词过程只是整个语言理解过程中比较小的一部分。理解性分词的基本思想就是在分词的同时进行句法、语义分析，利用句法信息和语义信息来处理歧义现象。它通常包括三个部分：分词子系统、

句法语义子系统、总控部分。在总控部分的控制下，分词子系统获得有关词、句子等的句法和语义信息来对分词歧义进行判断，即它模拟人对句子的理解过程。这种分词方法需要使用大量的词法、句法和语义知识[38]。由于汉语语言知识的笼统、复杂性，难以将各种语言信息组织成机器可直接读取的形式，因此目前基于理解的分词系统还处在试验阶段[34-36]。

2.3 机械分词词典机制研究

分词词典是机械分词系统中的一个重要组成部分[19]，其查询效率直接影响到分词系统的整体性能。而许多实际应用系统，如网络搜索引擎、语音识别等均需要不断对词典进行增、删、改、查等维护工作，这就要求分词词典必须有一个快速的更新机制。总之，快速查询是分词词典实用化时应满足的基本要求。

词典的查询是切分字串的基础，针对分词系统的特点，分词词典应具备以下三种基本的查询功能：

1）确定词查询

给定词 w，查询词典中是否存在词条 w。若有，则返回词条 w 在分词词典中的位置，以便得到 w 的各类附加信息。由于查询词 w 是预先确定的，一般称这类查询方式为“确定词查询”。这是一种基本的查询方式。

2）最长词查询

给定汉字串 S，根据分词词典查找 S 中从某一指定位置 i 开始的最长词（对应最大匹配分词方法）。由于最长词的长度是无法预先确定的，通常的解决办法是尝试查询从位置 i 开始的所有可能长度的词，经过多次“确定词查询”完成查询操作。

3）前缀词条查询

给定汉字串 S，根据分词词典查找 S 中从某一指定位置 i 开始的所有的词，这些词均为汉字串 S 中从 i 开始的子串 S_i 的前缀。该方式类似于“最长词查询”，但返回结果通常不唯一。

目前组织分词词典主要有整词二分、TRIE 索引树、逐字二分等三种主要的方法。

2.3.1 基于整词二分的分词词典机制

这种词典机制使用比较广泛。其结构通常分为三级，前两级为索引。词典结构如图 2-3 所示：

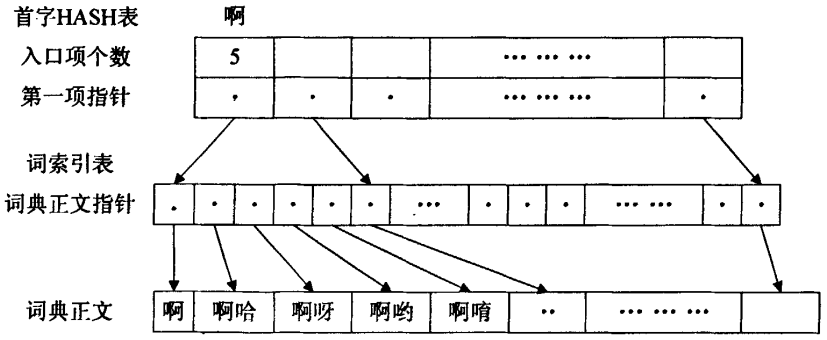

图 2-3 基于整词二分的词典机制

1．首字 HASH 表

词首字 HASH 函数根据汉字的国标区位码给出。通过一次哈希运算即可直接定位汉字在首字 HASH 中的序号。

首字 HASH 表的一个单元包括两项内容：

入口项个数：以该字为首字的词的个数；

第一入口项指针：指向第一入口项在词索引表中的位置。

2．词索引表

由于每个词的长度可能不一样（实际系统中还包括附属于该词的各类信息），所以采取不定长存储比较合适，但是这样势必不能实现对词的随机访问。为了能够对词进行随机访问，必须建立词索引表。词典正文指针指向词在词典正文中的位置。

3．词典正文

以词为单位的有序表。将词典正文和词索引表配合使用，很容易实现确定词在词典正文中的二分快速查找。

这种分词词典机制比较适合于“确定词条查询”。而在另外两种查询方式中，对任一位置 i，通常需要预先设定一个词的可能最长长度 M（一般 M 为词典中最长词的长度）。然后取汉字串 $S_{[i,\ i+M-1]}$作为匹配串，在词典中进行二分查找。如果没有查到，则匹配串循环递减一个字，直至匹配成功。这种由长词递减到短词的盲目尝试方法效率很低。

例如：查询汉字串 S“同学们大白天在学校操场上打了一个多小时的篮球”中从“大”字开始的最长的词。

假定词典中最长词的长度为 14，取从“大”字开始最长的匹配串：$W_{i,max}$=“大白天在操场上打了一个多小时”；

在分词词典中二分查找匹配串 $W_{i,max}$，查找失败；

去掉 $W_{i,max}$ 的末尾字，继续二分查找，查找失败；

依次减去末尾字，经过 11 次的错误尝试，匹配串减少到只剩下“大白天”，这时查找成功，于是返回 $W_{i,max}$=“大白天”。

2.3.2 基于 TRIE 索引树的分词词典机制

TRIE 索引树是一种以树的多重链表形式表示的键树[33]。面向英文的 TRIE 索引树一般以 26 个字母作为关键字，树节点包括个数相同的指针。而汉字差不多有 7000 来个，如果采用同样的策略构造中文词典，显然会造成指针的大量浪费。面向中文的 TRIE 索引树的节点应允许指针个数的变化。

基于 TRIE 树的分词词典由两部分组成，其结构如图 2-4 所示：

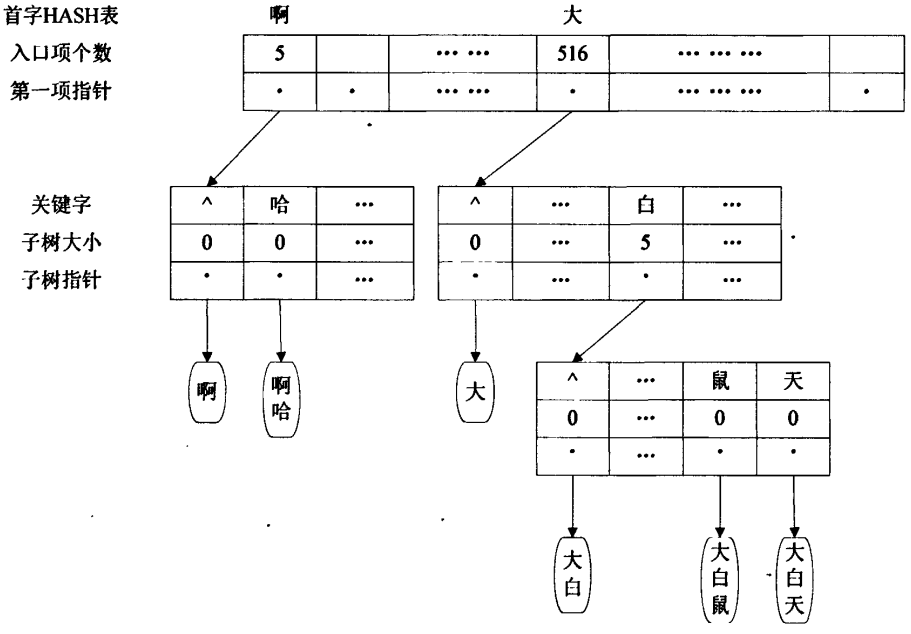

图 2-4 基于 TRIE 索引树的词典机制

1．首字 HASH 表

和基于二分的词典机制差不多，但是它的第一项指针指向对应的 TRIE 索引树的根节点。

2．TRIE 索引树节点

TRIE 索引树节点是以下面的结构为单元的、按关键字排序的数组：

关键字：一个汉字；

子树大小：从根节点到当前单元的关键字组成的子串为前缀的词的个数；

子树指针：当子树大小不等于零时，指针指向子树根节点；否则指针指向叶子节点。

由于 TRIE 索引树已经包含了词条信息，因此在词典中没有必要再显示地列

出词条，可以存储词条的一些附属信息（由叶子指针指向这些信息）。

按照 TRIE 索引树的分词词典机制来查询一个确定词 $W_{[1,n]}$的过程分为以下几个步骤：

1）根据首字 HASH 表取得汉字 $W_{[1]}$的索引树，沿相应指针移动到目标节点 Node；i=2；

2）在 Node 的关键字域中对 $W_{[i]}$进行二分查找。

If $W_{[i]}$与 Node 节点的第 j 个单元的关键字匹配成功

Then 沿该单元子树指针移至下级目标节点，用该节点重置 Node；i=i+1；

else 查询词 $W_{[1,n]}$失败，推出此过程。

3）继续第二步，直到 Node 为叶子节点为止。

4）If 匹配至叶子节点时 i>n

Then 查询成功，$W_{[1,n]}$是词典中的词；

Else 查询失败。

基于 TRIE 索引树的分词词典机制和整词二分不同的是，它每次只是比较一个汉字，不需要预先知道待查询词的长度，并且在对汉字串的一次扫描过程中，就能得到最长词查询和前缀词条查询两种方式的结果。这种从短词到长词的确定性工作方式避免了整词二分词典机制中的不必要的多次试探性查询。

例如：查询汉字串 S“同学们大白天在学校操场上打了一个多小时的篮球”中从“大”字开始的最长词以及所有词。

首先通过首字 HASH 表，经过一次查询，便可得到以“大”字开头的词的 TRIE 索引树节点 T；

由于节点 T 中第一项的关键字域为“^”（空字符），因此“大”可以成为一个词。在节点 T 中二分查找关键字“白”，其指针指向的子树节点假定为 T1。

节点 T1 的第一项的关键字域为“^”，“大白”也可以成词。在节点 T1 中继续二分查找关键字“天”，发现“天”的子树节点已经是叶子节点，所以“大白天”也能成词，至此查询结束。可以得出“大白天”是最长词，由中间过程查询出的“大”、“大白”、“大白天”都是以“大”字开头的词。

TRIE 索引树分词词典机制相比于整词二分的主要缺点是它在构造和维护方面都要复杂一些。

2.3.3 基于逐字二分的分词词典机制

针对整词二分和 TRIE 索引树的不足之处，一种改进方案——基于逐字二分的分词词典机制产生了。逐字二分和整词二分两种方式在数据结构上是完全一样的，但是查询过程不同。它在进行查询时不是将整个词作为关键字进行比较，而是和 TRIE 索引树差不多，每次只是比较一个汉字，所以其查询效果和 TRIE 索

引树是一样的。

2.4 本章小结

本章介绍了中文分词系统的理论模型及其评价准则，详细介绍了几种常用的分词方法，本文研究的新分词方法是在基于字符串匹配的分词方法基础上作了一定的改进，另外两种方法能在一定程度上提高分词的准确率，是未来的一个研究方向。机械分词方法的效率很大程度上依赖于一个良好的分词词典机制，因此本章对目前的几种分词词典的组织方法进行了详尽的阐述。

第三章 中文分词算法设计

中文分词的功能是将组成汉语句子的字序列切分成与原句子意思相同的词序列。通过对分词基础理论知识的学习，本章在分析汉字编码体系、中文词特点的基础上，提出了一种新的中文词表数据结构；并且基于这一结构设计了相应的机械分词算法，该算法支持首字 Hash 和二分查找，具有很高的分词效率。

3.1 分词算法的选取

在聊天机器人系统中，用户首先通过聊天界面向系统提出一个话题，然后系统对该话题进行分析处理，在系统知识库中寻找与该话题匹配的话语回复用户。用户在聊天时的一个显著特点是：所提出的话题一般都是比较短小的，而不是长篇大论，不具有段落篇章结构，绝大多数就是少数几句话。

