自动分词

高琰

目录

01

概述

分词概述

- 自动分词过程：从信息处理需要出发，按照特定规范，对分词单位进行划分的过程
- 分词单位：指汉语信息处理使用的、具有确定的语义或语法功能的基本单位，包括词和少量词组
- 词组：由两个或两个以上的词，按一定的语法规则组成，以表达一定意义的语言单位

71

分词:从字符串到词串

- 学生人数多又能保证质量的才是好学校。 （字串）
- 学生 人数 多 又 能 保证 质量 的 才 是 好学校 （词串）
- Schools which can both enroll more students and ensure the quality of instruction are the successful ones.

关于中文分词的形式化定义，可参看马婴，1991， 基于评价的汉语自动分词系统的研究与实现，黄昌宁、夏莹编《语言信息处理专论》，清华大学出版社1996年版

分词的意义

- 正确的机器自动分词是正确的中文信息处理的基础
  - 文本检索
    - 和服 | 务 | 于三日后裁制完毕，并呈送将军府中。
    - 王府饭店的设施 | 和 | 服务 | 是一流的。
      如果不分词或者“和服务”分词有误，都会导致荒谬的检索结果。
  - 文语转换
    - 他们是来 | 查 | 金泰 | 撞人那件事的。(“查”读音为cha)
    - 行侠仗义的 | 查金泰 | 远近闻名。 (“查”读音为zha)

71 7579846@qq.com, 2020/2/16

02

英文分词

英文分词

1. Dog's Let's
2. ad hoc and so on New York
3. strong stronger strongest
4. buy bought buying
5. eat ate eaten
6. try tried tries
7. treat treatment
8. news newspaper

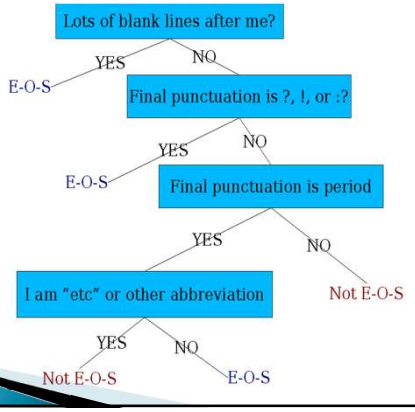
英文分词处理-术语

- Word Tokenization(词素切分)
  - Tokenization：把字符串变为词串
    - I'm a student -> I 'm a student
  - 英文的词素拆分是主要是两个步骤：首先将文本进行句子分割；其次对句子进行分割得到词串。
- Word Stemming/Lemmatization
  - Lemmatization：对词进行内部结构和形式分析，又叫**词形还原**
    - took -> take + ed (past tense)
  - Stemming:提取词干，Apple->Appl

Sentence Segmentation

- !, ? are relatively unambiguous
- Period "." is quite ambiguous
  - Sentence boundary
  - Abbreviations like Inc. or Dr.
  - Numbers like .02% or 4.3
- Build a binary classifier
  - Looks at a "."
  - Decides EndOfSentence/NotEndOfSentence
  - Classifiers: hand-written rules, regular expressions, or machine-learning

Determining if a word is end-of-sentence: a Decision Tree



Lemmatization(词形还原)

- Reduce inflections or variant forms to base form
  - *am, are, is → be*
  - *car, cars, car's, cars' → car*
- *the boy's cars are different colors → the boy car be different color*
- Lemmatization: have to find correct dictionary headword form

Stemming（词干提取）

- 英语词汇由两部分构成：词干和词缀，词缀又分前缀和后缀。
  - 如happiness->happy（stem）。
- 词干提取就是把具有词性变化的单词还原成词干形式，然后再查词典获取单词的基本信息

for example compressed and compression are both accepted as equivalent to compress.

→

for exampl compress and compress ar both accept as equival to compress

Stemming（词干提取）

为什么要词干提取

一种对同义词的处理方式:采用相同词干作为查询拓展，将包含查询词词干的所有文档返回过来

Stemming（词干提取）

- 波特词干器（Porter Stemmer，Porter1）是马丁.波特博士于1979年提出来的,是基于规则的方法的提取算法，应用最为广泛的、中等复杂程度的词干提取算法。比较热门的检索系统包括Lucene、Whoosh等中的词干过滤器就是采用波特词干算法。

在Porter算法中，v表示为一个元音字母（vowel），c表示一个辅音字母（consonant），C表示连续的辅音字母串，V表示连续的元音字母串。一个英文单词可以表示为如下形式：

[C][VC]m[V]

其中，原括号表示为必选项，方括号表示为可选项，m表示为括号内成分的重复次数。

Stemming（词干提取）

[C][VC]m[V]

- ***S：词干以字母S结尾（S也可以替换为其他字母）**
- ***v*：词干含有一个元音字母**
- ***d：词干以连续两个相同辅音字母为结尾**
- ***o：词干以cvc形式结尾，其中第二个辅音不是W、X或Y，例如：-WIL、-HOP**

