一种能够检测所有交叉歧义的汉语分词算法

王显芳, 杜利民
(中国科学院声学研究所语音交互信息技术研究中心, 北京 100080)

摘 要: 本文给出了一种能够检测句子中所有交叉歧义的汉语分词算法. 该算法基于"长词优先"的切分原则. 它解决了切分路径数随句子长度的增长而呈几何级数增长的问题, 并且提供了一种方法可将句子的覆盖歧义和交叉歧义分开处理. 算法的运算复杂度为 $O(N)$, N 为句子长度. 它的输出使得进行下一步处理的运算量大大减少.

关键词: 自动分词; 长词优先
中图分类号: TP301.6 文献标识码: A 文章编号: 0372-2112 (2004) 01-0050-05

A Method of Sentence Segmentation That Check All Overlapping Ambiguity

WANG Xiar-fang, DU Li-min
(*Center for Speech Interactive Information Technology Institute of Acoustics, Chinese Academy of Science, Beijing* 100080, *China*)

Abstract: We proposed a new method of Chinese automatic segmentation that can check all overlapping ambiguity in sentence. This algorithm is based on the principle of "Choose Longer Word". It solves the problem that the count of segmentation way is exponentially increasing with the sentence length, and provides a method to handle overlaying ambiguity and overlapping ambiguity separately. The time complexity of this algorithm is $O(N)$, where N is the length of sentence. Its output can greatly decrease the computing cost of post processing.

Key words: automatic segmentation; choose longer word

1 问题的提出

在书面汉语中, 字与字、词与词是连写的, 词在句中没有显式的标记. 因此, 理解汉语的首要任务就是把连续的汉字串分割成词的序列, 即自动分词. 近二十年来, 汉语自动分词研究取得了很大成就, 推出了一批有代表性的分词系统, 如清华大学的 SEGTAG 系统; 北京航空航天大学的 CDWS 系统; 北大计算语言学研究所分词系统等[1~3]. 同时提出了许多分词算法, 其中有一定代表性的主要有: 最大匹配法(又可分为正向、逆向、双向三种)、最优路径(+ 词频选择)法(最少分词法)、特征词库法、邻接约束法、人工神经网络方法、无词典分词法等等. 这些算法各具特色. 如果只考虑利用词条信息, 不使用词性、语义等复杂特征的话, 则有最大匹配法和全切分算法[1,2].

最大匹配法虽然有算法简单、速度快的优点, 但它仅从一种或两种切分路径中选择, 显然不能保证切分结果是全局最优的. 全切分算法可以遍历所有的切分路径, 但是切分路径的总数是随着句子长度的增长而成指数级增长的, 对于长度比较长的句子, 全切分算法可能需要非常长的时间才能遍历所有切分路径以选择出全局最优的切分结果.

举几个例子:

*S*1. 中国人民

*S*2. 中国人民万岁

*S*3. 公路局正在治理解放大道路面积水问题

*S*4. 上海专利事务所业务办公室主任丁惠敏为上海一位 50 多岁的专利技术人员代理申请了一项交通自动控制方面的发明专利

*S*5. 江泽民在北京人民大会堂会见参加全国法院工作会议和全国法院系统打击经济犯罪先进集体表彰

大会代表时要求大家要充分认识打击经济犯罪工作的艰巨性和长期性

这五个句子的切分路径个数分别为

*S*1	*S*2	*S*3	*S*4	*S*5
6	12	1424	3538944	3764387840

说明一下, 作者所使用的词典为北航的词典, 词条数为 67480 个, 最大词长为 7, 其中包含了所有的汉字单字.

文献[3]指出, 句子的切分路径个数是与其句子长度成几何级数的. 对于比较长的句子, 全切分算法必须需要很长时间才可能遍历所有切分路径. 无论是进行统计语言模型训练还是在句子分析、翻译等实用系统中句子的预处理用这么长时间都是难以忍受的.

其实这种切分路径个数随句子长度而呈几何级数增长的

收稿日期: 2002-01-09; 修回日期: 2002-05-18
基金项目: 国家 973 重点基础研究发展项目"图像、语音、自然语气理解和知识挖掘——汉语自然口语对话的理论和实验平台研究"基金(No. G1998030505)

现象主要是由于覆盖歧义现象引起的. 假设一个词又可以分成两个词, 如果不考虑这个词引起的覆盖歧义现象, 则句子的切分路径个数比原来少一半.