根据人们在聊天过程中的特点，并且通过第二章对中文自动分词方法的研究与分析，本文选择了基于字符串匹配的分词方法作为智能聊天机器人系统的分词方法。

本文选择机械分词而不是其它两种分词的原因是：由于在聊天过程中，用户输入不会很长，不存在段落及篇章结构，也就没有上下文关系，所以基于统计的分词方法在聊天系统中没有实际意义，对分词效果不会有任何提高，可以不予考虑；另外由于基于理解的分词系统目前还不是一种成熟的分词方法而且分词的过程比较复杂，时间复杂度比较高而不能满足系统的速度需要，所以也不予以考虑。因此系统采用基于字符串匹配的分词方法能取得比较好的切分效果。

基于字符串匹配的算法涉及到词典的设计和基于词典的分词算法两部分。本章首先提出了一种改进的分词词典存储结构，从而提高了词典的查找匹配速度；针对机械分词算法解决歧义困难的特点，采用了双向匹配及长词优先等方法进行歧义消解，从而提高分词的准确性。

3.2 词典结构设计

中文电子词表的研究是中文信息处理中基础性的工作，在中文自动分词、文本检索、机器翻译等诸多领域都有重要的应用价值。在对现有中文词表结构进行深入分析的基础上（详见 2.3 节），结合中文字词的特点，本节提出了一种改进的数据结构——基于首字 HASH、支持二分查找。

3.2.1 汉字编码体系介绍

在研究词库的设计之前，应当先讨论一下中文字词在计算机内是如何表示的。

中文字的数目目前还没有一个准确的数字，一种说法是汉字大概有 10 万个左右，而常用汉字只有几千个。目前汉字的代码体系由输入码、交换码、内码、区位码等构成，很多中文平台都采用内码来处理汉字信息。下面简单介绍一下输入码、交换码、内码、区位码的相关概念：

1）输入码

汉字的个数很多，字形复杂，常用的汉字有六七千个，比英文字母要多得多。在计算机系统中使用汉字，首先遇到的问题就是如何把汉字输入到计算机内。为了能直接使用西方标准键盘进行输入，必须为汉字设计相应的编码方法。汉字编码方法主要分为三类：数字编码、拼音编码和字形编码。

2）国标码和机内码

西文处理系统的交换码和机内码均为 ASCII，用一个字节表示，一般只用低七位。我国也制定了汉字交换码（简称国标码）。在国标码中，一个汉字用两个字节表示，每个字节也只用其中的七位，每个字节的取值范围和可打印的 ASCII 字符的取值范围相同。为了避免 ASCII 码和国标码同时使用时产生二义性问题，大部分汉字系统一般都采用将国标码每个字节高位置设为“1”作为汉字机内码。这样既解决了汉字机内码与西文机内码之间的二义性，又使汉字机内码与国标码具有简单的对应关系。

3）USC 编码

为了统一表示世界各国的文字，国际标准化组织公布了“通用多八位编码字符集”的国际标准 UCS(Universal Code Set)。UCS 包含了中、日、韩等国的文字，这一标准为包括汉字在内的各种正在使用的文字规定了统一的编码方案。该标准用四个字节来表示每个字符，并相应地指定组、平面、行和字位。

3.2.2 词典设计

随着时间的推进，中文词的数目也在不断增加，中文词是一个开放集。以不同字开头的词的数目变化很大，多的达到数百个，少的也有可能只有一个或者没有；词的长度变化也很大，有单字词，也有由六、七个字成词的。这就要求在设计词典时，除了考虑访问效率外，还得充分考虑存储利用率。

考虑到中文字的编码体系和中文词的特点，为了提高词表的查找效率和存储利用率，本节设计了一种新的数据结构：以词的首字作为索引，所有以该字为首字的词条组织在一起；另外由于词的长度不同，将同一首字下不同长度的词存放

在不同的区间，分别形成2字词表、3字词表等多字词表。这种结构在内存中的逻辑形式如图3-1所示：

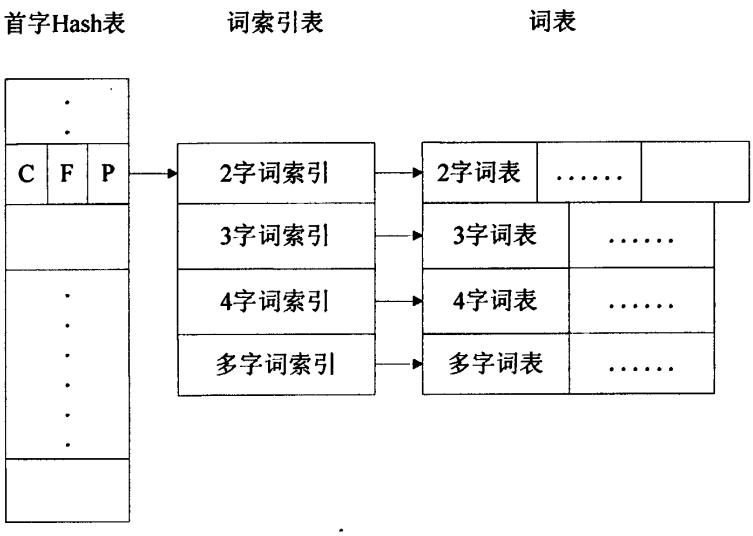

图 3-1 词典结构

由图3-1可知，此分词词典由三部分组成：首字Hash表、词表索引、词表。

1）首字Hash表

通过一次哈希运算就可以直接定位汉字在表中的位置。首字Hash表的一个单元包括三项内容：

C：存储首字

F标志位：存储以C为首字的最长词条的长度；

P：指向词表索引表

2）词表索引

由于词的长度可变，建立了每个首字下不同长度词条的词表。词表索引包括两项内容：词表词条长度和指向词表的指针。

3）词表

以词为单位的顺序表，按USC码大小有序排列，很容易实现指定词在词表中的二分快速查找。

这种分词词典机制在进行“确定词条查询”时具有很高的效率，并且由于已知某个首字下各词条的长度，由长词递减到短词的尝试方法不再是盲目的，所以在“前缀词条查询”和“最长词条查询”中的时间复杂度也不高。

以词表中所有“大”字为首字的词为例，其在内存中的逻辑表示如图3-2所示。

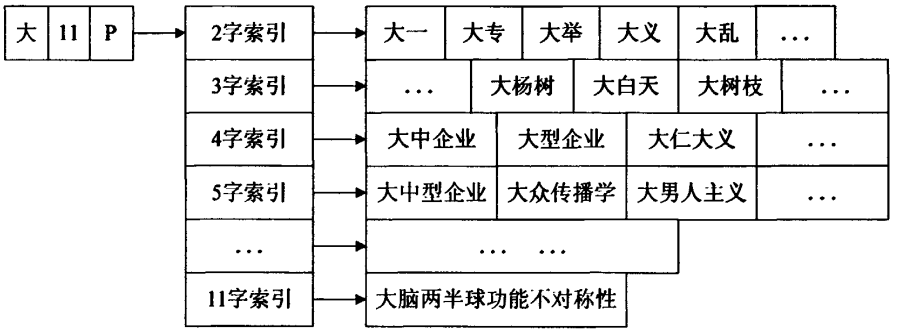

图 3-2 词典结构示例

此词典机制在进行“最长词条查询”时类似于 2.3 节介绍的基于整词二分的分词词典机制，但比后者效率更高。在匹配过程中，由于已知各词表长度，且每次二分查找都只在各词表中进行，所以切分效率比较高。

例如：查询汉字串 S“同学们大白天在学校操场上打了一个多小时的篮球”中从“大”字开始的最长的词。

已知词典中首字为“大”的最长词的长度为 11，取从“大”字开始最长的匹配串：Wi,max=“大白天在操场上打了一个”；

在分词词典 11 字词索引指向的词表中二分查找匹配串 Wi,max，查找失败；

在 Wi,max 中，取词表索引表中词条第二长的长度为子串，在相应的词表中继续二分查找，查找失败；

依次以词表索引表中各词条长度为准，取 Wi,max 中的子串，匹配串减少到只剩下“大白天”，这时查找成功，于是返回 Wi,max=“大白天”。

上面分析的是正向最大匹配下的词典结构，对于逆向最大匹配，可以很容易地构造类似的词典结构，同首字 Hash 相同，只需建立一个末字 Hash 词典即可。

3.2.3 词表访问效率

当词表载入内存后，就可以对其进行相关访问了。本小节主要讨论数据查找、增加、删除等算法的效率。

1）词表查找算法

查找效率是衡量词表访问性能的最重要的指标之一。词典结构采用了 Hash 方法，能够大大地降低查找次数。

查找某个特定的词条时，首先利用 Hash 方法，得出首字 C 的存储地址，再计算该词条的长度，在首字 C 对应的词表内进行二分查找。算法描述如下：

算法 1：FindWord

第一步：获取待查词条首字和它的长度；

第二步：通过 Hash 表得到首字下和该词条相同长度词表的入口地址；

第三步：在词表中对词条剩余部分进行二分查找；

第四步：返回结果

由此可以得出以下结论：若记$\overline{n_i}$为每个首字下的 i 字词的平均词数，而且每个词条被查询的机会均等，则查找一个词的时间复杂度为$O\left(\log \overline{n_i}\right)$。

2）词表增加算法

新增一个词条的过程要复杂一些，首先在词表中查找输入项，如果找到，则不用增加；否则根据查找词条获得的索引号 k，将第 k 项至第 n-1 项词条顺序后移，新增的词条作为第 k 项。增词算法描述如下：

算法 2：AddWord

第一步：查找输入项，如果找到，返回 false；否则获得索引号 k；

第二步：新增一个输入项词条；

第三步：将第 k 项至第 n-1 项词条顺序后移；

第四步：将新增词条存入词表 k 位置。

由 FindWord 可知，查找输入项时间复杂度为$O\left(\log \overline{n_i}\right)$；移动词条过程的时间复杂度为$O\left(\overline{n_i}\right)$，因此新增词条的时间复杂度为$O\left(\overline{n_i}\right)$。

3）词表删除算法

如果要删除某个词条，首先要在词表中找到相应的索引号 k，将第 k+1 项至 n-1 项顺序前移。删除词条算法描述如下：

算法 3：DeleteWord

第一步：查找输入项，如果没有找到，返回 false；否则获取索引号 k；

第二步：删除输入项词条；

第三步：将第 k+1 项至 n-1 项词条顺序前移；

和 FindWord 类似，删除词条的时间主要花在了移动词条上，因此删词的时间复杂度也为$O\left(\overline{n_i}\right)$。

3.3 基于词典结构的算法设计

本节提出的基于词典的算法是基于 3.2 节设计的词典结构的：在进行分词之前，首先需要把分词所依赖的词库按照 3.2 节设计的数据结构加载到内存中；然后对待切分文本进行分词预处理，将待切分串分成较小的子串，从而降低切分错误发生的概率；对子串中的汉字串进行双向最大匹配分词，该方法具有分词准确率高且简单有效的特点；由于正反向匹配分词得出来的结果可能不同，需要进行消除歧义的处理，从而统一分词结果。本节主要介绍该算法的详细设计方案，经分析它具有比较高的分词效率。