- **算法中每条规则表示为：**
  - **（condition）S1->S2**

其含义为：在condition条件下，后缀S1替换为S2。S2可以为空串，条件condition也可以为空（NULL）。

Stemming（词干提取）

[C][VC]m[V]

	规则	转换例子
第一组规则	sses->ss ies → i ss->ss s->NULL	addresses → address ponies → poni address → address dogs → dog
第二组规则	（m>0）eed→ee (*v*)ing → NULL (*v*)ed → NULL ……	feed → feed walking → walk sing → sing plastered → plaster
第三组规则	ational→ ate izer→ ize ator→ ate ize→NULL ……	relational→ relate digitizer → digitize operator → operate digitize → digit
第四组规则	（*v*）y->i	happy → happi sky → sky

Porter 词干还原工具的官方网址是:

http://www.tartarus.org/~martin/PorterStemmer/

基于NLTK的英文分词

用到的函数：

nltk.sent_tokenize(text) #对文本按照句子进行分割

nltk.word_tokenize(sent) #对句子进行分词

```
>>> import nltk
>>> text = 'PythonTip.com is a very good website. We can learn a lot form it.'
>>> #将文本拆分成句子列表
>>> sens = nltk.sent_tokenize(text)
>>> sens
['PythonTip.com is a very good website.', 'We can learn a lot form it.']
>>> #对句子进行分词，  nltk的分词是句子级的，因此要先分句，再逐句分词，否则效果会很差
>>> words = []
>>> for sent in sens:
        words.append(nltk.word_tokenize(sent))

>>> words
[['PythonTip.com', 'is', 'a', 'very', 'good', 'website', '.'], ['We', 'can', 'le
arn', 'a', 'lot', 'form', 'it', '.']]
```

http://textanalysisonline.com/nltk-word-tokenize

基于NLTK的其他语言的分词

```
>>> from nltk import SnowballStemmer >>>
SnowballStemmer.languages # See which languages
are supported ('danish', 'dutch', 'english',
'finnish', 'french', 'german', 'hungarian',
'italian', 'norwegian', 'porter', 'portuguese",
'romanian', 'russian', 'spanish', 'swedish') >>>
stemmer = SnowballStemmer("german") # Choose a
language >>> stemmer.stem(u"Autobahnen") # Stem a
word u'autobahn'
```

http://textanalysisonline.com/nltk-porter-stemmer

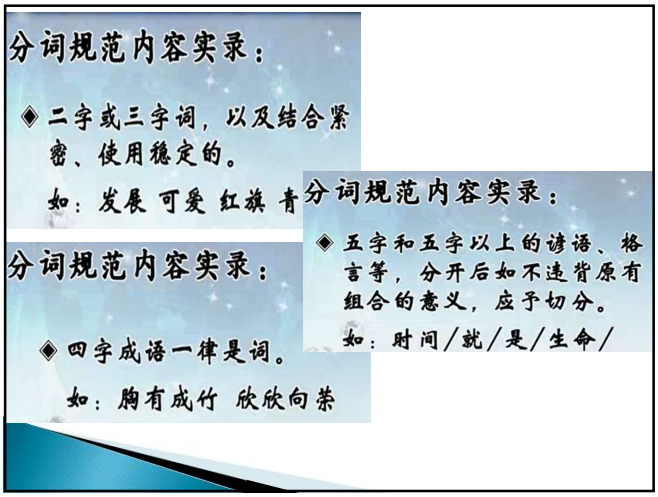
03

中文存在的问题

中文分词规范

- GB/T13715-1992《信息处理现代汉语分词规范》（1990）
- 《现代汉语语库加工规范—词语切分与词性标注》（1999，俞士汶）
- 分词的原则：

  结合紧密，使用稳定

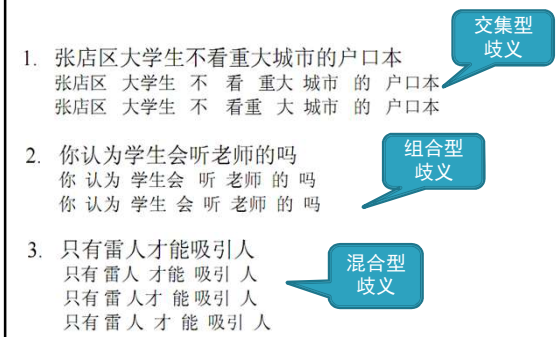


汉语分词存在的问题

- 自动分词存在2大难题：
  - 分词歧义
  - 未登录词识别

未登录词现象和歧义切分现象构成了影响分词系统准确率的两个因素。

切词中的歧义



基本定义

- 定义1：汉字串AJB称作交集型切分歧义，如果满足AJ，JB同时为词。此时的汉字串J称作交集串。
- 定义2:汉字串AB称作多义组合型切分歧义，如果满足A、B、AB同时为词。

