



[12] 发 明 专 利 说 明 书

[21] ZL 专利号 99123110.4

[45] 授权公告日 2004 年 9 月 22 日

[11] 授权公告号 CN 1168029C

[22] 申请日 1994.2.18 [21] 申请号 99123110.4
分案原申请号 94101382.0

[30] 优先权

[32] 1993. 3. 3 [33] US [31] 08/025,464

[71] 专利权人 国际商业机器公司

地址 美国纽约

[72] 发明人 安东尼奥·扎莫拉

审查员 王京霞

[74] 专利代理机构 中国国际贸易促进委员会专利
商标事务所
代理人 范本国

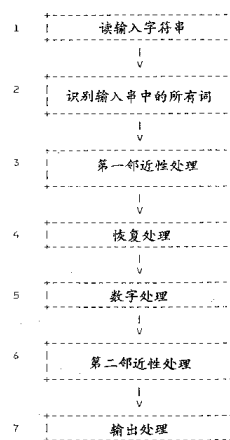
权利要求书 1 页 说明书 8 页 附图 8 页

[54] 发明名称 从连续的中文文本中分离出中文词
的方法

[57] 摘要

本发明一个处理过程, 该过程用于机器分析连续的中文文本并分离出组成文本的词。该处理过程使用一个词典、一些处理标点符号的直接规则、识别一串中文文本中全部词和通过依次更严格的过滤机制消除不合逻辑段从而将输入文本中的重叠词分解成一组相邻词的方法, 以及解除多义性的方法。

处理流程图



1. 一种在数据处理系统中从连续文本中分离出词的方法，其特征在于包括以下步骤：
 - 将输入文本字符串读入一个数据处理系统中；
 - 在上述系统中识别出输入字符串中的所有词；
 - 在上述系统中执行第一邻近性处理，以删除不与另一词相邻的词；
 - 在上述系统中执行恢复处理，从而恢复出包含着被上述的第一邻近性处理所删除的未被引用字符的一个最长的词，形成一系列分离出来的词；以及
 - 输出带有指定的分隔符的经分离的词系列。

从连续的中文文本中分离出 中文词的方法

本发明广义而言与数据处理系统有关,更具体地说,是涉及从连续的中文文本(text)中分离出中文词的语言学方法。

中文是写成“词符(logographs)”的,每个词符代表一个音节,而且通常是一个概念或有意义的单元。中文的传统写法是在这些词符之间没有间隔。一个中文“词(word)”可能由一个或多个这种词符组成,因此一个中文读者必须辨认出这些词的分界以便理解文本的意思。

电子形式的中文文件也是书写成不带间隔的,这使计算机应用(例如信息存贮和检索 IS/R)中难于识别机器可读索引中使用的项。当然,对于 IS/R 遇到的问题可以用蛮力(brute force)办法来解决,即把文本的每个字(character)编成索引从而能查寻这些字的每种组合,但这是效率很低的,因为它使用太大的索引空间并检索出大量无关的结果(即不准确)。

尽管 IS/R 应用能够解决而不必识别中文文本中的词,但有其

他应用(如计算机辅助翻译)需要对词准确识别以便能给出有意义的翻译结果。本发明的目的就是要定义一个识别出中文文本字符串(string)中全部词的处理过程,通过依次更严格的过滤机制排除掉不合逻辑的段,从而将重叠的词分解成一组相邻的词,并借助频率判据和语法约束来解除多义性。

所以,本发明的一个目的是提供一种从连续的中文文本中分离出中文词的方法,包括以下步骤:

将输入文本字符串读入一个数据处理系统中;

在上述系统中识别出输入字符串中的所有词;

在上述系统中执行第一邻近性处理,以删除不与另一词相邻的词;

在上述系统中执行恢复处理,从而恢复出包含着被上述的第一邻近性处理所删除的未被引用字符的一个最长的词,形成一系列分离出来的词; 以及

输出带有指定的分隔符的经分离的词系列。

本发明实现这些和其他目的、特点和优点。所描述的处理过程用于机器分析连续的中文文本并分离出组成文本的词。该处理过程使用一个词典、一些处理标点符号(*punctuation*)的直接规则、识别一串中文文本中全部词和通过依次更严格的过滤机制消除不合逻辑段从而将输入文本中的重叠词分解成一组相邻词的方法、以及解除多义性的方法。