3.3.1 算法流程

基于词典的分词算法分为词典加载、预处理、最大匹配、歧义消解几个阶段，其具体流程如下：

步骤一：预处理阶段，按照特殊字符（英文字母、数字、标点符号等）将待分析文本进行断句，将待切分的文本切分为只有中文的短句子，这些句子是下一步分词处理的基本单位；

步骤二：对断句出来的句子进行双向最大匹配分词，分词后的结果作为步骤三的输入；

步骤三：对上一步分词得到的结果进行比较，判断是否存在歧义，如果存在歧义，就进行一定的歧义消解；

步骤四：重复步骤二、步骤三，直到处理完步骤一中断句所切分出的所有句子单元。

算法流程如图 3-3 所示：

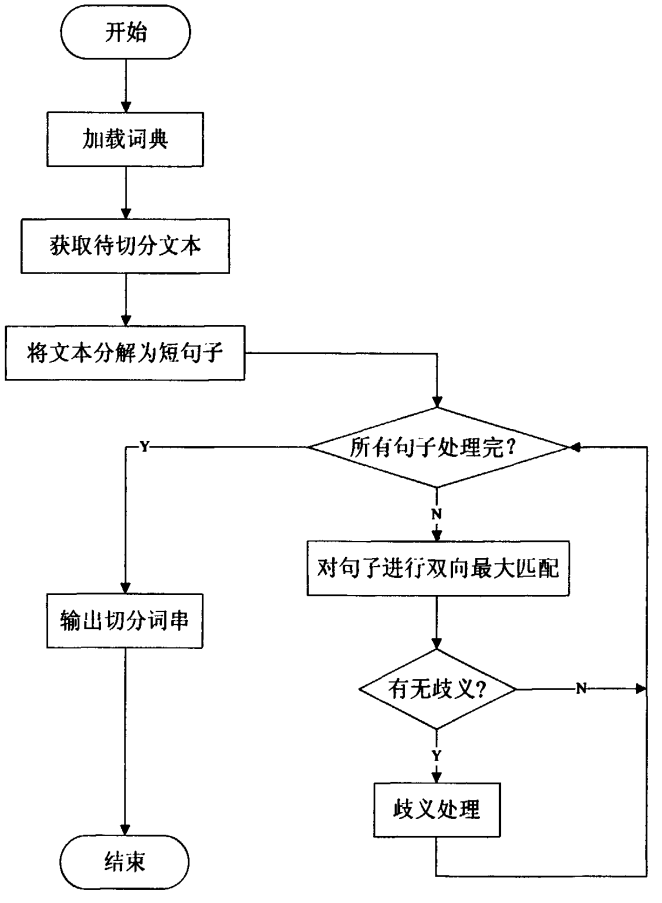

图 3-3 分词算法整体流程图

以上是对基于词典的分词算法的一个整体描述，其中的核心步骤是双向最大匹配阶段，词典加载及分词预处理是为其提供输入，歧义处理是对其输出进行操作。下面将对各个具体步骤进行详细介绍。

3.3.2 词典加载

机械分词都依赖于一个机器词典，在进行分词之前，首先得把词库载入内存。这里所选用的词库是根据中文词库素材分类整理而来的，按照词条长度分为多个小词库，每个子词库内词条按顺序存储在 txt 文件中，其中每个词条信息各占一行。

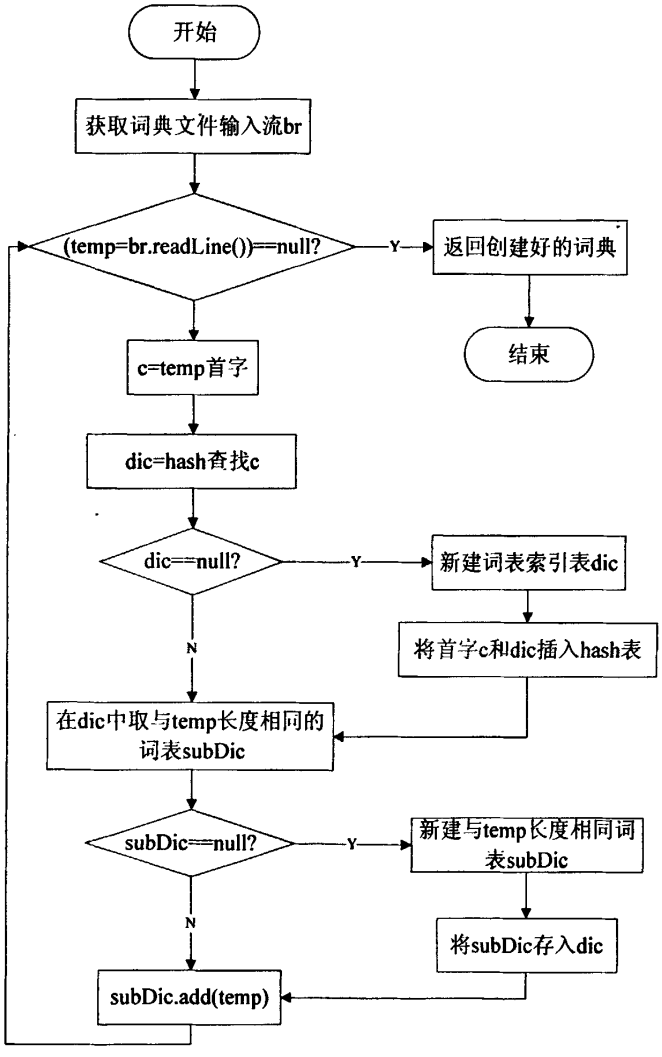

图 3-4 词典加载流程图

在加载过程中，通过对词库中的各个词条进行 hash 查找将词条插入到相应词表的正确位置。当词库载入内存后，词典以图 3-1 所示的存储结构来组织词库中的每个词条。

分词词典的加载流程见图 3-4，其相应的算法描述如下：

算法：分词词典加载

输入：词库文件

输出：分词词典

方法：

1）获取子词库文件，如果读取完则转 8)；否则转 2)；

2）打开子词库文件；

3）如果文件流读取到子词库文件最后，则转 1)；否则转 4)；

4）读取词条，获得词条首字和词条长度；

5）Hash 查找首字，如果没找到，则新建该字的词索引表和与词条长度对应的子词表，插入该词，添加 Hash 表项；否则转 6)；

6）获取与词条长度对应的子词表，如果没有，则新建子词表，插入该词；否则转 7)；

7）在子词表中插入该词，转 3)；

8）返回 Hash 表词典；

3.3.3 分词预处理

分词过程是一个非常复杂的过程，字符串越长，发生匹配错误的可能性就越大。如果能够将原字符串分解成较小的字串再进行扫描分词，匹配错误发生的概率就会减小。

在双向扫描之前，先将字符串按照特殊字符切分为较短的字串，以这些较短的串作为处理的基本单元，这样的处理能够保证分词的准确性，减少匹配的错误率。预处理的目的就是把待切分串分成不同类型的子串，从而只对其中的汉字串进行双向匹配。

在预处理阶段，如何识别字符类型是首要解决的问题。本文采用的解决办法是通过逐字扫描待切分串，借助一个字符类型判别函数，对每个字符进行判断，从而识别出特殊字。

下面是对字符进行类型判别的函数，它把待切分串的字符分为四种不同的类型：

```
//字符类型判断函数
public static int testChar(char c) {
  if (isChinese(c))              //汉字
    return CHINESE;
  if (isEnChar(c))               //英语字母
    return ENCHAR;
  if (isNumber(c))               //数字
    return NUMBER;
  else                           //标点符号等
    return OTHER;
}
```

得出待切分文本中每个字符的类型后，就可以根据不同的类型采用不同的处理方式了：

1）如果是数字，根据前一个字符的标记符判断是否与前一个字切分开，如果是数字标记符，则不与后面的字串切分开，先暂时保存当前字的标记，待对后一个字进行判断后再切分，主要解决数字串的问题；

2）如果是字母，根据前一个字符的标记符判断是否与前一个字切分开，如果是字母标记符，则不与后面的字串切分开，先暂时保存当前字的标记，待对后继字进行判断后再切分，主要是解决文本中可能出现的英文单词；

3）如果是标点符号，则直接将待切分串从标点处分为前后两个子串。

预处理算法描述如下：

```
//分词预处理，将待切分串分为不同类型的子串
preProcess(String str)
  String temp = "";
  char c ← str.charAt(0);
  int lasttype ← testChar(c);
  temp += c;
  for  i = 1  to  str.length()-1
     c ← str.charAt(i);
     newType ← testChar(c);
     //判断当前字符与上一个字符类型是否相同
     if  newType = = lasttype  then  temp += c;
     else 保存 temp 子串;
          lasttype ← newType;
```

```
            temp = "" + c;
        end if
    end for
    保存 temp 子串;
end preProcess
```

其算法流程如下图所示：

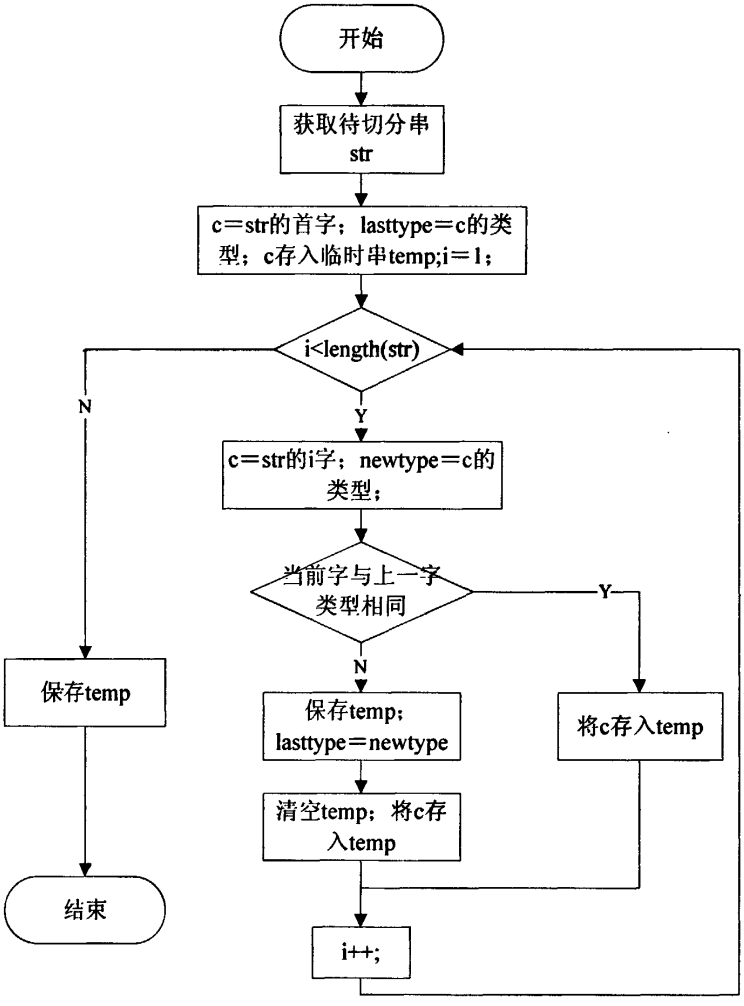

图 3-5 预处理算法流程图

3.3.4 双向最大匹配算法

经过上一节的分词预处理，将待切分串分成不同类型的文本串后，只需对其中的中文字串进行中文分词就可以了。

目前采用较多的机械分词方法是最大匹配法。基于词典的“双向最大匹配法”

是最基本的分词方法，Sun M.S.和 Benjamin K.T.统计出：汉语文本中 90%左右的句子，其与双向最大匹配的结果相吻合，而且是正确的分词结果。9.0%左右的句子正向最大匹配和逆向最大匹配不同，但其中有一个是正确的切分结果。只有不到 1.0%的句子，或者正向最大匹配和逆向最大匹配结果相同，但不正确；或者正向最大匹配和逆向最大匹配的结果不同且都不对。由此可见，双向最大匹配与词典相结合的方法是目前中文信息处理中最简单有效的方法。