交集型歧义的链长

- **交集型歧义字段中含有交集字段的个数，称为链长**
  - 链长为1： 和尚未
  - 链长为2： 结合成分
  - 链长为3： 为人民工作
  - 链长为4： 中国产品质量
  - 链长为5： 鞭炮声响彻夜空
  - 链长为6： 努力学习语法规则
  - 链长为7： 中国企业主要求解决
  - 链长为8： 治理解放大道路面积水
  - ……

具实文本中分词歧义的分布情况

交集型歧义：组合型歧义 = 1: 22 语料规模：17,547字 [1]

语料规模：500万字新闻语料 [2]

链长 / 歧义字段	1	2	3	4	5	6	7	8	总计
Token次数	47402	28790	1217	608	29	19	2	1	78248
比例%	50.58	47.02	1.56	0.78	0.04	0.02	0.00	0.00	100
Type种数	12686	10131	743	324	22	5	2	1	23914
比例%	53.05	42.36	3.11	1.35	0.09	0.02	0.01	0.01	100

[1] 刘挺、王开铸，1998，关于歧义字段切分的思考与实验。《中文信息学报》第2期，63-64页。

[2] 刘开瑛，2000，《中文文本自动分词和标注》，商务印书馆，65页。

13

汉语真实文本中的分词歧义情况(续)

- **真歧义**
  - 确实能在真实语料中发现多种切分形式
  - 比如“应用于”、“地面积”、“解除了”
- **伪歧义**
  - 虽然有多种切分可能性，但在真实语料中往往取其中一种切分形式
  - 比如“挨批评”、“市政府”、“太平淡”

汉语真实文本中的分词歧义情况(续)

78248个[1]交集型歧义字段

- 伪歧义：94%
- 真歧义：6%
  - 多种切分均匀分布 12% (甲)
  - 一种切分切分占优 88% (乙)

(甲) 将信息技术/应用/于/教学实践
信息技术/应用于/教学中的哪个方面

(乙) 上级/解除/了/他的职务
方程的/解/除了/零以外还有...

[1] 刘开瑛，2000，《中文文本自动分词和标注》，商务印书馆，66-67页。

汉语真实文本中的分词歧义情况（续）

分词歧义的四个层级（语料规模：**50883**字）[1]

- 词法歧义：84.1% （“用方块图形式加以描述”）
- 句法歧义：10.8% （“他一阵风似的跑了”）
- 语义歧义：3.4% （“学生会写文章”）
- 语用歧义：1.7% （“美国会采取措施制裁伊拉克”）

[1] 何克抗 等，1991，《书面汉语自动分词专家系统设计原理》，载《中文信息学报》，1991年第2期。

不同的人对“词”的认识有差异

6人对100句（4372）字进行人工分词，然后两两比较认同率

	M2	M3	T1	T2	T3
M1	0.77	**0.69**	0.71	**0.69**	0.70
M2		0.72	0.73	0.71	0.70
M3			**0.89**	0.87	0.80
T1				0.88	0.82
T2					0.78

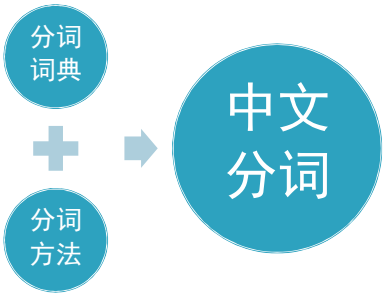
平均值
0.76

Sproat R. et al., 1996, A Stochastic Finite-state Word Segmentation Algorithm for Chinese. Computational Linguistics, Vol.22, No.3, pp377-404.

未登录词

未登录词就是在大规模真实文本处理中遇到的许多不能由词典识别的人名地名术语等词汇。

中文分词



分词词典-索引结构

- 整词索引
  - **支持整词索引**
- 分级索引
  - **支持前缀索引**
- 倒排索引
  - **支持词的片段检索**

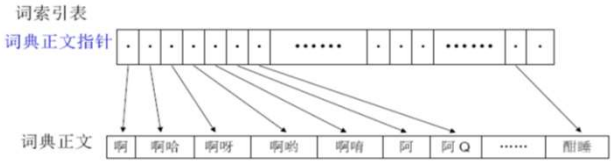
分词词典-索引结构

- 词典检索算法的性能评价
  - 时间复杂度
  - 空间复杂度.
  - 增量式索引:词典增加或修改时不用重建全部索引
- 整词索引：只支持整词查找
- 常见索引结构
  - 顺序索引
  - 散列索引

