# 专利文献翻译中并列结构的处理

赵 然,晋耀红
ZHAO Ran,JIN Yao-hong

中国科学院 声学研究所,北京 100190
Institute of Acoustics,Chinese Academy of Sciences,Beijing 100190,China
E-mail:zhaoran@zhaoran.net

**ZHAO Ran,JIN Yao-hong.Processing of parallel structures in patent articles translation.Computer Engineering and Applications,2009,45(24):125-129.**

**Abstract:** In order to process parallel structures during translating patent articles,a pre-processing and post-processing method (Separating-combining Method) is proposed.It fixes disadvantages of the statistical translation system in processing complex structures,while maintains advantages in processing idioms and saving cost of human labor.Experiments show that this method consistently improves the accuracy of translation system.Besides,it is independent of the particularity of statistical translation system,and can be conveniently applied on different systems.

**Key words:** machine translation;syntactic structure;patent article

**摘 要:**针对专利文献翻译中的复杂并列结构的处理,提出了一种预处理和后处理的方法——拆分合并法。它弥补了统计翻译系统在复杂结构方面的劣势,同时保留统计翻译在经验句式、人力成本等方面优于规则系统的特点。实验表明,这种方法稳定地提高了翻译系统的准确率。此外,它独立于具体的统计翻译系统,可以方便地移植到不同的统计翻译系统上。

**关键词:**机器翻译;语法结构;专利文献
**DOI:**10.3778/j.issn.1002-8331.2009.24.037 **文章编号:**1002-8331(2009)24-0125-05 **文献标识码:**A **中图分类号:**TP391

## 1 引言

怎样在系统中引用丰富的语言学信息,特别是语法结构的信息,是当前机器翻译研究的一个热点。因为不同的语言之间存在着用词和结构上的巨大差异,只有完整地分析出源句的结构和语义,才能保证翻译系统的质量。但是,目前的翻译系统还不能有效地利用所有语言信息,也不可能达到和人工翻译相比拟的结果,特别是针对专利文献这样结构复杂的句子,翻译质量很差。

大致来讲,现行的机器翻译系统主要有基于规则和基于统计两种方法。在规则系统中,人工可以编写出复杂的、有着明确语言学意义的规则,较容易利用语法信息,但是在人力成本、常用习语的翻译上不如统计系统。对于统计系统而言,虽然节省了大量的人力,也省去了用于解决规则冲突的复杂设计,但是却难以将语法结构等信息引入数学模型。

提出了在统计机器翻译基础上引入一个特定语法结构(并列结构)的一种方法,弥补了统计系统在复杂结构方面的劣势,同时保留统计翻译在经验句式、人力成本等方面优于规则系统的特点。通过对并列结构的拆分和翻译结果的合并,实现了对这种结构的处理。实验表明,相对于原本的统计机器翻译系统,这种方法提高了翻译的质量(BLEU 值提高了 1%)。这种方法在原来的翻译系统之上增加了预处理和后处理模块,在大体上独立于原来的算法,因此,可以比较容易地应用于不同的翻译系统。

## 2 相关的工作

传统的统计机器翻译系统一般采用基于短语的翻译方法,来源于一个噪声信道模型,这种方法由 IBM 在 1993 年提出[1],它只是将句子处理成一个平坦的顺序结构,不能体现任何层次性的语法结构信息。

在此之后,有很多工作尝试在统计翻译模型中考虑句法结构的信息。1997 年,DeKai Wu 在统计机器翻译的基础上,引入了自然语言中的句子具有层次性这个思想,但是,除此之外并没有考虑语言学意义上的因素[2]。在此之后也有一些工作在统计翻译中引入了语言学意义上的语法信息[3-5],他们使用语法标注的平行语料来训练,使用了语法树或者依赖树。

但是早期的基于句法结构的统计翻译系统效果并不好,也有人指出引进诸多与句法结构特别是语法意义上结构相关的特征并不能显著改善翻译质量[6]。这可能是因为加上这些语法的限制之后,大大缩减了训练短语的数目,所以损害了翻译系统的质量[7]。