最大匹配的原则就是认为最长出现在词典中的字串就是词，一旦确定一个词就不再分析该词内部是否包含长度较短的词。在最大匹配方法的指导下，中文分词可以从待切分文本的最左端或者最右端对文本进行切分，即“正向最大匹配”和“逆向最大匹配”。将同时使用两种匹配方法的分词方法称为“双向最大匹配法”。

最大匹配方法的基本思想在 2.2 节中已有介绍，在这一节就不再对其进行赘述，但是针对不同的词典结构需要对算法做出相应的改变。本文提出的双向最大匹配方法是基于 3.2 节所介绍的一种改进的中文分词词典机制的，下面分别对其正向最大匹配和逆向最大匹配两种方法进行阐述。

在进行正向最大匹配时，对待切分串的首个汉字在词典中进行 hash 查找，获得该首字下的不同长度的所有词表，按照词条长度递减的方式，取待切分串中各长度的字串，在各词表中对其进行二分查找，如果匹配成功，则切分出一个词，对待切分串的剩余部分进行相同操作，直到处理完待切分串的所有字符为止。算法具体描述如下：

```
算法：正向最大匹配
输入：汉字序列
输出：与汉字序列对应的词串
chnSegmentLR(String str)
//正向最大匹配算法，str 为待切分串
String temp ← "";        //str 中的临时子串
List list;               //存放切分出来的词条
int start = 0;           //str 剩余串的起始位置
int length ← length(str);     //str 的长度
//处理待切分串 str
while   start < length
    dictionary ← start 位置汉字的词表索引表
    //str 中 start 位置的汉字是单字词
    if   dictionary = = null
```

```
        then  start 处的汉字单独成词，存入 list;
              start ++;
        else
            maxReadLen ← dictionary 中词表的词条长度最大值;
            index ← maxReadLen 长度词条词表在 dictionary 中的位置;
            if  maxReadLen > (length - start)
                if  (length – start) = =1  then  maxReadLen ← 1;index ← -1
                else maxReadLen ← 最接近且小于或等于 length – start 的词表词
                                  条长度;
                     index ← maxReadLen 长度词条词表在 dictionary 中的位置;
                end if
            end if
            for  int i = index  to  0
                maxReadLen ← 词表 dictionary[i]的词条长度;
                temp ← str.substring(start, start + maxReadLen);
                在词表 dictionary[i]中二分查找 temp;
                if  查找成功  then  break;
                else  maxReadLen ← 1;
                end if
            end for
            temp ← str.substring(start, start + maxReadLen);
            temp 成词，存入 list;
            start += maxReadLen;
        end if
    end while
    return list;
    end chnSegmentLR
```

正向最大匹配算法流程见图 3-6。逆向最大匹配与正向类似，只是在切分过程中是从待切分传递的末尾开始的，切分时采用的词典是末字 hash 词典。其算法流程见图 3-7。

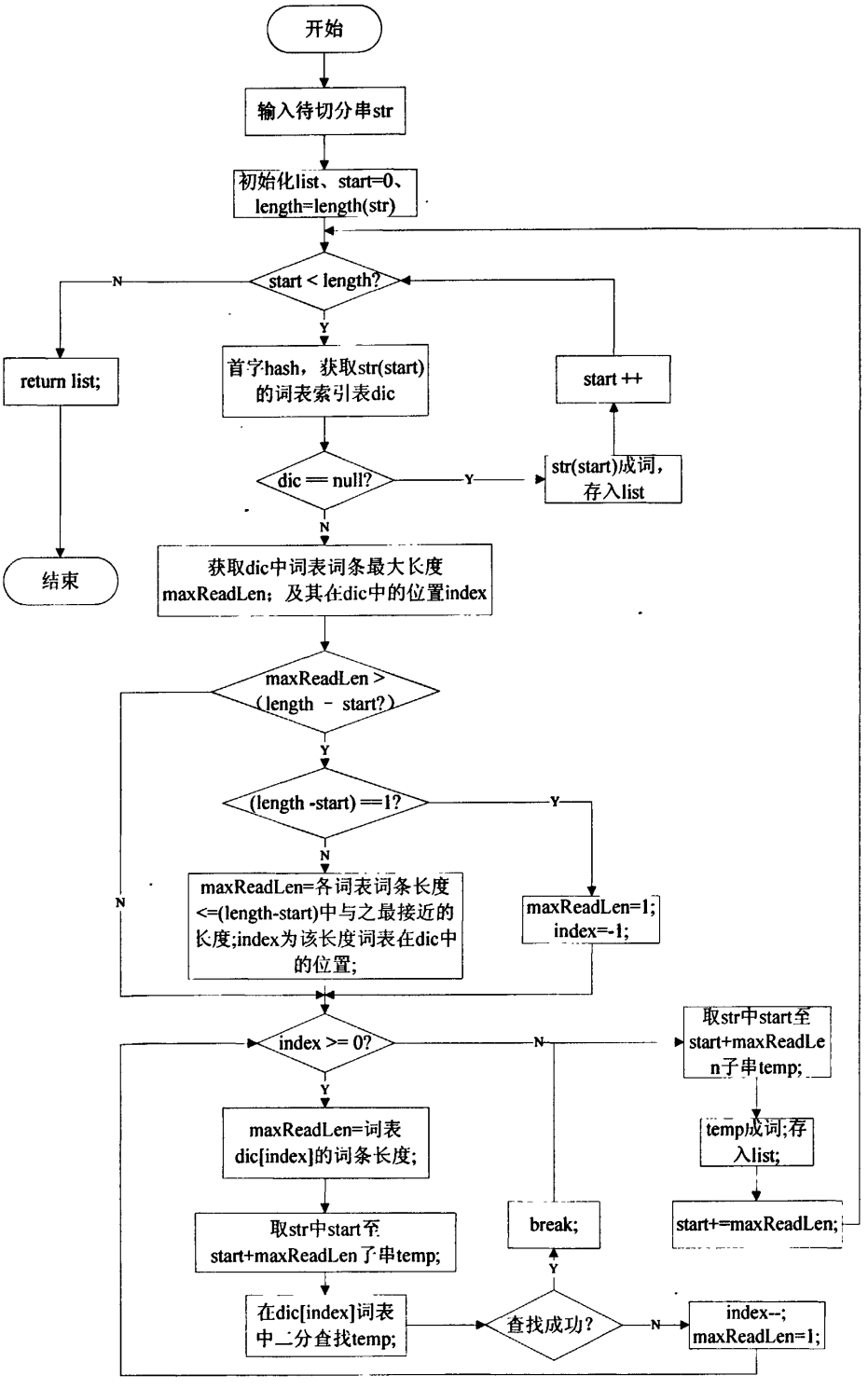

图 3-6 正向最大匹配流程图

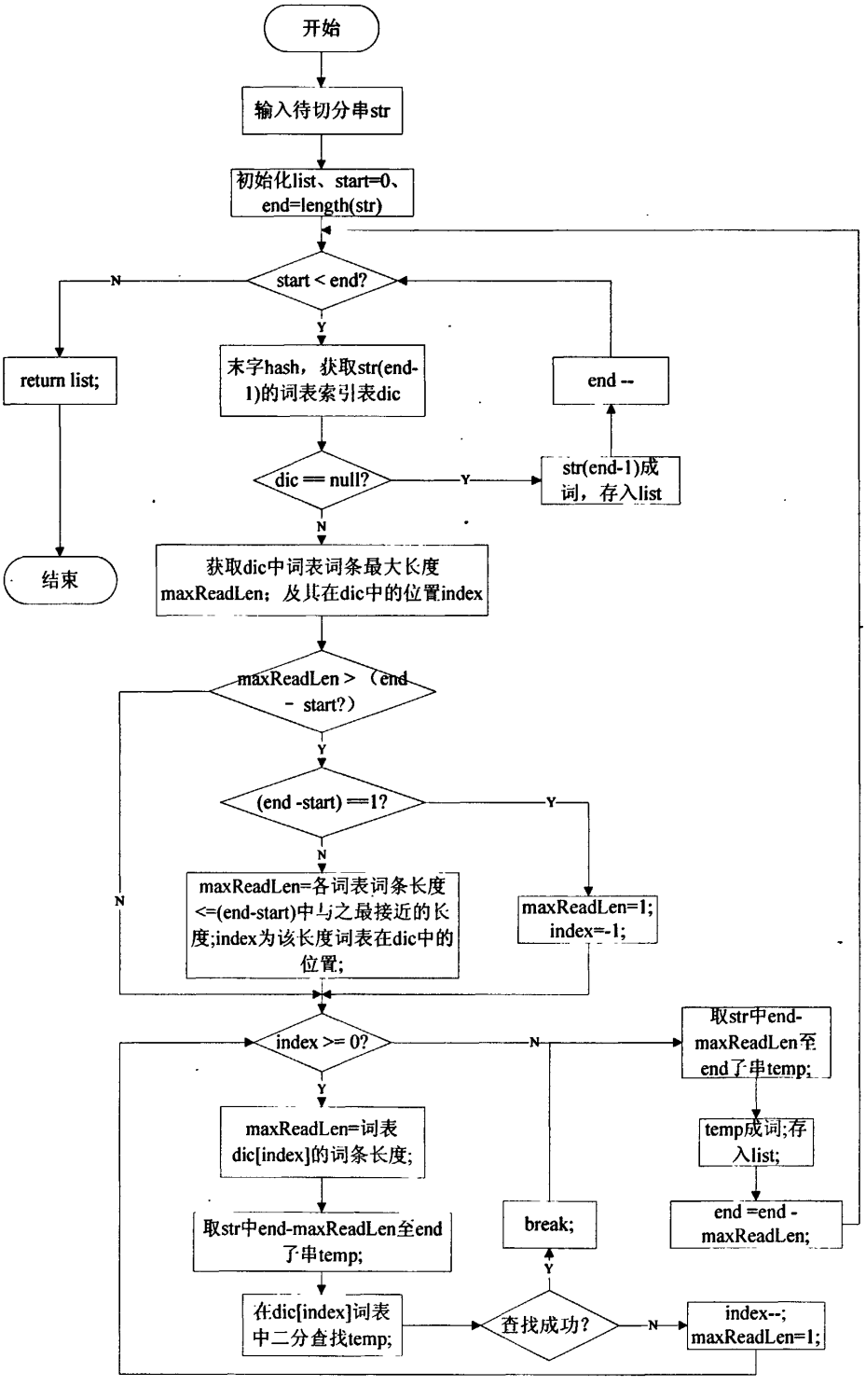

图 3-7 逆向最大匹配流程图

由于逆向最大匹配是从待切分串的后面开始进行切分，先切分出来的词是字

串末尾的词，也就先存入 list，所以在进行逆向最大匹配后，需对 list 内的元素进行倒序处理。至此，双向最大匹配处理完毕，但并不是所有的正向最大匹配结果和逆向最大匹配结果都相同，当出现不同情况时，其解决办法在下一节进行阐述。

3.3.5 歧义消解

当正、反向最大匹配算法得出来的切分结果不一样时，就必须对其进行歧义处理。要消解歧义首先必须找出歧义发生的地方，由于采用的是基于词典的最大匹配算法分词，因此本文很自然的选用双向扫描法进行歧义采集。在采集歧义时比较正、逆向最大匹配算法切分出的结果，如果切分结果不同即表明可能存在歧义就进行歧义处理。

如果正、逆向最大匹配分词切分出的字串的个数相同且对应的字串也相同则表明没有歧义直接返回切分结果。

如果正、逆向最大匹配分词切分出的字串的个数不同，则返回字串个数较少的切分结果。因为词长与词所含的信息量成正比，字段切分的个数越少，切分出的字段的词长就会比较长，信息量就越大，切分正确的可能性也就越大，所以要尽量优先保存词条长度较长的词，尽量不再切分长词。