词项的整数表示

- 词在内存中表示为一个不定长的字符串
- 在很多自然语言处理应用中，为了提高效率，通常将一个词项表示为一个整数，由于整数是定长的，这样可以大大减少存储空间。
- 另外，由于整数比较速度大大快于字符串比较，这样也可以提高系统运行速度。
- 词项整数化需要查表，这种查表可以用整词通过整词所以实现。

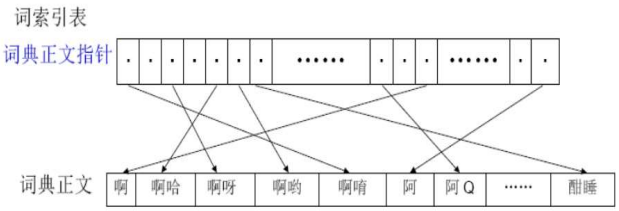
词典顺序索引



- 索引结构简单，占用空间小
- 不能实现增量式索引：每增加一个词需重新排序

整词二分查找:
时间复杂度O(logN)
无法按前缀查找:查找时精确匹配

词典散列索引



索引结构简单，占用空间小(比顺序索引稍大)
可以实现增量式索引

词典散列索引的检索算法

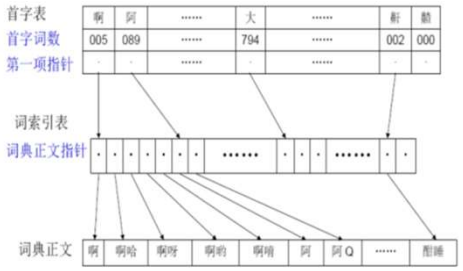
- 利用散列(hash)函数直接定位
- 效率高:常数
- 不能按前缀查找
- 冲突的解决
  - 使用冲突队列
  - 使用再散列
- 散列函数(hash)的选择

索引结构类型

- 整词索引:只支持整词检索
  - 必须知道整个词才能检索
- 分级索引:支持前缀检索
  - 知道词的前缀就可以检索
- 倒排索引:支持片段检索、模糊检索
  - 知道词的任意片段就可以检索

首字索引

- 对于汉语而言，第一级索引一般使用词语的首字，所以又常称为首字索引



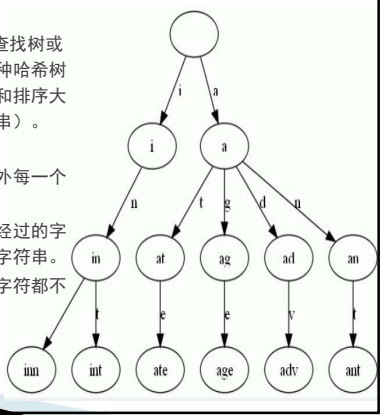
首字二分查找

- 时间复杂度: 0(logN)
- 空间复杂度: 0(N)
- 可以按前缀查找
- 不能增量式索引:每次要重新排序

增加操作：查找时间+指针移动时间
删除操作：查找时间+指针移动时间

TRIE 树索引

- Trie树，即字典树，又称单词查找树或键树，是一种树形结构，是一种哈希树的变种。典型应用是用于统计和排序大量的字符串（但不仅限于字符串）。
- 基本性质：
根节点不包含字符，除根节点外每一个节点都只包含一个字符。
- 从根节点到某一节点，路径上经过的字符连接起来，为该节点对应的字符串。
- 每个节点的所有子节点包含的字符都不相同。

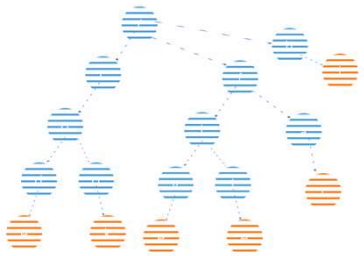


TRIE树索引

- Trie树的插入操作
Trie树的插入操作很简单，其实就是将单词的每个字母逐一插入Trie树。插入前先看字母对应的节点是否存在，存在则共享该节点，不存在则创建对应的节点。

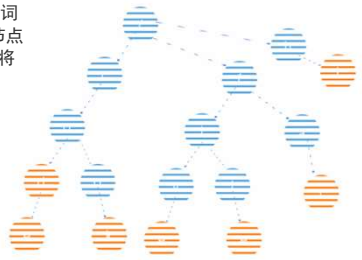
TRIE树索引

- **Trie树的删除操作：**
删除整个单词（比如hi）



TRIE树索引

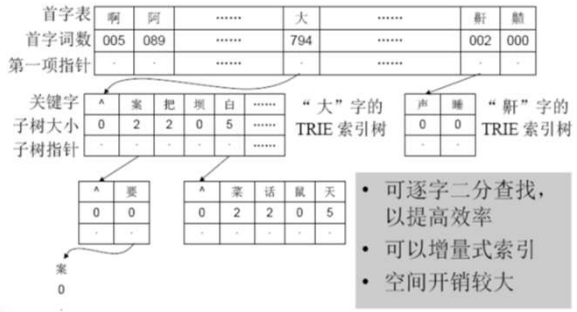
- **Trie树的删除操作**
删除前缀单词（比如cod）：只需要将cod单词整个字符串查找完后，d节点因为不是叶子节点，只需将其单词标志去掉即可。