为方便描述, 我们称词典包含的句子的子串为候选词条. 句子 $S2$ 共有 11 个候选词条:

中 中国 中国人 国 国人 人 人民 民 万 万岁 岁

其中, 会引起交叉歧义的词条为"中国"、"中国人"、"国人"、"人民", 这些词条都和别的词条存在交叉现象, 例如,"中国"和"国人","中国人"和"人民","国人"、"人民". 其他的词条"万岁"、"中"、"国"、"人"、"民"、"万"、"岁"不和其他的词存在交叉现象.

句子 $S1$ 比句子 $S2$ 缺少子串"万岁", 而这个子串不会在句子 S2 当中引起交叉歧义, 只引起覆盖歧义. 如果不考虑这个子串引起的覆盖歧义现象, 句子 $S1$ 和句子 $S2$ 的切分路径个数将是相同的. 但如果考虑了, 则句子 $S1$ 的切分路径个数为句子 $S2$ 的切分路径个数的一半.

汉语的覆盖歧义目前还没有理想的解决方法."长词优先"的准则是一种切实可行的解决覆盖歧义问题的切分准则[4, 5]. 所谓"长词优先", 就是尽可能地用最长的词匹配句子中的汉字串. 比方说"中国人"和"中国"都是词, 但当我们在句子中遇到"中国人"这个汉字串时, 就用"中国人"去匹配它, 使得切出来的词尽可能长, 切出来的词条数尽可能少.

最大匹配法从句子的起始位置开始, 依次在已经得到的最后一个词的结束位置使用"长词优先"的准则. 但某一位置开始的最长的词条有可能和从该词条内某一位置开始的一个词条交叉. 比如说第一句"中国人民", 其前向最大匹配切分路径为"中国人_民", 但此时"中国人"和"人民"相交叉. 因此它忽略了另一种切分路径"中国 人民", 所以最大匹配法无法检测到所有的交叉歧义的. 最优路径(+ 词频选择) 法(最少分词法) 使用了动态规划方法[1]. 但动态规划方法有一个假设, 就是句子在某一位置前的切分路径的选择和此位置后的句子的内容无关. 这个假设在使用词条数最少准则和"费用"最小准则是适用的, 但从句法、语法以及语用层面上考虑则是不正确的. 动态规划方法虽然可以输出所有词条数为最少词条数的切分路径, 但是它仍然无法检测并得到所有的交叉歧义. 这样在后续的处理中先行失去了一些信息. 交叉歧义占切分歧义现象的 86%[1], 处理歧义的重点就是要处理交叉歧义问题. 而最大匹配法和动态规划方法在进行相应的判断和选择之前就忽略了一些交叉歧义.

2　最大无覆盖歧义切分路径集

"长词优先"的准则可描述为: 如果一个候选词条 w_n 覆盖了其他一些候选词条$w_{n1}, w_{n2}, \cdots, w_{nm}$, 即 w_n和w_{n1}的起始位置相同, 和 w_{nm}的结束位置相同, 且 $w_{n1}, w_{n2}, \cdots, w_{nm}$ 依次首尾相连, 则在考虑切分路径的时候, 只考虑 w_n出现在切分结果时的情况, 而不将 w_n拆开.

"长词优先"的准则和检测交叉歧义并不矛盾. 在使用"长词优先"的准则的时候仍然能够检测所有的交叉歧义. 为了给出算法, 我们首先引入一个定义.

给定词典 L, 一个句子的所有切分路径构成了一个集合 P, 它必然存在一个不包含覆盖歧义的切分路径的子集合 $Q \subseteq P$, 而对任给句子的一种切分路径 $x \in P$, 都能够找到一种切分路径 $y \in Q$, 使得 y 与 x 之间只存在覆盖歧义而不存在交叉歧义. 称满足这种条件的子集合 Q 为最大无覆盖歧义切分路径集.