如果正、逆向最大匹配分词切分出的字串的个数相同但对应的字串有所不同，表明仍存在歧义，根据长词优先规则，如果正向最大匹配切分出的字串中最长词条的长度大于逆向最大匹配切分出的字串中最长词条的长度，则返回正向匹配的切分结果；如果小于则返回逆向匹配的切分结果；如果相等，则根据逆向最大匹配切分准确率要高于正向最大匹配的切分准确率，同样返回逆向匹配的切分结果。

消歧算法具体描述如下：

```
getRightMean(List list1, List list2)
//list1 正向分词词串，list2 逆向分词词串
boolean equal ← true;  //正向和逆向分词结果是否相同
List list;             //保存 list1 或者 list2
String result;         //将 list 转换为字符串
if  list1 = = null  then  return;
if  list2 = = null  then  return;
//判断正、逆向匹配结果是否相同
if  length(list1) != length(list2)  then  equal ← false;  //分词个数不同
else                                                      //分词个数相同
```

```
    for  i = 0  to  length(list1) - 1
    //比较正、逆向结果中每个词条是否相等
        if  list1[i] != list2[i]
        then     equal ← false;
                 break;
    end for
end if
if  equal = = false   then  输出正、逆向分词结果;
//根据长词优先原则选择分词结果
if  length(list1) < length(list2)  then  list ← list1;
else
    if  length(list1) > length(list2)  then  list ← list2;
    else
        len1 ← list1 中最长词长度;
        len2 ← list2 中最长词长度;
    if  len1 > len2  then  list ← list1;
    else  list ← list2;
end if
//返回经过歧义处理后的字符串
for  i = 0  to  length(list) - 1
     result += list[i] + " ";
end for
return result;
end getRightMean
```

3.3.6 算法性能分析

一般观点认为，时间复杂度是衡量分词方法优劣的一个非常重要的标准。在其它条件相同的情况下，如果时间复杂度越低，则说明这种分词方法的性能越好，分词速度越快。计算分词方法的复杂度时，一般选择匹配比较次数作为衡量基本单位。分词方法的时间复杂度，确切指的是这种分词方法进行的平均每切分出一个词所需要的匹配比较次数。

进行机械匹配分词，必须有自动分词词典。现假设有这样一个分词词典，是以首字索引式结构的形式组织起来的、其收集的词汇量为 N，最长的词的长度为 i，1、2、3、...、i 字词分别有 N_1、N_2、N_3、...、N_i 个，它们在语言中出现的频度分别为 P_1、P_2、P_3、...、P_i，词典中收集的字的数目是 k，即首字索引中一共

有 k 个首字，那么，平均每个首字下有 N/k 个词，其中 1 字、2 字、3 字、…、i 字词分别有 N_1/K、N_2/K、N_3/K、…、N_i/k 个，切分出一个 1 字词、2 字词、3 字词、…、i 字词的平均比较次数分别为 T_1、T_2、T_3、…、T_i 次。这样平均每切分出一个词要进行的比较次数是：$T=\sum_{j=1}^{i}P_j\cdot T_j$，即分词算法的时间复杂度。

根据文献[21]提到的词频统计数据，如表 3-1 所示：

表 3-1 汉语词统计信息

词条字数	1	2	3	4	5	6	7
词条数	9919	65891	26352	21699	5124	2446	980
出现频率(%)	56.75	39.65	2.21	1.19	0.144	0.083	0.023

由于本文提出的词表结构支持首字 Hash 和二分查找，其时间复杂度是很低的。该词表结构类似于传统结构上的整词二分，却又有不同之处，两者的比较结果见表 3-2：

表 3-2 与传统整词二分对比分析

词典结构	占用空间	切分效率	匹配方式
传统整词二分	需要为词表中每个词条建立索引，占用空间比较大	在某个首字下的所有词条中进行二分查找，效率不高	整词二分查找
本文结构	只用为每个首字下相同词条长度的词条组成的词表建立索引，占用空间较小	只需要在某个首字下特定长度的词表内进行二分查找，查找范围减小，效率提高	整词二分查找

按照上述时间复杂度的计算方法，本文提出的分词算法的时间复杂度计算结果如下：

查找单字词不用比较，则平均比较次数为 0；

查找一个两字词的平均比较次数为：

$$0.3965*\log_2(65891/9919)\approx 1.09\text{。}$$

同理得到查找三字词或高于三字词的平均匹配次数为约 0.05。

因此本文提出的算法切分出一个词条的平均时间复杂度为 1.09＋0.05＝

1.14。这一时间复杂度要优于本文所研究的各种基于词典的分词算法的时间复杂度。下面对本文所提及的一些分词方法与本文提出的分词方法作个大致的比较，如表 3-3 所示：

表 3-3 几种分词方法的比较

分词方法	比较次数	空间复杂度	主要特征
文献[30]方法	2.89	需要为每个词条建立索引	支持首字 hash，同一首字下的词条实行顺序查找
文献[40]方法	1.66	需要为每个词条建立索引	支持首字 hash，同一首字下的词条支持标准的二分查找，使用近邻匹配法
文献[41]方法	1.25	需要为每个词条建立索引	支持首字 hash，支持二分查找，由于同一首字下前缀相同的词条组织在一起，近邻匹配效率更高
本文方法	1.14	只用为首字下各词表建立索引，但引入了标志位	支持首字 hash，同一首字下的词条按长度分类组织，二分查找范围减小

通过对比分析可知，本文提出的算法在没有增加空间复杂度的前提下，平均比较次数更小，切分效率更高。

3.4 本章小结

本章针对聊天过程中用户输入的特点，提出了基于词典的分词方法；在对汉字编码体系和现有分词词典机制研究的基础上，提出了一种改进的分词词典结构，此词典结构类似于整词二分却又不同于整词二分：首字下各词条按照词条长度分类组织在一起；基于此词典结构，提出了相应的字符串匹配算法。通过对算法性能进行分析，可知本章提出的中文分词方法在分词过程中具有很高的效率。

第四章 分词系统在聊天机器人中的应用

本章通过对聊天机器人 ALICE 系统的研究，实现了聊天机器人模块；按照第三章设计的中文分词算法，实现了中文分词功能；通过将中文分词模块集成到聊天机器人模块中去，成功地实现了聊天机器人的中文聊天。

4.1 聊天机器人系统研究

随着社会的日益信息化，人们希望能用自然语言与计算机进行交流。自然语言人机接口的研究从 20 世纪 60 年代开始，涌现出了一批该领域内有代表意义的作品，如 60 年代 ELIZA 程序、70 年代的 LUNAR 系统和 LIFER 系统、90 年代的 ALICE 聊天机器人等。本文的研究是在 ALICE 系统的基础上展开的，以下是对 ALICE 原理的阐述。

4.1.1 人工智能标记语言 AIML

ALICE 采用 AIML(Artificial Intelligence Markup Language)作为它的知识描述语言。AIML 是利用 XML 标准定义的一种服务于人工智能领域需要的特定语言，在整个聊天系统中的作用至关重要，可以说，ALICE 系统能否取得成功的关键，除了有一个好的推理机制外，更重要的还是要看 AIML 格式的知识库建设的好坏。ALICE 各种版本的核心实现是围绕方便、有效、快捷地组织和检索 AIML 知识分类进行的，因此，在论述 ALICE 内部推理机制之前，应该对 AIML 做一个简单的介绍。

1.AIML 语法构成要素

在 AIML 中，基本的知识单元是由分类<category>构成的，而每一个分类又是由用户输入的问题、ALICE 输出的答案和可选上下文环境组成[42]。一个简单的分类如下所示：

```
<category>
    <pattern>你好</pattern>
    <template>你也好！ </template>
</category>
```

其中模式<pattern>部分代表用户输入的问话，模板<template>部分则代表用户输入这一问句后，系统应该给出的答案。当然，AIML 还有其它许多重要标记，如递归调用标记<srai>，随机选择标记<random>等。AIML 规范中还定义了星号

(*)和下划线(_)两个通配符，模式中出现这两个通配符的地方代表此处可以与一个或任意多个单词匹配成功，匹配时优先匹配下划线。对上面的例子来说，当用户输入“你好”，系统在找到对应的模板后返回“你也好！”作为结果。有关 AIML 的详细用法可以参阅文献[43]。

2.AIML 知识库结构

AIML 知识库是由多个以 AIML 为后缀名的文件组成，每个 AIML 文件可以代表一个领域知识，比方说关于生物方面的知识，我们可以把它组织到 biological.aiml 文件中，这样可以便于知识库的分类管理。一个简单的 AIML 文件内容如下所示：

```
<?xml version="1.0" encoding="gb2312"?>
<aiml>
  <category>
    <pattern>how do you do</pattern>
    <template>how do you do</template>
  </category>
  <category>
    <pattern>How are you</pattern>
    <template>Fine, thanks!And you?</template>
  </category>
  ……
</aiml>
```

显然，一个 AIML 文件就是一个结构良好的 XML 文件[44]，通过文字处理工具就可以增、删、改、查知识分类。

AIML 就像一个关于问题和答案集的简单数据库，其中模式部分的查找和 SQL 查询相似，甚至更简单，由于模板中还有可能包含递归调用标记，因此一个问题的答案输出并不一定仅仅依赖于第一个匹配上的分类，还与递归调用标记中的内容有关。

4.1.2 ALICE 内部推理机制

1.ALICE 知识组织

ALICE 把所有的知识以树的形式组织到一个称为 Graphmaster[44]的对象中，Graphmaster 是由一系列称为 Nodemapper 的节点集组成的。每一个 Nodemapper 节点都有一些从该节点出来的分支，这些分支可以是一个单词，也可以是一个通配符。如果加载官方公布的英文 AIML 知识库，Graphmaster 根节点的 Nodemapper 有大约 2000 个分支，每一个分支代表一个 pattern 的第一个单词。虽然 ALICE

存储了 40000 多条 category，但由于许多 pattern 的第一个单词相同，这些不同的 pattern 在 Nodemapper 中就指向了同一个分支节点，这样的话根节点也就只有 2000 多个子分支了，从而大大节省了内存空间。第二级分支节点的 Nodemapper 是 pattern 中的第二个单词，它们的子分支指向紧跟在其后的第三个单词或通配符。依此类推，直到子节点的分支到了 pattern 的末尾为止。此时，把 category 中该 pattern 对应的 template 内容存入该节点，这样 Graphmaster 就构成了一棵内存树，树的叶子节点存储 template 信息，叶子节点的数目正好与知识库 pattern 的数目相同。

下面举个简单的例子，有以下 3 个句子：①where is it；②where are you；③how are you，其内部组织形式如图 4-1 所示：

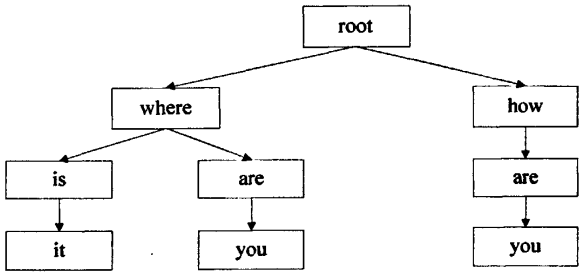

图 4-1 ALICE 知识组织示例图

2.ALICE 核心推理

根据用户的输入查找对应答案(template)的过程就是 ALICE 的推理过程。假设用户输入的问句以单词 X 开头，那么首先将 X 与 Graphmaster 中第一级节点的内容逐个比较，整个推理过程分为三步：

1) 当前节点是否包含通配符“_”？如果有，则搜索以当前节点为根节点的子树，以 X 后面剩余部分组成的子句在该子树中继续该过程，如果没有相匹配的节点，则转 2)；