对于这些及其他目的、特别和优点,将结合所附图予以更充分的评述。

图1描绘出经过一次字典查寻处理过程之后的数据结构。

图2描绘出经过第一次相邻性(*adjacency*)处理过程之后的数据结构。

图 3 描绘出第二次相邻性处理过程之后的数据结构。

图 4 描绘出第二次相邻性处理过程之后的未删除词。

图 5 描绘出实现本发明方法的操作步骤序列的流程图。

图 6 给出要被处理的字串的第一个实例。

图 7 给出要被处理的字串的第二个实例。

图 8 给出要被处理的字串的第三个实例。

从文本中分离出词的处理过程所要求的数据结构要能识别出由相邻字组成的文本的子串(substring)。这些子串可以代表彼此重叠或彼此相邻的中文词。再有,该数据结构应能包容伴随每个词的数据,如词类或频率。

作为本发明实施例的举例说明,其数据结构由一个至少有三个字段(field)的数组(array)来表示,这三个字段是:位置、长度和标记(flag)。“位置”指出一个字段的第一个字的位置,“长度”确定这个字串有多长。“标记”用于标明词条(work entry)是“被删除的”的,并提供一种恢复被删除词的机制。对于该数据结构可以增加附加字段以容纳频率信息或词类以解除各种多义性。

图 1 给出字串“softwaredevelopment(软件开发)”在使用字典查寻识别出所有词之后的数据结构的内容。尽管这个例子是英文的,类似的处理过程适用于中文文本字串。对该数据结构中各词的检验揭示出不能由人立即发现却被计算机成功发现的那些词,此时对文本的所有可能的子串都对照词典进行了检验。标记值为零表明该词未被删除。请注意字的位置从零开始而不从 1 开始计算。

词典查寻处理过程包括识别文本的全部子串和与词典匹配。然而,为了使处理的效率更高和防止词典复盖失效,使用了下列判据:

1)不产生含有标点符号的子串,2)当文本的一个字符不被词典中发现的任何词所包含时,对这单个字符构成一个数据结构条目。

邻近性限制的应用

第一邻近性处理将一个词的标记置成非零值以删除不与另一词相邻或不与字串开头或末尾相邻的词。图2显示出标记1来标志其末端不与另一词开头相邻的词,用标记2标志其开头不与另一词相邻的词。这样,词“so”被标志为删除,因为没有以“ft...”开头的词跟在它后面,而词“oft”被删除是因为它前面的“s”不是一个有效的词。请注意,第一邻近性处理从清单中列出的19个词中删出了8个。

重叠字串和邻近性限制存在的问题

尽管第一邻近性处理显著减少了词的数量,但它有一个缺陷得由第一恢复处理来校正。例如,考虑字串“thexresult”。在识别出词“the”,“hex”,“re”,以“result”之后,第一邻近性处理删除了词“the”,因为它的末端不与另一个词的开头相邻,“hex”被删除是因为它的开头不与另一个词的末尾相邻。词“re”也被删除了,因为它的末端不与另一个词的开头相邻。只有词“result”保留下来,因为它在词“hex”的末端与字串的末尾之间。这就造成了文本串被数据结构复盖的缺口。进入字符“thex”的所有词条目都被第一邻近性处理给删除了,因为字串“the”和“hex”重叠。

从重叠字串中造成相邻字串的处理过程

第一恢复处理通过拷贝文本串和擦掉被引用的全部字符来识别出数据结构复盖文本串时出现的缺口。任何剩余字符都是由于各文本串重叠造成的。第一恢复处理选出第一个未被引用字符,并将标记置回到零来恢复那个包含这一字符作为词的第一字符的最长的词。如果该字符没有作为任何被删除字的第一字符出现,则对该单个字符造成一个新的数据条目。在被恢复词或新条目中出现的字符从文本字串的拷贝中抹去,然后再重复这一寻找未引用字符的处理过程,直至所有字符都被引用了为止。