在此之后 Chiang David 提出了一种基于同步上下文无关文法的翻译,一定程度上提高了翻译的质量,但也没有加入语法意义的结构信息[8-9]。

该文试图在一个具体的方面利用语法意义的结构信息,从而增强翻译系统处理复杂结构的能力。

**作者简介:**赵然(1985-),男,硕士研究生,主要研究领域为机器翻译,信息检索;晋耀红(1973-),男,副研究员,主要研究领域为机器翻译,信息检索。
**收稿日期:**2008-10-17 **修回日期:**2009-02-17

## 3 机器翻译中的并列结构

在专利文献中,经常有许多复杂的句子,它们往往有许多嵌套或者不嵌套的并列结构。翻译系统如果将并列的辖域和层次关系分析错误的话,就会严重影响翻译的结果。以英文翻译为例,描述一些英文句子中含有一个或多个“and”的现象(含有“or”等其他连接词的情况类似。)

### 3.1 并列结构的例子和几个基本概念

在英文句子中,研究的并列结构是指以一个连接词“and”为核心的,几个并列的词或短语组成的结构。例如:

**例句** 1 The invention relates to [a softened foodstuff] *and* [the preparing method]。

例句 1 中含有一个并列结构。斜体的 and 是这个结构的“连接词”,and 旁的两个名词短语用方括号标出来,表示这个并列结构的各“元素”。所有的元素组成了这个 and 的“辖域”。and 和它的辖域组成了一个“并列结构”。

对于一个英文句子,在含有两个或两个以上 and 的时候,根据各自辖域范围,可以大致分为两种:辖域之间相互独立,如例句 2;也可以是有嵌套关系,即一个辖域包含另一个,如例句 3。句中的方括号和圆括号分别表示出两个并列结构各自的元素。

**例句** 2 Disclosed is a jig for stamping a primary board for a liquid crystal display circuit,which is used to perform [stamping] *and* [trimming] on a primary board for liquid crystal display circuit,comprising(a chassis),(a movable plate),*and* (a stamping device)。

**例句** 3 [A set of fixing pieces is provided with an annular catching groove at the inner side],*and* [a set of annular grooves is provided with (a set of hollows) *and* (an opening)]。

### 3.2 翻译系统对并列结构的翻译

下面举例说明翻译系统对并列结构的处理情况:

**例句** 4 [A set of fixing pieces is provided with an annular catching groove at the inner side],*and* [a set of annular grooves is provided with (a set of hollows) *and* (an opening)]。

**译句** 4 一套固定块在内侧具备一个环形的扣槽,一套环形槽具备一套圆角面和一个开放。

这个译句被认为是基本正确。原句由两个并列的字句组成,这里翻译为两个逗号隔开的句子。后面一个“and”连接两个名词词组,翻译成“和”,并且连接关系明显正确。

**例句** 5 A core line [is accommodated] *and* [freely moves (back) *and* (forth)] inside an outer tube of flexible wire.

**译句** 5 一个果心线被容纳,并且在软线的一个外管内和向前自由地往后退。

这个译句被认为翻译错误。第一个 and 的后半部分辖域判断过长。第二个 and 辖域明显错误。

### 3.3 引入句法分析结果的可能性

从专利文献中分别选取了几十句含有若干个“and”的句子。分别测试了目前的翻译系统和句法分析系统的效果,如表 1 所示。选用了 Google Translation 作为测试翻译系统。Google Translation 是 Google 公司开发的一个机器翻译的商业产品(http://translate.google.com/)。SP 是 Stanford Parser 系统,是一个基于统计的句法分析系统,可以用它做输入句子的分析工作。RMBT 是一个基于规则的翻译系统,用于参考。

**表 1 含有若干个“and”句子的分析结果**

| 系统 | 基本正确率/(%) | 错误率/(%) | 总数 |
|---|---|---|---|
| Google | 58.97 | 41.03 | 78 |
| SP | 76.92 | 23.08 | 78 |
| RMBT | 73.08 | 26.92 | 78 |
| Google(SP) | 70.00 | 30.00 | 60 |
| RMBT(SP) | 86.67 | 13.33 | 60 |

从结果中可以看出,基于统计翻译(SMT)的 Google 翻译系统在处理句子并列结构时效果很差。而现有的句法分析却能达到较好的结果(Stanford Parser 正确率为 76.92%)。作为参照,RBMT 系统达到了和句法分析相当的结果,这是因为它也用到了很多与句子结构有关的规则形式。