最大无覆盖歧义切分路径集的意思就是如果向该集合当中加入一种不属于该集合的切分路径, 则此切分路径必然和集合中一种切分路径存在覆盖歧义; 而如果从此集合中删除一种切分路径, 必然会导致句子的一些切分路径无法在该集合中找到与之只存在覆盖歧义而不存在交叉歧义的切分路径.

最大无覆盖歧义切分路径集其实将一个句子的切分信息分成了两部分: 可由词典直接得到的信息和仅能从正在处理的句子中得到的信息. 这种分离对于任何语言处理工作都将是有帮助的.

下表列出上述五个例句的全切分集和最大无覆盖歧义切分路径集中切分路径的个数.

表 1　五个句子的全切分路径个数和最大无覆盖歧义切分路径个数比较

	$S1$	$S2$	$S3$	$S4$	$S5$
全切分集	6	12	1424	3538944	3764387840
最大无覆盖歧义切分路径集	2	2	12	2	16

我们知道, 句子的全切分集和句子的全切分词图相对应. 那么, 我们怎样才能得到与最大无覆盖歧义切分路径集相对应的词图呢?

3　覆盖歧义检测法

回顾构造句子的全切分词图的过程. 我们称以某一位置为结束位置的候选词条为该位置的前驱词条, 称以某一位置为开始位置的候选词条为该位置的后续词条.

构造句子的全切分词图的过程可以分为两步. 首先找到句子中每一个位置的所有候选词条. 然后在该位置的所有前驱词条和后续词条之间建立弧.

全切分算法之所以会遍历如此多的切分路径, 主要原因就是它不仅考虑了整个词条在切分路径中出现时的情况, 它还考虑了将词条"切碎"时的情况. 正是这些将词条"切碎"的形式导致了覆盖歧义, 也导致了切分路径数的指数级增长. 比如说在句子 $S2$ 中, 我们不仅考虑了词条"万岁"出现时的情况, 还考虑了"万　岁"出现时的情况. 也就是说, 在考虑位置 4, 也就是"民"的后续词条时, 不仅考虑了词条"万岁", 还考虑了词条"万". 在位置 4 的后续词条为"万"时, 引入了含有"万　岁"的切分形式.

因此, 如果我们想得到句子 $S2$ 的最大无覆盖歧义切分路径集, 我们就不需要考虑位置 4 的后续词条为"万"时的情况.

引入几个定义. 设句子的某个候选词条 w_{ml}的开始位置为 m, 长度为 l(也就是说该词条的结束位置也就是该词条后面的下一个候选词条的开始位置为 $m+l$). 另一个候选词条

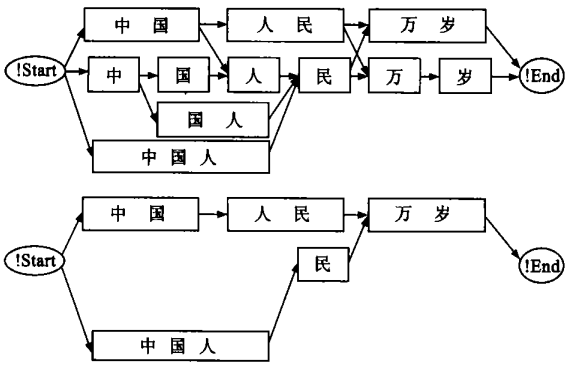

图 1 句子 $S2$ 的全切分词图和最大无覆盖歧义词图

w_{mn}的开始位置也为 m, 长度为 n, 但 $n< l$. 切分路径 1 含有词条 w_{mi}, 切分路径 2 含有词条 w_{mn}, 但切分路径 2 中不存在开始位置为 $m+ l$ 的词条, 则称切分路径 1 和切分路径 2 存在着因词条 w_{ml}引起的交叉歧义.

称候选词条 w_{ml}为不可导致交叉歧义候选词条, 如果它满足如下条件: 对于所有含有候选词条 w_{mi}的切分路径 s, 任给句子的一种切分路径 t, $t \neq s$, s 和 t 之间不存在着因词条 w_{ml}引起的交叉歧义.

称候选词条 w_{ml}为可导致交叉歧义候选词条, 如果它不是不可导致交叉歧义候选词条.