2) 当前节点是否包含 X？如果是，则搜索以当前节点为根节点的子树，以 X 后面剩余部分组成的子句在该子树中继续该过程，如果未发现相匹配节点，则转 3)；

3) 当前节点是否包含“*”？如果是，则搜索以当前节点为根节点的子树，以 X 后面剩余部分组成的子句在该子树中继续该匹配过程。如果没有发现匹配节点，回溯到节点的父节点，把 X 重新加到子句子的首部，继续该匹配过程。

当进行匹配的句子达到句尾，并且匹配的最后一个节点包含 template 内容时，则表明已经找到了相匹配的 pattern，此时，终止搜索匹配过程，返回该节点，取出 template 内容，把结果返回用户。从以上推理过程可以看出，如果知识树中

第一级节点含有星号(*)关键字，并且该节点有 template 信息，则无论输入什么问句，都可以得到一个相同的结果。

3.ALICE 工作流程

ALICE 系统工作流程如图 4-2 所示：

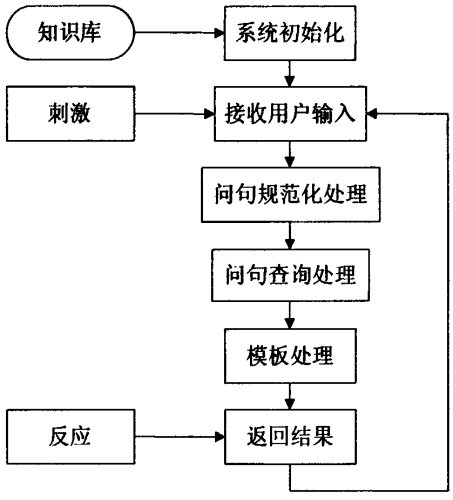

图 4-2 ALICE 系统工作流程

1) 系统初始化：ALICE 系统在启动时，首先根据配置文件进行系统的初始化操作，把需要替换的词串、自身的相关信息、人称转换信息以及此前的对话情景变量读入内存当中，形成内存知识树。加载完成后，等待用户输入问句。

2) 接收用户输入，进行问句规范化处理：当 AIML 解析器接收到用户的输入后，首先把输入的文字分成单独的句子，进行问句规范化处理。之后，以规范问句到内存知识树中查询推理答案。

3) 问句推理查询：这一过程是 ALICE 的核心部分，将规范化处理后的问句与内存知识树中的模式进行匹配，寻找最佳匹配结果，找到后，读出该匹配模式对应的模板信息，进行下一步处理。

4) 模板处理：也就是答案的后处理，模板中可能包含一些特殊标记需要处理，比如包含跳转标记，还需要在内存知识树中以跳转部分的内容做进一步的推理。

模板处理完后把结果返回给用户，等待用户输入新语句。

4.2 聊天系统总体结构

在对 ALICE 系统的原型进行分析和研究的基础上，针对中文句子和西方语言句子的不同，通过在 ALICE 系统中加入一个中文分词模块，使其能够支持人

机的中文聊天。加入分词模块后的系统结构图如下所示：

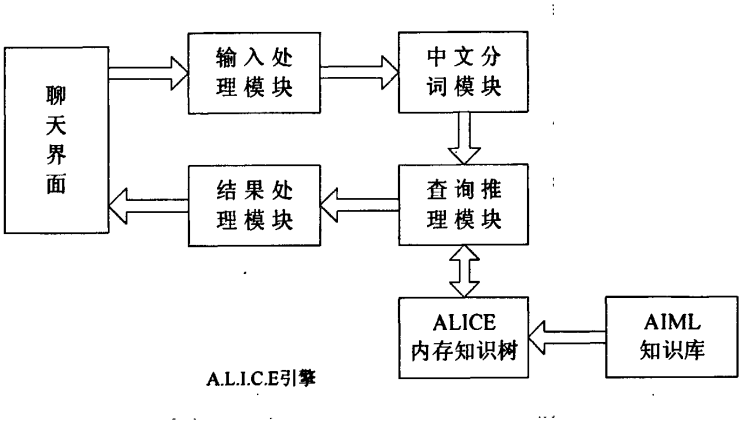

图 4-3 系统结构图

由于加入了中文分词处理模块，在系统初始化时，需要加载聊天所需的问答知识库及分词所需的词库；聊天过程中，在对用户所输入的句子进行一般规范化后，需要再进行中文分词处理，才能满足中文聊天的要求。因此加入中文分词后，聊天系统的工作流程如图 4-4。

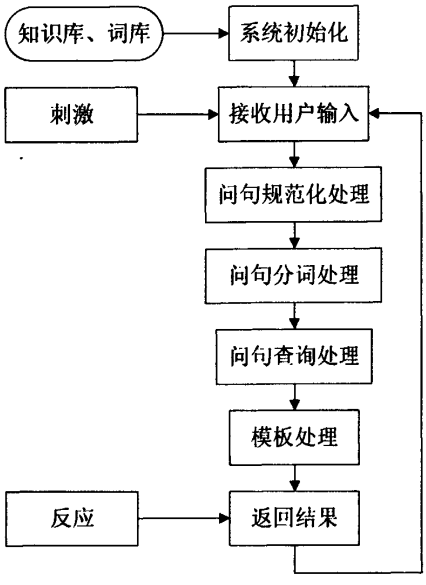

图 4-4 改进后的系统工作流程

4.3 聊天机器人模块

系统采用的开发语言是 JAVA[46]语言，开发环境为 JBuilder 9。系统把聊天机器人模块组织在 csu.bot 包中，主要包括以下几个类：

AIMLLoader 类：主要负责 AIML 文件的读取、解析；

GraphMaster 类：存储知识树的根节点，提供知识库加载和模式推理接口；

Nodemapper 类：负责知识树的构建，模式推理功能；

Bot 类：负责系统初始化工作，反馈聊天结果；

Globals 类：提供系统的全局信息，包括替换词串、人物转换、知识库的存放路径等信息。

根据对聊天机器人原型系统的分析，系统的核心类是 Nodemapper。本节主要围绕 Nodemapper 类介绍 ALICE 知识库的加载，及搜索推理机制的实现。其 UML 图如图 4-5 所示：

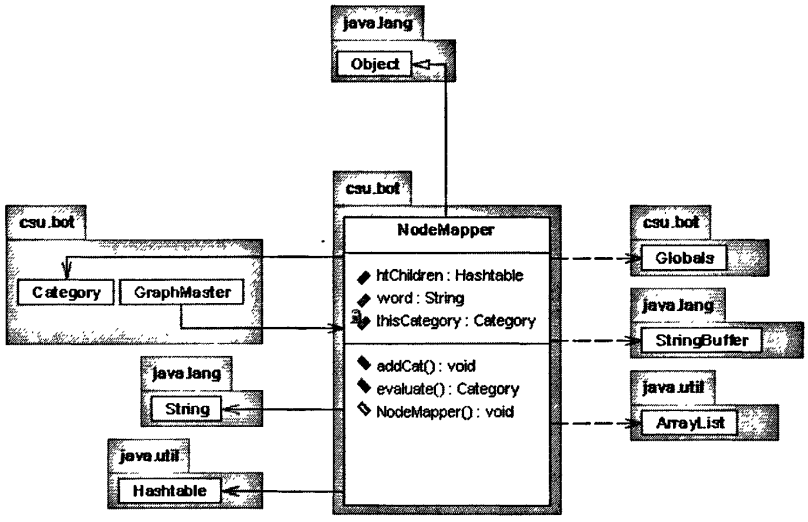

图 4-5 Nodemapper UML 图

ALICE 知识树的节点用 Nodemapper 类型表示。由图可知，Nodemapper 对象包括三个成员属性：其中 htChildren 存储子树根节点；word 保存该节点的关键词；thisCategory 用于存储一条知识，当节点是非叶子节点时 thisCategory 为空，也即所有的知识都是存储在叶子节点中。

ALICE 知识树的根节点存储在 GraphMaster 对象中。从根节点到叶子的路径上所经过的所有节点中的关键词序列便是叶子节点中所存储知识的模式部分。

4.3.1 知识树构建

由于组成 AIML 知识库的 AIML 文件是结构良好的 XML 文件，系统采用 JDOM[45]对 AIML 文件进行解析。JDOM 是一种解析 XML 的独特 JAVA 工具包，用于快速开发 XML 应用程序。在 JDOM 中，XML 元素是 Element 的实例，XML 属性是 Attribute 的实例，XML 文档本身是 Document 的实例。

在加载知识库的过程中，系统先获得 XML 文档，通过

Document.getRootElement 方法得到文档的根节点元素，最终能得到标签为<category>的所有知识。在构建 ALICE 知识树之前，需要对<category>中<pattern>的文本进行规范化、分词等处理，从而得到与<pattern>中的文本相对应的词串（词与词之间以空格隔开），然后通过调用 Nodemapper.addCat(String , Category)方法即可构建出一棵 ALICE 知识树。

Nodemapper 构建知识树的思想是：判断 String 是否为空串，如果条件为真，则当前 Nodemapper 节点为一个叶子节点，把 Category 代表的知识存入该叶子节点；否则获取 String 中的首词 first，在当前节点中查找是否存在包含 first 的子节点，如果条件为真，则调用子节点的 addCat 方法，子节点方法的输入串为当前节点中去掉 first 后的剩余串；否则新建一个 Nodemapper 节点，新建节点的关键词即为 first，将新建节点作为当前节点的子节点，调用新建节点的 addCat 方法，同样新建节点方法的输入串也是当前节点去除 first 后的剩余串。至此则完成了知识树中一条知识的插入。

4.3.2 推理模块的实现

根据改进后的系统工作流程可知在进行问句查询推理之前，需要先对输入问句进行规范化、分词等处理。在对问句经过一系列处理之后，得到的是与问句对应的词序列（同上节处理 pattern 元素一样）。根据问句词序列在知识树中查找模式的过程就是系统的推理过程。推理功能由 Nodemapper.evaluate()方法实现，其在模式搜索时采用了带有回溯的递归过程，该方法的具体思想在 4.1 节中已详细阐述，由于子节点是存储在 Hashtable 类型的 htChildren 对象中，通过一次 hash 就能很方便地找到其子树根节点。

4.4 中文分词模块

分词模块组织在 csu.seg 包中，对外提供一个分词接口，其中几个主要的类如下：

HashDictionary 类：实现词典 hash 表部分，提供词典加载和查找词表索引表；

Dictionary 类：实现词表索引表及提供查找、插入等操作；

SubDictionary 类：实现词表，及基于词表的查找、插入等操作；

MaxMatchSegment 类：提供中文分词功能；

通过上述类所提供的方法，本节主要介绍词典的实现和基于词典的分词算法实现。

4.4.1 词典实现

根据 3.2 节所介绍的词典结构，系统分词模块采用的词典是一个三级结构。

Hash 表由 HashDictionary 类中所定义的 Hashtable 类型的成员属性表示，其存储的键值对中的键表示首字或末字，值即一个 Dictionary 对象；处于中间级的词表索引表由 Dictionary 类中定义的 List 类型的成员属性表示，其元素类型为 SubDictionary；词表由 SubDictionary 类中 ArrayList 类型的成员属性表示，其存储的即为汉语词条。

设计好词典结构的框架之后，在词典加载过程中，对于一个特定词条，通过 Hashtable.get()方法可以获得该词条首字或末字的 Dictionary 对象，如果没有该对象，则创建一个 Dictionary 对象，与词条首字或末字形成键值对存入 Hashtable 中；然后在词典中加载一个词条的工作便由找到的或者新创建的 Dictionary 对象去完成，Dictionary 对象在 List 类型的成员属性中查找有无与这个词条长度对应的 SubDictionary 对象，如果没有则创建一个 SubDictionary 对象，将此对象加入到词表索引表中；最后，只需将待加载词条加入到 SubDictionary 对象的词表中去。