在第一恢复处理过程结束时,数据结构中包含了一组展布在整个文本字串上的不重叠条目。这样,字串“*thexresult*”被恢复成“*the*”,“*x*”和“*result*”。这一处理过程可能会用来优先产生另一组条目“*t*”、“*hex*”和“*result*”作为最后串。

附加邻近性限制

在应用了第一邻近性处理和第一恢复处理之后,有可能借助第二邻近性处理从数据结构中删除多余的条目。第二邻近性处理删除不与另一个未被删字相邻或不与字串开头或结尾相邻的词。图3显示出用标记3标志其末尾不与另一未被删除词开头相邻的词,用标记4标志其开头不与前面的未被删除词相邻的词。

第一邻近处理有助于建立词的边界,而第二邻近处理实施一次更严格的逻辑一致性检验。

数字串处理

在中文文本中包含各种数字串需要作为一个单元处理，而它们不能期望出现在词典中，因为对能遇到的数字组合的数目是无限制的。数字处理过程识别出只含有数字调符的所有字串并建立包含任何相邻数字串的单一数据结构条目。

消除多义性

图 4 显示出在第二邻近性处理之后仍保留了某些多义性。应该是“soft”（软）和“ware”（器件）还是应该为“software”（软件）？在这里正可以使用附加词典数据来解决这一问题。可以使用频率信息来判定它是两个词而不是一个词的可能性。对于某些应用，如 IR/S，甚至可能希望对全部这三个词检索。数据结构使得有可能保留或消除这种多义性。

下面是图 5 的流程图中进行的步骤，这是用于实现本发明的方法的一系列步骤：

步骤 1. 从输入设备输入一个字符串并存贮于计算机内部存储器。

步骤 2. 逐个字符地扫描内存贮器中存贮的字符串。造成一个数据结构，它包含能在词典中找到的每个子字符的位置和长度。在数据结构中的每个这样的条目叫做“词”，并伴有一个状态指示，使得可能逻辑删除该词或恢复一个被删除词。

步骤 3. 第一邻近性处理删除任何前面没有词或后面没有词的那些词。就是说，如果一个词不在字串开头或者前面没有一个未被

删除词,而且它不在该字符串末尾或者没有跟随一个未被删除词,那么这个词便被删除。重复这一处理过程直至再没有可被删除词为止。

步骤 4. 恢复处理识别出在输入字符串中没有被数据结构中删除词复盖的部分。实现这种识别的作法是造成一个输入字符串拷贝并从这一拷贝中去掉属于数据结构中未删除词的那些字符。然后恢复处理完成恢复字符串拷贝中任何剩余字符构成的被删除词。当字符串拷贝中的字符不能由恢复被删除词来复盖时,对该单个字符建立新的数据结构条目。当恢复一个词或建立一个新的数据结构条目时,便去掉字符串拷贝中的相应字符。重复这一处理过程直至字符串拷贝中的所有字符都被去掉为止。到这时,输入字符串中的所有字符被至少一个数据结构条目所复盖。

步骤 5. 数字处理将相邻的数值字符集合成单一数据结构条目。数据结构被扫描以找出只包含数值字符的数据结构条目。当发现几个这种相邻条目时,则将它们全部删除而恢复其中第一个条目,但其长度包括了所有这些相邻字符。

步骤 6. 这一个邻近性处理等效于步骤 3。它保证在做成数据结构后没有任何词前面没有词或后面没有词。

步骤 7. 输出处理是选择数据结构条目供输出到打印设备、检索处理、或数据库处理。输出处理可以在选择数据结构条目时使用统计信息。例如,由于二字符中文词出现频率高于单字符词或有更多字符的词,因此当需要在数据结构中进行选择时输出处理将会给二字

符词以优先。对于打印信息可能希望有这种选择。然而,对于数据库中检索信息,可能对数据结构中的全部词建立索引更有利于最大限度地检索数据。

图6的部分A和部分B给出被处理的两个字符串实例。图7的部分A和部分B给出被处理的另外两个实例。图8中的部分A和部分B又给出两个被处理的字符串实例。图6、7、8给出本发明的操作和实现的结果。所实现的发明提供了从连续中文文本中分离出中文词的一种改进的方法。