在句子 $S2$ 中, 词条“万岁”就是不可导致交叉歧义候选词条. 它不会和其他词条产生交叉歧义. 显然, 最大无覆盖歧义切分路径集当中的任何一种切分路径都不会包含不可导致交叉歧义候选词条的子词条, 比如说句子 $S2$ 的最大无覆盖歧义切分路径集的任何一种切分路径都不会包含有词条“万”. 所以我们只需要考虑位置 4 的后续词条为“万岁”时的情况.

称位置 i 为可后续位置, 如果在最大无覆盖歧义词图中存在跨越位置 i 的弧.

如果位置 i 的后续最大长度候选词条是可导致交叉歧义候选词条, 则称其后续最大长度候选词条和句子其他子串存在交叉歧义时的无覆盖歧义内部切分路径为位置 i 的无覆盖歧义局部切分路径, 称该无覆盖歧义局部切分路径的结束位置 j 为位置 i 的可导致交叉歧义位置.

称位置 i 的后续最大长度候选词条和其无覆盖歧义局部切分路径的首词条为该位置的无覆盖歧义后续词条.

设位置 m 的后续最大长度候选词条的结束位置为位置 n, 则称该词条为位置 n 的无覆盖歧义前驱词条. 如果位置 m 的某一条无覆盖歧义局部切分路径的结束位置为位置 n, 则称该无覆盖歧义局部切分路径的首词条为位置 n 的无覆盖歧义前驱词条.

因此, 构造最大无覆盖歧义词图的方法为: 依次处理句子的各个位置 i, $0 \leq i \leq N$. 首先判断句子的位置 i 的后续最大长度候选词条是否为可导致交叉歧义候选词条. 如果该候选词条是可导致交叉歧义候选词条, 我们就找出它的无覆盖歧义局部切分路径. 然后我们在位置 i 的无覆盖歧义前驱词条 w_i 所引导的无覆盖歧义局部切分路径的尾词条w_m和该位置的无覆盖歧义后续词条 w_n 之间建立弧. 如果与词条 w_j 起始位置相同的候选词条w_s 与无覆盖歧义后续词条w_n 的结束位置相同, 则说明此种切分形式和其他切分形式存在覆盖歧义, 不需要在该位置之前的词和该位置之后的词建立弧, 否则就建立弧.

4 算法流程

将位置 0 设为可后续位置; 其他位置设为不可后续位置;
将虚词条! Start设为位置 0 的无覆盖歧义前驱词条;
将虚词条! End位置 n 的无覆盖歧义后续词条;

```
for(位置 i= 位置 0; i< n; i+ + ){//判断位置 i
  if(位置 i 不是 可后续位置)  continue; //判断下一个位置
  //判断位置 i 的后续最大长度候选词条是否为可导致交叉歧义候选词条
  设 w_i 为位置 i 的后续最大长度候选词条, nEndPos 为 w_i 的结束位置.
  将位置 nEndPos 设为位置 i 的后续位置. 词条 w_i 为位置 i 的无覆盖歧义后续词条
  CheckWordItem(i, i, nEndPos);
  //在位置 i 的无覆盖歧义前驱词条和无覆盖歧义后续词条之间建立弧
  for(位置 i 的无覆盖歧义后续词条 w_i){
    for(位置的无覆盖歧义前驱词条 w_k){
      bAddArc= TRUE;
      for(与词条 w_k 相同起始位置的词条 w_m)
        if(w_m 的结束位置= = w_i 的结束位置){
          bAddArc = FALSE; break;}
      if(bArcExist){
        for(w_k 的起始位置的每条无覆盖歧义切分路径 p){
          j= p 的尾词条的结束位置; 在 p 的尾词条和 w_l 之间建立弧;
          将位置 j 置为可后续位置; 将 w_k 加入位置 j 的无覆盖歧义前驱词条}}}}
CheckWordItem(位置 i, 上次判断的结束位置 j, 结束位置 nEndPos){
  for(每一个起始位置为 j 的候选词条 w_k){
    设 w_k 的结束位置为 m;
    if (m > nEndPos){//位置 i 的后续最大长度后续词条为可导致交叉歧义候选词条,
      将临时切分路径设为位置 i 的无覆盖歧义局部切分 f 方式;
      将临时切分路径的首词条设为位置 i 的无覆盖歧义后续词条}
    else if(m< nEndPos){
      if(位置 m 的上次判断位置! = 位置 i){
        将 w_k 加入位置 i 的临时切分路径
        CheckWordItem(i, m, nEndPos);
        从位置 i 的临时切分路径中将 w_k 删除}}
    位置 m 的上次判断位置= 位置 i;}}
```

以句子 $S2$ 为例, 说明一下算法流程.