4.4.2 分词算法实现

系统所采用的分词算法是基于上一节中实现的词典而设计的。整个分词方法封装在 MaxMatchSegment 类中，该类的 UML 图如图 4-6。

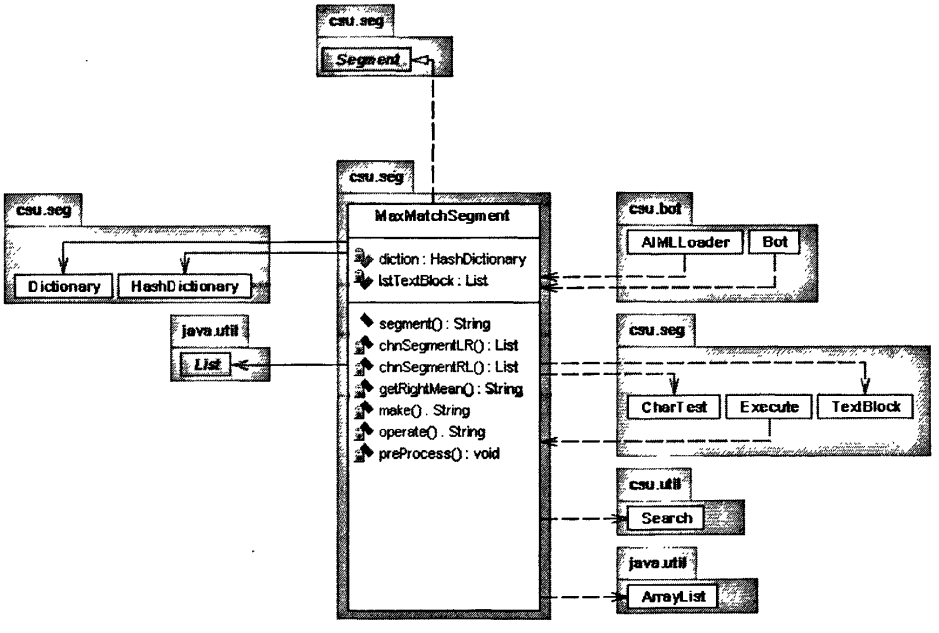

图 4-6 MaxMatchSegment UML 图

由图 4-6 可以看出，MaxMatchSegment 实现了 Segment 接口，此接口包含一个 segment()方法。以下对 MaxMatchSegment 类的各个方法进行一一介绍。

preprocess()方法：该方法实现对待切分文本的预处理，将文本分为不同类型的文本块；并将各文本块存入 List 类型的 lstTextBlock 对象中；

chnSegmentLR()方法：实现汉字串的正向最大匹配；

chnSegmentRL()方法：实现汉字串的逆向最大匹配；

getRightMean()方法：对正反向匹配得出的不同结果进行处理，返回其中较好的结果；

make()方法：获取预处理阶段产生的不同类型文本块；

operate()方法：对不同类型的文本块采用不同的处理方法，如果是数字或英文，则原样返回；如果是汉字类型，则对其进行分词处理；如果是其它类型，则以空格符代替原文本块；

segment()方法：调用上述方法，实现分词功能；

其中 segment()方法是提供给聊天模块的接口，对外开放；而其它方法都是 private 类型的，不对外公开。

4.5 实验结果

与聊天系统交互效果的好坏，与 AIML 知识库建设的规模和知识的覆盖面至关重要，目前关于西方语言方面的 AIML 知识库已出具规模。

```
<category>
<pattern>What's your name</pattern>
      <template>
          My name is ALICE.
      </template>
</category>
<category>
<pattern>你是谁</pattern>
      <template>
          我是ALICE，你呢?
      </template>
</category>
<category>
<pattern>你好啊</pattern>
      <template>
          嗯，你也好
      </template>
</category>
<category>
<pattern>How are you</pattern>
      <template>
          Fine, thanks
      </template>
</category>
<category>
<pattern>很高兴认识你</pattern>
      <template>
          呵呵，我也是
      </template>
</category>
```

图 4-7 AIML 知识库样本

针对中文编写的AIML尚未发现相关资料，单凭个人的力量很难建设好一个具有一定规模的AIML知识库。因此本文为了试验系统的可行性，只设计了一个用于实验的AIML文件。其部分内容如上图所示。

4.5.1 分词效果

基于词典的分词方法，其切分精度很大程度取决与词库词条的覆盖面，系统所采用的词库是根据中文词库素材分类整理而来的。下面是随便选取的一段文字，对其进行分词后，其效果如下所示。

待切分串：一位英国朋友来中国旅行，在我们家住了一个星期。最初，我们专门请了一个从上海来的西餐厨师给他做面包蛋糕。英国朋友直率地说："我不是为了吃西餐或者名为西餐实际上四不像的东西而来的，请把你们的具有古老传统和独特魅力的饭给我弄一点吃吧，可以吗？"怎么办呢？只好很不好意思地招待他吃稀饭和咸菜了。

切分结果："一位 英国 朋友 来 中国 旅行 在 我们 家住 了 一个 星期 最初 我们 专门 请了 一个 从 上海 来的 西餐 厨师 给他 做 面包 蛋糕 英国 朋友 直率 地 说 我不是 为了 吃 西餐 或者 名为 西餐 实际上 四不像 的 东西 而 来的 请 把 你们的 具有 古老 传统 和 独特 魅力 的 饭 给我 弄 一点 吃 吧 可以吗 怎么办 呢 只好 很 不好意思 地 招待 他 吃 稀饭 和 咸菜 了"。

其中正反向分词不同的文本块分词如下：

正向分词："我们 专门 请了 一个 从上 海 来的 西餐 厨师 给他 做 面包 蛋糕 "

反向分词："我们 专门 请了 一个 从 上海 来的 西餐 厨师 给他 做 面包 蛋糕 "

正向分词："我不是 为了 吃 西餐 或者 名为 西餐 实际上 四不像 的 东西 而来 的 "

反向分词："我不是 为了 吃 西餐 或者 名为 西餐 实际上 四不像 的 东西 而 来的 "

正向分词："请 把你 们 的 具有 古老 传统 和 独特 魅力 的 饭 给我 弄 一点 吃 吧 "

反向分词："请 把 你们的 具有 古老 传统 和 独特 魅力 的 饭 给我 弄 一点 吃 吧 "

字符数：143

处理时间：31毫秒

4.5.2 聊天结果

在没有加入中文分词之前，ALICE 不能支持中文聊天，虽然 AIML 知识库中包含有与用户输入对应的知识；在聊天系统中加入中文分词模块后，系统具有了中文聊天的功能。其聊天结果见图 4-8 和图 4-9。

以下是没加入中文分词之前的聊天界面，此系统是用 C#开发实现的。

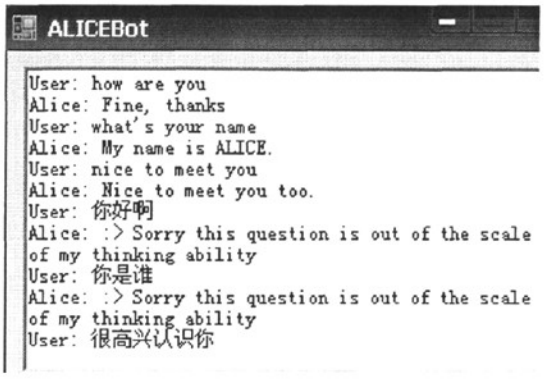

图 4-8 改进前的聊天界面

以下是本文实现的聊天系统界面图。

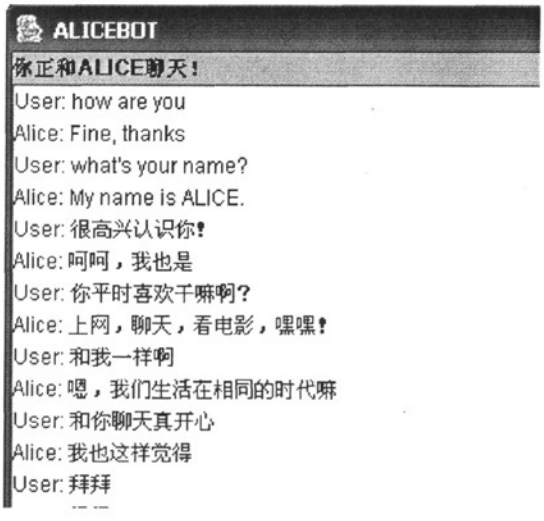

图 4-9 改进后的聊天界面

对比两种聊天结果，可知在没加入中文分词模块之前，ALICE 系统不支持中文聊天，对用户输入的中文句子，只返回一个默认结果；本文在 ALICE 系统中加入中文分词功能之后，系统能在知识库中检索匹配到用户输入的正确结果并返回给用户。

4.6 本章小结

本章对聊天机器人原型系统进行了研究分析，得出其在中文聊天方面存在先天缺陷；通过把中文分词模块集成到 ALICE 系统中去，在能进行西方语言聊天的基础上，实现了聊天系统的中文聊天功能。在分词过程中，当正反向最大匹配结果不尽相同时，分词模块能选取其中切分正确率高的结果返回。通过实验分析表明，本课题的研究成果是确实可行的。

第五章 总结与展望

5.1 论文总结

随着自然语言处理技术的日趋成熟，使得我们应用计算机分词技术来解决智能聊天机器人系统中的关键问题成为可能。这种技术和基于关键词的搜索技术有很大不同，它通过利用自然语言处理中的词语切分技术使用户可以直接使用自然语言与计算机交互，得到简洁、准确的回复，而不是抽取搜索问句中的关键词。本文在理论研究和实际应用方面结合得很好，具有重要的现实意义。

本文以中文自动分词技术为核心，在聊天过程中通过对用户输入进行一系列的处理，研究了分词技术如何提高切分的效率和准确率，得到与输入相对应的词串，使计算机能更好的理解用户意图，并初步实现了分词技术在智能聊天系统中的应用。

论文的具体工作如下：

1、分析了中文自动分词技术的发展状况，研究中文分词技术的意义，分词技术的困难所在，分词系统的评价准则，以及常用的分词技术方法和分词词典机制。简要说明了几个现有的分词系统以及它们使用的分词方法和取得的成果。

2、分析了智能聊天机器人系统的发展现状，以及系统中存在的缺陷：在聊天过程中，对用户输入的处理以词为单位，而中文句子中词与词之间没有明显的分隔符，因此聊天机器人系统不支持中文聊天。

3、本文根据现有的中文分词技术及分词词典机制的研究成果，提出了一种改进的分词词典机制，在此词典的基础上辅以相应的中文分词算法，提高了分词效率；在分词过程中结合长词优先原则，能解决一定的歧义问题。通过对算法进行性能分析，可知分词算法具有比较低的时间复杂度。

4、根据智能聊天系统中用户输入的特点，选取了一种具有针对性的分词方法，本文提出了将中文自动分词技术应用到尚且不能支持中文聊天的智能聊天系统中去，有效地实现了中文分词和聊天系统的集成，解决聊天系统不能支持中文聊天的不足，并取得了初步的成效。

虽然如此，本文的研究还是存在不足之处，比方说只注重分词效率，忽略了分词精度这一重要指标，使得分词精度不高，而分词精度直接影响到聊天系统的查询推理工作；论文尚未涉及词性词义问题、未登录词识别方面，这些都还有待于进一步的研究。