虽然已披露了本发明的一个具体实施例,但精通本门技艺的人们将会理解,对这一具体实施例可进行许多改变而不偏离本发明的精神和范围。

<softwaredevelopment>			
标记	位置	长度	词
0	0	2	<so>
0	0	4	<soft>
0	0	8	<software>
0	1	2	<of>
0	1	3	<oft>
0	4	3	<war>
0	4	4	<ware>
0	5	1	<a>
0	5	3	<are>
0	6	2	<re>
0	6	3	<red>
0	6	9	<redevelop>
0	6	13	<redevelopment>
0	8	7	<develop>
0	8	11	<development>
0	9	3	<eve>
0	12	3	<lop>
0	15	2	<me>
0	15	3	<men>

图 1—词典查寻处理后的数据结构

<softwaredevelopment>			
标记	位置	长度	词
1	0	2	<so>
0	0	4	<soft>
0	0	8	<software>
1	1	2	<of>
2	1	3	<oft>
1	4	3	<war>
0	4	4	<ware>
2	5	1	<a>
2	5	3	<are>
0	6	2	<re>
0	6	3	<red>
0	6	9	<redevelop>
0	6	13	<redevelopment>
0	8	7	<develop>
0	8	11	<development>
0	9	3	<eve>
0	12	3	<lop>
1	15	2	<me>
1	15	3	<men>

图 2—第一邻近性处理后的数据结构

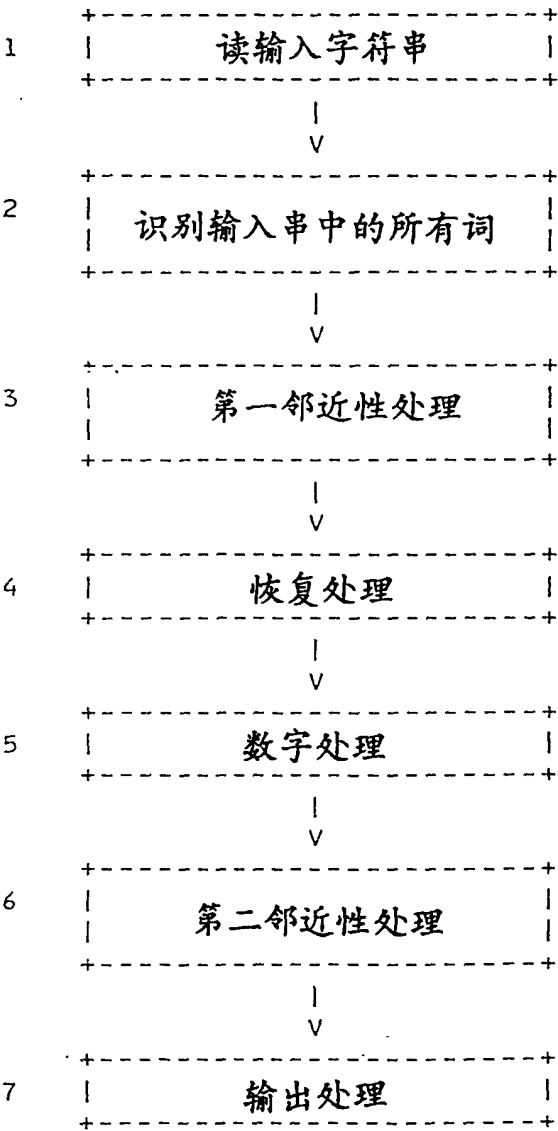
<softwaredevelopment>			
标记	位置	长度	词
1	0	2	<so>
0	0	4	<soft>
0	0	8	<software>
1	1	2	<of>
2	1	3	<oft>
1	4	3	<war>
0	4	4	<ware>
2	5	1	<a>
2	5	3	<are>
4	6	2	<re>
4	6	3	<red>
3	6	9	<redevelop>
4	6	13	<redevelopment>
3	8	7	<develop>
0	8	11	<development>
4	9	3	<eve>
3	12	3	<lop>
1	15	2	<me>
1	15	3	<men>

图 3—第二邻近性处理后的数据结构

<softwaredevelopment>			
标记	位置	长度	Wc词
0	0	4	<soft>
0	0	8	<software>
0	4	4	<ware>
0	8	11	<development>