首先检查位置 0, 其后续最大长度候选词条为“中国人”, 位置 0 的后续词条还有两个,“中国”和“中”.“中国”的后续最大长度候选词条为“人民”, 其结束位置在词条“中国人”之后. 因此, 位置 2 为位置 0 的可导致交叉歧义位置,“中国人”为可导致交叉歧义候选词条, 切分路径“中国”为位置 0 的无覆盖

歧义局部切分路径. 再检查词条“中”, 其结束位置为位置 1, 位置 1 的后续词条为“国人”和“国”.“国人”的结束位置为位置 3, 正好是位置 0 的后续最大长度候选词条的结束位置. 词条“国”的结束位置为位置 2. 位置 2 已经判断过, 不需要再对位置 2 的后续词条进行判断. 位置 2 为位置 0 的可导致交叉歧义位置. 此时得到的局部切分路径为“中 国”, 需要判断此局部切分路径是否是已经得到的位置 0 的无覆盖歧义局部切分路径的覆盖歧义. 因为“中 国”为“中国”的一种覆盖歧义切分路径. 因此, 不将此切分路径加入位置 0 的可导致交叉歧义内部切分路径. 位置 0 的无覆盖歧义后续词条为“中国人”和“中国”, 无覆盖歧义前驱词条为虚词条! Start. 在虚词条! Start 和“中国人”之间和虚词条! Start 和“中国”之间建立弧. 至此, 位置 0 判断完毕.

位置 1 不是可后续位置, 不需要对它做任何判断. 位置 2 为可后续位置. 位置 2 的后续最大长度候选词条为“人民”, 位置 2 的后续词条还有“人”.“人”的后续词条是“民”, 其结束位置为 4, 正好是词条“人民”的结束位置. 因此,“人民”是不可导致交叉歧义侯选词条. 因此位置 2 的无覆盖歧义后续词条为“人民”. 然后我们考虑跨越位置 2 的弧. 位置 2 的无覆盖歧义前驱词条为“中国”, 其起始位置为位置 0, 以位置 0 为起始位置的词条的最大长度为 3. 因此在“中国”和“人民”之间建立弧. 位置 2 判断完毕.

位置 3 为可后续位置. 其后续词条只有一个,“民”.“民”是不可导致交叉歧义侯选词条. 因此位置 3 的无覆盖歧义切分后续词条为“民”, 其结束位置为 4. 然后我们考虑跨越位置 3 的弧. 位置 3 的无覆盖歧义前驱词条为“中国人”, 其起始位置为位置 0, 以位置 0 为起始位置的词条的结束位置为 3, 小于 4. 因此在“中国人”和“民”之间建立弧. 位置 3 判断完毕.

位置 4 为可后续位置. 位置 4 的后续最大长度候选词条为“万岁”, 位置 4 的后续词条还有“万”.“万”的后续词条是“岁”, 其结束位置为 6, 正好是词条“万岁”的结束位置. 因此,“万岁”是不可导致交叉歧义词条, 位置 4 的无覆盖歧义后续词条为“万岁”. 然后我们考虑跨越位置 4 的弧. 位置 4 的无覆盖歧义前驱词条为“人民”和“民”.“人民”的起始位置为位置 2, 以位置 2 为起始位置的词条的最大长度为 4, 小于位置 6. 因此可以在“人民”和“万岁”之间建立弧.“民”的起始位置为位置 3, 以位置 3 为起始位置的最大长度词条的结束位置为 4, 小于位置 6. 因此可以在“民”和“万岁”之间建立弧. 位置 4 判断完毕.