TRIE树索引

- **删除分支单词（比如cook）：**与删除整个单词情况类似，区别点在于删除到cook的第一个o时，该节点为非叶子节点，停止删除，这样就完成cook。

汉语词典TRIE树索引



TRIE树索引

- Trie的核心思想是空间换时间，利用字符串的公共前缀来降低查询时间的开销以达到提高效率的目的。
- 假设字符的种类数有m个，有若干个长度为n的字符串构成了一个Trie树，则每个节点的出度为m（即每个节点的可能子节点数量为m），Trie树的高度为n。
  - 空间复杂度：O（m^n）
  - 查询的时间复杂度：O（n）

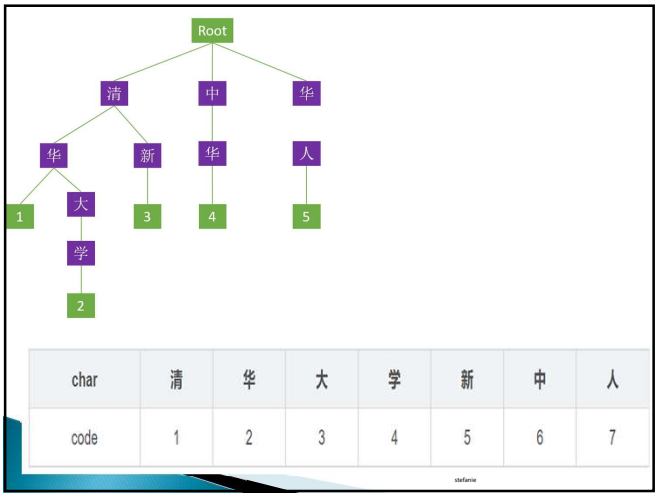
解决方案：双数组Trie树

TRIE树索引

- 双数组Trie树：用base[]和check[]两个数组就将整个trie的信息储存了起来，这两个数组的构建规则是：

  base[i]+code(x)=j

  check[j]=i
  - base[]数组用于记录跳转结构， base array中index为i的那个节点，如果按照字符x转移，会转移到index为j的节点
  - check[]数组用于 标识出 base array 中每个状态的前一个状态，其主要作用是检验按base做转移的转移正确性
  - code(x)是字符x的编码，现实中为了方便通常直接用char code



char	清	华	大	学	新	中	人
code	1	2	3	4	5	6	7

char	清	华	大	学	新	中	人
code	1	2	3	4	5	6	7

初始化root的base index为0，base值记为1。首先看root的所有子节点"清，中，华"

目前base array的情况如下：

position	0	1	2	3	4	5	6	7	8	9	10
char	root		清	华				中			
base	1										

position	0	1	2	3	4	5	6	7	8	9	10
char	root		清	华				中			
base	1										

- **每次遍历完一个节点的所有子节点，只可以确认当前节点的base值，以及它的子节点的index位置**
- **子节点的base值此时会默认继承当前节点的base值，但在遍历子节点的子节点时，一旦有冲突，子节点的base值就会做相应修改**

position	0	1	2	3	4	5	6	7	8	9	10
char	root		清	华				中			
base	1										

接下来遍历第二层的节点"华，新，华，人"：

char	清	华	大	学	新	中	人
code	1	2	3	4	5	6	7

position	0	1	2	3	4	5	6	7	8	9	10
char	root		清	华	中华	清华		中	清新	华人	
base	1		3	2				2			

position	0	1	2	3	4	5	6	7	8	9	10
char	root		清	华	中华	清华	清华大	中	清新	华人	清华大学
base	1		3	2	2	3	6	2	3	2	6
check	-2		0	0	7	2	5	0	2	3	6

TRIE树索引

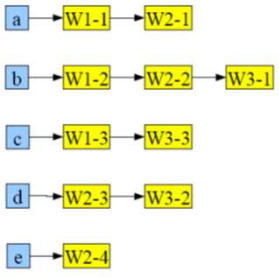
TRIE树索引

- 题目：给你100000个长度不超过10的单词。对于每一个单词，我们要判断他出没出现过，如果出现了，求第一次出现在第几个位置。
- 解法：从第一个单词开始构造Tire树，Tire树包含两字段，字符与位置，对于非结尾字符，其位置标0，结尾字符，标注在100000个单词词表中的位置。对词表建造Tire树，对每个词检索到词尾，若词尾的数字字段>0,表示单词已经在之前出现过，否则建立结尾字符标志，下次出现即可得知其首次出现位置，便利词表即可依次计算出每个单词首次出现位置复杂度为O（N×L）L为最长单词长度，N为词表大小

