## [Trie（前缀树/字典树）及其应用](https://www.cnblogs.com/justinh/p/7716421.html)

 Trie，又经常叫前缀树，字典树等等。它有很多变种，如后缀树，Radix Tree/Trie，PATRICIA tree，以及bitwise版本的crit-bit tree。当然很多名字的意义其实有交叉。

## **定义**

在计算机科学中，trie，又称前缀树或字典树，是一种有序树，用于保存关联数组，其中的键通常是字符串。与二叉查找树不同，键不是直接保存在节点中，而是由节点在树中的位置决定。一个节点的所有子孙都有相同的前缀，也就是这个节点对应的字符串，而根节点对应空字符串。一般情况下，不是所有的节点都有对应的值，只有叶子节点和部分内部节点所对应的键才有相关的值。

trie中的键通常是字符串，但也可以是其它的结构。trie的算法可以很容易地修改为处理其它结构的有序序列，比如一串数字或者形状的排列。比如，bitwise trie中的键是一串位元，可以用于表示整数或者内存地址

**基本性质**

1，根节点不包含字符，除根节点意外每个节点只包含一个字符。

2，从根节点到某一个节点，路径上经过的字符连接起来，为该节点对应的字符串。

3，每个节点的所有子节点包含的字符串不相同。

**优点**：

可以最大限度地减少无谓的字符串比较，故可以用于词频统计和大量字符串排序。

　　跟哈希表比较：

　　　　1，最坏情况时间复杂度比hash表好

　　　　2，没有冲突，除非一个key对应多个值（除key外的其他信息）

　　　　3，自带排序功能（类似Radix Sort），中序遍历trie可以得到排序。

## ****缺点**：**

1，虽然不同单词共享前缀，但其实trie是一个以空间换时间的算法。其每一个字符都可能包含至多字符集大小数目的指针（不包含卫星数据）。

每个结点的子树的根节点的组织方式有几种。1>如果默认包含所有字符集，则查找速度快但浪费空间（特别是靠近树底部叶子）。2>如果用链接法(如左儿子右兄弟)，则节省空间但查找需顺序（部分）遍历链表。3>alphabet reduction: 减少字符宽度以减少字母集个数。,4>对字符集使用bitmap，再配合链接法。

2，如果数据存储在外部存储器等较慢位置，Trie会较hash速度慢（hash访问O(1)次外存，Trie访问O(树高)）。

3，长的浮点数等会让链变得很长。可用bitwise trie改进。

**bit-wise Trie**

类似于普通的Trie，但是字符集为一个bit位，所以孩子也只有两个。

可用于地址分配，路由管理等。

虽然是按bit位存储和判断，但因为cache-local和可高度并行，所以性能很高。跟红黑树比，红黑树虽然纸面性能更高，但是因为cache不友好和串行运行多，瓶颈在存储访问延迟而不是CPU速度。

**压缩Trie**

压缩分支条件：

1，Trie基本不变

2，只是查询

3，key跟结点的特定数据无关

4，分支很稀疏

若允许添加和删除，就可能需要分裂和合并结点。此时可能需要对压缩率和更新（裂，并）频率进行折中。

**外存Trie**

某些变种如后缀树适合存储在外部，另外还有B-trie等。

## ****应用场景**：**

**（1） 字符串检索**  
事先将已知的一些字符串（字典）的有关信息保存到trie树里，查找另外一些未知字符串是否出现过或者出现频率。  
举例：  
1，给出N 个单词组成的熟词表，以及一篇全用小写英文书写的文章，请你按最早出现的顺序写出所有不在熟词表中的生词。  
2，给出一个词典，其中的单词为不良单词。单词均为小写字母。再给出一段文本，文本的每一行也由小写字母构成。判断文本中是否含有任何不良单词。例如，若rob是不良单词，那么文本problem含有不良单词。

3，1000万字符串，其中有些是重复的，需要把重复的全部去掉，保留没有重复的字符串。

**（2）文本预测、自动完成，see also，拼写检查**

**（3）词频统计**

1，有一个1G大小的一个文件，里面每一行是一个词，词的大小不超过16字节，内存限制大小是1M。返回频数最高的100个词。

2，一个文本文件，大约有一万行，每行一个词，要求统计出其中最频繁出现的前10个词，请给出思想，给出时间复杂度分析。

3，寻找热门查询：搜索引擎会通过日志文件把用户每次检索使用的所有检索串都记录下来，每个查询串的长度为1-255字节。假设目前有一千万个记录，这些查询串的重复度比较高，虽然总数是1千万，但是如果去除重复，不超过3百万个。一个查询串的重复度越高，说明查询它的用户越多，也就越热门。请你统计最热门的10个查询串，要求使用的内存不能超过1G。  
(1) 请描述你解决这个问题的思路；  
(2) 请给出主要的处理流程，算法，以及算法的复杂度。

==》若无内存限制：Trie + “k-大/小根堆”（k为要找到的数目）。

否则，先hash分段再对每一个段用hash（另一个hash函数）统计词频，再要么利用归并排序的某些特性（如partial\_sort），要么利用某使用外存的方法。参考

　　“[海量数据处理之归并、堆排、前K方法的应用：一道面试题](http://www.dataguru.cn/thread-485388-1-1.html" \t "https://www.cnblogs.com/justinh/p/_blank)” [http://www.dataguru.cn/thread-485388-1-1.html](http://www.dataguru.cn/thread-485388-1-1.html" \t "https://www.cnblogs.com/justinh/p/_blank)。

　　“[算法面试题之统计词频前k大](http://blog.csdn.net/u011077606/article/details/42640867" \t "https://www.cnblogs.com/justinh/p/_blank)” [http://blog.csdn.net/u011077606/article/details/42640867](http://blog.csdn.net/u011077606/article/details/42640867" \t "https://www.cnblogs.com/justinh/p/_blank)

[算法导论笔记——第九章 中位数和顺序统计量](http://www.cnblogs.com/justinh/p/6518354.html)

**（4）排序**

Trie树是一棵多叉树，只要先序遍历整棵树，输出相应的字符串便是按字典序排序的结果。  
比如给你N 个互不相同的仅由一个单词构成的英文名，让你将它们按字典序从小到大排序输出。

**（5）字符串最长公共前缀**Trie树利用多个字符串的公共前缀来节省存储空间，当我们把大量字符串存储到一棵trie树上时，我们可以快速得到某些字符串的公共前缀。  
举例：  
给出N 个小写英文字母串，以及Q 个询问，即询问某两个串的最长公共前缀的长度是多少？  
解决方案：首先对所有的串建立其对应的字母树。此时发现，对于两个串的最长公共前缀的长度即它们所在结点的公共祖先个数，于是，问题就转化为了离线（Offline）的最近公共祖先（Least Common Ancestor，简称LCA）问题。  
而最近公共祖先问题同样是一个经典问题，可以用下面几种方法：  
1. 利用并查集（Disjoint Set），可以采用采用经典的Tarjan 算法；  
2. 求出字母树的欧拉序列（Euler Sequence ）后，就可以转为经典的最小值查询（Range Minimum Query，简称RMQ）问题了；

**（6）字符串搜索的前缀匹配**  
trie树常用于搜索提示。如当输入一个网址，可以自动搜索出可能的选择。当没有完全匹配的搜索结果，可以返回前缀最相似的可能。  
Trie树检索的时间复杂度可以做到n，n是要检索单词的长度，  
如果使用暴力检索，需要指数级O(n2)的时间复杂度。

**（7） 作为其他数据结构和算法的辅助结构**  
如后缀树，AC自动机等