位置 5 不是可后续位置. 不需要对它做任何判断.

位置 6 为句子的结束位置. 位置 6 的无覆盖歧义前驱词条为“万岁”, 在“万岁”和虚词条! End之间建立弧. 位置 6 判断完毕. 判断过程结束.

5　运算复杂度分析

设句子的长度为 N, 词典的最大词长为 T.

步骤 1. 对于位置 i, $0 \le i \le N$, 找到位置 i 开始的候选词条, 每个位置最多需要找的候选词条数为词典的最大词长 T. 因此本步的运算复杂度为 $O(N)$.

步骤 2. 对于位置 i, $0 \le i \le N$. 首先判断该位置的后续最大长度候选词条, 是否为无交叉歧义词条. 判断的次数为该位置的后续词条个数, 由于每次判断仅在后续最大长度候选词条的长度范围内进行判断, 和句子的长度 N 没有关系. 然后我们考虑该位置的无覆盖歧义后续词条和无覆盖歧义前驱词条之间建立弧. 很显然, 该位置的无覆盖歧义后续词条必然是该位置的后续词条, 所以个数将不超过 T. 该位置的无覆盖歧义前驱词条的起始位置将在该位置的前 T 个位置的范围之内. 以该位置前面第 i 个位置的无覆盖歧义前驱词条的长度显然不可能超过 i, 因此其个数最多为 i 个, 所以一个位置的无覆盖歧义前驱词条的个数最多为 $T*(T-1)/2$. 对于每个无覆盖歧义前驱词条, 最多需要判断 T 次, 因此判断的次数不超过 $T*T*(T-1)/2$. 所以本步骤的运算复杂度为 $O(N)$.

因此, 算法的运算复杂度为 $O(N)$.

6　算法性能

在整个句子是一个交叉歧义链时, 覆盖歧义检测法仍然是有效的. 句子“结合成分子时”的交叉歧义链长为 6. 其全切分路径个数为 13 个, 而其最大无覆盖歧义切分路径数为 4 个.

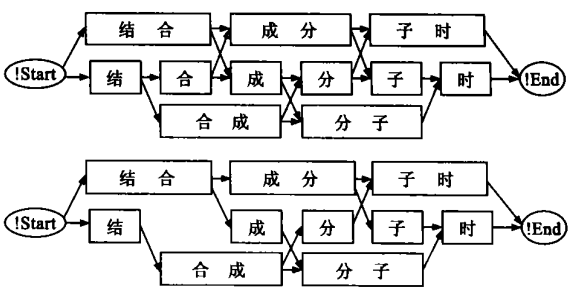

图 2　句子“结合成分子时”的全切分词图和最大无覆盖歧义词图

最大无覆盖歧义切分路径集当中的切分路径数是很少的. 在 1990-2000《人民日报》共十一年的全文和 1994 年《市场报》全文以及 1994 年《中国百家报刊精选》的 520M 语料中, 总共有 17494391 句, 所有的切分路径数为 22459830. 平均 1.284 种/句.

下图给出最大无覆盖歧义切分路径个数为 1 到 8 和大于 8 的句子个数.

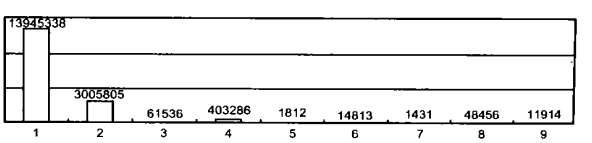

图 3　最大无覆盖歧义切分路径个数为 1 到 8 和大于 8 的句子个数

切分路径个数不为 1 的句子是包含交叉歧义字段的, 这样的句子总数为 3549053, 约占句子总数的 20%. 这些句子的最大无覆盖歧义切分路径总数为 8514492, 平均 2.34 种/句. 在这些句子当中, 最大无覆盖歧义切分路径个数超过 128 的

句子个数为31句, 最大无覆盖歧义切分路径个数最大为 384.

覆盖歧义检测法的运算效率是很高的, 下图给出了在 CPU 为 PⅢ 800 的计算机处理上述数据, 最大匹配法、覆盖歧义检测法和全切分算法所需要的时间.