从翻译系统和句法分析正确率的差距来看,统计翻译系统(Google)可能有较大的性能提升的空间。而 RMBT 系统并没有太大的提升空间。

## 4 拆分合并法处理并列结构

### 4.1 拆分合并法

针对上述并列结构处理的问题,提出了一种方法——拆分合并法。这种方法基于一个认识:并列的句子结构表达了几个并列的意思,可以用几个并列的句子分别表达。

考虑到并列结构中各元素的独立性,将并列结构退化为并列结构的各个元素,然后分别产生各自的“退化句”。将这些预处理过后的退化句送入翻译系统,它们的翻译结果就完整地表达了原来的意思。而且在这种情况下,翻译系统只需要对简单句做处理,避免了直接面对并列结构的处理,会有较高的准确率。

以例句 5 为例。它含有两个嵌套的并列结构,可以由此生成三个退化句,如下所示:

(1)A core line [is accommodated] inside an outer tube of flexible wire.

(2)A core line [freely moves (back)] inside an outer tube of flexible wire.

(3)A core line [freely moves (forth)] inside an outer tube of flexible wire.

这三个句子被翻译为:

(1)一个果心线在软线的一个外管内[被容纳]。

(2)一个果心线在软线的一个外管内[自由地(往后退)]。

(3)一个果心线在软线的一个外管内[自由地(向前移动)]。

三个句子分别表达了各自独立的意思,然后合起来完整地表达了原句的意思。虽然其他的一些词汇翻译并不准确,但是很好地处理了并列结构。

一般情况下,期待翻译结果只输出一个完整的句子,因此,需要将结果组合起来,将不同的元素合并成并列结构。上述的这三个译句,根据层次性,先合并 2 和 3,然后是 1。合并后的译句为:

**译句** 6 一个果心线在软线的一个外管内被容纳并且自由地往后退和向前移动。

可以明显看出,译句 6 比译句 5 更好地体现了辖域和层次性。

该文提出的拆分合并法,是通过对源句的预处理和目标句的后处理,来实现对复杂句子中的并列结构的处理。这些处理

和翻译系统的具体翻译策略是独立的，它们只需要向翻译系统传入源语言的句子，得到目标语言的句子。因此，这种方法可以方便地应用于不同的翻译系统。

下面，详细介绍句子拆分和合并的技术。

## 4.2　句子的拆分

### 4.2.1　基本阐述

拆分模块的主要作用是将一个句子拆分成若干个句子，保持意思的完整性。对于并列结构来说，拆分基本上类似于一个乘法分配率的过程：把并列结构中的每一个元素单独提出来，代替原来的整个并列结构，就生成了一个退化句。拆分的基本模式可以表示为下面的式子：

$$S_1+(e_1+e_2+\cdots+e_{n-1}+\text{conj}+e_n)+S_2\rightarrow\begin{cases}S_1+e_1+S_2\\S_1+e_2+S_2\\\cdots\\S_1+e_n+S_2\end{cases}\qquad(1)$$

式(1)的左侧表示了一个拆分前的句子。括号中的部分是一个并列结构，它由 $n$ 个元素($e_1,e_2,\cdots,e_n$)和 1 个连接词(conj)组成，前后各有一个部分($S_1,S_2$)。式子右侧表示了拆分后的 $n$ 个退化句。

如果句子不只是一个并列结构，那么拆分后的 $n$ 个句子可能仍然含有并列结构。需要把含有并列结构的子句再次拆分。当并列结构发生嵌套时，先处理上层的并列结构，再处理下层的并列结构，这样可以避免生成重复的字句。

如果所有的并列结构都是嵌套的，那么最终得到的句子和并列结构的数目呈线性关系；如果所有的并列结构都是不嵌套的，那么最终得到的句子和并列结构数目呈指数关系。当非嵌套并列结构数目增加时，得到字句的数目也会显著增加。所幸一个句子的并列结构并不会太多，因此计算的复杂度仍然在可以接受的范围之内。

### 4.2.2　确定并列结构

句子的拆分需要首先确定句子中的并列结构，文中使用句法分析系统对句子的分析结果来实现这个目的。具体使用了 Stanford Parser 作为句法分析系统。但是其他类似的系统也可以完成这个工作。

