# 中文分词

1.搜索引擎是如何理解我们的搜索语句的？

Like ‘%中国%’ 没有索引的 不能模糊，比如我搜湖南省 就搜不到湖南相关的

NLP处理

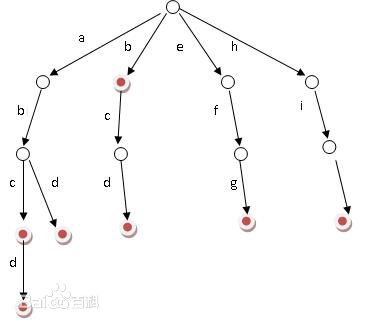
1.什么是Trie树：trie树就是我们平常说的字典树，它是一种专门用来处理字符串匹配的数据结构。特别适合用来在很多字符串中快速查找某一个特定的字符串。前缀树，赫夫曼树，前缀编码

2.Trie的数据结构：假设我们有以下几个英文单词：my name apple age sex，假如我们要查找里面某一个字符串是否存在，你怎么去找呢？

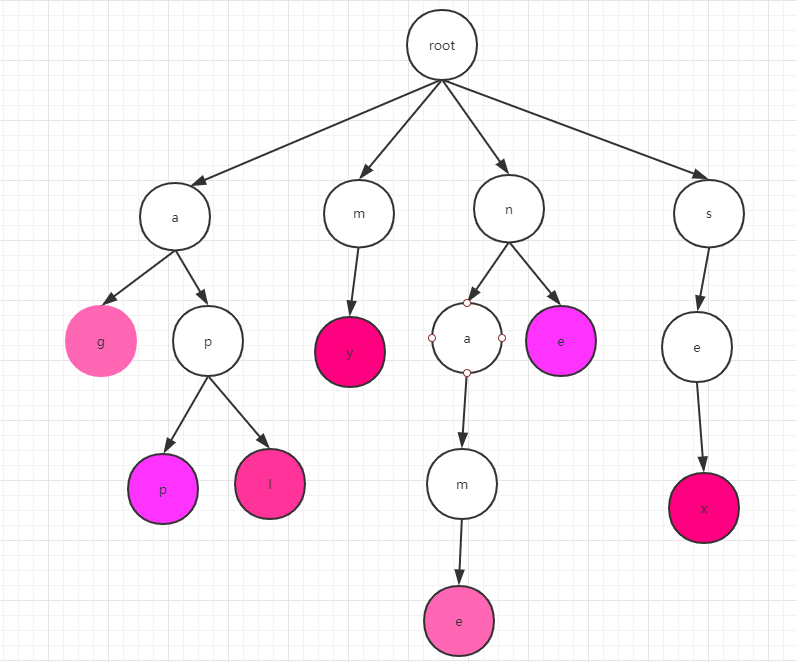
散列表：hashMap

如果利用Trie树我们该怎么来解决上面的查找问题呢？

我们可以将上面的字符串变成以下结构，利用字符串的公共前缀，将重复的合在一起组成一颗树，即为我们所要讲的trie树。



3.Trie树的构建：我们要先将词分成一个个的字母，然后再依次插入到树中。如右图所示，根节点root，如果我们要插入app



则首先将app分成：a,p,p，然后从root点开始，一层层的插入，注意的是

P会挂在a下面，后面的一个p会挂在前面的p上。单词的末尾我们就用紫色表示。

这里需要注意我们插入的时候每一层的字母都是有序的。

4.Trie的查找：

查找我们就从root点开始，再第一层找第一个字母，依次往下找到我们所要的单词。

注意要找到末尾的标记才算完成一个单词的查找。比如app，我们要找ap

虽然字典树里面有ap，但是这个p不是紫色那么ap还是不存在字典树中的。

5.Trie树的实现:其实Trie树就是一颗多叉树。这里我们应该要想到B+Tree&B-Tree，是有些类似的。

Trie树又是巧妙的利用了数组的下标，因为英文字母刚好是26个，所以我们可以开一个26长度的数组

A[] = new int[26];

A[0] = ‘a’ => 下标就是’a’-97 这里刚好就是0，利用的是ascii计算。

所以它的数据结构应该是

Class TrieNode{

Char c; //存储当前这个字符

TrieNode child[26]; //存储这个字符的子节点

}

6.Trie树的分析：

时间复杂度：非常高效 O(单词的长度)

空间复杂度：

以空间来换效率的数据结构。因为每个单词理论上都有26个子节点，所有它的空间复杂度就是26^n，n表示的是树的高度。

优化：

1.重复的字母不要重复建

2.因为我们每个node都开了26个空间来存储节点。但实际情况可能不需要这么多，所以这里其实我们可以考虑用散列表来实现，

这里大家可以去看IK的源码，当子节点少的时候是用的数组，但是节点大于3个是它是用的hashMap，这个再一定的程度上是可以节省很多的空间的。

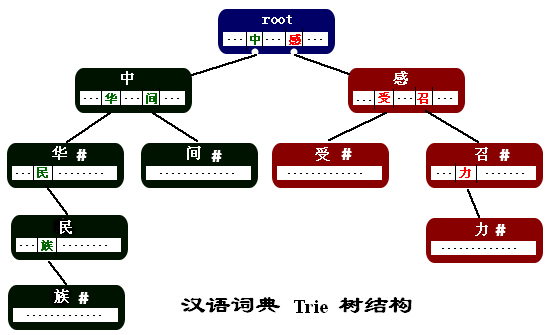
IK分词

1.分词的原理：

（1）英文分词：my name is zhaoyun

（2）中文分词：我是中国人

中文分词是所有自然语言处理以及搜索引擎的核心基础，试想下如果你都不能准确的分出用户所要表达的意思那么怎么可能可以弄出好的检索效果呢？以及怎么可能理解用户的意图呢？



2.中文分词器实战：

3.中文分词需要特别注意的是它有两种方式：

Smart：智能分词，这里不会分出一句话的所有情况，比如我是中国人，会分成 我 / 是 /中国人

最小颗粒（非Smart）：会分出一句话里面的所有情况，比如 我是中国人 会分成 我 / 是 / 中国 / 中国人 /国人

4.中文的歧义问题：

武汉市长江大桥

可分成：武汉市/长江大桥

可分成:武汉/市长/江大桥

以上两种分词就是完全不一样的理解，这个就是中文的博大精深所在，因为没有空格，所以没办法准确的知道用户的意图，这个在分词领域就叫做歧义问题。这种解决其实还是蛮难的，最简单的就是利用词库。复杂的还会运用到AI技术 机器学习算法等。