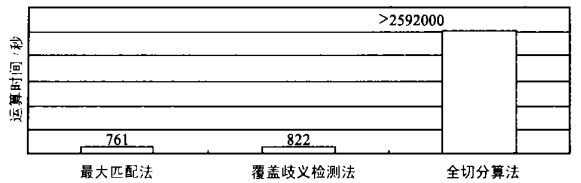

图 4 最大匹配法、覆盖歧义检测法和全切分算法所需要的时间

7 讨论

覆盖歧义检测法是基于“长词优先”原则的, 就是尽可能用更长的词来匹配句子中的字串, 使得切出来的词尽可能长, 切出来的词的数量尽可能少. 这个切分原则也可以说成是忽略覆盖歧义, 其实这也是最大匹配法的匹配原则[1]. 其本质区别是, 最大匹配法只考虑特定的扫描方向, 在上一次得到的词的结束位置使用“长词优先”原则, 它在忽略覆盖歧义的同时也忽略了交叉歧义, 从而导致它对切分路径判断的不完全性, 而覆盖歧义检测法在句子的每一个位置都进行检测, 在忽略覆盖歧义的同时保留了所有的交叉歧义, 从而提供了一种能够对覆盖歧义和交叉歧义分开处理的方法.

覆盖歧义检测法并没有直接输出句子的唯一切分结果, 还需要利用其他知识进行切分排歧. 它的输出使得进行下一步处理的运算量大大减少, 并且没有忽略任何切分信息. 据统计, 交叉歧义大约有 94% 是伪歧义. 只有 6% 的交叉歧义是真正存在歧义的[6]. 我们可以首先直接排除伪歧义, 然后利用上下文信息来排除真正的交叉歧义. 对于覆盖歧义, 也可以采用相同的方法. 对于未录入词问题, 在切分句子的时候, 未录入词通常被切分成单字[7]. 因此未录入词一般不和句子的其他部分存在交叉歧义. 所以此算法将交叉歧义的处理和未录入词的处理分开.

采用最大无覆盖歧义切分路径集作为下一步处理的输入不会使系统资源的消耗增加很多. 由上面给出的数据可知句子的最大无覆盖歧义切分路径集中切分路径的平均个数仅为 1.284. 采用此方法时系统资源的消耗是很小的. 最大无覆盖歧义切分路径集中切分路径的个数的最大值也不大. 因此采用此方法, 系统资源的消耗不会在处理一部分句子时很小, 而在处理另外一部分句子时很大.

参考文献:

[1] 陈小荷. 现代汉语自动分析 [M]. 北京: 北京语言文化大学出版社, 1999.
[2] 马晏. 基于评价的汉语自动分词系统的研究及实现 [M]. 语言处理专论, 北京: 清华大学出版社, 1996.
[3] 王雪松. 汉语语言的多层面优化统计语言模型研究 [D]. 北京: 中科院声学所硕士论文, 1997.
[4] 刘开瑛. 中文文本自动分词和标注 [M]. 北京: 商务印书馆, 2000.
[5] 侯敏, 孙建军, 陈肇雄. 汉语自动分词的歧义问题 [M]. 计算语言学进展与应用, 北京: 清华大学出版社, 1995.
[6] 孙茂松等. 高频最大交集型歧义切分字段在汉语自动分词中的运用 [J]. 中文信息学报, 1999, 13(1): 27- 34.
[7] 孙茂松, 等. 中文姓名的自动辩识 [J]. 中文信息学报, 1995, 9(2): 16- 27.

作者简介:

王显芳 男, 1975 年 7 月生于安徽萧县, 1997 年毕业于北京大学数学科学学院信息科学系, 获理学学士学位, 现为中国科学院声学研究所博士生, 主要研究领域为语音识别, 对话系统.

杜利民 男, 1957 年 1 月生于四川南充, 1983 年、1987 年、1991 年分别于北京大学、中国科大研究生院、中国科学院声学研究所获理学学士、工学硕士、理学博士学位; 1996 年美国麻省理工学院(MIT) 高级访问科学家, 现为中国科学院声学研究所研究员、博士研究生导师、IEEE 高级会员、中国电子学会理事、电子学报编委.