5.2 课题展望

虽然我国在中文分词领域达到了一个非常先进的水平，但是对于分词中所涉及的一些关键问题，仍然没有得到很好的解决方案。因此中文信息处理技术的进步和中文信息处理系统的广泛应用，有待于对分词中的关键问题进行更深入的研究，如机械分词中通用分词词表的制定，研究歧义切分字段，增强歧义判别能力，提高专有名词的识别率，研究汉语构词规则和词法规则等。

聊天机器人是问答系统的一种，通过对 ALICE 系统进行改造，设计实现的中文聊天机器人，在网络教学、智能答疑、Web 服务等应用中都有广泛的应用前景。同时对于如何更有效地组织、检索和获取知识；如何合理扩充 AIML 语言以增强它的知识表达能力；汉语句子句序自由，存在大量同义句的问题；受知识库大小的限制，存在匹配不到用户输入的知识的情况，如何提供聊天机器人自主学习的功能等等，这些都值得作进一步深入的研究。

参考文献

[1] Turing A.M. Computing machinery and intelligence[J]. Mind, 59(236): 433-460

[2] John R Searle. Minds, brains, and programs[J]. Behavioral and Brain Sciences, 1980, (3): 417-424

[3] J. Weizenbaum. ELIZA-A Computer Program for the Study of Natural Language Communication between Man and Machine. Communications of the ACM, 9(1): 36-45

[4] Wallace RS. The Anatomy of ALICE[EB/OL]. http://www.alicebot.org /anatomy.html

[5] 夏天, 樊孝忠, 刘林. ALICE 机理分析与应用研究[J]. 计算机应用, 2003, 23(9): 1-5

[6] 刘颖. 计算语言学[M]. 北京: 清华大学出版社, 2002

[7] 陈力为. 汉语书面语的分词问题——一个有关全民的信息化问题. 中文信息学报, 1996, 10(1): 11-13

[8] 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): 28-31

[9] 冯书晓, 徐新, 杨春梅. 国内中文分词技术研究新进展. 中文信息处理, 2005, (11): 29-30

[10] 孙晓, 黄德根. 基于动态规划的最小代价路径汉语自动分词. 小型微型计算机系统, 2006, 27(3): 516-519

[11] 冯冲, 陈肇雄, 黄河燕等. 基于 Multigram 语言模型的主动学习中文分词. 中文信息学报, 2006, 20(1): 50-58

[12] 韩客松, 王永成, 陈桂林. 汉语语言的无词典分词模型系统. 计算机应用研究, 1999, 31(10): 8-9

[13] Huang De-gen, Zhu He-he, Wang Kun-lun. Chinese Automatic Words Segmentation Based on Maximum-matching and Second-maximum Matching. Journal of Dalian University of Technology, 2005, 39(6): 831-835

[14] Liang Nan-yuan. CDWS:A Word Segmentation System for Written Chinese Texts. Journal of Chinese Information Processing, 1987, (2): 44-52

[15] Huang Chang-ning. Chinese Word Segmentation in Chinese Information Processing. Applied Linguistics, 1997, 21(1): 72-78

[16] 韩客松, 王永成, 陈桂林. 无词典高频字串快速提取和统计算法研究[J].

中文信息学报, 2000, 15(2): 23-29

[17] 刘开瑛. 歧义切分与专有名词识别软件. 语言文字应用, 2001, 8(3): 16-23

[18] 牛耘, 朱献有. 神经网络技术在汉语歧义切分中的应用. 情报学报, 1999, 18(3): 37-45

[19] 孙茂松, 邹嘉彦. 汉语自动分词研究中的若干理论问题. 语言文字应用, 1995, (4): 32-35

[20] Sproat R, Shih C. A Stochastic Finite-state Word Segmentation Algorithm for Chinese. Computational Linguistics, 1996, 22(3): 377-404

[21] Lai B.Y, Sun M.S. Chinese Word Segmentation and Part-of-speech Tagging in One Step. Research on Computational Linguistics, 1997: 229-236

[22] 孙茂松, 黄昌宁, 高海燕. 中文姓名的自动辨识. 中文信息学报, 1995, 9(2): 16-22

[23] 郑家恒, 李鑫, 谭红叶. 基于语料库的中文姓名识别方法研究. 中文信息学报, 2000, 14(1): 38-44

[24] 马哲, 姚敏. 一种改进的基于 PATRICIA 树的汉语自动分词词典机制. 华南理工大学学报(自然科学版), 2004, (32): 28-32

[25] 孙茂松, 左正平, 黄昌宁. 汉语自动分词词典机制的实验研究. 中文信息学报, 2000, 14(1): 1-7

[26] 冯书晓, 徐新, 杨春梅. 国内中文分词技术研究新进展[J]. 情报杂志, 2002, (11): 29-30. (12): 21-24

[27] winter. 中文搜索引擎技术揭密：中文分词(网络文章). http://www.fullsearcher.com

[28] 梁南元. 书面汉语的自动分词与一个自动分词系统. 北京航空学院学报, 1984, (1): 25-29

[29] Huang De-gen, Zhu He-he, Yang Yuan-sheng. Chinese Automatic Words Segmentation Based on Word Reliability and Double Words Reliability. Journal of Computer Research and Development, 2001, 19(7): 132-135

[30] 吴胜远. 一种汉语分词方法. 计算机研究与发展, 1996, 33(4): 30-32

[31] 孙茂松, 黄昌宁, 邹嘉彦等. 利用汉字二元语法关系解决汉语自动分词中的交集型歧义[J]. 计算机研究与发展, 1997, 34(5): 332-339

[32] 丁承, 邵志清. 基于字表的中文搜索引擎分词系统的设计与实现[J]. 计算机工程, 2001, 27(2): 191-193

[33] 梁南元. 书面汉语自动分词系统——CDWS. 中文信息学报, 1987, 2(2)

[34] 肖红, 许少华, 李欣. 具有三级索引词库结构的中文分词方法研究. 计算机应用研究, 2006, 25(8): 49-51

[35] 徐华中, 徐刚. 一种新的汉语自动分词算法的研究和应用. 计算机与数字工程, 2006, 34(2): 135-138

[36] 张民, 李生, 王海峰. 基于知识评价的快速汉语自动分词. 系统情报学报, 1996, 15(2): 4-13

[37] Cheng K.S, Young G, Wong K.F. "A study on word-based and integral-bit Chinese text compression algorithms". Journal of the American Society for Information Science, 1999, 50(3): 218-228

[38] Ye Wang, Shang-Teng Huang, "Chinese word segmentation based on a priori and adjacent characters". Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, Guangzhou, 18-21 August 2005

[39] Sproat R, Shih C. "A Statistical Method for Finding Word Boundaries in Chinese Text". Computer Processing of Chinese and Oriental Languages, 1990, (4): 336-351

[40] 陈桂林, 王永成, 韩客松, 王刚. 一种改进的快速分词算法[J]. 计算机研究与发展, 2000, 37(4): 418-423

[41] 张海营. 全二分快速自动分词算法构建[J]. 现代图书情报技术, 2007, (4): 52-55

[42] Wallace RS. AIML Overview[EB/OL]. http://www.pandorabots.com /pandora /pics /wallaceaimltutorial.html

[43] aiml 规范研究文档. http://blog.csdn.net/IceSun963/archive/2008/05/21 /2466205.aspx

[44] 戴开宇, 张申生, 王淼. 分布式虚拟环境中的聊天机器人的研究[J]. 计算机工程与应用, 2002, (7): 13-16.

[45] 用 JDOM 简化 XML 编程. http://www.ibm.com/developerworks/cn/java /j-jdom/

[46] Bruce Eckel. Thinking in Java[M]. 北京: 机械工业出版社, 2005, 291-378

致 谢

本文从题目的确定到框架的构建，直至正文的写作，无不倾注了导师的大量心血。感谢我的导师廖志芳副教授三年来对我的悉心指导和严格要求，使得我的理论水平有很大提高，同时还让我学到了许多为人治学的道理，廖老师严谨的治学态度、循循善诱的教诲、渊博的学识都让我受益匪浅。在学习和写论文期间，许多学术上的疑问都是得到了廖老师的细心回答和启发才得以解决。在此谨向廖老师致以诚挚的谢意！感谢导师在做人做事上给我的教诲，感谢导师在科研上用长者的睿智给予我方向性的指导。

感谢樊晓平老师一直以来对我的指导和帮助。三年来，每周四下午樊老师主持的学术研讨活动使我有了发表学术看法、请教师长疑难问题的机会，更使我了解到学术研究的最新动态，促使我努力去深入研究学术问题。没有樊老师给我创造的一个学术研究的环境，不会有我三年的研究成果。衷心的感谢樊老师！

感谢智能信息与控制实验室的刘少强老师、杨胜跃老师、张亚鸣老师、郁松老师、黎燕老师，谢谢他们给予我热情的关心与指导！

感谢同窗罗浩、蒋玉娟、周筠、周明伟、唐泉、胡松、胡志华、刘铮、周智洋，有缘做同学，一起学习生活、一起找工作的日子历历在目，我们相互帮助和勉励完成各自的学业！

感谢师兄刘映辉、刘克准、熊科、熊哲源、李奇、杨石磊、陈宇宙，师姐王双维、杨帆、陈洁洁无私的帮助和指点！师妹周婷、张玉娟，师弟陈磊、洪丹龙的帮助和支持！

感谢百忙之中抽空审阅我论文的老师！感谢信息院、研究生院的领导及相关老师！

最后，我要感谢我的家人，他们的关怀、督促和鼓励是我前进的最大动力。

攻读硕士学位期间完成的学位论文

完成论文：

廖志芳，李鹏，刘克准等. 数据聚类分析新方法研究. 计算机工程与应用，2009，45(10): 147-150.

廖志芳，李鹏，樊晓平. 中文分词在智能聊天机器人中的研究与应用，已投稿

中文分词在聊天机器人中的应用研究

作者： 李鹏
学位授予单位： 中南大学

相似文献(2条)

1.期刊论文 易顺明.胡振宇.YI Shun-ming.HU Zhen-yu 中文聊天机器人原型系统的设计 -沙洲职业工学院学报

2007,10(2)

探讨了中文聊天机器人原型系统的设计方式,提出了采用中文分词与词,形成规则库与规则树.聊天时,采用带回溯的深度优先算法,找到最佳匹配路径,给出回答.

2.学位论文 于晓燕 基于自然语言接口的虚拟参考咨询系统设计与实现 2004

近几年来，虚拟参考咨询(VirtualReferenceServices，简称VRS)或称数字参考咨询(DigitalReferenceServices，简称DRS)成为国内外图书馆学界关注的一大热点，它是网络环境下图书馆参考咨询服务的主流发展方向，甚至被认为是未来图书馆的核心工作之一。

本文通过对当前各种参考咨询服务模式优缺点的比较，提出了一种新的设计思想：基于自然语言接口的虚拟参考咨询系统。该系统将计算机人工智能领域中的模式识别技术、自然语言处理技术引入图书馆虚拟参考咨询系统中，以人工智能标记语言(AIML)为基础，加入了中文分词模块，利用有效的知识库构建方法，设计实现了一套具有小型知识库的、基于WEB的聊天机器人系统。

本文首先描述了参考咨询服务的产生与发展，并对各种虚拟参考咨询服务的模式进行了分析比较；接着具体描述了基于自然语言接口的虚拟参考咨询系统的实现原理以及人工智能标记语言(AIML)的语法构成；然后结合本系统的具体实现分别阐述了中文分词技术、ISAPI技术、AIML知识库的构建方法和系统运行评价；最后，本文总结了该系统目前存在的问题以及未来的发展方向。

在国内，虚拟参考咨询系统的研究和实施还处于初级阶段，特别是开发基于自然语言模式识别的智能化系统工作尚在摸索之中，本文在这方面作了有益的尝试。