图 4—第二邻近性处理后未被删除词

处理流程图
图 5



请输入要处理的字符串

<在舉行的第一天會議中，>

分离结果：

```
loc: 0, len: 2, <在>
loc: 2, len: 2, <舉>
loc: 2, len: 4, <舉行>
loc: 4, len: 2, <行>
loc: 6, len: 2, <的>
loc: 8, len: 4, <第一>
loc: 12, len: 2, <天>
loc: 14, len: 4, <會議>
loc: 18, len: 2, <中>
loc: 20, len: 2, <, >
```

请输入要处理的字符串

<誠摯地促請美方一本以往合作的精神，>

分离结果

```
loc: 0, len: 2, <誠>
loc: 2, len: 2, <摯>
loc: 4, len: 2, <地>
loc: 6, len: 2, <促>
loc: 8, len: 2, <請>
loc: 10, len: 2, <美>
loc: 12, len: 2, <方>
loc: 14, len: 2, <一>
loc: 16, len: 2, <本>
loc: 18, len: 2, <以>
loc: 18, len: 4, <以往>
loc: 20, len: 2, <往>
loc: 22, len: 2, <合>
loc: 22, len: 4, <合作>
loc: 24, len: 2, <作>
loc: 26, len: 2, <的>
loc: 28, len: 4, <精神>
loc: 32, len: 2, <, >
```

图 6

请输入要处理的字符串

<據中華社二十六日華盛頓電，>

分离结果

```
loc: 0, len: 2, <據>
loc: 2, len: 4, <中華>
loc: 6, len: 2, <社>
loc: 8, len: 6, <二十六>
loc: 14, len: 2, <日>
loc: 16, len: 6, <華盛頓>
loc: 22, len: 2, <電>
loc: 24, len: 2, <,>
```

请输入要处理的字符串

<專程來美參加中美保護智慧財產權問題諮商的國府代表團，>

分离结果

```
loc: 0, len: 2, <專>
loc: 2, len: 2, <程>
loc: 4, len: 2, <來>
loc: 6, len: 2, <美>
loc: 8, len: 4, <參加>
loc: 12, len: 2, <中>
loc: 12, len: 4, <中美>
loc: 14, len: 2, <美>
loc: 16, len: 4, <保護>
loc: 20, len: 4, <智慧>
loc: 24, len: 4, <財產>
loc: 24, len: 6, <財產權>
loc: 28, len: 2, <權>
loc: 30, len: 4, <問題>
loc: 34, len: 2, <諮>
loc: 36, len: 2, <商>
loc: 38, len: 2, <的>
loc: 40, len: 2, <國>
loc: 42, len: 2, <府>
loc: 44, len: 2, <代>
loc: 44, len: 4, <代表>
loc: 44, len: 6, <代表團>
loc: 46, len: 2, <表>
loc: 48, len: 2, <團>
loc: 50, len: 2, <,>
```

图 7

请输入要处理的字符串

<共同設法解決有關智慧財產權問題，>

分离结果

```
loc: 0, len: 4, <共同>
loc: 4, len: 2, <設>
loc: 6, len: 2, <法>
loc: 8, len: 4, <解決>
loc: 12, len: 2, <有>
loc: 12, len: 4, <有關>
loc: 14, len: 2, <關>
loc: 16, len: 4, <智慧>
loc: 20, len: 4, <財產>
loc: 20, len: 6, <財產權>
loc: 24, len: 2, <權>
loc: 26, len: 4, <問題>
loc: 30, len: 2, <,>
```

请输入要处理的字符串

<並將國府從美方的優先國家名單中除名。>

分离结果

```
loc: 0, len: 2, <並>
loc: 2, len: 2, <將>
loc: 4, len: 2, <國>
loc: 6, len: 2, <府>
loc: 8, len: 2, <從>
loc: 10, len: 2, <美>
loc: 12, len: 2, <方>
loc: 14, len: 2, <的>
loc: 16, len: 4, <優先>
loc: 20, len: 2, <國>
loc: 20, len: 4, <國家>
loc: 22, len: 2, <家>
loc: 24, len: 4, <名單>
loc: 28, len: 2, <中>
loc: 30, len: 4, <除名>
loc: 34, len: 2, <。>
```

图 8