倒排索引

- 根据字母(汉字)反查单词，
- 查询方式灵活，在信息检索中被广泛采用

编号	词语
W1	abc
W2	abde
W3	bdc

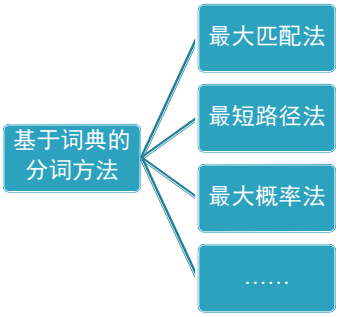


05

分词方法

- *the boy's cars are different colors*
  - → *the boy ‘s car are different colors*
  - → *the boy car be different color*
  - → *the boy car ar differ color*
- 提高中国产品质量？

分词方法



最大匹配算法

- 最大正向匹配

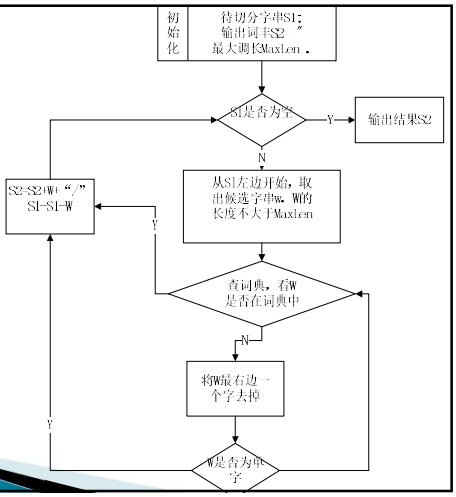
  例：“后天我们去北京”， 最大词长：4

后天我们　后天我　后天　我们去北　我们去　我们

去北京　去北　去　北京

最大匹配算法

- 最大正向匹配



逆向最大匹配

- 基本思想与正向最大匹配相同，唯一的区别是最大匹配的顺序不是从首字开始，而是从末尾开始。

  例：后天我们去北京 最大词长：4

最大匹配的问题

- **有意见分歧？**

  FMM　有意/见/ 分歧/

  BMM　　有/意见/ 分歧/

- **原子结合成分子时？**

  FMM&BMM 原子/结合/成分/子时/?

最大匹配法+规则

规则示例

IF W = "个人", W_{Left} = 数词 THEN W = "个/ 人/" ENDIF

最大匹配法+规则

分词原则:

- 颗粒度越大越好
- 切分结果中非词典词越少越好，单字字典词数越少越好
- 总体词数越少越好

双向最大匹配+规则

- 我们在野生动物园玩 ,maxLen=5
  - 正向:我们/在野/生动/物/园/玩
  - 逆向:我们/在/野生动物园/玩
  - 选择
    - 非字典词：正向(1)>逆向(0)（越少越好）
    - 单字字典词：正向(2)=逆向(2)（越少越好）
    - 总词数：正向(6)>逆向(4)（越少越好）

????

增加回溯机制

- 对于某些交集型歧义，可以通过增加回溯机制来修改最大匹配法的分词结果。
- 例如:“爱人民英雄”
  顺向扫描的结果是:“爱人/民/英雄/”,

通过查词典知道“民”不在词典中，于是进行回溯，将“爱人”的尾字“人”取出与后面的“民”组成“人民”，再查词典，看“爱”，“人民”是否在词典中，如果在，就将分词结果调整为:“爱/人民/英雄/”

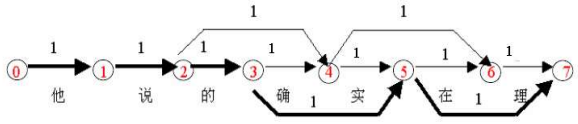
最短路径匹配算法

建立词图：

设分词串S=$v_1v_2…v_n$, 其中v_i为单个字，n为串的长度，n≥1.

根据词典，顺序匹配出在中文字串中所有可能的出现的词的集合，构造成为一个有向无环图

- 他说的确实在理



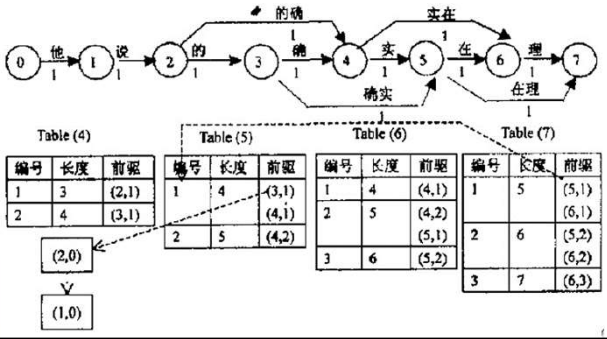
在词图上选择---条词数最少的路径

最短路径匹配算法

- 最短路径方法基本步骤：
  - 每个节点记录K个最短路径值，并记录相应路径上当前节点的前驱。
  - 如果同一长度对应多条路径，必须同时记录这些路径上当前节点的前驱。最后通过回溯求解N类最短路径。