对于例句 4，Stanford Parser 分析出如图 1 所示的结果。

```
(ROOT
  (S
    (NP (DT A) (NN core) (NN line))
    (VP
      (VP (VBZ is)
        (VP (VBN accommodated)))
      (CC and)
      (VP
        (ADVP (RB freely))
        (VBZ moves)
        (ADVP (RB back)
          (CC and)
          (RB forth))
        (PP (IN inside)
          (NP
            (NP (DT an) (JJ outer) (NN tube))
            (PP (IN of)
              (NP (JJ flexible) (NN wire)))))))
    (. .)))
```

**图 1　Stanford Parser 的输出结果**

下面要做的，就是从这个结果里面提出并列结构，并将其表示为连接词 conj 和各元素{$e_1,e_2,\cdots,e_n$}。Parser 并没有给出显式的这样的结果，只是把连接词和一些元素并列放在一个分析树的节点中。在图 1 中有两个这样的节点：

(1)(VP (VP) (CC and) (VP))

(2)(ADVP (RB) (CC and) (RB))

它们含有两个并列结构：{$e_1$=(VP)，$e_2$=(VP)，conj=(CC and)}和{$e_1$=(RB)，$e_2$=(RB)，conj=(CC and)}。一般的，并列结构的连接词就是子节点(CC and)，但是其他的子节点不一定都是元素。整个节点的形式可能有很多种，使用几个简单的规则从这个结果中确定并列结构：

(1)如果此层次上没有分割标点(逗号，分号等)，那么只有两个元素。

①获取元素核心标签。一般情况下，是“and”前一个节点的标签。

②如果该节点的标签是一个修饰成分(JJ 等)，那么寻找“and”后面的最后一个修饰成分，和前面的最前一个修饰成分。这两个之间就是并列结构。

③如果该标签是“NN”或其他非修饰成分，那么前面后面各为一个元素。

(2)如果有分割标点，则以连接词和分割标点为分割，每部分各为一个元素。

### 4.2.3　合并类型

仅仅确定并列结构是不够的，在翻译的结果的合并中，需要考虑 and 翻译成什么，元素之间怎样连接等问题。这些信息主要来源于源句，所以，在拆分阶段确定以后合并的方式。

事实上在目标语言(中文)中有多种多样的合并方式，依赖的因素也很多，而且没有唯一的解。这里选用几种合并方式，作为候选的合并类型，基本上可以包括中文里面的大多数情况。该文确定的合并类型包含下面几个因素：

(1)连接词可以是“和”，“并且”，也可以是没有；

(2)元素之间可以是逗号、顿号；

(3)连接词前面可以有标点符号，也可以没有。

使用几个规则来确定合并类型：

(1)默认连接词为“和”，元素之间为逗号，连接词前面没有标点。

(2)元素的核心标签为 S 时，连接词为无，连接词前面有逗号。

(3)元素的核心标签为 VP 等动词结构，且元素较长时，连接词为“并且”。

(4)元素的核心标签为名词短语时，且元素较短时，元素间使用顿号。

## 4.3　句子的合并

### 4.3.1　基本阐述

一般情况下，期待翻译结果只输出一个完整的句子，因此，需要将结果组合起来，将不同的元素合并成并列结构。合并和拆分是正好相反的过程，对于翻译后的 $n$ 个句子，采用类似提公因式的方法，过程如下式所示：

$$\begin{cases}S_1+e_1+S_2\\S_1+e_2+S_2\\\cdots\\S_1+e_n+S_2\end{cases}\rightarrow S_1+(e_1+e_2+\cdots+e_{n-1}+\text{conj}+e_n)+S_2\qquad(2)$$

式(2)左侧是 $n$ 个并列的句子,每个句子还有一个并列结构的元素,另外前后各有一个相同部分 $S_1$ 和 $S_2$。合并后的句子如右侧所示。

但是,翻译系统是一个复杂的系统,经常不能保证翻译结果的句子结构和原来的完全相同。比如下面的句子:

(1)它通过混合被生产[到烟草里的野菊花组成部分撕碎]做过程。

(2)它被生产通过混合[随后的普通的香烟]制作过程。

上面两句的翻译结果无法写成式(2)左侧的严格形式,因此,也就无法直接进行组合。对于这些情况,要么回到原来未拆分的翻译结果,要么就弱化式(2)左侧的严格条件,进入如下所述的恢复模块。

#### 4.3.2 恢复模块

如果翻译后的各退化句并不能完全对齐,那么就需要调用恢复模块。恢复的基本思想是,在一定条件下,选择一个句子作为基准句,其他的句子按照基准句的格式,朝基准句上合并。但如果字句间结构差别过大,那么就直接输出未切分的结果。这个差别过大的容忍程度由变量 $L$(恢复等级 Recovery Level)控制。$L$ 等级越大,容忍的程度就越大。系统按照下面的顺序尝试恢复。

(1)如果左侧有两个或两个以上的退化句符合规则,剩下的句子基本结构符合,则选择基准句强行恢复。要求 $L\geqslant 1$。

(2)如果左侧只有两个句子,但基本结构符合,选择标准句强行恢复。要求 $L\geqslant 2$。

(3)若左侧有 3 个或 3 个以上的句子,且没有任意两个符合规则,但它们的基本结构符合,选择基准句强行恢复。要求 $L\geqslant 3$。

(4)以上所有恢复程序都失败,则输出未切分翻译结果。要求 $L\geqslant 0$。

以上所有的恢复体系都要求句子结构基本符合。举例说明基本符合的标准:

(1)它通过混合被生产[到烟草里的野菊花组成部分撕碎]做过程。

(2)它被生产通过混合[随后的普通的香烟]制作过程。

这种情况下,所有并列元素的翻译结果连续,且每句只有一个并列结构。假设以第一句为基准句,只需要将第二句的并列元素并到第一句的并列元素就可以了。

(1)独立权利要求还包括[一个运动]的[男性的保护的运动]的一个附着的方法。

(2)一个独立的声明也为[男性的保护的运动]的一个附着的方法被包括到[运动衣]。

这种情况下,所有并列元素的翻译结果连续。每句有两个不相交的并列结构,且顺序不一致,但是有一个并列元素是相同的。那么就和前面的例子类似,合并不同的并列元素就行了。

综上所述,句子结构的基本符合应当满足:

(1)每个并列元素的翻译结果都是连续的,没有间断。

(2)每个句子含有若干个互不相交的并列元素,且数目相同。这些并列元素中,除了一个之外,其他的在每个句子中都是一样的。

#### 4.3.3 基准句的选择

恢复的时候,需要一个基准句来进行恢复。选择标准如下:

在恢复模块 1 中,由于已经有两个符合严格匹配条件的句子了,那么基准句就选择这两个句子中的任意一个。

对于恢复模块 2 和 3,采取两种不同的策略选择基准句。

(1)根据复杂度

对于两个并列的句子,原句唯一不同的就是关注的并列元素。如果并列元素的结构越简单,翻译系统处理成功的可能性就越大。因此,选择并列元素最简单(用长度最短近似)的句子作为基准句。

(2)根据语言模型

语言模型描述了句子的概率,在语言生成中被广泛地采用。这里可以假设任意一个句子为基准句,根据语言模型计算各合并后句子的概率,从候选中选择最优的结果。

## 5 实验和讨论

### 5.1 数据集和实验方法

测试集包含了大约 1 000 个含有若干个“and”的句子,这些句子均来源于英文的专利文献,并且都有 2 份人工翻译的参考。另外有大约 2 000 个不含“and”的句子,用来训练语言模型。

采用 Google Translate 的系统作为基准系统,与经过拆分合并法预处理和后处理之后的系统相比较。采用 BLEU 值来衡量翻译系统的质量。

同时,在基于规则的系统 RBMT 上做同样的测试,对比拆分合并法对统计系统和对规则系统的影响。

### 5.2 实验结果

首先在整个数据集上测试 BLEU 的值,如表 2 所示。第一行是原始的 Google 系统,后两行是加了拆分合并处理后的系统。SC_length 在恢复模块根据并列元素复杂度选择基准句,SC_lm 是根据语言模型选择基准句。表 2 中列出了这些系统在不同恢复等级下的 BLEU 值。