最短路径匹配算法

- 张华平等（2002）基于N-最短路径的中文词语粗分模型，中文信息学报



最短路径分词法

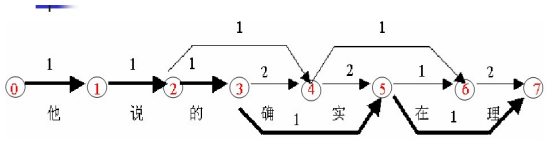
- 优点：好于单向的最大匹配方法
  - 最大匹配：独立自主 和平 等 互利 的 原则 (6 words)
  - 最短路径：独立自主 和 平等互利 的 原则 (5 words)
- 缺点：同样无法解决大部分交集型歧义
  - 结合 成分 子时
  - 他 说 的 确实 在理 （都是最短路径）
    他 说 的确 实在 理
    他 说 的确 实 在理

最短路径分词法

基本观察	大多数单字在语境里如果能组成合适的词就不倾向于单独使用。	
基本概念	半词	如果一个字不单独作为词使用，就是半词。半词既包含了成词语素，也包含了不成词语素，后者肯定是半词，比如“民”，前者则要看它作为语素的使用频度高，还是作为单字词的使用频度高，比如“见”。
	整词	如果一个字更倾向于自己成词而不倾向于和别的字组成词，这类“单字词”就称之为“整词”。这类词就是一般说的单字高频成词语素，比如“人、说、我”等。
基本思路	充分利用半词和整词的差别，尽量选择没有半词落单的分词方案。	

半词法分词的实现

- 在词图的路径优劣评判中引入罚分机制
- 罚分规则：
  1 每个词对应的边罚1分。
  2 每个半词对应的边加罚1分。
  3 一个分词方案的评分为它所对应的路径上所有边的罚分之和。
  4 最优路径就是罚分最低的分词路径。



他 说 的 确实 在理 （1+1+1+1+1 = 5分）
他 说 的确 实 在理 （1+1+1+2+1 = 6分）
他 说 的确 实在 理 （1+1+1+1+2 = 6分）

“有意见分歧”的问题

基于统计的分词方法

- 基于统计的消歧方法
  - 最大概率法分词
  - 基于互信息
  - t-测试差的歧义切分方法

最大概率法分词

输入　字符串S: 有意见分歧

输出　词串 W1: 有/ 意见/ 分歧/

词串W2: 有意/ 见/ 分歧/

$$Max(P(W1|S), P(W2|S)) \ ?$$

$$P(W|S)=\frac{P(S|W)\times P(W)}{P(S)}\approx P(W)$$

独立性假设，一元语法

$$P(W)=P(w_1,w_2,...,w_i)\approx P(w_1)\times P(w_2)\times ...\times P(w_i)$$

$$P(w_i)=\frac{w_i在语料库中的出现次数n}{语料库中的总词数N}=\frac{Freq(w_i)}{N}$$

最大概率法分词

词语	概率
…	…
有	0.0180
有意	0.0005
意见	0.0010
见	0.0002
分歧	0.0001
…	…

P(W1) = P(有) * P(意见) * P(分歧)

= 1.8 × 10^{-9}

P(W2) = P(有意) * P(见) * P(分歧)

= 1.0 × 10^{-11}

P(W1) > P(W2)

- 动态规划算法：最优路径中的第i 个词Wi 的累积概率等于它的左邻词Wi-1的累积概率乘以Wi自身的概率。

$$P'(w_i)=P'(w_{i-1})\times P(w_i)$$

- 为方便计算，一般把概率转化为路径费用（代价）

$$C=-\log(P)$$

$$C'(w_i)=C'(w_{i-1})+C(w_i)$$　　公式1

最大概率分词法的实现：动态规划算法

1) 对一个待分词的字串 S，按照从左到右的顺序取出全部候选词$w_1, w_2, \dots, w_i, \dots, w_n$；
2) 到词典中查出每个候选词的概率值$P(w_i)$，转换为费用$C(w_i)$，并记录每个候选词的全部**左邻词**；
3) 按照公式1计算每个候选词的**累计费用**，同时比较得到每个候选词的**最佳左邻词**；
4) 如果当前词w_n是字串S的尾词，且累计费用$C'(w_n)$最小，则w_n就是S的终点词；
5) 从w_n开始，按照从右到左顺序，依次将每个词的最佳左邻词输出，即为S的分词结果。