**表 2 各系统的 BLEU 对比**

| 系统 | 恢复等级 0 | 恢复等级 1 | 恢复等级 2 | 恢复等级 3 |
|---|---|---|---|---|
| Google | 0.249 7 | 0.249 7 | 0.249 7 | 0.249 7 |
| SC_length | 0.251 3 | 0.251 8 | 0.251 7 | 0.250 3 |
| SC_lm | 0.251 3 | 0.251 8 | 0.252 2 | 0.251 0 |

为了检验拆分合并法的稳定性,将数据集随机切分成 5 个集合,在不同的集合上测试 Google 系统与 SC_lm 系统(恢复等级 2)的 BLEU。实验结果如图 2 所示。

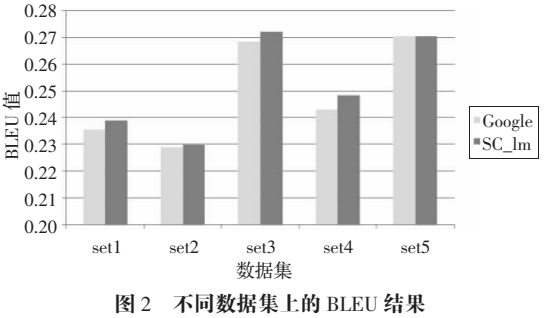


**图 2 不同数据集上的 BLEU 结果**

同样将拆分合并法用于 RBMT 之上,测试在整个测试集上的 BLEU 值。结果比原来的 RBMT 没有明显增长,反而略低(原系统为 0.265 1,加上拆分合并处理后为 0.264 4)。

同样统计了经拆分合并处理后,表 2 的结果,新计算的结果如表 3 所示。

### 5.3 分析

原本的 Google Translate 对复杂并列结构的处理效果很

表 3　拆分合并后并列结构处理情况

| 系统 | 基本正确率/(%) | 错误率/(%) | 总数 |
|---|---|---|---|
| Google | 70.51 | 29.49 | 78 |
| SP | 76.92 | 23.08 | 78 |
| RMBT | 71.79 | 28.21 | 78 |
| Google(SP) | 90.00 | 10.00 | 60 |
| RMBT(SP) | 91.67 | 8.33 | 60 |

差,正确率不到 60%。增加的拆分合并的处理可以改善对并列结构的翻译结果。从并列结构翻译情况的统计来看,这个正确率上升到了 70%,而在句法分析正确的情况下,这个正确率达到了 90%。而整体的翻译指标来看,系统的 BLEU 值提高了 1%左右。在随机选取的数据集上重复实验,拆分合并处理都一致地增加了 BLEU 值。这都充分表明了使用句法分析的结果使翻译系统的质量得到稳定提高。

在合并时,并列句子的翻译可能结构上不一致,这时需要选择不同的基准句进行恢复操作。实验表明,恢复等级为 2 的时候,系统的效果达到最优。而在选取基准句的时候,使用语言模型选取的效果要比根据并列元素复杂度选取的效果好。

统计机器翻译系统通常很难在模型里面加入丰富的语言学信息,因此在加入拆分合并处理之后能提高处理并列结构的能力,从而提高翻译质量。但是规则系统一般都进行了比较深层的语法分析,然后根据语法分析的结果人工构建出复杂的规则形式。可以看到其对并列结构的处理正确率已经和专门的语法分析工具差不多,所以拆分合并的过程并没有给这样的系统带来明显的改进。

## 6　结语

针对专利文献中复杂的并列结构提出了一种拆分合并的方法。这种方法可以看作是在统计机器翻译中引入语言学意义的语法结构信息的一种尝试。

依据句子中的并列结构,将句子拆分成若干个退化句后,送入翻译系统,并将翻译结果进行合并。整个过程相当于在翻译系统的基础上增加了预处理和后处理模块,因此可以独立于翻译系统,可以方便地移植到不同的翻译系统上。

实验结果表明,拆分合并法稳定地提高了翻译系统的准确率,并且在恢复等级为 2 的时候达到最少。作为参考,在已经进行了深层语法分析的规则翻译系统中,引入拆分合并法并没有带来明显的效果。

在该文的基础上,今后的工作可能会深入统计翻译系统的内部,研究怎么样产生出更好的翻译结果,使得翻译后的各退化句尽可能地结构一致。

## 参考文献:


[1] Brown P E.The mathematics of statistical machine translation:Parameter estimation[J].Computational Linguistics,1993,19:263-311.