最大概率法分词示例（续）

序号	候选词	费用	累计费用	最佳左邻
0	结	3.573	3.573	-1
1	结合	3.543	3.543	-1
2	合	3.518	7.091	0
3	合成	4.194	7.767	0
4	成	2.800	6.343	1
5	成分	3.908	7.451	1
6	分	2.862	9.205	4
7	分子	3.465	9.808	4
8	子	3.304	10.755	5
9	子时	6.000	13.451	5
10	时	2.478	12.286	7

最大概率分词中的思政

- 最大概率法分词采用了动态规划
  - 知行合一

给出问题具备最优子结构和重叠子问题性质的定理是“知”，自底向上求解各阶段子问题的最优值是“行”。

  - 不积跬步,无以至千里.不积小流,无以成江海

基于互信息和t-测试差的歧义切分方法

互信息 对有序字符串xy，汉字x、y之间的互信息定义为：

$$I(x;y)=\log_2\frac{p(x,y)}{p(x)P(y)}$$

其中p(x,y)是x，y的邻接同现概率，p(x),p(y)分别表示x、y的独立概率。

若在字容量为N的汉语语料库中，x、y的邻接同现此时为r(x,y)，x、y的独立出现次数分别为r(x)，r(y)，则上式各量下列各式估计：

$$p(x,y)=\frac{r(x,y)}{N}\qquad p(x)=\frac{r(x)}{N}\qquad p(y)=\frac{r(y)}{N}$$

互信息反映了汉字对间结合关系的紧密程度：

① 当I(x,y)>>0时，则p(x,y)>>p(x)p(y),此时x、y间有紧密结合关系， I(x,y)值越大，结合度越强。

②当I(x,y) ≈ 0时，则p(x,y) ≈ p(x)p(y),此时x、y间有结合关系不确定。

③ 当I(x,y)<<0时，则p(x,y)<<p(x)p(y),此时x、y间有基本没有结合关系， I(x,y)值越小，结合度越弱。

基于互信息和t-测试差的歧义切分方法

t-测试 对有序汉字串xyz，汉字y相对于x及z的t-测试定义为：

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

其中p(x|y) ，p(y|z)分别是y关于x，z关于y的条件概率，$\delta^2(p(y|x)+p(z|y))$则代表各自方差。上式各量可用下列各式估计。

$$p(y|x)=\frac{p(x,y)}{p(x)}=\frac{r(x,y)}{r(x)}\qquad p(z|y)=\frac{p(y,z)}{p(y)}=\frac{r(y,z)}{r(y)}$$

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

t-测试表示的意义如下：

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

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与前驱字x有相连的趋势，值越小，相连趋势越强。

基于互信息和t-测试差的歧义切分方法

t-测试差 对有序汉字串vxyw，汉字x 、 y之间的t-测试差为：

$$\Delta t(x;y)=t_{v,y}(x)-t_{x,w}(y)$$

t-测试差表示的意义如下：

① 当 $t_{v,y}(x)>0$，$t_{x,w}(y)<0$时，x.v之间相互吸引，必有 $\Delta t(x;y)>0$，x,y之间倾向于连，且趋势比单独使用$t_{v,y}(x)$或 $t_{x,w}(y)$ 更显得突出。

② 当$t_{v,y}(x)<0$，$t_{x,w}(y)>0$ 时， x,y之间相互排斥，必有 $\Delta t(x;y)<0$，x,y之间倾向于断。

③当 $t_{v,y}(x)>0$，$t_{x,w}(y)>0$时，y吸引x同时w吸引y，产生竞争。若$\Delta t(x;y)>0$，则倾向于连，若$\Delta t(x;y)<0$，则倾向于断。

④当 $t_{v,y}(x)<0$，$t_{x,w}(y)>0$时，x吸引y同时v吸引x，产生竞争。若$\Delta t(x;y)>0$，则倾向于连，若$\Delta t(x;y)<0$，则倾向于断。

分词系统评测

- 评测的主要指标为分词的准确率(P)、召回率(R)、覆盖率(COV)：

$$P=\frac{\text{系统输出中正确的标记个数}}{\text{系统输出的全部标记个数}}\times 100\%$$

$$R=\frac{\text{系统输出中正确的标记个数}}{\text{金标语料中全部正确的标记个数}}\times 100\%$$

$$COV=\frac{\text{金标语料中被系统标记的测试项的个数}}{\text{金标语料中测试项的总数}}\times 100\%$$

- “金标语料(goldstandardcorpus)”是指由人工标注或校对的质量很高的评测集的答案语料。

分词系统评测

- **黄金标准**:“结婚 的 和 尚未 结婚 的”
- 分词结果:“结婚 的 和尚 未结婚 的 ”

	分词区间	
黄金标准	[1,2],[3,3],[4,4],[5,6],[7,8],[9,9]	A
分词结果	[1,2],[3,3],[4,5],[6,7,8],[9,9]	B
重合部分	[1,2],[3,3],[9,9]	A∩B

Thank You !