[2] Wu De-kai.Stochastic inversion transduction grammars and bilingual parsing of parallel corpora[J].Computational Linguistics,1997,23:377-404.

[3] Yamada K,Knight K.A syntax-based statistical translation model[C]//Proceedings of the 39th Annual Meeting of the ACL(ACL-2001),2001:523-530.

[4] Yuan Ding,Palmer M.Machine translation using probabilistic synchronous dependency insertion grammars[C]//Proceedings of the 43rd Annual Meeting of the ACL(ACL-2005),2005:541-548.

[5] Quirk C,Menezes A,Cherry C.Dependency treelet translation:Syntactically informed phrasal SMT[C]//Proceedings of the 43rd Annual Meeting of the ACL(ACL-2005),2005:271-279.

[6] 熊得意,刘群,林守勋.基于句法的统计机器翻译综述[J].中文信息学报,2008,22(2):28-39.

[7] Koehn P,Och F J,Marcu D.Statistical phrase-based translation[C]//Proceedings of HLT-NAACL 2003,2003:48-54.

[8] Chiang D.A hierarchical phrase-based model for statistical machine translation[C]//Proceedings of the 43rd Annual Meeting of the ACL(ACL-2005),2005:263-270.

[9] Chiang D.Hierarchical phrase-based translation[J].Computational Linguistics,2007,33:201-228.


(上接 101 页)

表 3　*K*-means 算法和 *K*-Wmeans 算法在 Web 服务集上的比较

| Web 服务类别 | Web 服务样品个数 | 决策值的数目 | 条件属性的数目 | *K*-Wmeans 准确性/(%) | *K*-means 准确性/(%) |
|---|---|---|---|---|---|
| 数码类 | 300 | 3 | 4 | 93.2 | 89.7 |
| 电脑类 | 215 | 3 | 5 | 95.4 | 94.3 |
| 汽车类 | 288 | 7 | 9 | 88.9 | 82.8 |
| 图书类 | 40 | 2 | 4 | 88.2 | 0 |
| 服装类 | 94 | 4 | 35 | 97.4 | 0 |
| 食品类 | 202 | 7 | 16 | 95.3 | 0 |

## 6　结束语

研究了 P2P 网络环境下的 Web 服务发现机制，并运用基于加权改进的 *K*-Wmeans 进行分布式聚类分析,从而保证服务请求者能够按类别查找 Web 服务，提高了查找的准确度和效率,实验证明了此种方法的有效性。然而随着 Web 中服务数量的剧增,此种方法能否继续适用,有待进一步研究。

## 参考文献:


[1] Clement L,Hately A,Riegen C V,et al.Universal description discovery & integration(UDDI) 3.0.2.2004[EB/OL].http://uddi.org/pubs/uddi_v3.htm.

[2] 孙士宝,秦克云.改进的 K-平均算法研究[J].计算机工程,2007,33(13):200-201.

[3] 李勇.分布式 Web 服务发现机制研究[D].北京邮电大学,2007.

[4] 吴集,王晓川,金士尧.集群 Web 服务器预取机制中用户会话聚类的研究与实现[J].计算机工程与科学,2005,27(12):4-6.

[5] 张晓燕.基于 Web Services 的分布式服务发现系统的研究[D].燕山大学,2006.

[6] 邓晶晶,蒋玉明,傅静涛.基于 Web 使用挖掘的实时聚类算法[J].四川大学学报,2007,44(4):803-806.

[7] 杨国威,王志坚,许峰.基于 P2P 的 Web 服务部署与发现框架[J].微计算机信息,2008,24(15):154-156.

[8] 朱树人,贺株莉.一种分布式 Web 服务发现方法[J].计算机工程与应用,2008,44(15):121-123.

[9] Verma K.A scalable P2P infrastructure of registries for semantic publication and discovery of Web services[J].Information Technology and Management,2005,6(1):17-39.

[10] Bandyopadhyay S.Clustering distributed data streams in peer-to-peer environments[J].Information Sciences,2006(176):1952-1985.